MACHINE LEARNING
Science and Technology

PURPOSE-LED PUBLISHING™ Part of the MACHINE LEARNING Series

ACCEPTED MANUSCRIPT • OPEN ACCESS

Towards a self-driving trigger at the LHC: Adaptive response in real time

To cite this article before publication: Shaghayegh Emami *et al* 2026 *Mach. Learn.: Sci. Technol.* in press https://doi.org/10.1088/2632-2153/ae631f

Towards a Self-Driving Trigger at the LHC: Adaptive Response in Real Time

Shaghayegh Emami,[1,*] Cecilia Tosciri,[2] Giovanna Salvi,[1]
Zixin Ding,[3] Yuxin Chen,[3] Abhijith Gandrakota,[4] Christian Herwig,[1]
David W. Miller,[2,5] Jennifer Ngadiuba,[4] and Nhan Tran[4]

[1] *University of Michigan, Ann Arbor, Michigan 48109, USA*
[2] *Enrico Fermi Institute, University of Chicago, Chicago, IL 60637, USA*
[3] *Department of Computer Science, University of Chicago, Chicago, IL 60637, USA*
[4] *Fermi National Accelerator Laboratory, Batavia, IL 60510, USA*
[5] *Department of Physics, Kavli Institute for Cosmological Physics, University of Chicago, Chicago, IL 60637, USA*

Real-time data filtering and selection – or *trigger* – systems at high-throughput scientific facilities such as the experiments at the Large Hadron Collider (LHC) must process extremely high-rate data streams under stringent bandwidth, latency, and storage constraints. Yet these systems are typically designed as static, hand-tuned menus of selection criteria grounded in prior knowledge and simulation. In this work, we further explore the concept of a *self-driving trigger*, an autonomous data-filtering framework that reallocates resources and adjusts thresholds dynamically in real-time to optimize signal efficiency, rate stability, and computational cost as instrumentation and environmental conditions evolve. We introduce a benchmark ecosystem to emulate realistic collider scenarios and demonstrate real-time optimization of a menu including canonical energy sum triggers as well as modern anomaly-detection algorithms that target non-standard event topologies using machine learning. Using simulated data streams and publicly available collision data from the Compact Muon Solenoid (CMS) experiment, we demonstrate the capability to dynamically and automatically optimize trigger performance under specific cost objectives without manual retuning. Our adaptive strategy shifts trigger design from static menus with heuristic tuning to intelligent, automated, data-driven control, unlocking greater flexibility and discovery potential in future high-energy physics analyses.

* shghygh@umich.edu; Author to whom any correspondence should be addressed.

CONTENTS

1. INTRODUCTION

In scientific domains characterized by high data throughput, algorithms tasked with filtering and selecting data for storage and analysis – frequently referred to as *trigger algorithms* – must operate under extreme bandwidth, computation, and storage capacity requirements: data rates of 10^6–10^9 per second (MHz–GHz), bandwidths of 10^{12}–10^{15} bits per second (Tb/s–Pb/s), and

storage volumes of 10^{15}–10^{18} bytes per year (PB/y–EB/y). To meet these demands, data filtering algorithms must be highly selective, often operating with selection rates at the level of 1 part in 10^5. With sensors numbering from the millions to the billions, these data processing tasks are also moving ever closer to the edge (i.e., closer to the sensor itself), further stratifying and complexifying the data filtering process.

Despite these complex environments, many data filtering paradigms for discovery science domains, such as at high-energy particle colliders like the Large Hadron Collider (LHC), rely heavily on detailed prior knowledge of the feature space being probed, precise simulations of the system and instrumentation, and highly modularized approaches to optimization. Consequently, the design and operation of these experiments require enormous manual effort and human expertise, and also introduce inherent inefficiencies such as redundant feature labeling schemes and cost-ineffective algorithm execution. Furthermore, both known and unknown operator biases present in these algorithms raises the fundamental concern of whether rare or new phenomena are being missed entirely.

In this work, we further explore the concept of a *self-driving trigger system*: a hardware-aware autonomous data processing and filtering system. The core objective of this work is to establish systems capable of continuously and autonomously learning what data to collect and record, and identifying and responding dynamically to changing conditions and new observations. Such a system aims to provide a *cost-effective explanation* for why data are saved or discarded, subsequently updating its decision framework in a hardware- and resource-aware manner. This effort builds on advances in reinforcement learning, explainable artificial intelligence (AI), and control optimization to autonomously adapt and refine selection policies.

In the context of triggers for collider detectors, machine learning (ML) algorithms have become increasingly widespread in recent years [1–3]. Most notably, real-time anomaly detection algorithms have been implemented by the ATLAS and CMS collaborations, which use static neural networks to evaluate and record interesting events without dependence on physical priors [4, 5]. On the other hand, ML techniques have been used to address changes in detector response over time in the context of data-quality monitoring in CMS [6–8], but lack the ability to reconfigure the experiment on the fly. However, precedent for real-time control does exist in related experimental contexts, including accelerated beam control and adaptive optics in telescopes [9, 10]. Recent studies [11, 12] have proposed the integration of adaptive algorithms into the trigger decision chain to dynamically reallocate bandwidth and adjust thresholds in response to real-time changes in experimental conditions.

Building on these efforts, we introduce a benchmark framework to evaluate autonomous trigger control strategies under realistic scenarios at collider experiments. Specifically, we focus on the sets of data selection rules, so-called *trigger menus*, that encode the selection criteria and constraints. These menus are often designed based on predefined science goals and expected detector performance. While robust under stable conditions, such menus are effectively static and are inherently inflexible: they do not automatically adapt to time-dependent variations in the operation of the instrumentation, the conditions that evolve throughout a data taking period, or even in the potential appearance of new features in the data that were previously absent or not statistically significant. Consequently, the system may fail to identify rare and potentially interesting data or misinterpret otherwise uninteresting data. In the context of large scientific facilities, updates and adjustments to these complex filtering rules to account for these issues typically necessitate extensive human intervention and can take weeks or even months to assess and implement. This motivates the development of smarter, more autonomous systems that can dynamically respond to real-time changes in experimental conditions and adapt to minimize biases and maximize the effectiveness of the data collection and discovery potential.

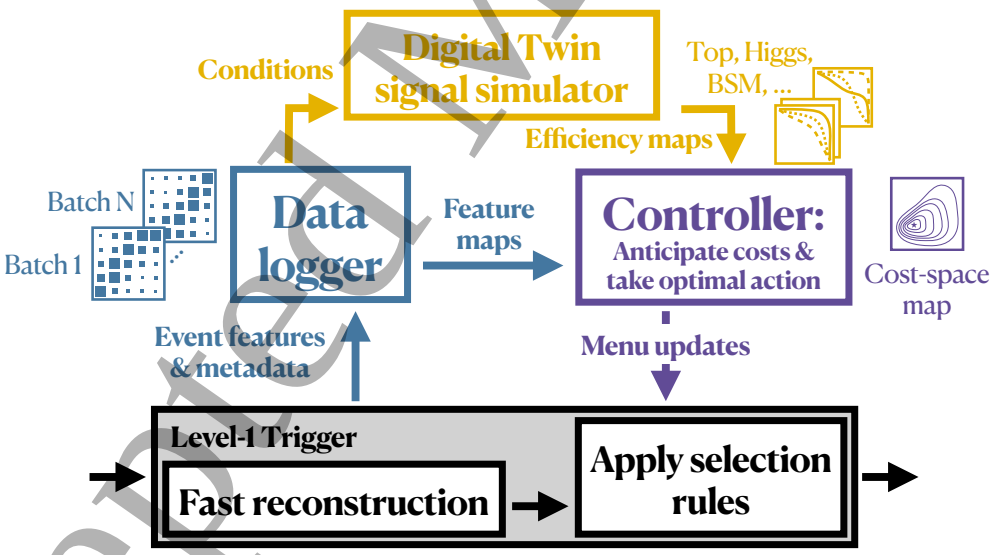


FIG. 1: Schematic overview of the adaptive trigger control system. A cost-aware agent receives batches of collision data, evaluates trigger performance and computational costs, and updates L1 trigger thresholds and bandwidth allocations in real time through a feedback-driven control loop.

We envision the self-driving trigger as an *agent* that interacts directly with the *environment* of the hardware-based Level-1 (L1) trigger system operations, as sketched in Fig. 1. L1 trigger algorithms build particle candidates in each event in real-time, after which a Global Trigger system

applies a set of selection rules that determine which collision events will be permanently recorded. An agent should generically log the inputs to these decisions, including the reconstructed event features but also potentially metadata related to trigger and even the detector and accelerator configuration. This data source allows the agent to evaluate many factors, including short- and long-term trends in the selection rates observed in data, correlations among the decisions of the active trigger algorithms, and auxiliary factors such as the computation cost of processing the selected events. In addition, the agent might consult a real-time simulation of the experimental setup – often referred to as a *digital twin* – to understand its response to hypothetical signal processes of interest. This could be implemented using a modest library of Monte Carlo (MC) events or parametrized responses that encompasses a range of expected experimental conditions. Access to such simulation would allow the agent to estimate the trigger efficiency of a candidate menu for key signatures, based on events with similar conditions to those currently seen in data. A flexible reward mechanism allows each of these considerations to factors into the agent's real-time prescriptions. In this work, we focus on updated menu thresholds, but such prescriptions could also be generalized to include wholesale replacement or retraining of selection algorithms themselves.

We test this approach using CMS Open Data and corresponding Monte Carlo (MC) simulations [13], focusing on the task of selecting events with interesting jet activity. A simplified trigger menu includes a traditional algorithm to select the most energetic events, as well as a modern approach for unsupervised anomaly detection (AD) with neural networks. Our results establish a proof-of-concept that an adaptive trigger systems present a powerful and robust innovation over the status quo, which can be easily tailored to prioritize a range of different operational objectives including the efficiency for known or expected signals, sensitivity to anomalies, and expected burden on computational resources.

The remainder of this paper is organized as follows: Section 2 introduces the datasets, simulation framework, and benchmark tasks. Section 3 describes the baseline H_{T} and anomaly detection trigger algorithms. In Section 4, we characterize the performance and limitations of a fixed trigger menu in simulated event streams. Sections 5–7 develop and compare real-time control strategies, starting from the independent adjustment of each algorithm and extending to a single controller that allocates bandwidth across several triggers based on different cost and physics objectives. Section 8 then demonstrates the behavior of these controllers on publicly available collision data from the CMS experiment at the LHC. Section 9 presents a direct comparison between the Monte Carlo simulation and the real data's approach, assessing the robustness and transferability of the proposed control strategies. Finally, Section 10 summarizes the main conclusions of this work and

outlines the possible directions for future developments.

2. DATASETS AND SIMULATION FRAMEWORK

The CMS [13] and ATLAS [14] Collaborations have each released datasets of reconstructed collision events that are publicly accessible through the CERN Open Data portal [15]. The availability of the data is essential for the study of real-time detector control described in this manuscript, as it relies heavily on variations in detector conditions that may only be present in the real data. Monte Carlo (MC) simulations of various physical processes are also provided to facilitate the interpretation of this dataset and are generated to reflect, on average, the experimental conditions in which the collision data were recorded.

Our studies make use of the CMS Open Data dataset corresponding to proton-proton collisions recorded at a center-of-mass energy $\sqrt{s} = 13$ TeV in 2016. Specifically, we use data collected during periods G and H of Run 2, amounting to a total of 16.4 fb^{-1} [16, 17]. To date, this is the most recent major dataset that has been released by either of the experimental collaborations. The collision datasets and MC simulation samples are processed with detector simulation and reconstruction algorithms matching the 2016 data taking conditions, including the pileup profile, beam energy, and detector calibration parameters.

In this benchmark scenario, we implement and test our approach using two representative trigger paths:

- H_{T} trigger, collecting events with large scalar sums of jet transverse momenta (p_T), $H_T = \sum_{j \in \mathcal{J}} p_T^j$, $\mathcal{J} = \left\{ j \,\middle|\, p_T^j > 20 \text{ GeV},\ |\eta^j| < 2.5 \right\}$.
- Anomaly detection trigger, AD, trained to identify events that deviate from known background topologies.

Together, these triggers capture the dual challenge of selecting rare signals and carrying out this task based on a set of rich but inherently shifting (pileup sensitive) observables. While the H_{T} trigger follows the classic heuristic of controlling rates by selecting the most energetic events, the AD trigger serves as an alternate strategy that may be particularly promising in selecting novel or poorly modeled processes. A detailed description of each algorithm is presented in Section 3.

We validate our strategy using a combination of CMS data and simulation and provide a reproducible framework for evaluating adaptive trigger control strategies. The CMS Open Data used in this study are taken from the `ZeroBias` stream, which records events triggered solely by the

presence of colliding bunch crossings, without physics-based selection. These events are therefore largely unbiased with respect to specific physics signatures, although they are still recorded after the CMS trigger system; for example, the effective event rate for the run used in this study (Run 283408) is approximately 34 Hz.

In the following, we refer to the `ZeroBias` sample as the "background data" sample. Such triggers are heavily 'pre-scaled', whereby only a small fraction of events passing the trigger logic are actually recorded. Because this stream is typically used for experimental diagnostics rather than physical measurements, the selected events make up only a small fraction of the total trigger rate.

At the LHC, a fill refers to a period of time during which beams are injected, accelerated, and kept in stable collisions, typically lasting several hours. Within each fill, data are collected in multiple runs, which correspond to contiguous periods of data taking defined by stable detector and trigger configurations. Considering sequential data from a single run within a fill allows us to track the gradual evolution of beam conditions and pileup over time with relatively consistent settings. Thus, in order to mimic a system capable of adjusting trigger rules in real-time, we primarily consider the largest run contained in the dataset (Run 283408), which includes 1.99 million events in the background data stream.

Monte Carlo samples of events corresponding to Standard Model (SM) and Beyond Standard Model (BSM) processes were also used to design the real-time control strategy and assess its performance. Specifically, the following processes have been considered:

- Simulated minimum bias collision events, corresponding to an inclusive sample of inelastic pp interactions, with pileup and detector conditions matching the 2016 data-taking period [18], simulated with Pythia8 [19].

- Production of top quark pairs ($t\bar{t}$), used as a representative Standard Model (SM) benchmark process for the control design [20]. Top-quark pair events provide a realistic and sufficiently complex topology, featuring sizable hadronic activity and a variety of common analysis objects such as (b-)jets, leptons, and missing transverse momentum. This makes them a convenient reference scenario to study the behaviour of the control system in the presence of nontrivial event-by-event fluctuations. The hard-scatter process is simulated at next-to-leading order in the strong coupling constant using Powheg [21], interfaced with Pythia8 for parton showering and hadronization.

- Exotic decays of the Standard Model Higgs boson (h) to new pseudo-scalar particles (a),

simulated using MadGraph [22] interfaced with Pythia8, for a range of a masses between 12 and 60 GeV. Both pseudo-scalars are assumed to decay promptly to b-quark pairs with a 100% branching ratio, resulting in a $h \to aa \to b\bar{b}b\bar{b}$ (simply referred to as $h \to 4b$ in the following sections) final state [23].

To design and evaluate the real-time control strategy, it is essential to model how the collision environment evolves during a typical LHC fill. In real data, the number of primary vertices (N_{PV}) typically decreases over time, as the number of protons in the beam drops due to inelastic collisions and beam losses. This reduction in luminosity leads to lower pileup and fewer reconstructed vertices per event. Figure 2 illustrates this behavior in CMS Run 283408 by showing the average N_{PV} reconstructed as a function of time, shown as the fraction of the run that has elapsed.

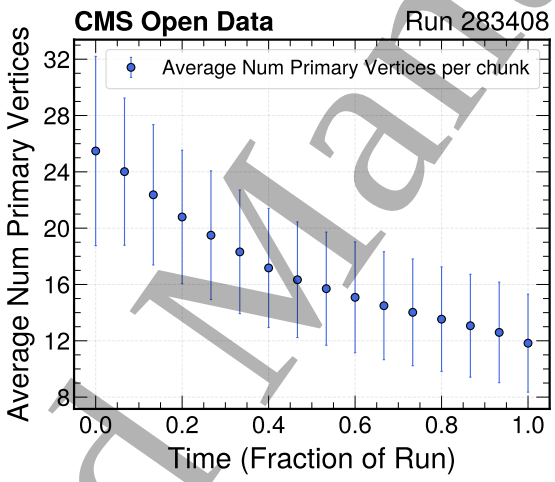
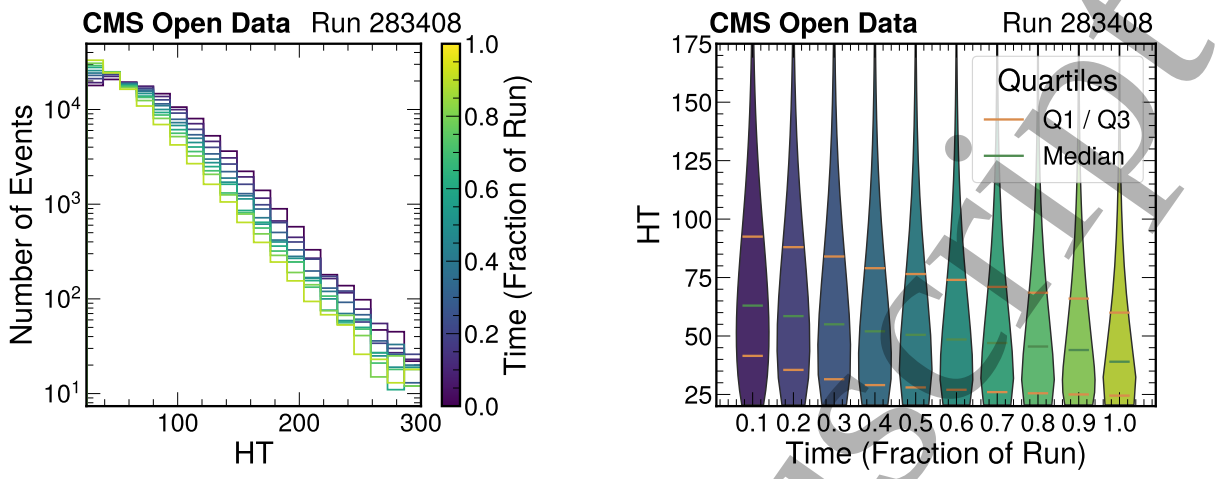


FIG. 2: The average number of primary vertices (N_{PV}) reconstructed at various points throughout Run 283408 collected by the CMS Experiment in 2016. Time is measured as the fraction of the run that has elapsed, with the gradual decrease of N_{PV} reflecting the gradual drop in luminosity and pileup as the fill progresses.

A similar time evolution pattern is observed in the variable H_T, the scalar sum of trigger jet p_T, defined in detail in Section 3. As shown in Fig. 3a, the energy distribution gradually shifts toward lower values as the fill progresses. To visualize this trend more clearly, Fig. 3b shows the same distributions as a series of violin plots, highlighting the evolution of the median and quartiles of H_{T} as a function of the run-time fraction. These distributions convey the extent to which the experimental trigger data undergo significant changes (in both scale and shape) over the course of a single run, and the inherent time-dependence of an H_{T} trigger algorithm.

Events produced with MC simulation are inherently unordered, with each event's value of N_{PV}

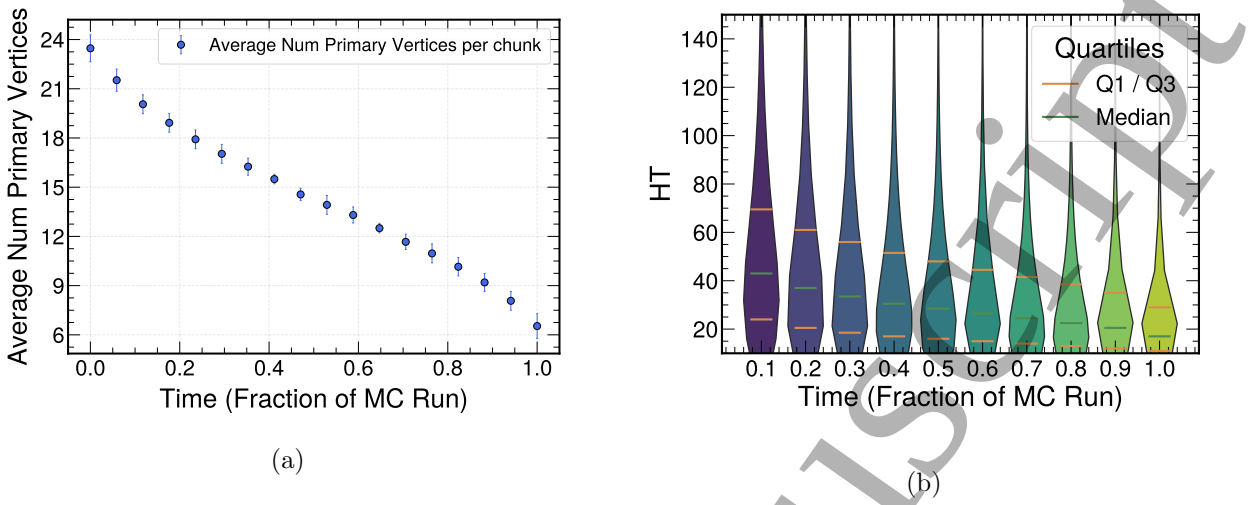


(a) (b)

FIG. 3: Evolution of H_{T} during a 2016 LHC run recorded by CMS (Run 283408). (a) Histograms show the progressive shift of the H_{T} distribution toward lower values over the course of the fill. (b) Violin plots display the evolution of the median and quartiles of H_{T} as a function of the run-time fraction, highlighting the correlation between H_T, time, and N_{PV} through variations in the pileup energy density.

having been randomly sampled from the aggregate distribution measured over the full 2016 data sample. To develop and test a real-time control strategy, an ad hoc ordering must be enforced on the simulated datasets that captures the general features observed in collision data. To this end, we emulate the gradual decrease in the average number of pileup collisions over time within a fill by enforcing an approximate N_{PV}-sorting of the simulated events. However, because the true value of N_{PV} follows a Poisson distribution and therefore fluctuates on an event-by-event basis, the same effect is naturally imposed in our simulation samples.

Arbitrary levels of variation can be studied in simulation with this approach. Figures 4a and 4b demonstrate that the time variable induces time-dependent trends in simulation samples similar to those observed in data. To summarize our benchmark strategy, the input to the hardware-based L1 trigger logic is treated as background, and the trigger menu is configured to pass 99.75 percentile of events using thresholds on HT and AD menu items. These thresholds are dynamically adjusted by a controller layer operating on top of the L1 logic, which autonomously adapts the menu as a function of time. The present setup focuses on hadronic final states, which represent one of the most challenging environments for trigger design due to their high rates and complex backgrounds. However, the framework is not intrinsically tied to jet signatures. The control strategy operates at



(a) (b)

FIG. 4: Introducing a time index in simulated events by ordering them according to the N_{PV} distribution, smeared with Poisson noise to mimic realistic fluctuations. (a) N_{PV} versus time, illustrating how the constructed index reproduces the characteristic pileup evolution seen during an LHC fill. (b) Violin plots of H_{T} versus time, showing the expected decrease of hadronic activity over the fill as the pileup drops.

the level of trigger paths and their associated rates and efficiencies, and could therefore be applied to triggers targeting a wide range of physics signatures, including those based on leptons, photons, or missing transverse momentum. Two types of background are considered: simulated minimum bias events and real `ZeroBias` data. While real data naturally carries time information, simulated events do not; therefore, we exploit the time evolution using the number of primary vertices, N_{PV}, as a proxy for time. Simulated background events are therefore sorted and smeared according to N_{PV}, producing a time-indexed background stream that mimics realistic running conditions.

The controller additionally has access to a simulated signal library, consisting of $t\bar{t}$ and $h \to 4b$ samples, which is used exclusively to evaluate the performance of the evolving trigger menu. The treatment of signal samples depends on the choice of background. When using a simulated background, signal events are ordered using the same N_{PV}-based time indexing, and for the case of real data as background, each data event is randomly matched to signal events with the same N_{PV}. Our benchmark, therefore, begins with studying the simulated background to develop and validate the framework and controller strategies, and subsequently transitions to the data background with matched signal samples to evaluate methods under real experiment conditions.

3. FEATURES OF THE TRIGGER DATA AND ALGORITHMS

We employ two representative trigger algorithms that represent distinct operational challenges to benchmark the behavior of static versus adaptive strategies. The first is the H_{T} trigger, a standard algorithm to collect events with significant hadronic activity in the ATLAS and CMS experiments, which generally suffers from sizable pileup dependence. The second is an anomaly detection (AD) trigger, based on a machine learning approach designed to capture rare or *a priori* unknown features in collision events with minimal bias toward any specific new-physics model. These triggers reflect complementary experimental considerations and physics priorities: one is based on known signatures and relative operational stability, while the other aims to capture new signatures even if an end-to-end analysis workflow with a fully optimized trigger strategy has not yet been fully defined.

In the current ATLAS and CMS experiments, hardware-based L1 trigger systems perform the initial stage of real-time event reconstruction and selection. The L1 trigger processes data with latencies of a few microseconds and reduces the raw 40 MHz collision rate to around 100 kHz. This drastic reduction is achieved through coarse event reconstruction based on calorimetric and muon data. For hadronic events, L1 jets are reconstructed from regional energy deposits using simplified clustering algorithms. Due to the presence of overlapping pp collision byproducts, raw jet energies must be corrected, typically by estimating and subtracting pileup contributions on an event-by-event basis. Further technical details on jet reconstruction and pileup mitigation at L1 can be found in Refs. [24, 25].

Building on the datasets described in the previous section, comprising both MC simulated background and signal samples, as well as real collision data, we extract the event-level jet information. This includes up to eight L1 jets, with per-jet features encoded in an array encompassing detector pseudorapidity (η), azimuthal angle (ϕ), and transverse momentum (p_T), together with the number of primary vertices associated with the event N_{PV}.

These features serve as inputs for the two trigger algorithms, and enable a direct comparison between a fully-supervised trigger and a data-driven approach. Figs. 5a, 5b, 5c, 5d, show the distributions of key observables for different datasets, including H_T, jet multiplicity, jet p_T and missing transverse hadronic energy H_T^{miss}, defined as the magnitude of the vector sum of jet

momenta in the (x, y) plane:

$$H_{\mathrm{T}}^{\mathrm{miss}} = \|\vec{p}_{\mathrm{miss}}^{\,\mathrm{T}}\| = \sqrt{\Big(\sum_j p_{T,j}\cos\phi_j\Big)^2 + \Big(\sum_j p_{T,j}\sin\phi_j\Big)^2},$$

$$\vec{p}_{\mathrm{miss}}^{\,\mathrm{T}} = \sum_j p_{T,j}\big(\cos\phi_j\,\hat{x} + \sin\phi_j\,\hat{y}\big).$$

The two triggers allow us to isolate the effects of different trigger strategies. In particular, while the H_{T} trigger prioritizes highly energetic events with a simple energy threshold, the AD trigger is more flexible, selecting events with the most anomalous scores as evaluated by a custom machine learning model.

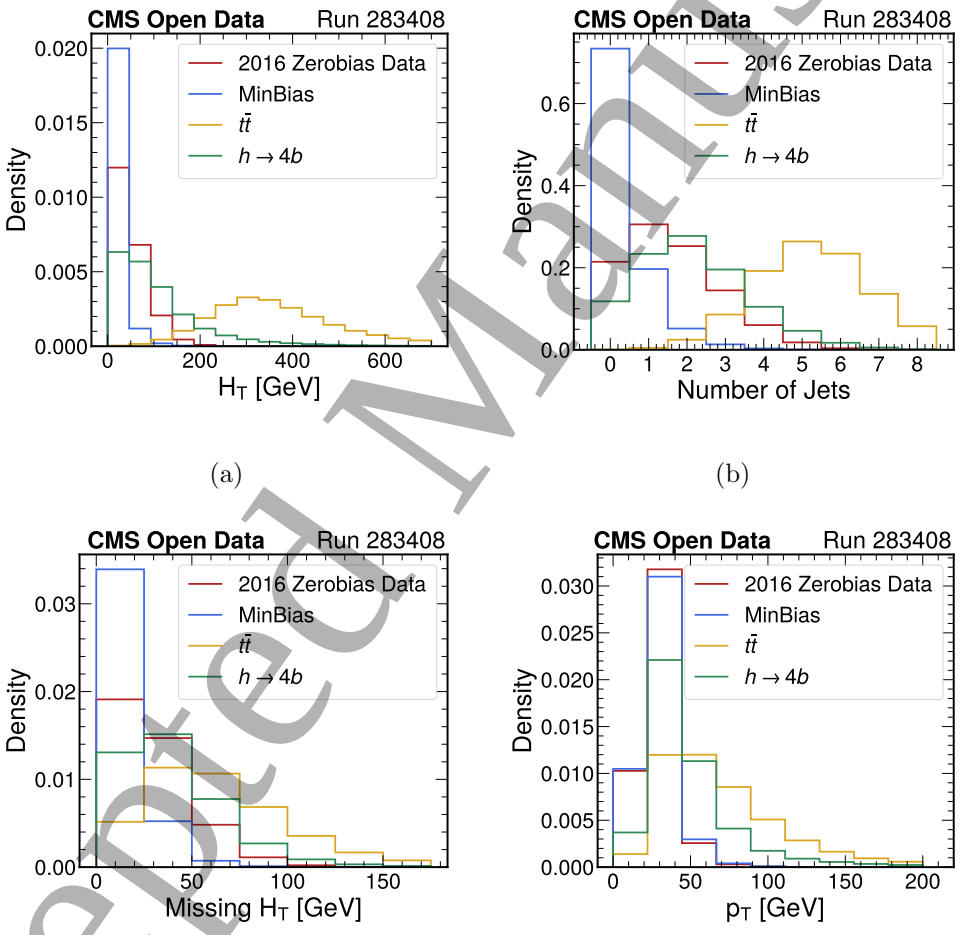


FIG. 5: Comparison of key per-event features across different samples: real collision data, minimum bias simulation, SM $t\bar{t}$ events, and an exotic benchmark process $h \to 4b$. These distributions provide insight into the kinematic and topological differences that the trigger strategies aim to capture.

3.1. Anomaly Detection Algorithm

We use an autoencoder (AE) model to compute the anomaly score for each event. As the core component of the AD trigger, the AE is essential for assessing the trigger's sensitivity to rare physics signals and its robustness under varying experimental conditions. Our approach to anomaly detection at the L1 is inspired by state-of-the-art methods employed by CMS [5] and ATLAS [4] that leverage AE models.

Autoencoders [26, 27] are self-supervised neural networks trained to reproduce their inputs at the output layer. They consist of an encoder f_θ that maps an input feature vector $x \in \mathbb{R}^D$ to a lower-dimensional latent representation $z = f_\theta(x) \in \mathbb{R}^d$ with $d \ll D$, and a decoder g_ϕ that maps back to a reconstruction $\hat{x} = g_\phi(z)$. The parameters (θ, ϕ) are optimized to minimize a reconstruction loss over background events, such that the latent space captures the structure of the most frequent (background) patterns. At test time, events that are well represented by this learned background manifold are reconstructed with small error, while rare (i.e., anomalies) yield larger reconstruction error. We use the reconstruction error as the anomaly score, defined as the mean squared error (MSE) between each input array and its reconstruction.

We adopt a fully connected multi-layer perceptron with a single encoding and decoding layer, consistent with prior work and compatible with a compact hardware implementation (e.g., with `hls4ml` [28]) for trigger systems across a variety of scales. In our implementation, the AD model takes a 25-dimensional input representation of each event, compresses it to a d-dimensional latent space, and then reconstructs it back to 25 dimensions. The model can be trained with either real collision data or MC simulation of minimum bias events to learn the typical background features. Events that deviate from these learned features are flagged as anomalous from the L1-trigger perspective and provide promising candidates for offline follow-up studies, where additional information and more powerful analysis techniques are available.

A key aspect in our input representation is the choice to provide the model with a proxy for the time variable, in this case N_{PV}. This helps the model learn the detector response to pileup variations more effectively. This is a natural choice, as all standard hadronic trigger algorithms must attempt to 'calibrate away' pileup dependence (through energy density corrections, for example), even though ultimately some residual dependence almost always remains. A schematic diagram of the AD model is presented in Fig 6. Each event is represented by up to eight jets, characterized by their kinematic properties (η, ϕ, and p_T), together with the event's N_{PV}, resulting in a total of 25 input features.

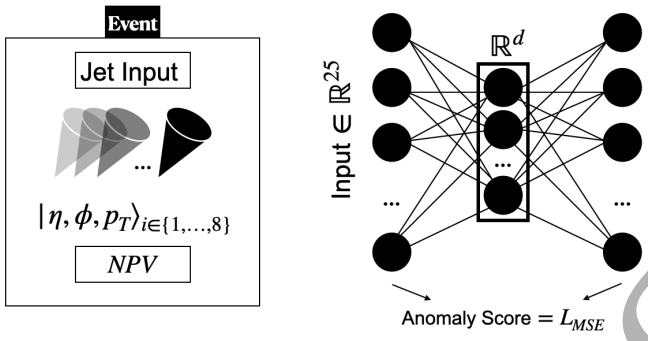


FIG. 6: Schematic representation of the autoencoder model used for anomaly detection.

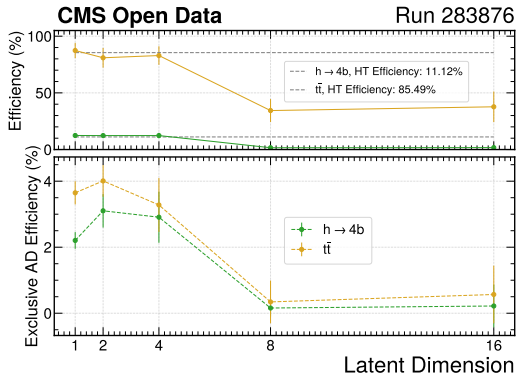
A guiding consideration for this study is to develop an AD model that is comparable in size and scope to the actual models that have been implemented so far in the trigger hardware of the ATLAS and CMS experiments. We thus seek to construct a compact and efficient model that can effectively perform AD tasks on the datasets described above, motivating a simple architecture that still satisfies the fundamental requirements of the project, and has a sufficient amount of data to be trained.

Subject to these conditions, the anomaly detection model used in the trigger menu is trained on background samples. In simulation studies, the model is trained on an independent simulated background sample, while the data model is trained using the second longest run available for that year (Run 283876 [16]). This is to ensure sufficient statistics while avoiding overlap with the evaluation sample for control studies [17].

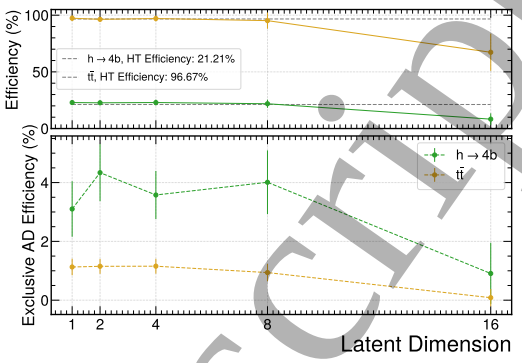
To this end, we trained a suite of models with varying latent dimensions and determined an AD score threshold for each that would achieve a rejection factor of 400 the test background sample, commensurate with typical L1 trigger requirements. The efficiency for signals to exceed these thresholds is shown in Figs. 7a and 7b for both $t\bar{t}$ and $h \to 4b$ samples. These figures also show the fraction of signal events selected exclusively by the AD path by applying the same background rejection threshold to both H_{T} and AD.

It is observed that models with smaller dimensions show higher signal efficiency and larger exclusive contributions relative to the H_{T} trigger.

While the available training statistics can influence the performance of higher-dimensional models, the presence of detector noise and data-driven fluctuations means that learning detailed representations does not necessarily improve discrimination. In practice, more compact latent spaces provide a more stable and robust separation between signal and background in data. Therefore,



(a) AD model trained on data background.



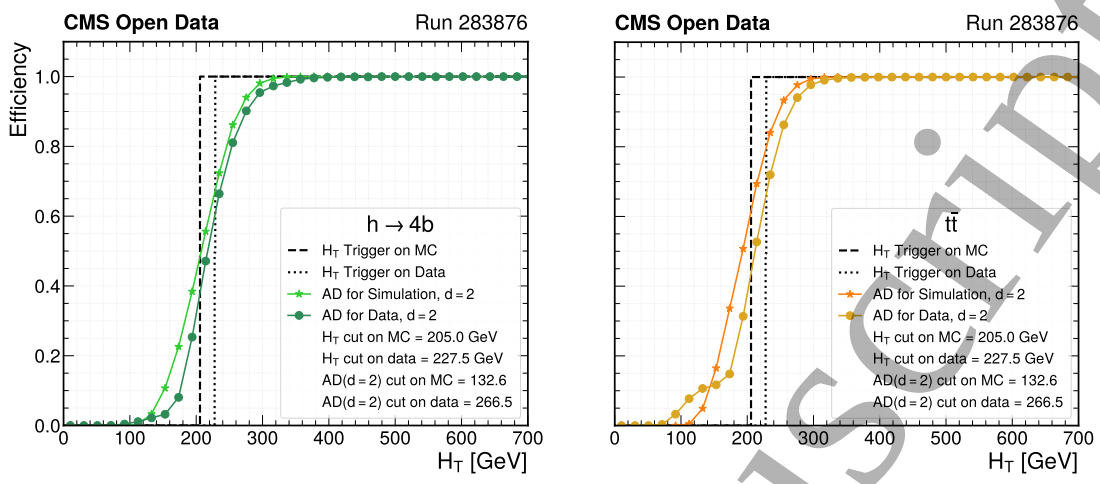
(b) AD model trained on simulation background.

FIG. 7: Signal efficiency as a function of latent dimension for $t\bar{t}$ (yellow) and $h \to 4b$ (green) simulated signals.

given the consistent performance, lower resource requirements, and resilience to noise, for the purposes of this benchmark study, the configuration with $d = 2$ is adopted as the baseline.

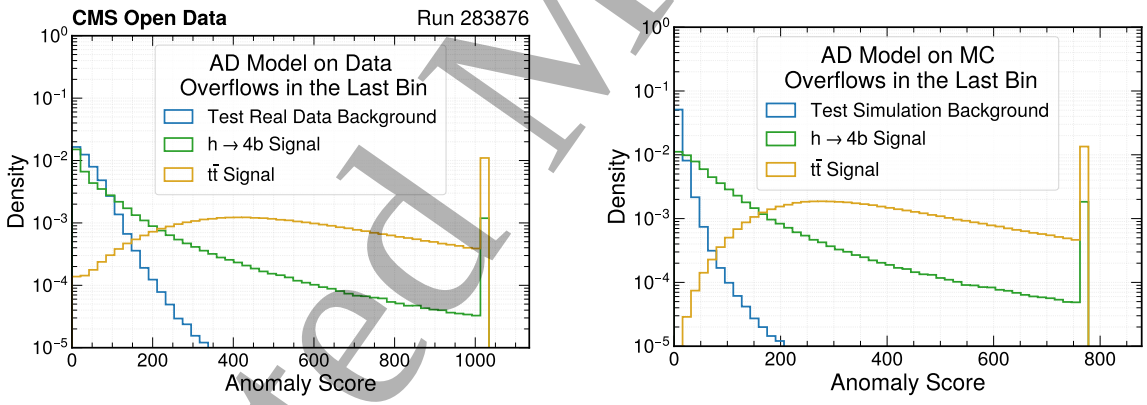
Figure 8 compares the signal efficiency obtained using an H_{T} trigger versus an AD trigger on simulated or real data events. The results show that for an H_{T} threshold and an AD score cut chosen to yield a consistent L1 rate reduction, the AD trigger recovers signal efficiency in the region below the H_{T} threshold, where the H_{T} trigger rejects essentially all signal events. When operated in parallel with H_{T}, the AD trigger therefore retains additional sensitivity at lower energy scales. As shown in Fig. 8, this efficiency recovery is observed for both signal models, reaching up to about 50% below the H_{T} cut. The effect is more pronounced for the $t\bar{t}$ sample, where the efficiency gain extends to slightly lower H_{T} values. This reflects the fact that fully hadronic $t\bar{t}$ events typically contain a higher jet multiplicity and a topology that is easier for the anomaly detection algorithm to distinguish from background. In contrast, the $h \to 4b$ signal more closely resembles the background in terms of hadronic activity in the very low-H_{T} region, resulting in a comparatively smaller efficiency recovery there.

Finally, Fig. 9 shows the anomaly score distributions for the models trained on simulation and real data. The high-score tail is most relevant for signal efficiency: improved separation between signal and background in this region translates directly into higher signal acceptance.



(a) $h \to 4b$ (b) $t\bar{t}$

FIG. 8: Efficiency as a function of H_{T} for (a) $h \to 4b$ and (b) $t\bar{t}$ simulated events. This is a demonstration of signal efficiency improvement in the lower p_T kinematic region when using anomaly detection.



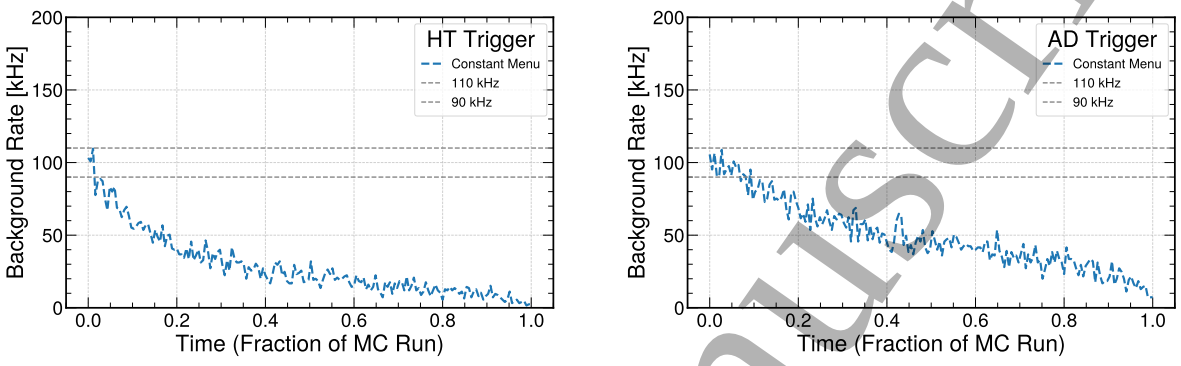
(a) Data background. (b) Simulation background.

FIG. 9: In both cases, there is a clear separation between the background and the two signal samples: $t\bar{t}$ and $h \to 4b$.

4. PERFORMANCE OF A FIXED TRIGGER MENU IN SIMULATED EVENTS

In this section, we analyze the behavior of a fixed trigger menu on the simulation background without any control mechanisms. The study is done by allocating all of the bandwidth to only one item, which is either the H_{T} or the AD. The goal is to investigate how their distributions evolve over time and how the trigger rates and signal efficiencies are affected by evolving pileup

conditions in our MC samples. This is the first step to formulate the problem before introducing any solutions. The trigger menu encompasses a set of rules that determine whether an event is saved or rejected, which are determined in our setup by thresholds on an AD score or H_{T}. To begin, we consider a constant trigger menu, where these thresholds do not vary with time.



(a) H_{T} trigger rate. (b) AD trigger rates.

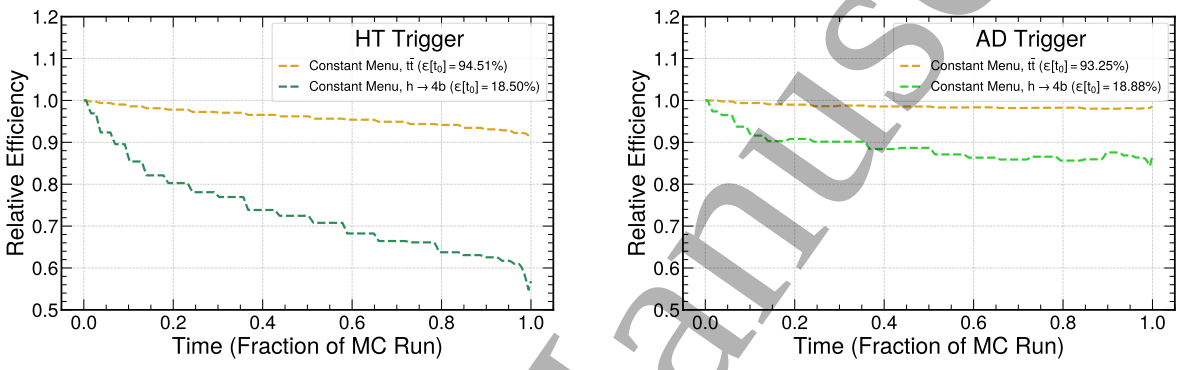
FIG. 10: Evolution of background trigger rates under a constant threshold strategy on MC background sample. (a) H_{T} trigger. (b) AD trigger. Dashed lines provide visual guides. Significant rate drifts are observed due to evolving detector conditions.

In Fig. 10, the fixed menu thresholds are determined by requiring a 400x rejection factor for events in the first few batches of data. In this work, a *batch* refers to a fixed number of events from the data stream, processed sequentially. As the focus of this study is L1 triggers, we target an acceptance rate of maximally 100 kHz from the background events, assuming 40 MHz collisions. Assuming that this is the maximum tolerable rate given external constraints on the system (bandwidth and storage, for example) and our available resources, then ensuring it does not exceed the limit in the first batch serves as a reasonable safeguard that it will remain within bounds for the rest of the run.

For the H_{T} trigger, we observe a significant rate decrease as the N_{PV} decreases over time. This behavior reflects the expected sensitivity of H_{T} to pileup conditions: fewer interactions per bunch crossing lead to reduced jet activity and lower H_{T} values, an increasing number of which ultimately fall below the fixed threshold. This familiar decrease in trigger rates over the course of a fill is a well-documented phenomenon in LHC operations [29, 30]. A similar trend is observed for the AD trigger. This suggests that its anomaly score inherits a conventional pileup dependence, analogous to traditional trigger items.

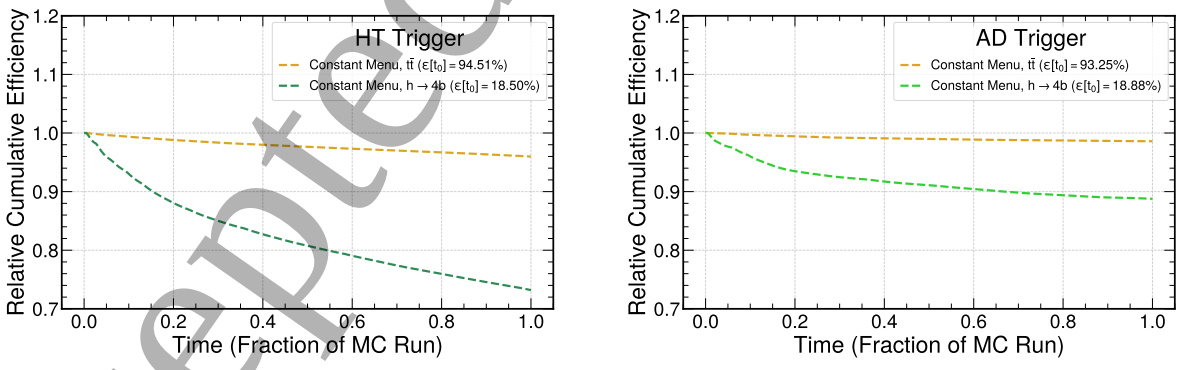
Figure 11 shows the relative change in signal efficiency as a function of time, $\frac{\epsilon(t)}{\epsilon(t=0)}$, computed

per batch of 50,000 events. This quantity reflects how the instantaneous trigger efficiency evolves, which is highly sensitive to changes in event characteristics such as pileup conditions. Figure 12 complements this view by showing the relative change in cumulative signal efficiency instead, i.e., the cumulative fraction of signal events passing the trigger up to a given point in the run. This metric smooths out short-term fluctuations and highlights longer-term trends in trigger performance. These figures show how the ability of H_{T} and AD triggers to capture signal events degrades over time due to pileup evolution.



(a) H_{T} trigger. (b) AD trigger.

FIG. 11: Relative signal efficiency (a) H_{T} trigger, (b) AD trigger. This quantity reflects the instantaneous trigger efficiency and is highly sensitive to changes in event characteristics such as pileup conditions.



(a) H_{T} trigger. (b) AD trigger.

FIG. 12: Relative cumulative signal efficiencies as a function of time (a) H_{T} trigger, (b) AD trigger. The plots show how static thresholds lead to degraded performance as pileup decreases.

Ultimately, the loss of efficiency highlights a key limitation of static trigger menus: without adaptation, valuable signal events are increasingly missed as detector and beam conditions evolve.

5. REAL-TIME CONTROL FOR INDIVIDUAL TRIGGER RULES

The primary objective for a trigger controller is to achieve a stable data collection rate over time (with varying detector conditions) through the dynamic adjustment of trigger rules. Without a perfect model for all aspects of the changing experimental context (i.e., the environment with which the controller interacts), we invoke the classic solution from control theory of a linear feedback mechanism. Here, observed deviations from the target event rate in one batch of data can lead to proportionate adjustments to trigger thresholds in the subsequent batch.

The Proportional-Integral-Derivative (PID) controller is a natural extension of this approach found across a broad range of applications in daily life where parameters must be regulated in evolving environments [31, 32]. The PID algorithm computes corrections based on the deviation $e(t)$ from a target value:

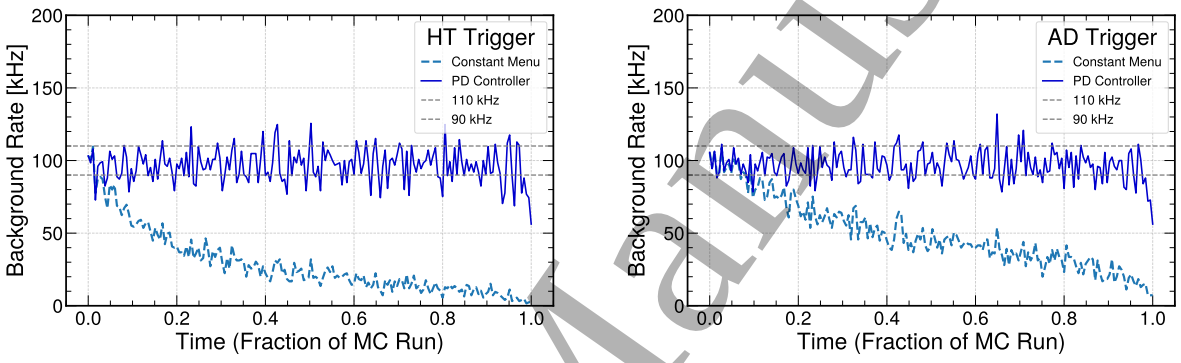
$$\Delta u(t) = K_p\, e(t) + K_d \frac{de(t)}{dt} + K_I \int_0^t e(t')dt',$$

where K_p, K_d, and K_I are tunable constants governing proportional, derivative, and integral corrections. Typically, the proportional term sets the dominant behavior, while the derivative and integral terms contribute to predictive adjustments and error accumulation corrections, respectively. In our toy model, the regulated parameter is the trigger threshold for a single algorithm (either H_{T} or AD), which seeks to maintain a fixed background rate, taken to be $r_t = 100$ kHz as in Section 4.

Through comparisons of controllers with various gain value configurations, we find no significant impact from the presence of integral term feedback ($K_I[\sum_{t=t_0}^{t=t_N} e(t)]$). Consequently, we implement the simpler PD loop to regulate menu items dynamically, noting that even the improvement from including a derivative ($K_d[e(t_N) - e(t_{N-1})]$) is relatively modest. Optimal values of the controller gain constants K_p and K_d are obtained from grid searches, which are performed separately for each trigger algorithm.

Figures 13, 14, and 15 summarize the impact of the PD controller on background stability and signal efficiency for both the H_{T} and AD triggers. Figure 13 shows the background trigger rates under PD control compared to fixed menus. In both the H_{T} (Fig. 13a) and AD (Fig. 13b) cases, the controller compensates for the time dependence of the collision environment and maintains the rate within the desired tolerance band, while the fixed-threshold menus exhibit a clear drift. The batch-to-batch variations in rate observed in the controller scenarios are due to the statistical uncertainty associated with the finite batch size.

The effect on the signal is illustrated in Fig. 14, which presents the relative change in instantaneous signal efficiency as a function of the fraction of run time. For both the H_{T} and AD triggers, the PD-controlled menus maintain higher and more stable efficiencies than their fixed counterparts, mitigating the degradation that would otherwise occur as pileup decreases. Figure 15 reports the relative change in cumulative signal efficiency over the full run. Here, the benefit of the dynamic strategy becomes particularly pronounced: PD-controlled thresholds enhance the total number of recorded signal events by up to about 70% with respect to fixed menus, while still maintaining the background rate constraint.

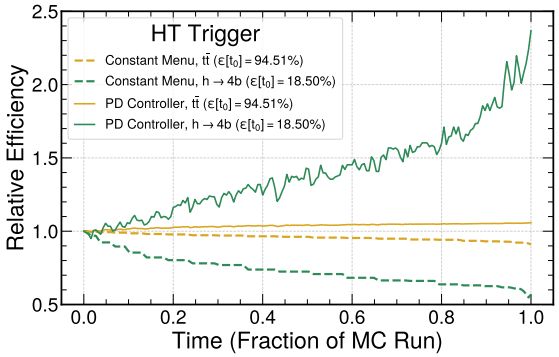


(a) H_{T} trigger. (b) AD trigger.

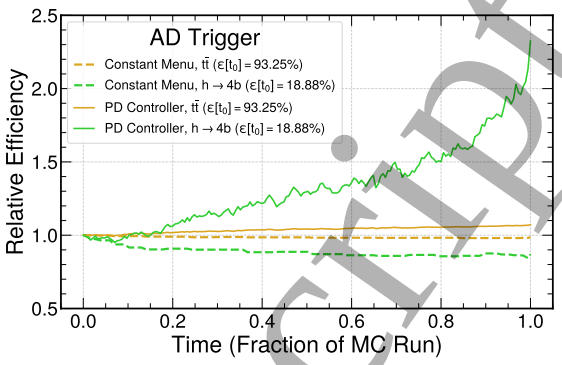
FIG. 13: Background trigger rates under PD control. (a) H_{T} trigger, (b) AD trigger. The controller stabilizes the background rate within the target tolerance band and mitigates the drift observed with fixed menus.

6. MULTI TRIGGER CONTROL FRAMEWORK

While the PID loop approach demonstrated effectiveness in maintaining steady trigger rates for intrinsically unstable input features, this simple framework is inherently limited to control of a single target parameter using a single tunable parameter. It furthermore relies on tuning several gain constants of the model, which can themselves vary over time in principle, and may also belie a deeper dependence on multiple effects contributing on different timescales. In a real experimental setup, multiple trigger paths operate concurrently, deciding whether to accept a common stream of events interpreted under different hypotheses, requiring a more sophisticated control strategy. Viewing each trigger algorithm as a simple function of a single parameter, the action space for a controller manipulating an n-path trigger system is an n-dimensional space of thresholds. The

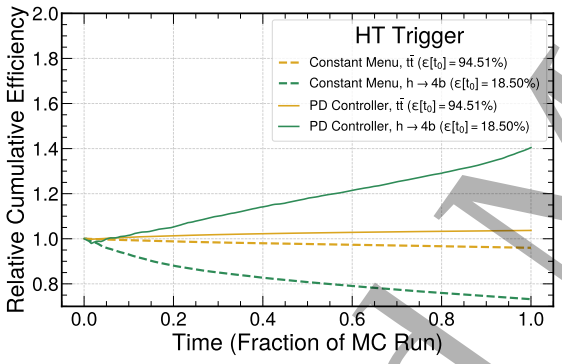


(a) H_{T} trigger.

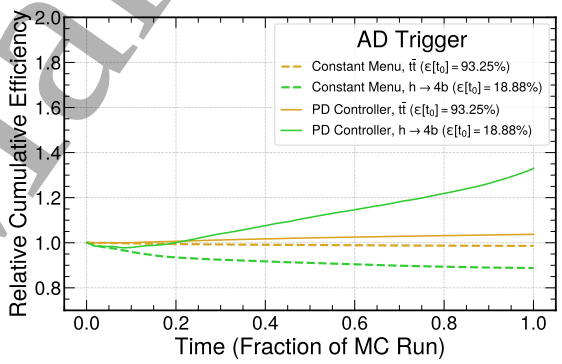


(b) AD trigger.

FIG. 14: Relative change in instantaneous signal efficiency over time. (a) H_{T} trigger, (b) AD trigger. PD-controlled menus maintain higher and more stable signal efficiencies compared to fixed menus.



(a) H_{T} trigger.



(b) AD trigger.

FIG. 15: Relative change in cumulative signal efficiency over time. (a) H_{T} trigger, (b) AD trigger. PD-controlled strategies substantially (up to 70%) improve the accumulation of signal events compared to fixed threshold menus, especially for pileup-dependent triggers.

primary operational objective of constraining the total data taking rate as near as possible to some predetermined target cuts is an $n-1$ hypersurface upon which the controller's prescription should lie.

Though one may consider constructing independent PID loops for each trigger, this approach is cumbersome and challenging to define in a straightforward way. It is not trivial to define distinct target rates for each trigger independently, or more generally assign value to their positive and negative correlations. Instead, inspired by PID's principle of minimizing the error term at each time step through corrective actions, we extend this concept to defining a global cost function

C, as a function of a set of variables and hyperparameters, that can be dynamically assessed and minimized throughout a run. Though a neural network is not required to perform this optimization, such a cost function would be a natural candidate to direct the training of an agent-based controller.

The cost function can be defined based on a set of constraint functions, $f_i(\vec{x};\theta)$, each representing a system requirement (e.g., background rate below a threshold). A strict control strategy would enforce that all constraints are exactly satisfied, mathematically corresponding to requiring $f_i = 0$ for all i, at $\vec{x} = \vec{x_0}$, where each function is defined to be zero when the corresponding condition is fulfilled. However, exact satisfaction may not always be feasible in practice. Instead, a soft-constrained approach is adopted, where deviations from the ideal values are tolerated and prioritized through a set of weights. In other words, each requirement is formulated as a scalar function of control variables and associated hyperparameters (constraint functions), and the cost decreases as the constraints are better satisfied. For this purpose, it seems natural to define the terms such that negative cost is not allowed and each term is dimensionless.

To this end, each constraint function f_i is assigned a weight w_i, which reflects the relative importance of that constraint. Another natural interpretation of the weights is to consider their inverses as setting a tolerance scale $\sigma_i = \frac{1}{w_i}$, with σ_i representing the size of an 'acceptable' deviation for the i-th constraint. Any sub-constraint f_i that varies from the ideal target by this magnitude will result in an identical penalty, i.e., contribution to the total cost function:

$$\mathrm{C} = \sum_{i=1}^{N} w_i f_i = \sum_{i=1}^{N} \frac{f_i}{\sigma_i}.$$

This formulation enables the controller to balance multiple competing objectives, each contributing according to an acceptable tolerance scale, while avoiding hand-tuning the many gain constants that would be necessary in a generalized PID approach.

In this scheme, the first cost function that one might consider is $C_0 = |r_b - r_t|/\sigma_b$, in which r_b is the total data taking rate dominated by background processes and r_t is its target. The scale σ_b would serve to normalize the contribution of C_0 if it were summed with other contributions to the total cost. However, in the presence of multiple trigger paths, C_0 is minimized by many degenerate actions of the controller. It is natural to break this degeneracy by extending C to incorporate more sophisticated operational priorities into the full control strategy.

The remainder of this work is dedicated to studying this framework in simulated and real collision data sets, making use of our toy model of a hadronic trigger menu that is based on an H_{T} and an AD path. In this context, the controller design must consider trade-offs between competing goals, such as maximizing signal efficiency from the MC library, allocating resources to exploratory

paths, and respecting resource limits. To investigate how different priorities influence controller behavior, we define a set of representative case studies, each built around a distinct objective function:

- *Case 1* focuses on maximizing the efficiency to collect target signal processes while also maintaining a fixed total background rate.
- *Case 2* introduces dedicated bandwidth allocation for an exploratory path (e.g., anomaly detection), while ensuring that the trigger performance for a known benchmark signal (here $t\bar{t}$) is not sacrificed.
- *Case 3* considering the highly asymmetric computational cost of each trigger path in the optimization, highlighting trade-offs between idealized physics performance and finite computational resources.

Each case illustrates how the controller's overall menu configuration and response to shifting conditions are shaped by the chosen optimization strategy. To decouple the impact of the cost function selection from the design of the control algorithm itself, we first study the behavior of an (impossibly) ideal controller, which 'looks into the future' to set trigger thresholds for each batch that will minimize each cost. This step of the study makes sure the cost function definition and format are reasonable to be used in a real control framework.

6.1. Case 1: Background-Signal Tradeoff

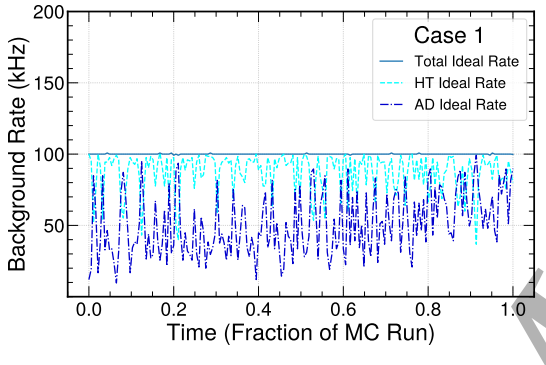
In a multi-path trigger configuration, a fundamental goal is to optimize signal collection while maintaining a stable background rate within bandwidth constraints. This trade-off can be formalized through a simple cost function that simultaneously prioritizes both objectives:

$$C_1 = \frac{|r_b - r_t|}{\sigma_b} + \frac{1 - \epsilon_s}{\sigma_s}.$$

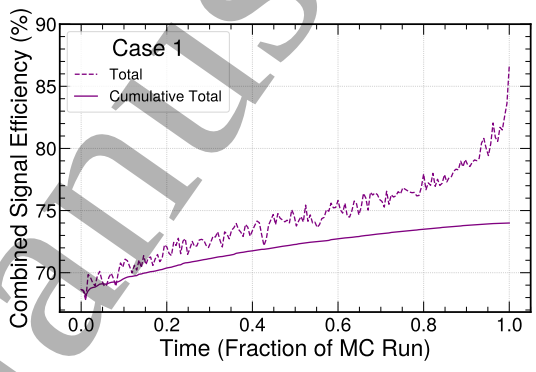
Here, r_b is the total rate of background events, r_t is the target rate, and ϵ_s denotes the overall signal efficiency, calculated on the total set of signal processes ($t\bar{t}$ and $h \to 4b$). The scales σ_b and σ_s represent allowed deviations for each constraint term that determine the relative importance of background stability and signal efficiency. As such, these weights inherently encode experimental priorities, for which we must make representative choices.

Figure 16 summarizes trigger performance under the Case 1 cost, for a choice of $\sigma_b = 4\,\text{kHz}$ and $\sigma_s = 0.05$. Panel 16a shows the total background rate and its decomposition into the H_T and

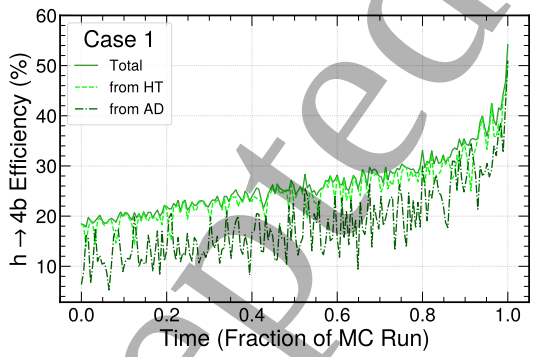
AD trigger paths, illustrating how the ideal controller holds the combined background near the target. Panel 16b reports the instantaneous and cumulative signal efficiencies, indicating how the policy prioritizes signal acquisition over time. Panel 16c focuses on the BSM benchmark $h \to 4b$, illustrating its total efficiency and the split between H_{T} and AD. Panel 16d similarly breaks down the $t\bar{t}$ signal efficiency, showing contributions from both paths and the overall high efficiency for this well studied process.



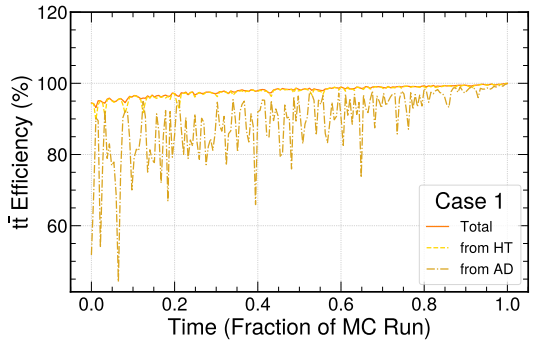
(a) Background rates: total (solid blue) and contributions from H_{T} (cyan dashed) and AD (blue dash-dotted), showing stable control near the target.



(b) Signal efficiency: instantaneous (purple dashed) and cumulative (solid), illustrating the trade-off between prompt response and overall yield.



(c) BSM signal $h \to 4b$ efficiency breakdown by path, increasing over time.



(d) $t\bar{t}$ signal rate, decomposed by path, confirming high efficiency across both triggers.

FIG. 16: Performance of the Ideal multi-path trigger system under Case 1 across four panels: (a) Background rates, (b) Signal efficiency, (c) $h \to 4b$ efficiency, (d) $t\bar{t}$ signal rate.

6.2. Case 2: Specific Path Bandwidth Allocation

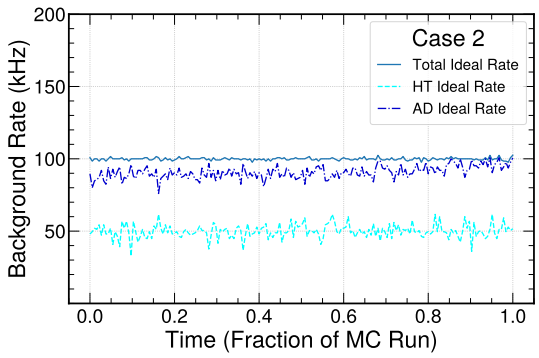
For a new AI-enhanced trigger whose selection behavior is still uncertain, the path should operate in parallel with proven triggers, perhaps with a fixed bandwidth allocation earmarked for exploration. At the same time, to ensure continued coverage of high value SM processes, one might also like to enforce some minimum baseline efficiency on these signals.

For instance, we may require the trigger to maintain a high, but not necessarily maximal, efficiency for an important signal such as $t\bar{t}$, allowing the remaining bandwidth to be flexibly prioritized toward the AD path. To implement such a strategy, we define the following cost function:

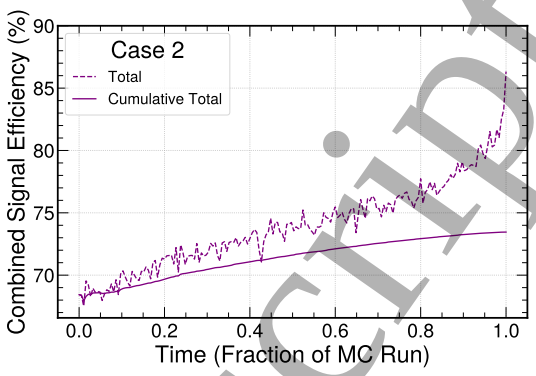
$$C_2 = \frac{|r_b - r_t|}{\sigma_b} + \frac{|\epsilon_{t\bar{t}} - 0.9|}{\sigma_{t\bar{t}}} + \frac{|r_{\mathrm{AD}}^{(\mathrm{ex})} - p \cdot r_t|}{\sigma_{AD}}$$

where: $\epsilon_{t\bar{t}}$ is the total efficiency for $t\bar{t}$ events, $r_{\mathrm{AD}}^{(\mathrm{ex})}$ is the exclusive rate for the AD path and p represents the allocated fraction of bandwidth. As an example configuration, we consider $\sigma_b = 4\,\mathrm{kHz}$, $\sigma_{t\bar{t}} = 0.05$, $\sigma_{\mathrm{AD}} = 16\,\mathrm{kHz}$, and $p = 0.5$. While this formulation focuses on exclusive AD rate to capture events that would not otherwise be collected, the inclusive AD rate may also be prioritized to better study correlations between trigger paths, as one example.

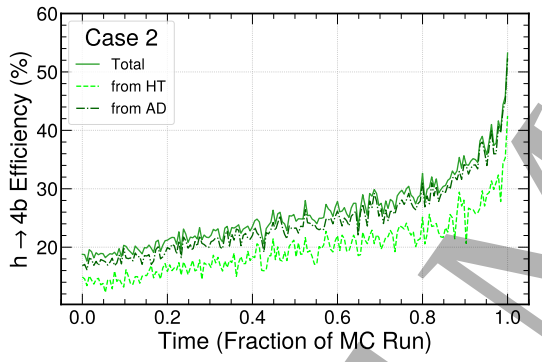
In Fig. 17, we observe the heightened AD trigger activity in collecting both signal and background events compared to Case 1, as a result of the 50 kHz ($p = 0.5$) exclusive allocation to this path. The cost function is also designed to maintain the $t\bar{t}$ efficiency at 90%. However, this term is correlated with the total bandwidth constraint in the first part of the cost. Specifically, we find in Fig. 11 that the $t\bar{t}$ efficiency exceeds 90% for both H_{T} and AD in a fixed menu, and controlling it requires very tight cuts. However, if both triggers were forced to adopt such cuts, the total rate would deviate from the target, and this is penalized in the cost function through a larger sensitivity to bandwidth variations. With this more complex set of priorities, it is not possible for the 'ideal' controller (separately minimizing the cost for each batch) to prescribe actions that perfectly satisfy each item at all times. This is a good example of demonstrating that the hierarchical priority setup in the cost function is effective. On the other hand, this richer problem space opens the opportunities for improvements by more sophisticated controllers that may globally optimize cost over longer timescales.



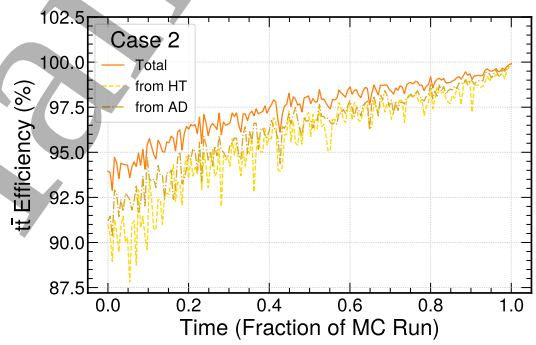
(a) Background rates: total controlled near 100 kHz, with H_{T} held close to 50 kHz to ensure ∼50 kHz exclusive rate for the AD trigger ($p = 0.5$).



(b) Signal efficiency: changing the cost format has not significantly altered the trend shown in Case 1, as it correlates mostly with total bandwidth accessibility.



(c) BSM signal $h \to 4b$ efficiency breakdown by path, showing increased AD contribution due to bandwidth allocation.



(d) $t\bar{t}$ signal rate by path, dominated by AD but nearly failing the 90% efficiency criterion imposed by the cost.

FIG. 17: Ideal Optimization outcome for Case 2 across four panels: (a) Background rates, (b) Signal efficiency, (c) $h \to 4b$ efficiency, (d) $t\bar{t}$ signal efficiency.

6.3. Case 3: Computational Cost Awareness

In parallel with our physics-driven goals for data collection, it is important to recognize that sending an event through a given trigger path incurs a considerable computational cost. To attempt to quantify how this could impact data-taking decisions, we recall that the L1 trigger is only the first step in the two-stage trigger architectures used by ATLAS and CMS. Events selected by the L1 trigger are sent to the High Level Trigger (HLT), which performs a more comprehensive event reconstruction in a commodity PC farm, to further reduce the population of selected events for

permanent storage. The processing steps that are required to be carried out in this software-based trigger vary, depending on which L1 trigger item has already accepted the event. As an example, the calorimeter subdetector system may be ignored by the HLT for events that only pass muon-related triggers in the L1 step.

As such, one accounting of the 'computational cost' of an L1 trigger is the level of resources required to execute a given algorithm, such as the fraction of a Field-Programmable Gate Array (FPGA) required to compute an anomaly score. However, perhaps even more critical is the cumulative cost of all downstream HLT algorithms that must be executed when the event is accepted by that path. If a certain trigger path allows an event to be reconstructed and accepted for relatively low computational cost, the freed resources can instead be utilized to evaluate more sophisticated trigger rules. Therefore, the computational cost one expects to incur for each accepted event can be an important factor for the trigger agent to consider during optimization.

While the active monitoring of computational costs during data taking is a high priority, at this point such constraints are only implicitly reflected in the handmade trigger menus by the experimental collaborations [29, 30, 33]. If the optimization were to minimize only this computational cost, the solution would be trivial: the agent would favor filling the bandwidth with the "cheaper" path, disregarding physics objectives. To study a more realistic scenario, we incorporate the computational cost into a composite cost function that also includes physics goals, such as signal efficiency. This introduces a trade-off between maximizing physics performance and maintaining computational feasibility, allowing for a more meaningful outcome.

For the context of this open data benchmark, we utilize a toy model of these computational costs that does not aim to emulate the underlying technical details relevant to any given experiment. A more realistic and dynamic integration of these factors could be developed and incorporated into the framework in future work. As such, we consider two subcategories of these costs that represent different loads on the HLT CPU farm to further process events selected by the L1 trigger:

Event Level Computational Cost: reflects the 'complexity' of the event and is quantified here by its jet multiplicity. While a high number of jets could indicate interesting physics, it could equally be a pileup artifact. Therefore, event cost can be defined as the average number of jets in accepted events per batch. For example, if 2 events pass in a batch, with 2 and 3 jets respectively, the event-level cost associated with that batch is equal to 2.5.

Trigger-Path Level Computational Cost: accounts for differences in the class of algorithms required to process an event in the HLT depending on the L1 seed, irrespective the event level complexity. Each L1 path may invoke a distinct set of HLT reconstruction algorithms, resulting in

varying per-event processing requirements.

H_{T} trigger targeting highly energetic jets may only require reconstruction of hadronic and electromagnetic calorimeter information, bypassing track reconstruction completely. However, AD trigger inherently seek to test a wide range of event hypotheses, using all subdetector systems to precisely measure low-p_T jets, perform b-tagging, and evaluate more powerful AD models.

Based on recent HLT timing estimates from CMS [34] where jet-related algorithms accounted for roughly 150 of the average 600 ms processing time, our toy model assigns a cost to AD-selected events that is four times larger than those selected by the H_{T} path alone, i.e., an accepted event costs 4 units if passed by AD, otherwise 1, and the trigger-path cost is computed as the average path-associated cost per batch.

Note that events passing both paths are not double counted in the cost calculation, since the HLT executes each algorithm only once, and the computational cost is calculated solely based on background events. The final form of the cost that we introduce is:

$$C_3 = \frac{|r_b - r_t|}{\sigma_b} + \frac{1-\epsilon_s}{\sigma_s} + \frac{\max(C_{\mathrm{evt}} - C_{\mathrm{evt}}^{\mathrm{ref}},\, 0)}{\sigma_{\mathrm{evt}}} + \frac{\max(C_{\mathrm{algo}} - C_{\mathrm{algo}}^{\mathrm{ref}},\, 0)}{\sigma_{\mathrm{algo}}}$$

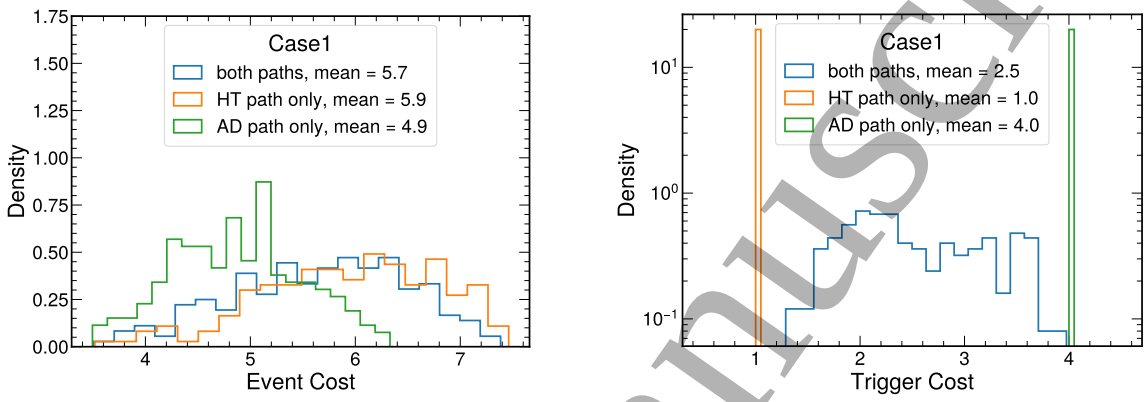
where r_b and r_t denote the background and target rates, and ϵ_s is the signal efficiency. The values of σ_b and σ_s are chosen to reflect typical variations of background rate and signal efficiency. The hyperparameters could be made dynamic to reflect changes in experimental priority over time, however, implementing such a scheme would require further studies and evaluation of hardware and software limitations. Therefore, in the present study, they are kept fixed for simplicity.

The quantities C_{evt} and C_{algo} represent the average event-level and trigger-path-level computational costs per batch, respectively. Figure 18 displays histograms of the average event level and trigger-path level costs per batch in Case 1, a scenario where optimization is guided solely by physics performance, and the computational cost is not explicitly constrained.

The scaling parameters σ_{evt} and σ_{algo} are chosen to be comparable to the width of the corresponding cost distributions in Fig. 18, while the reference values $C_{\mathrm{evt}}^{\mathrm{ref}}$ and $C_{\mathrm{algo}}^{\mathrm{ref}}$ are set equal to the mean values of the corresponding distributions. These choices define a baseline corresponding to the typical computational cost of an unconstrained configuration and ensure that only configurations exceeding these reference costs are penalized.

A notable feature of Fig. 18a is that the AD-only configuration yields a lower average event-level cost than the H_{T}-only configuration. This reflects the different selection logic of the two paths. The H_{T} trigger preferentially accepts events with larger hadronic activity, including high-multiplicity jet topologies that are typically more expensive to reconstruct, whereas the AD path is

not driven solely by overall hadronic, but by the degree to which an event deviates from the typical background patterns. However, this does not imply that the AD path is automatically favoured. As shown in Fig. 18b, the AD path itself carries a substantially larger trigger-path computational cost than the H_{T} path.



(a) Event-level cost for three configurations: both paths active (blue), only the H_{T} path (orange), only the AD path (green).

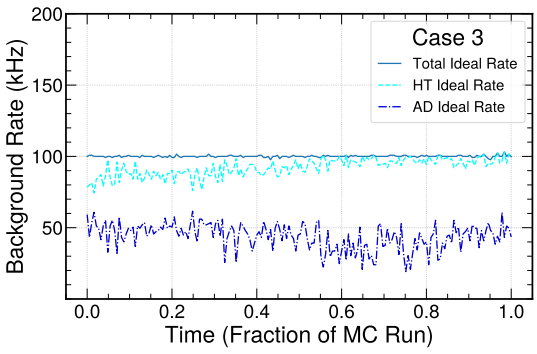
(b) Trigger-path level cost distributions for the same three configurations, showing the relative computational load associated with each path.

FIG. 18: Distributions of computational costs of accepted background events for Case 1. Panel (a) shows that the AD-only configuration selects events with a lower average event-level cost than the H_{T}-only configuration, reflecting the different selection logic of the two paths. Panel (b) shows, however, that the AD path itself has a higher trigger-path cost than the H_{T} path. Together, these distributions define the reference cost scales used in the Case 3 optimization.

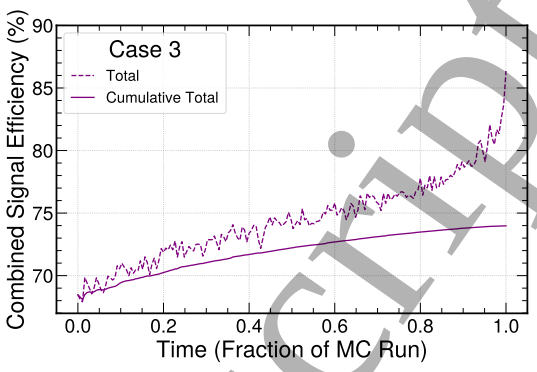
The resulting behavior under this strategy is illustrated in Fig. 19 with below set of parameters, chosen as described above:

$$\sigma_b = 4\,\mathrm{kHz}, \quad \sigma_s = 0.05, \quad \sigma_{\mathrm{evt}} = 0.5, \quad \sigma_{\mathrm{algo}} = 0.5, \quad C_{\mathrm{evt}}^{\mathrm{ref}} = 5.7, \quad C_{\mathrm{algo}}^{\mathrm{ref}} = 2.5.$$

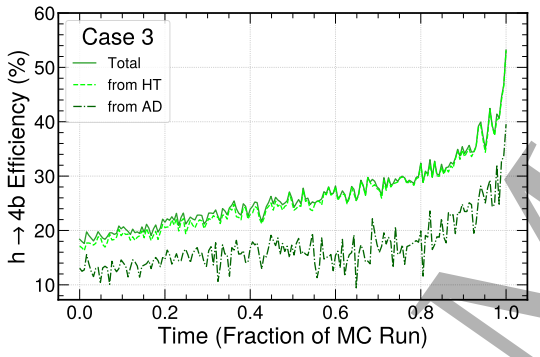
In addition to stable total background rates and gain in signal efficiencies with time, Fig. 19 is also consistent with the choice of H_{T} trigger as the "cheaper" path since the collection of events in background and signal samples is dominated by this trigger while no significant loss is observed in total signal efficiency.



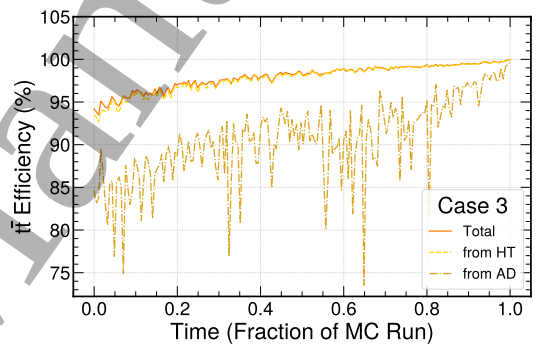
(a) Background rates, demonstrating stable control near the target while showing a different rate decomposition compared to earlier cases.



(b) Total signal efficiencies, illustrating sustained gain over time similar to previous cases.



(c) BSM $h \to 4b$ efficiency by trigger path, highlighting the controller's allocation strategy.



(d) $t\bar{t}$ efficiencies by path, showing the H_{T} trigger becoming dominant in event collection.

FIG. 19: Ideal Trigger performance under Case 3 (cost-aware controller): (a) Background rates, (b) Total signal efficiencies, (c) $h \to 4b$ efficiency by path, (d) $t\bar{t}$ efficiency by path. Together these panels highlight how the controller balances physics goals while managing computational cost, leading to the H_{T} path becoming the dominant contributor.

7. A SIMPLE LOCAL CONTROLLER

In Section 6, we have set the table for three benchmark tasks by codifying competing experimental priorities into explicit functions that can be optimized. We illustrated the evolution of these cost functions over time in the benchmark MC dataset by invoking an *ideal* controller that acted based on complete knowledge of the upcoming batch of data. This agent delineates an important upper bound on performance, achieving the best outcome in each event batch under perfect foresight.

While informative, such an approach is impossible to implement in practice. We therefore introduce a simple and realistic control strategy that relies only on past and current information. In principle, a global grid search across the action space performed on the current batch should approximate the optimal parameter set for the upcoming batch in the adiabatic limit. Though this approach may be practical for a simple two-component trigger menu, scaling of the grid volume for a scenario with many paths limits its applicability. On the other hand, observations of the ideal controller (Fig. 20) indicate that the optimal parameters evolve smoothly over time and generally remain localized within a constrained region of the parameter space. As a result, a local search strategy becomes both feasible and effective.

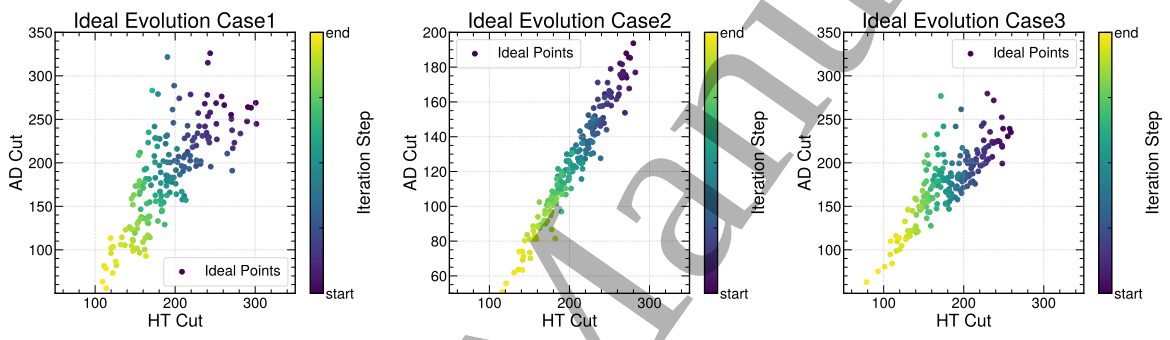


FIG. 20: Real time evolution of optimal point of the ideal controllers (minimum of the cost) in the parameter space.

Figure 20 generally illustrates how the global minimum of the cost function evolves over time. While we observe a trend toward lower cut values, there is random wandering in the parameter space as well. Based on these observation, a candidate simple controller can be defined to operate through a local grid search over a 10×10 window (versus 100×100 in the ideal case), using a search range of ± 20 units and a memory depth of five batches. Here, memory depth means that each update is computed as an average over the optimal local points from the previous five batches, which improves robustness and mitigates fluctuations. This enables the controller to follow historical trends while preventing abrupt parameter shifts, as shown in Fig. 21.

Figures 22, 23 and 24 compare the performance of this simple local controller with the ideal benchmark across the three case studies introduced earlier. In each case, the time evolution of rates and efficiencies obtained with the controller is shown together with the corresponding ideal curves, allowing a direct comparison. We observe that, when moving from the ideal case to the realistic controller algorithm, the overall behavior of rates and efficiencies, as well as their decomposition across the different trigger paths, is largely unchanged. Importantly, the cumulative signal efficiency

in the realistic controller also remains close to that of the ideal case (panels 22b, 23b, 24b). In summary, this controller yields a smooth evolution through the action space and achieves efficiencies very close to the ideal benchmark, demonstrating that the simplified, realistic algorithm does not compromise physics performance.

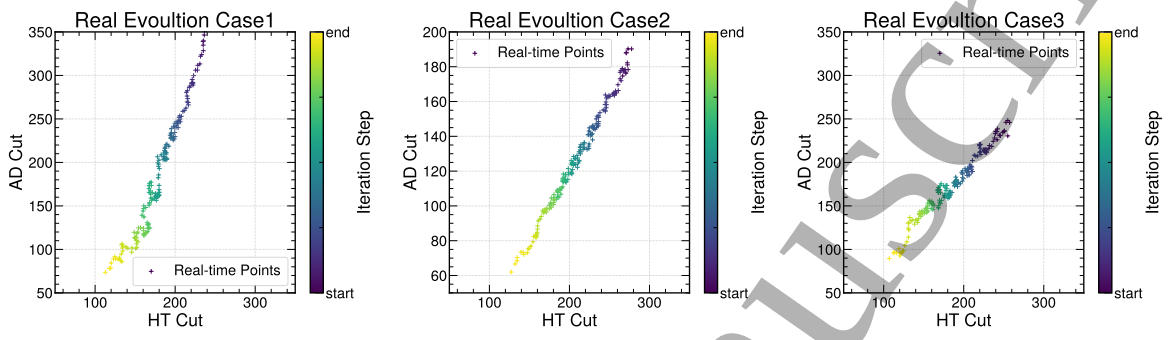
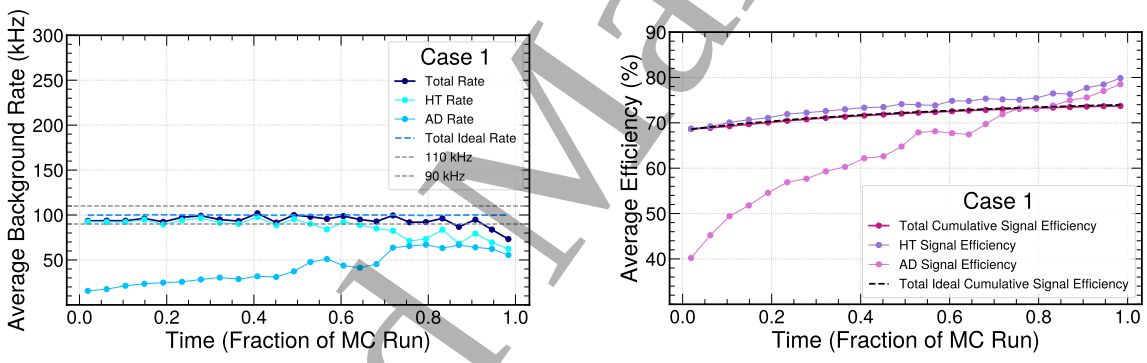


FIG. 21: Real time evolution of the cost functions with the simple controller.



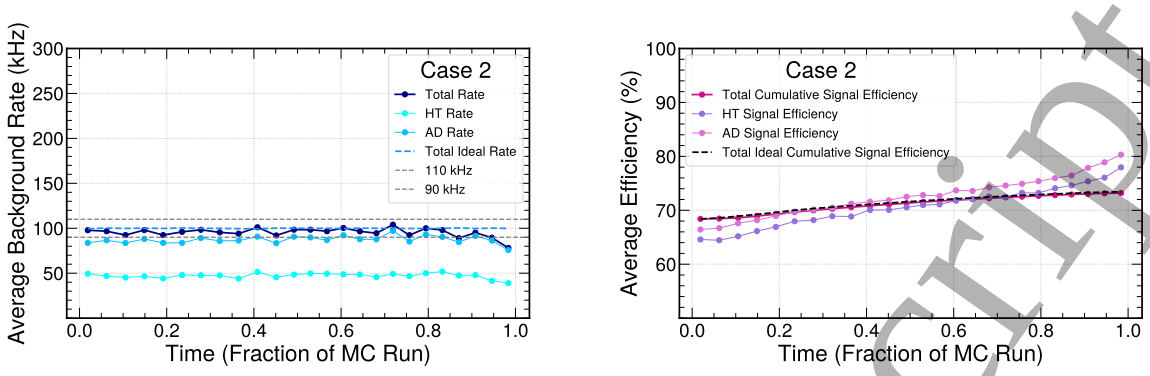
(a) Background Rate.

(b) Signal Efficiency.

FIG. 22: Trigger rates for Case 1 under the simple controller. The controller maintains the background rate close to the target while improving signal efficiency over time.

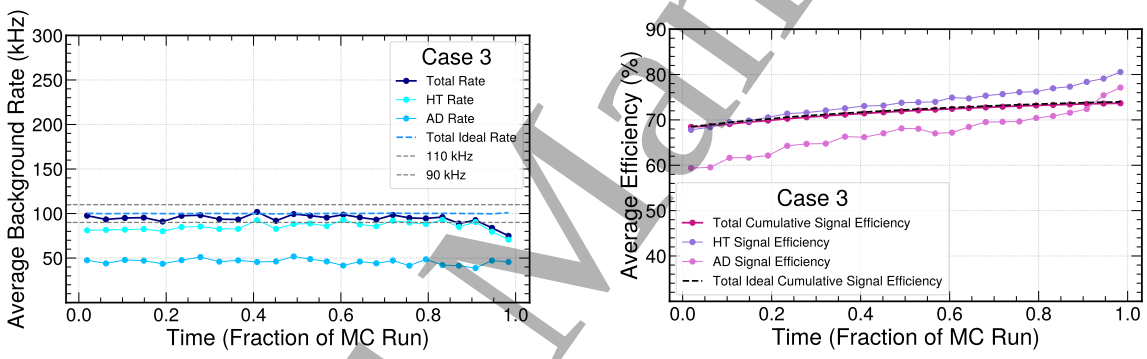
8. CONTROLLER PERFORMANCE WITH DATA

Having established the performance of the autonomous trigger framework on simulated datasets, we now turn to real collision data, which provide the decisive test of the controller's robustness under realistic detector conditions. We consider large samples of randomly triggered proton–proton collision events recorded by the CMS experiment during the 2016 data-taking period. These events are taken from the `ZeroBias` stream introduced in Section 2, which we refer to as the *background data* sample.



(a) Background Rate. (b) Signal Efficiency.

FIG. 23: Trigger rates for Case 2. The controller successfully achieves specific bandwidth allocation to the anomaly detection path, while keeping the total background rate within limits.



(a) Background Rate. (b) Signal Efficiency.

FIG. 24: Trigger rates for Case 3. The controller incorporates computational cost into the optimization, maintaining physics performance high while keeping the trigger level resource usage under control.

In Sections 5-6, simulated background samples were used to develop and validate the control strategy in a controlled setting. While essential at the design stage, simulation can only approximate real detector behavior and does not fully capture time-dependent effects such as evolving noise patterns, changing hardware configurations, and other subtle features of data taking. By contrast, the background data provide a direct and unbiased snapshot of detector activity, as events are recorded solely on the presence of colliding bunch crossings, without any physics-based selection. This makes them particularly well suited for testing real-time trigger strategies.

The anomaly detection autoencoder is retrained directly on background data, avoiding assumptions inherent to simulated samples, and the controller performance is evaluated under realistic

operating conditions. Although these data may contain rare physics signals, their contribution is statistically negligible for both the training of the anomaly detection model and the background rate studies presented here.

The autoencoder is trained on the second-longest run available in the dataset (Run 283876), while the controller performance is evaluated on the longest run (Run 283408), ensuring a clear separation between training and evaluation samples. Run 283408 spans 16 h 28 min and contains 1.99 million events in the random, prescaled stream used in this study, corresponding to an effective event rate of about 34 Hz. The controller processes events in batches of 2×10^4 per update, such that each batch corresponds to an update every ~ 12 min. This rate is orders of magnitude lower than the rate seen by an online agent in a real trigger system (40 MHz at the bunch-crossing level), and should therefore be interpreted as a realistic, publicly available proxy for time-evolving conditions, rather than a faithful model of online throughput. The target trigger rates used in the control studies (e.g. 100 kHz in the single-path control studies) refer instead to the simulated trigger bandwidth constraint and are independent of the rate of the recorded dataset.

In realistic operation, MC simulation and collision data play complementary roles. Simulated samples, including both background and benchmark signal processes, are used offline to design the anomaly detection model, determine initial thresholds, and tune controller gains. Once deployed on collision data, the controller continuously updates rate estimates and dynamically adjusts trigger thresholds in response to evolving conditions such as changes in instantaneous luminosity. Within this framework, simulation provides predictive expectations, while the data-driven operation constitutes the definitive test of the system under realistic detector and beam conditions.

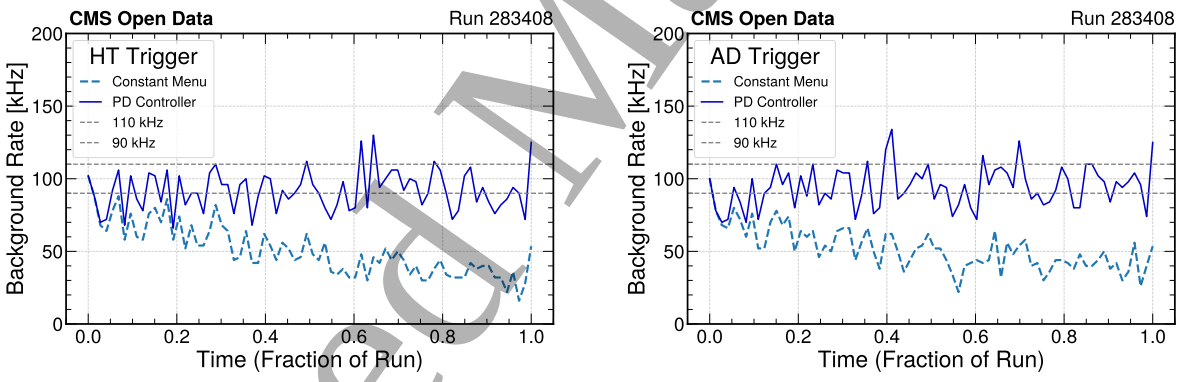
8.1. Single-Path Real-Time Control on Data

We begin by studying the simplest control scenario, in which a single trigger path is regulated in real time to maintain a fixed background rate. As in the simulation studies, we consider the H_{T} and AD triggers independently, each operating under a proportional–derivative PD feedback loop. For both triggers, the controller target is set to a background acceptance rate of 100 kHz, corresponding to a representative Level-1 bandwidth constraint. The controller updates the trigger threshold after each batch of 20 000 events, based solely on the observed deviation of the background rate from the target.

As shown in Fig. 25, when applied to real collision data, the PD controller successfully compensates for the gradual decrease in pileup throughout the run. In contrast to a fixed-threshold menu,

which exhibits a clear and monotonic rate drift as luminosity decreases, the controlled triggers maintain a stable background acceptance within the desired tolerance band. Residual batch-to-batch fluctuations are dominated by statistical uncertainties associated with the finite batch size rather than systematic trends.

The impact of dynamic control on signal performance is assessed by evaluating the relative signal efficiency as a function of time. For both the $t\bar{t}$ and $h \to GlobalHLTGlobalHLTRate[Hz]RateError[Hz]FullMen$ benchmark processes, the controlled menus preserve a substantially higher instantaneous efficiency than fixed menus, particularly in the later stages of the run when pileup has decreased significantly (Fig. 26). This improvement is further reflected in the cumulative signal efficiency, shown in Fig. 27, which quantifies the total fraction of signal events accepted over the full run. In all cases studied, real-time control mitigates the progressive loss of signal acceptance that is unavoidable with static thresholds. These results demonstrate that the PD controller, originally tuned on simulation, transfers robustly to real data without retuning, effectively stabilizing rates and preserving signal efficiency under realistic operating conditions.

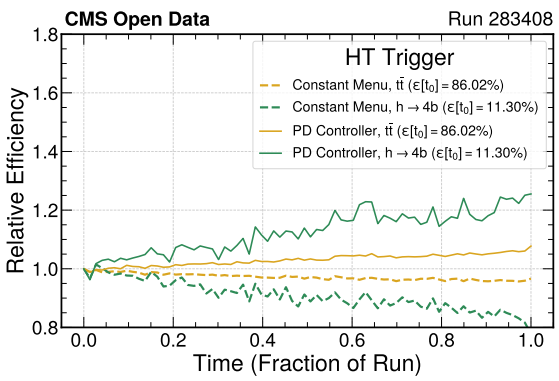


(a) H_{T} trigger. (b) AD trigger.

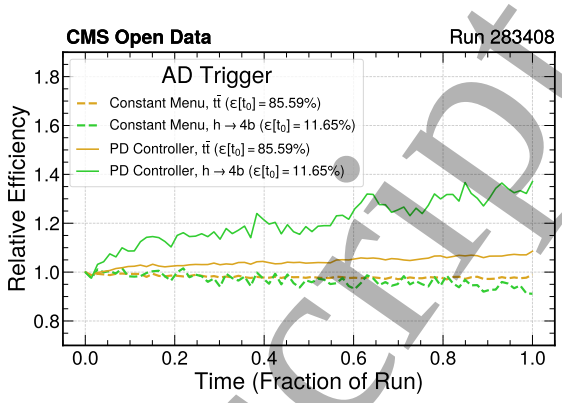
FIG. 25: Background trigger rates in real data under PD control. (a) H_{T} trigger, (b) AD trigger. The PD controller maintains the background rate within the target tolerance band while the constant menu decreases with time.

8.2. Multi-Path Real-Time Control on Data

We now extend the study to the more realistic scenario in which multiple trigger paths operate simultaneously and compete for a shared bandwidth. In this configuration, the controller adjusts the thresholds of both the H_{T} and AD triggers in concert, guided by a global cost function that

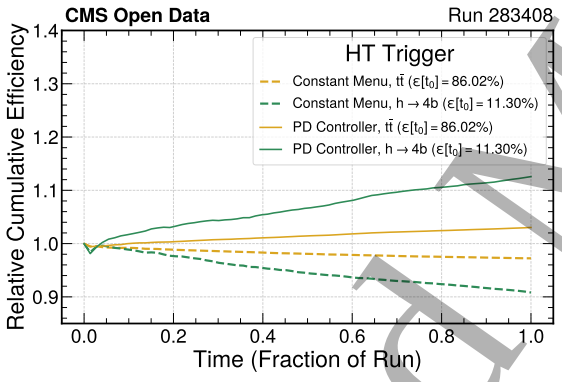


(a) H_{T} trigger.

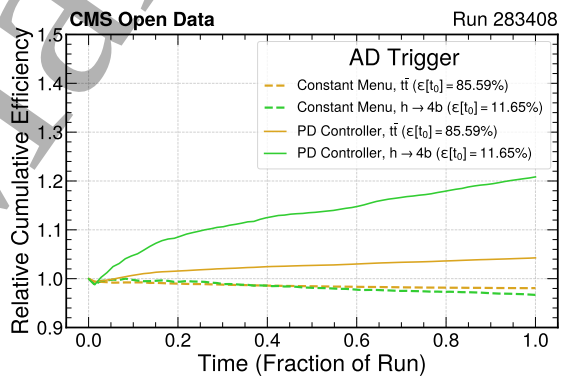


(b) AD trigger.

FIG. 26: Relative change in instantaneous signal efficiency over time, in data. (a) H_{T} trigger, (b) AD trigger. PD controlled menus achieve higher and more stable signal efficiencies compared to static menus.



(a) H_{T} trigger.



(b) AD trigger.

FIG. 27: Cumulative signal efficiency in real data. (a) H_{T} trigger, (b) AD trigger. The PD controller preserves a higher integrated signal yield (up to 25% gain in comparison to the fixed menu) while keeping background rates stable throughout the run.

encodes multiple objectives. As in Section 7, the ideal controller, assuming full knowledge of future data, though not physically realizable, serves as a valuable baseline for assessing how closely practical strategies approach the theoretical optimum. For this reason, we also use it on data as a benchmark reference, however, we focus on evaluating the behavior of the simple controller in the three representative configurations previously introduced in Section 6, namely Case 1, Case 2, and Case 3. The corresponding results are shown in Figs. 28, 29 and 30, with each plot illustrating a different control strategy:

- *Case 1* represents the simplest configuration, where the H_{T} and AD paths are treated symmetrically, with no explicit prioritization. The controller responds effectively to background fluctuations, maintaining stable rates over time and achieving improved signal.

- *Case 2* introduces an exclusive fixed bandwidth allocation for the AD trigger. In this case, it corresponds to 50% of the total bandwidth.

- *Case 3* implements the full cost-aware strategy, simultaneously accounting for computational cost, signal efficiency, and bandwidth constraints.

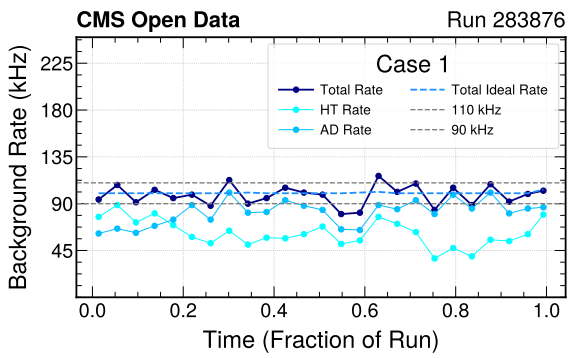
  Figure 31 illustrates the cost reference distribution derived from the Case 1 configurations for data. Based on this distribution, appropriate cost references are defined for data, while the remaining cost function's hyperparameters are kept identical to those used in the simulation studies. The resulting reference values and cost weights used in the real data optimization are:

$$\sigma_b = 4\,\mathrm{kHz}, \quad \sigma_s = 0.05, \quad \sigma_{\mathrm{evt}} = 0.5, \quad \sigma_{\mathrm{algo}} = 0.5, \quad C_{\mathrm{evt}}^{\mathrm{ref}} = 4.3, \quad C_{\mathrm{algo}}^{\mathrm{ref}} = 3.5.$$

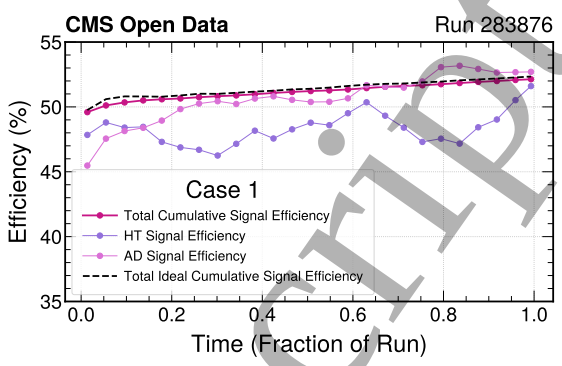
The results, shown in the aforementioned plots, provide compelling evidence that the autonomous control framework, initially developed in simulation, remains effective when applied to real data. Although moving to data introduces distributional shifts and amplifies the intrinsic volatility of the anomaly detection, the controller exhibits robust and interpretable behavior across configurations, dynamically adapting to detector fluctuations while consistently achieving key objectives such as rate stabilization, signal efficiency, and cost optimization. Even in the presence of realistic noise and operational variability, it maintains reliable performance, demonstrating flexibility in accommodating diverse experimental and system constraints. By enabling real-time feedback-driven optimization, the framework offers a path toward more autonomous, intelligent, and efficient trigger systems, capable of reshaping data acquisition at the LHC and meeting the evolving demands of future high-energy physics experiments.

9. SUMMARY PLOTS AND DATA–SIMULATION COMPARISON

This section condenses the main outcomes of the benchmark into two compact summaries: one showing the results using the *simulated background* in Fig. 32 and one showing the results using the *data background* in Fig. 33. Rather than showing each metric separately as a function of time, these plots represent the trajectory traced by a given control strategy as operating conditions evolve

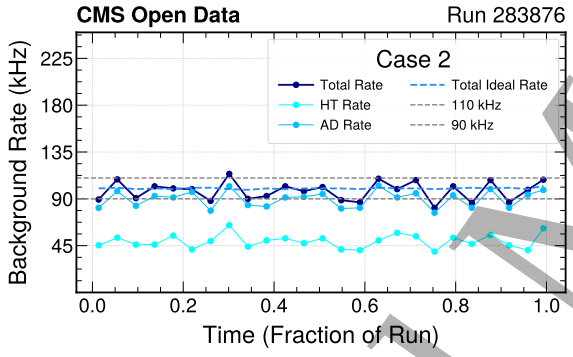


(a) Background Rate.

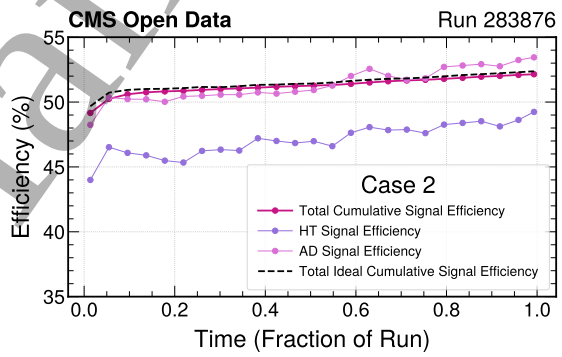


(b) Signal Efficiency.

FIG. 28: Trigger rate and efficiency for Case 1. The controller maintains the background rate close to the target while improving signal efficiency over time.



(a) Background Rate.



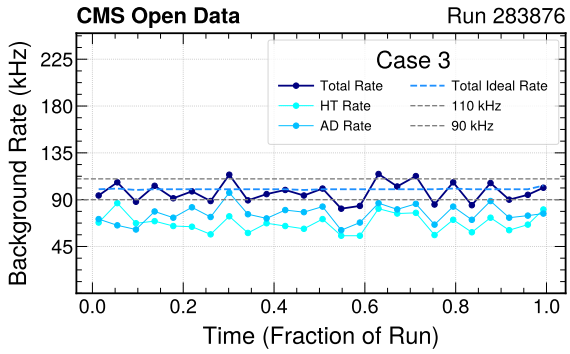
(b) Signal Efficiency.

FIG. 29: Trigger rates for Case 2 in data. The controller successfully achieves specific bandwidth allocation to the anomaly detection path, while keeping the total background rate within limits.

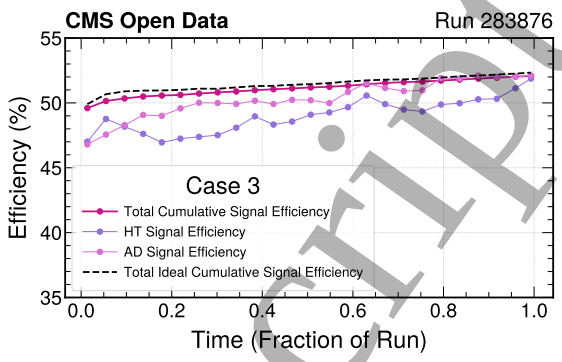
during a run. Each marker corresponds to a time slice, and the color gradient encodes progression through the run (dark → light), with the start-of-run point explicitly indicated. The Fixed Menu shown in these figures provides a static reference, while the three adaptive agents, corresponding to Cases 1, 2, and 3, implement feedback control with different optimization objectives, denoted as *Standard*, *Anomaly Focused*, and *Low-Comp Focused*, respectively.

9.1. Performance Summary on Simulated Events

Figure 32 summarizes the simulated case study, where the time evolution is emulated through the event ordering to mimic a gradual decrease in instantaneous luminosity. Each of the panels

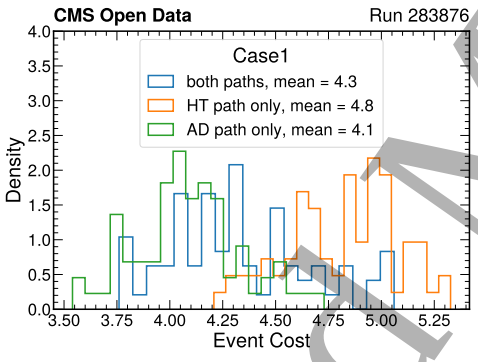


(a) Background Rate.

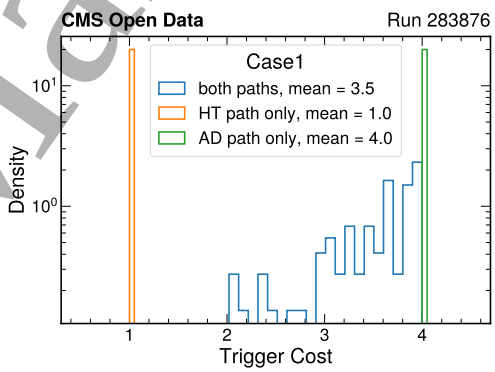


(b) Signal Efficiency.

FIG. 30: Trigger rates for Case 3 in data. The controller incorporates computational cost into the optimization, maintaining physics performance while keeping the trigger-level resource usage under control.



(a) Event-level cost for three configurations: both paths (blue), only the H_{T} path (orange), only the AD path (green).



(b) Trigger-path level cost distributions for the same three configurations, showing the relative computational load associated with each path.

FIG. 31: Distributions of computational costs of accepted background events for Case 1. Panels (a) and (b) define the reference cost values used in Case 3 optimization.

illustrates the evolution of paired trigger performance metrics over time as measured in signal and background events, which is strongly dictated by the policy adopted by each control agent.

Panel 32a visualizes the joint evolution of rate control and signal performance: the horizontal axis quantifies the deviation from the target background rate, while the vertical axis captures signal inefficiency. The Fixed Menu drifts away from the desired operating point as conditions change,

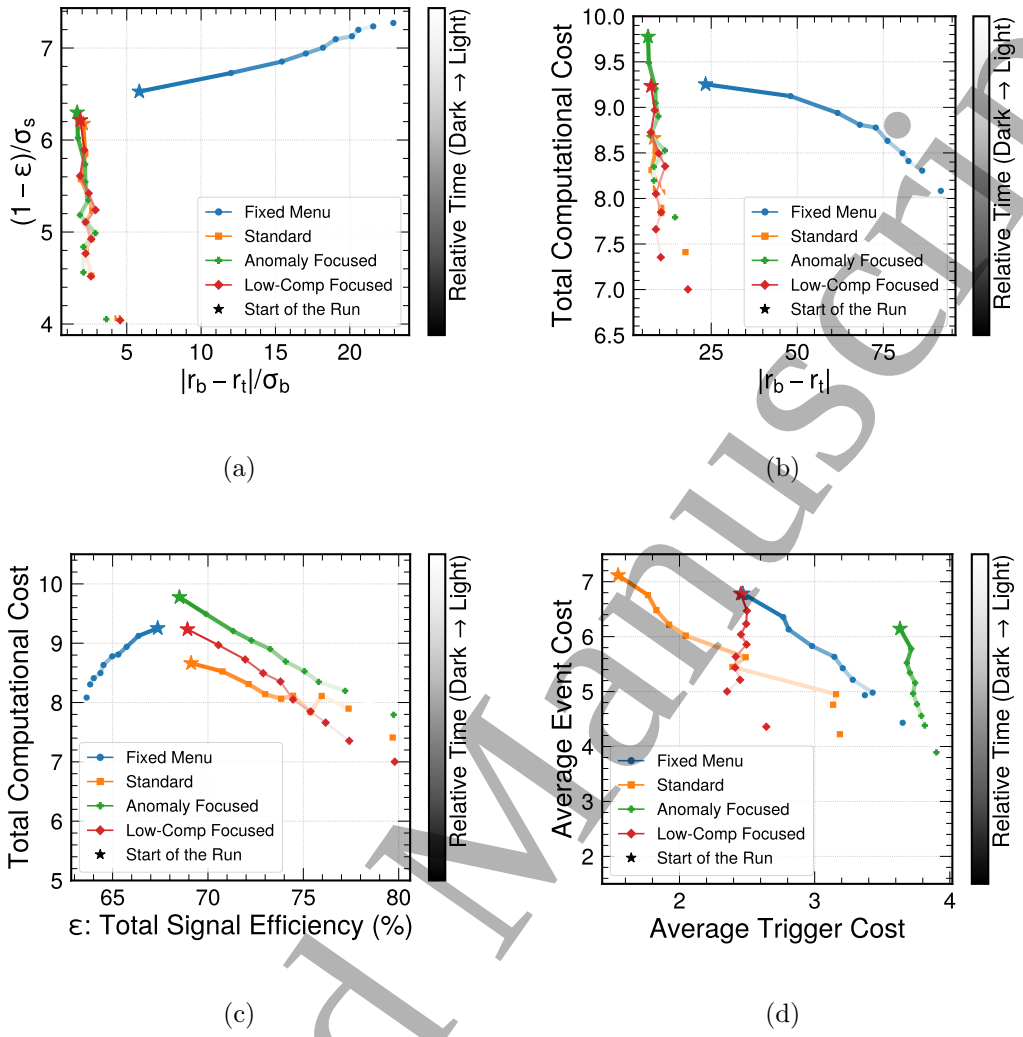


FIG. 32: Summary of different agents' performance on simulation samples.

exhibiting both increasing rate mismatch and degraded signal efficiency. By contrast, all adaptive agents remain confined to a region of small rate deviation while simultaneously improving signal efficiency, indicating that active control stabilizes the background response without sacrificing signal acceptance. Panel 32b adds the resource dimension by showing total computational cost versus background-rate deviation. The total computational cost is defined as the sum of trigger-path and event-level contributions. The adaptive agents optimize costs over the course of the run while operating in a comparatively low-$|r_b - r_t|$ regime. The Fixed Menu also exhibits a monotonic cost decrease, but along a trajectory in which $|r_b - r_t|$ grows substantially.

Panel 32c makes the efficiency–cost trade-off explicit: total signal efficiency (over $t\bar{t}$ and $h \to 4b$ samples) is plotted against total computational cost. As conditions evolve, the adaptive trajectories

tend to move toward a more favorable region with higher efficiency at lower cost, showing that the controllers exploit the changing environment to reduce average complexity while maintaining or improving signal yield. By the end of the run, the Low-Computational-Cost agent achieves the most aggressive cost reduction at comparable efficiency. Finally, panel 32d decomposes the computational cost by separating the trigger-level contribution (e.g. H_{T} versus AD composition) from the event-level contribution (proxied by jet multiplicity). All strategies follow the global trend of decreasing event complexity over the course of the run.

Overall, the simulation summary presents a coherent picture across all four projections: static menus drift and lose efficiency under time-dependent conditions, whereas adaptive control keeps the background stable and improves the position of the menu in the efficiency–cost trade space. Importantly, the differences among agents reflect their respective cost-function priorities, demonstrating how the framework can encode experiment-specific objectives.

9.2. Deployment on Collision Data

We now apply the same control strategies using CMS Run 283408, and summarize the four agents' trajectories over the full run in Fig. 33. The intent of this comparison is not to reproduce simulation point-by-point, but to verify whether the qualitative control behavior remains consistent under real operating conditions, including rate regulation, efficiency preservation, and cost adaptation. While our overall findings are similar in each setup, some trends are not as significant in the real collision data. This results from the underlying variation in the real 2016 collision data being generally less than the synthetic variations imbued in the simulated dataset, as seen in Figs. 3 and 4.

Panel 33a shows the deviation of the background rate from the target versus signal inefficiency. The Fixed Menu shows a large drift away from the target background rate, while maintaining poorer signal performance than the adaptive approaches. By contrast, all three agents cluster at smaller, more stable rate deviations and exhibit a clear reduction in signal inefficiency as the run evolves. Panel 33b shows total computational cost versus background-rate deviation. The adaptive agents still occupy a region with a substantially better rate control than the Fixed Menu. However, unlike in simulation, the adaptive strategies do not exhibit a perfectly monotonic decrease in total computational cost over the course of the run. While the final points generally lie at a lower cost than the initial ones, the intermediate evolution is not uniformly decreasing due to statistical variations within each batch of events and, for some strategies, includes an initial increase before

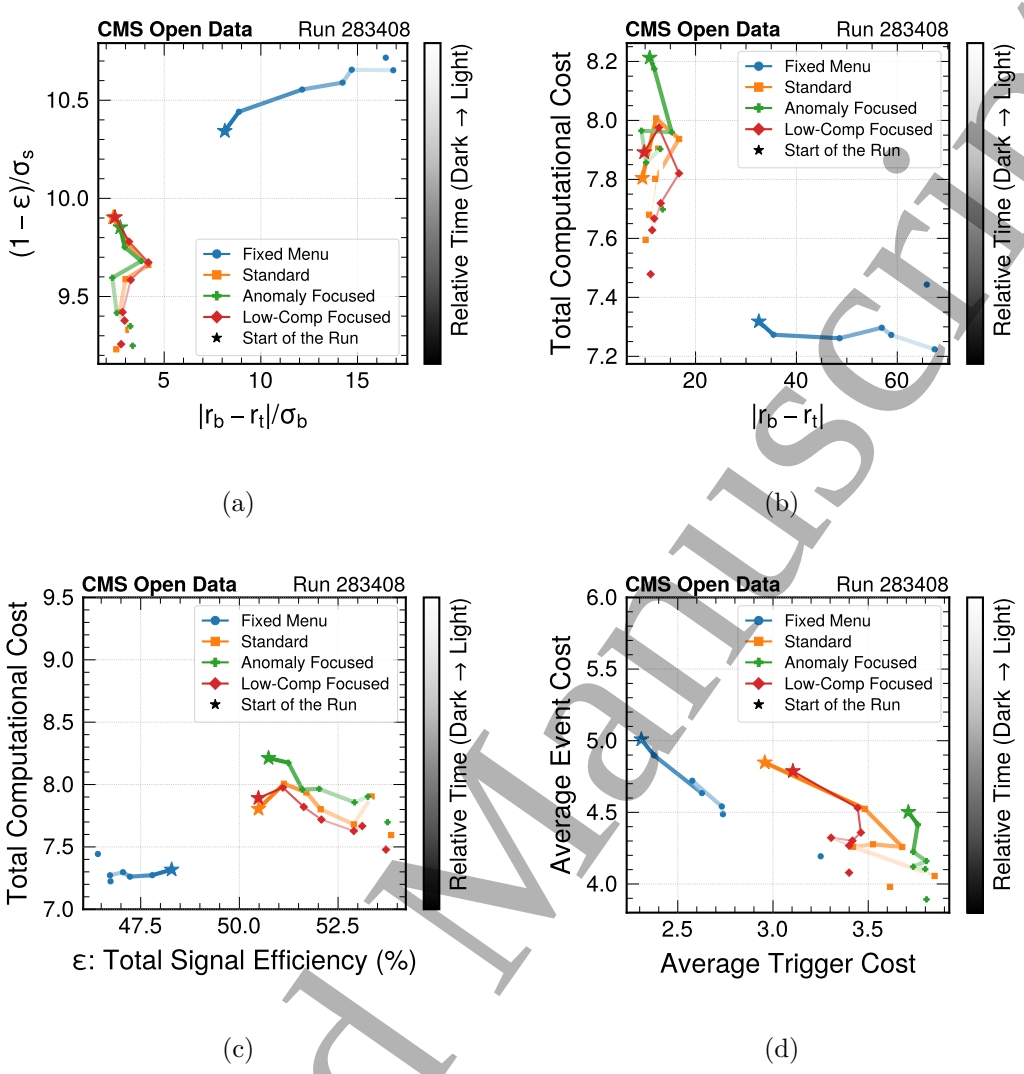


FIG. 33: A summary of each trigger menu control agents' performance across various metrics over the course of CMS Run 283408.

the subsequent reduction. The low-computational-cost agent shows the lowest computational cost at the end of the run among the adaptive strategies, consistent with what is observed in simulation.

Panel 33c relates total signal efficiency to total computational cost. The adaptive strategies evolve toward a more favorable operating region, achieving higher signal efficiency at reduced total cost as the run progresses. This indicates that the efficiency–cost improvement is not an artifact of the simulated stream ordering, but persists in collision data. Finally, panel 33d compares average trigger cost and average event cost. All strategies follow the expected decrease in event-level cost as the run evolves, reflecting the changing event activity conditions. In the low average event-cost region, the adaptive trajectories overlap substantially, limiting any stronger conclusion about the

relative behavior of the different agents.

Overall, Fig. 33 confirms the core conclusion suggested by simulation: adaptive trigger control stabilizes the background response and preserves or improves signal acceptance, while automatically exploiting the time evolution of the run to reduce computational burden. The qualitative agreement between MC and data trajectories supports the robustness of the benchmark and motivates the applicability of feedback-based trigger-control strategies in realistic operational environments.

10. CONCLUSIONS AND OUTLOOK

In this work, we have presented the framework of a dynamic, *self-driving* trigger system, which incorporates real-time feedback to maximize scientific and operational objectives. We set up a benchmark ecosystem that captures the tools, objectives, and constraints that are typical of LHC experiments: a combination of traditional and machine learning algorithms are employed to collect the most promising collision events while simultaneously controlling the rate of false positives and respecting limited computational resources. All aspects have been tested with historical trigger data from collisions recorded by the CMS experiment, focusing on events with interesting hadronic jet activity that could indicate the presence of rare Standard Model processes or yet-to-be discovered phenomena. To this end, we empowered an autonomous control agent to reconfigure the trigger selection criteria at regular time intervals, acting based on both external policies and in-situ performance estimates.

This benchmark began by studying simulated events that were given a temporal component, namely an arrow of time, characterized by a consistently decreasing instantaneous luminosity delivered to the experiment. First, we examined independent control of each trigger path based on a PD feedback loop. Then, we introduced and evaluated an ideal, generalized agent that could adjust all paths of the trigger menu at once, taking advantage of the time-varying correlations among all active algorithms. Depending on the specified reward, the controller's policy prioritized actions to collect signal processes, explore novel event topologies with AD, or reduce computational costs associated with further trigger processing. We showed that a simple control algorithm based on modest, adiabatic changes to the prescribed menu performed near the theoretical ideal in each of these cases.

After considering this synthetic dataset, the controller's performance was evaluated using historical CMS collision data, which can provide a unique window into time-varying factors that a real agent would encounter. Here, stable collection rates and improved efficiencies to candidate

signals were observed, as in the pure simulation setup. Details of the agents' evolving actions over time were presented in Sec. 9, illustrating the tradeoff in priorities that were ultimately dictated by each control policy. Despite having only an approximate emulation of time in our simulation samples, their observed trends largely coincided with those seen in data.

This study provides a proof-of-concept that an adaptive trigger system can be designed in a relatively simple manner while realizing dramatic improvements over the current experimental baseline of static menus. We further hope that providing all details of this work as an open benchmark task will further stimulate the community to produce new ideas that extend the self-driving trigger concept to even more powerful control systems. Extensions of this study could also explore more advanced decision-making agents, including reinforcement learning and active learning strategies that promise even more predictive and efficient means of control.

At the same time, additional considerations will need to be addressed before a fully autonomous system can be deployed in a realistic experimental environment; the level of integration of the controller with the trigger and data acquisition system, the ideal methodology for the book-keeping of trigger configuration data, and how occasional human intervention might be incorporated into this system could all provide interesting questions for future work.

The development of such systems will likely proceed through a multi-stage validation program, which incorporates both software- and hardware-based control components while allowing progressive validation of the system in increasingly realistic conditions. Initial demonstrations of adaptive control strategies could therefore take place through dynamic adjustments of trigger rates within software-based trigger environments, such as the High-Level Trigger. In addition, dedicated hardware demonstrations, such as prototype implementations on FPGA-based trigger boards, would provide an important intermediate step toward assessing the feasibility of these ideas in operational environments. Such studies would also help clarify how adaptive trigger strategies perform under the substantially higher event rates and more rapidly evolving conditions expected in future high-luminosity collider environments.

In conclusion, our results paint a clear picture of the novel sensitivity and numerous operational advantages over legacy systems, ideally opening the door to a new generation of autonomous triggers for high-throughput environments like the LHC. Self-driving triggers are well within reach of modern experimental capabilities, and their implementation offers a uniquely focused strategy to navigate the landscape of complex and wide-ranging constraints while maximizing our ultimate goal of sensitivity to interesting physics.

Data availability statement: The code that supports the findings of this study is openly

available at the following URL: https://github.com/Shaghayegh-E/Adaptive-ParticlePhysics-Triggers [35]. The datasets generated in this study are openly available at the following URL: https://doi.org/10.5281/zenodo.17399948 [36].

Acknowledgments: This work is supported by the University of Chicago Joint Task Force Initiative (JTFI) – Partnerships for Emerging Technologies Seed Funding. CT and DWM are supported by the National Science Foundation under Grant No. PHY-2310094. AG, JN, and NT are supported by FermiForward Discovery Group, LLC under Contract No. 89243024CSC000002 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

[1] T. Bose, P. Elmer, and V. I. M. Outschoorn, Building an ai-native research ecosystem for experimental particle physics: A community vision (2026), arXiv:2602.17582 [hep-ex].

[2] J. Gonski and J. Ott, Machine Learning on Heterogeneous, Edge, and Quantum Hardware for Particle Physics (ML-HEQUPP) (2026), arXiv:2602.22248 [physics.ins-det].

[3] Laura Boggia and Carlos Cocha and Fotis Giasemis and Joachim Hansen and Patin Inkaew and Kaare Endrup Iversen and Pratik Jawahar and Henrique Pineiro Monteagudo and Micol Olocco and Sten Astrand and Martino Borsato and Leon Bozianu and Steven Schramm, Perspective on machine learning for real-time analysis at the Large Hadron Collider experiments ALICE, ATLAS, CMS and LHCb (2026), arXiv:2506.14578 [hep-ex].

[4] M. M. Cohen and the ATLAS Collaboration, *GELATO: A Generic Event-Level Anomalous Trigger Option for ATLAS*, Technical Report CERN-EP-2025 (CERN, 2025).

[5] A. Gandrakota (CMS), PoS **ICHEP2024**, 1025 (2025), arXiv:2411.19506 [hep-ex].

[6] CMS ECAL Collaboration, Comput. Softw. Big Sci. **8**, 11 (2024), arXiv:2309.10157 [physics.ins-det].

[7] CMS HCAL Collaboration, Sensors **25**, 3475 (2025), arXiv:2408.16612 [cs.LG].

[8] CMS Muon Collaboration, Comput. Softw. Big Sci. **10**, 4 (2026), arXiv:2501.13789 [hep-ex].

[9] K. Rajput, S. Lin, A. Edelen, W. Blokland, and M. Schram, Mach. Learn. Sci. Tech. **6**, 031001 (2025), arXiv:2504.03793 [cs.LG].

[10] E. Liuzzo, R. Falomo, S. Paiano, A. Treves, M. Uslenghi, C. Arcidiacono, A. Baruffolo, E. Diolaiti, J. Farinato, M. Lombini, A. Moretti, R. Ragazzoni, R. Brast, R. Donaldson, J. Kolb, E. Marchetti, and S. Tordo, The Astronomical Journal **152**, 38 (2016).

[11] C. Mahesh, K. Dona, D. W. Miller, and Y. Chen, in *34th Conference on Neural Information Processing Systems* (2021) arXiv:2104.06622 [cs.AI].

[12] T. Evans, C. Fitzpatrick, and J. Horswill, Comput. Softw. Big Sci. **9**, 7 (2025), arXiv:2502.09557 [hep-ex].

[13] CMS Collaboration, About CMS Open Data (2024), accessed: March 9, 2025.

[14] ATLAS Collaboration, 2015-2016 Open Data for Research from the ATLAS experiment (2024), accessed: November 4, 2025.

[15] CERN, Open Data Portal (2014), accessed: November 4, 2025.

[16] CMS Collaboration, ZeroBias primary dataset in MINIAOD format from RunG of 2016 (/ZeroBias/Run2016G-UL2016_MiniAODv2-v1/MINIAOD) (2024).

[17] CMS Collaboration, ZeroBias primary dataset in MINIAOD format from RunH of 2016 (/ZeroBias/Run2016H-UL2016_MiniAODv2-v1/MINIAOD) (2024).

[18] CMS Collaboration, Neutrino_E-10_gun in PREMIX format for 2016 collision data (2024).

[19] C. Bierlich, S. Chakraborty, N. Desai, L. Gellersen, I. Helenius, P. Ilten, L. Lönnblad, S. Mrenna, S. Prestel, C. T. Preuss, T. Sjöstrand, P. Skands, M. Utheim, and R. Verheyen, SciPost Phys. Codebases , 8 (2022).

[20] CMS Collaboration, Simulated dataset TTToHadronic_TuneCP5_13TeV-powheg-pythia8 in MINIAODSIM format for 2016 collision data (2024).

[21] S. Alioli, P. Nason, C. Oleari, and E. Re, JHEP **06** (2010), 043, arXiv:1002.2581 [hep-ph].

[22] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H. S. Shao, T. Stelzer, P. Torrielli, and M. Zaro, JHEP **07** (2014), 079, arXiv:1405.0301 [hep-ph].

[23] CMS Collaboration, SUSY_GluGluH_01J_HToAATo4B_M-12_TuneCP5_13TeV_madgraph_pythia8 in MINIAODSIM format for 2016 collision data (2024).

[24] ATLAS Collaboration, European Physical Journal C **82**, 1 (2022).

[25] CMS collaboration, Journal of Instrumentation **15** (10), P10017.

[26] G. E. Hinton and R. R. Salakhutdinov, Science **313**, 504 (2006).

[27] J. An and S. Cho, Special lecture on IE **2**, 1 (2015).

[28] J. Duarte, S. Han, P. Harris, S. Jindariani, E. Kreinar, B. Kreis, V. Loncar, J. Ngadiuba, M. Pierini, D. Rankin, R. Rivera, S. Summers, N. Tran, and Z. Wu, in *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '19 (Association for Computing Machinery, New York, NY, USA, 2019) p. 305.

[29] ATLAS Collaboration, Journal of Instrumentation **19** (06), P06029.

[30] CMS Collaboration, Journal of Instrumentation **19** (11), P11021.

[31] B. Bradu, E. B. Viñuela, R. Marti, and F. Tilaro, in *Proc. of International Conference on Accelerator and Large Experimental Control Systems (ICALEPCS'17), Barcelona, Spain, 8-13 October 2017*, International Conference on Accelerator and Large Experimental Control Systems No. 16 (JACoW, Geneva, Switzerland, 2018) pp. 1017–1023, https://doi.org/10.18429/JACoW-ICALEPCS2017-WEAPL02.

[32] R. Steinhagen, *LHC Beam Stability and Feedback Control*, Tech. Rep. (CERN, 2007).

[33] ATLAS Collaboration, Journal of Instrumentation **15** (10), P10004.

[34] CMS Collaboration, *2024 HLT timing and throughput results*, Tech. Rep. (CERN, 2024).

[35] S. Emami, G. Salvi, and Z. Ding, Adaptive-particlephysics-triggers (2025), gitHub repository.

[36] Zenodo, Zenodo – Research. Shared. (2024), accessed: March, 2026.