Journal Pre-proof

Online Change Point Methods for Structural Shock Detection on Resource-Constrained Edge Devices: An Analytical Benchmarking Study

Zhymir Thompson[a,b], Yanzhou Fu[a,c], Justice Adjei Owusu[d], Joud N. Satme[a], Austin R.J. Downey[a,e,*], Gurcan Comert[d] and Jason D. Bakos[b]

[a]Department of Mechanical Engineering, University of South Carolina, Columbia, 29208, SC, USA
[b]Department of Computer Science and Engineering, University of South Carolina, Columbia, 29208, SC, USA
[c]Department of Ocean and Mechanical Engineering, Florida Atlantic University, Boca Raton, 33431, FL, USA
[d]Department of Computational Data Science and Engineering, North Carolina A&T State University, Greensboro, 27411, NC, USA
[e]Department of Civil Engineering, University of South Carolina, Columbia, 29208, SC, USA

ARTICLE INFO

Keywords:
Structural health monitoring
Time-series analysis
Real-time change point detection
High-rate dynamics
Performance benchmarking

ABSTRACT

Timely detection of structural shock events is critical for real-time health monitoring and system protection. This study evaluates five online detection methods for identifying shock-induced changes in drop-tower acceleration data: Bayesian Online Change Point Detection (BOCPD), Cumulative Summation (CUSUM), Expectation Maximization (EM)-based Gaussian mixture classification, Grey Systems Modeling (GM), and Isolation Forest. Each method is benchmarked against an offline detection baseline using precision, recall, F1 score, and detection delay. BOCPD achieved the best balance of accuracy and responsiveness (F1 score: 0.602; delay: 0.277 ms); Isolation Forest provided the second-highest F1 score (0.513), whereas GM achieved the shortest clean-signal detection delay (0.208 ms). The F1-score ranking is unchanged across four alternative offline reference definitions derived from the same acceleration signal. BOCPD, CUSUM, EM, and Isolation Forest retain their clean-signal performance under white Gaussian noise down to a signal-to-noise ratio of 0 dB, whereas the precision of GM degrades from 0.97 to 0.54. Parameter sweeps identify the reported BOCPD setting as its best tested configuration, but also show that cross-algorithm F1 rankings depend on the chosen parameters, particularly for GM and Isolation Forest. Repeating the evaluation on all five experimental runs, including a reference-light early-impact check, confirms the ordering BOCPD > Isolation Forest > CUSUM > EM on every run, whereas the rank of GM varies. Results highlight the effectiveness of lightweight online detection methods for shock detection and their algorithmic suitability for embedded structural health monitoring and control systems, pending validation on embedded hardware. Data and code are publicly available.

1. Introduction

Structures across engineering domains, including aerospace components [1], civil infrastructure [2], and electronic assemblies [3], are often subjected to sudden, high-intensity forces such as impacts, collisions, or transient vibrations [4]. These shock events can significantly exceed normal operating loads and may cause irreversible damage if not addressed promptly. To mitigate such risks, it is essential to detect these dynamic events in real time and accurately classify the structural state [5]. Timely detection is a prerequisite for future real-time controllers that close the loop to enable vibration and shock mitigation [6, 7], thereby protecting the integrity and functionality of the structure [8].

*Corresponding author
austindowney@sc.edu (A.R.J. Downey)
ORCID(s): 0000-0002-6704-0221 (Z. Thompson); 0000-0002-5471-9972 (Y. Fu); 0009-0004-1232-5838 (J.A. Owusu); 0000-0002-6769-8234 (J.N. Satme); 0000-0002-5524-2416 (A.R.J. Downey); 0000-0002-2373-5013 (G. Comert); 0000-0002-0821-6258 (J.D. Bakos)

One promising technique for real-time shock event detection is change point detection (CPD), which identifies points in sequential data that indicate abrupt changes in the sequence's statistical properties. CPD has been widely applied in fields such as finance [9], biomedical science [10], social and behavioral data science [11, 12], and climate science [13]. In the context of structural health monitoring (SHM), CPD provides a powerful approach to identifying the onset and cessation of shock events by monitoring changes in sensor measurements. Fig. 1 shows the general change point detection process in SHM. CPD is formally defined as the identification of abrupt changes in the generative parameters of a data sequence [14], where the statistical distribution after a change point differs from that before it. Algorithms process streaming data either as individual points or windows and use probabilistic or statistical models to assess whether the input belongs to the expected distribution or indicates a shift. CPD methods are typically classified as online or offline. Online algorithms operate sequentially, are capable of real-time detection, and do not require knowing the total number of change points in advance to recognize when one occurs. Offline methods, in contrast, process complete data sequences only after collection, so while often more accurate, they are impractical for real-time and online use.

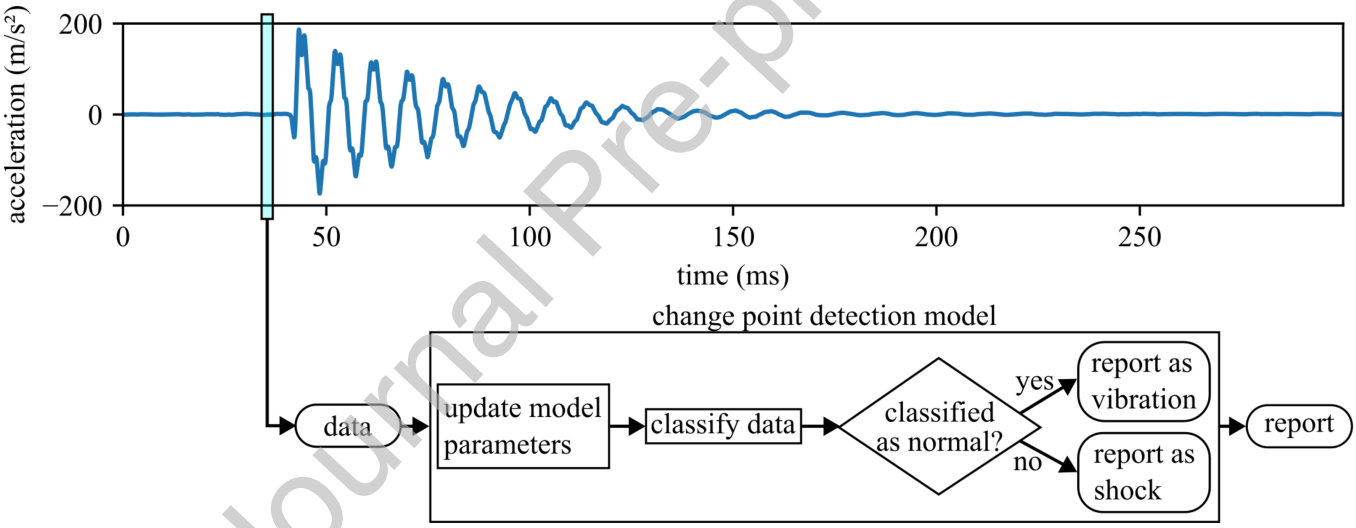


Figure 1: The general change point detection process.

Although CPD has been extensively studied in statistical and data science contexts, its application to engineering systems is gaining increasing attention. For instance, Kawahara et al. proposed a recursive change-point detection algorithm based on subspace identification, which was applied to detect both abrupt and gradual faults in a simulated spacecraft attitude control system using reaction wheels [15]. The method successfully identified both a commanded orientation change and a slowly developing failure in the roll-axis reaction wheel, demonstrating its ability to capture evolving anomalies in high-dimensional control systems. Shi et al. combined change point detection with deep learning in a dual-LSTM framework designed to detect transition points in degradation trajectories and estimate the remaining useful life of turbofan engines using NASA CMAPSS data [16]. This approach highlights CPD's utility not only for

event detection but also as a trigger for downstream prediction tasks, an approach relevant to real-time structural shock monitoring. Nigro et al. employed statistical Change Point Analysis (CPA) to detect abrupt changes in modal parameters of finite element models, enabling the localization and timing of stiffness degradation in beams and plates [17]. Their work demonstrates CPA's effectiveness for vibration-based SHM even in noisy environments. Wu et al. integrated multiple change-point models with particle filters to track transitions in reactor conditions without relying on labeled data [18]. Their method performed robustly in online settings, reinforcing CPD's feasibility in real-time engineering applications. Sharma et al. proposed a fast, semi-automatic change point detection method for deformation monitoring based on Kullback-Leibler divergence in a Gaussian-transformed space [19]. Requiring only a sliding window size as input, the method achieved over 99% faster computation than existing techniques with similar accuracy. It was validated on several real-world datasets, including drilling logs, InSAR data from the Galápagos Islands, and multiple slope deformation cases, demonstrating effective detection of critical change points such as the onset of slope movement. Ge et al. introduced a segmental semi-Markov model for time-series change-point detection, targeting semiconductor manufacturing applications [20]. By combining segment modeling (e.g., linear trends) with semi-Markov state duration modeling, the method addressed limitations of standard Hidden Markov Models. It employed Bayesian inference and Expectation Maximization to estimate change points efficiently and accurately, showing improved performance under noisy conditions in both simulated and real plasma etch data.

Within structural health monitoring specifically, change point methods have been applied across a range of structures and timescales. Liao et al. developed a sequential change-point method for damage detection and localization that accommodates an unknown post-damage feature distribution, verified on shake-table data from a steel frame [21]. Mariani and Cawley employed generalized-likelihood-ratio change detection to improve the sensitivity of guided-wave monitoring systems [22]. Li et al. combined offline and online change point detection to identify anomalous crack behavior in concrete dams in long-term service [23], and Lei et al. formulated structural condition assessment as a distributional change-point problem for damage-sensitive features of bridge monitoring data [24]. A recent systematic review by Dao examines the integration of trend monitoring with change point detection, including BOCPD and CUSUM, for wind turbine diagnostics [25], and robust and scalable variants of BOCPD remain an active methodological research direction [26]. Complementing these application studies, van den Burg and Williams established a benchmark of a large set of detection algorithms on 37 annotated real-world time series, and found binary segmentation and BOCPD to be among the best-performing methods [27]; their series are, however, general-domain data at comparatively low sampling rates. In the high-rate regime, dedicated test beds were developed precisely because benchmark experiments for validating detection algorithms at these timescales were lacking [28]. Table 1 summarizes the scope of representative prior studies alongside the present work. Because the underlying datasets, event types, and ground-truth definitions differ, the detection metrics reported by these studies are not numerically comparable;

Journal Pre-proof

Journal Pre-proof

the comparison therefore concerns the scope of each evaluation. To the authors' knowledge, no prior study combines online algorithms, high-rate experimental structural data, standardized metrics, robustness analyses of the evaluation itself, and openly released data and code.

Table 1
Scope of representative change point detection studies in engineering and structural health monitoring, compared with the present work. A dash indicates that the item is not reported in the cited publication. Reported detection metrics are not numerically comparable across studies because the datasets and ground-truth definitions differ.

study	application and data	online	multi-algorithm	robustness analyses	open data and code
Kawahara et al. [15]	simulated spacecraft attitude control	yes	no	—	—
Nigro et al. [17]	modal parameters of beams and plates	no	no	measurement noise	—
Liao et al. [21]	shake-table steel frame vibration	yes	no	unknown post-damage distribution	—
Mariani & Cawley [22]	guided-wave monitoring	yes	no	operational variations	—
Li et al. [23]	concrete dam cracks, long-term	offline, online	no	—	—
Lei et al. [24]	bridge damage-sensitive features	no	no	false-positive control	—
Sharma et al. [19]	deformation monitoring (slopes, InSAR)	semi-automatic	no	multiple real datasets	—
van den Burg & Williams [27]	37 general-domain time series	offline, online	yes	multiple human annotations	data, code
Thompson et al. [29]	drop-tower shock (preliminary)	yes	yes	—	data
this work [30]	drop-tower shock at 1 MS/s	yes	yes	ground truth, noise, parameters	data, code, library

Despite its potential, current research on CPD in structural applications faces several limitations. Numerous studies concentrate solely on offline detection methods. Despite their accuracy, these methods are impractical for real-time applications because of their significant computational demands and latency. While such methods remain valuable for benchmarking or post-event analysis, they are less practical for embedded or time-critical applications. Online CPD approaches, though more responsive, often require prior knowledge of system models or extensive training data, requirements that may not be feasible in real-world deployments. Furthermore, few studies systematically compare multiple online CPD algorithms using standardized evaluation metrics and high-rate experimental data from physical structures. Taken together, these limitations motivate three specific needs: (i) online methods whose computational and memory footprints permit deployment on resource-constrained edge devices, in contrast to the latency and cost of offline methods; (ii) methods that operate without detailed system models or extensive training data; and (iii) a standardized experimental benchmark that also quantifies how detection conclusions depend on measurement noise, algorithm parameter choices, and the offline reference used for evaluation. This work addresses these needs directly.

This study evaluates five online detection methods (i.e., Bayesian Online Change Point Detection (BOCPD), Cumulative Summation (CUSUM), Expectation Maximization (EM), Grey Systems Modeling (GM), and Isolation

Journal Pre-proof

Forest) for their effectiveness in detecting structural shock events based on acceleration data collected from drop tower experiments. The goal is to assess the applicability of CPD techniques for real-time structural health monitoring, focusing on both detection accuracy and computational efficiency. The contributions of this work are threefold: (1) A comprehensive algorithmic benchmark of five online detection methods is conducted using real-world high-rate acceleration data, with performance evaluated through standardized metrics such as precision, recall, F1 score, and detection delay [30], complemented by sensitivity analyses that quantify the robustness of the benchmark conclusions to the ground-truth definition, to additive measurement noise, and to algorithm parameter choices; (2) Strengths and limitations of the algorithms are analyzed in terms of both detection performance and computational cost, offering guidance for selecting suitable methods for real-time deployment; (3) A publicly available dataset of high-rate structural acceleration data is provided to support reproducibility and enable comparative evaluation of CPD techniques in structural shock detection [31].

The novelty of this work lies in the benchmarking framework rather than in a new detection algorithm. To the authors' knowledge, this is the first systematic comparison of online CPD methods on high-rate (1 MS/s) experimental structural data in which the evaluation itself is stress-tested: shock boundaries are compared across four alternative offline reference definitions (not independent physical ground truth), the noise robustness of every algorithm is characterized down to a signal-to-noise ratio of 0 dB, and the influence of principal detection parameters is quantified. Together with the openly released dataset, benchmarking pipeline, and reusable algorithm library (including compiled Rust implementations of BOCPD, CUSUM, and EM), this provides a reproducible reference against which future online detection methods for high-rate structural health monitoring can be measured.

This work extends the authors' preliminary conference study [29], which evaluated only CUSUM and an EM-based classifier using qualitative detection plots. The present study benchmarks five methods quantitatively across five runs, examines sensitivity to the offline reference, noise, and detector parameters, analyzes computational cost, and releases the data and code.

The methodological contribution is a reproducible evaluation protocol for online detectors under uncertain shock-boundary labels, rather than a new detection algorithm. Since event boundaries are inferred from the measured response, the protocol examines four alternative offline references, a reference-light check on clearly separated normal and shock samples, controlled noise injection, detector-parameter sweeps, and replication across five runs. At the nominal settings, BOCPD > Isolation Forest > CUSUM > EM is consistent across runs, whereas the ranking of GM varies. These findings characterize which comparisons are repeatable under the stated conditions, without establishing a universally optimal algorithm.

Journal Pre-proof

Journal Pre-proof

2. Experimental and Detection Methods

To evaluate the selected online detection methods in terms of speed and computational efficiency, acceleration data collected from drop tower experiments were used. The implementation code for these algorithms is publicly available in an online repository [30]. The acceleration data from the experiment can be found in an online repository [31]. For brevity, the main results and discussion focus on Test 1. Results for the remaining tests (Tests 2–5) are included in the Appendix, and Section 3.5 examines the consistency of the benchmark findings across Tests 2–5. Additionally, a library implementation is available in a public repository to facilitate future use of the algorithms [32].

2.1. Experiment setup

The experiment depicted in Figure 2 was designed to simulate an impact applied to an electronics package undergoing steady forced vibration, an operating condition relevant to applications that require real-time shock event classification [5]. This test setup was initially reported by Thompson et al. [29], together with preliminary results on change point algorithms for shock detection. To develop the desired conditions, the test structure needed to vibrate in a steady state, and an impact force needed to be applied at some point during the experiment. The click wheel actuator was used to induce constant vibration in the printed circuit board (PCB), while the shock test system supplied the impact. Although the structure under test is a PCB, this setup replicates dynamic conditions relevant to real-world active shock mitigation control applications and enables reproducible evaluation of change point detection algorithms for real-time shock detection.

The test subject was a PCB, which was fixed in a cantilever beam configuration. An accelerometer was attached near the tip of the free end to collect acceleration data throughout the experiment. To excite the PCB, an actuator equipped with a click wheel was used. The actuator was placed in a housing mounted on the drop table. A metal beam was clamped to the fixed end of the PCB so that the free end of the beam vibrated under the click wheel's excitation, thereby transferring vibrations to the PCB. Before data collection, the table was raised to a starting height. The experiment began with the actuator vibrating the PCB. After a set period, the drop table and all mounted components were released and dropped to the base level. The system was left undisturbed for a short duration after impact to observe its response. Finally, data collection was stopped, and the experiment concluded. The data are provided through an online repository [31], which includes raw and filtered acceleration data for each of the five experimental runs. The present analysis uses the filtered data from Test 1, while evaluations for the remaining filtered runs are provided in the Appendix and repeated with the settings of the main text in Section 3.5.

2.2. Online detection methods

In this study, we focus on five representative online detection methods (BOCPD, CUSUM, EM, GM, and Isolation Forest), selected to span diverse detection principles and computational characteristics. Compared with alternatives

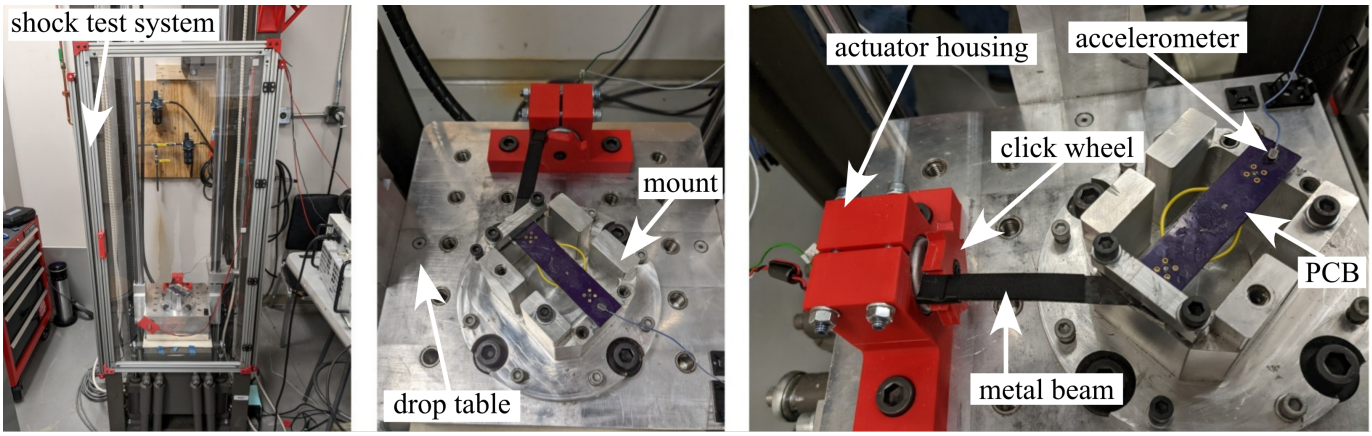


Figure 2: Experimental setup: (a) entire shock test system, (b) top-down view of the drop table and PCB fixed to the mount, and (c) close-up of the experimental components, including the motor with click wheel actuator, PCB, and accelerometer.

such as kernel-based methods, neural-network models, or segmentation-based strategies, these methods offer a favorable balance of real-time performance, algorithmic diversity, and implementation accessibility. Importantly, each method can process streaming data without requiring prior knowledge of the number of change points, making the set suitable for evaluating practical real-time structural monitoring approaches.

2.2.1. Bayesian online change point detection

Bayesian online change point detection (BOCPD) is an online change point detection algorithm based on Bayesian inference. The model assumes that the observation sequence consists of one or more partitions, with all observations within a partition drawn from the same distribution. Adjacent partitions are assumed to arise from different distributions; otherwise, they could be treated as a single partition. All observations must belong to the same distribution family, although the algorithm does not require a specific family in general. The implementation used here assumes a normal distribution with unknown mean and variance. The algorithm requires some initial parameters for the model. Generally, one may choose a naive model with no prior knowledge about the data or a model with some amount of prior knowledge. The naive approach was chosen for this model.

BOCPD was introduced by Adams and Mackay [33] and has subsequently been applied to satellite and traffic data [34]. The algorithm uses the current observation together with information from the previous step to update predictions recursively, which improves computational efficiency. It requires an underlying predictive model (UPM), a hazard function, and initial model conditions. The UPM is used to compute $p(X_t|X_{(t-i)};\theta_m = [\mu,\kappa,\alpha,\beta])$ for any $i \in [1,\dots,t-1]$ at time t [35]. This predictive Bayesian model uses the current run length to make predictions. The hazard function $h(r|\lambda)$ describes the probability that a change point occurs given the current run length r. The set $\theta = [\mu,\kappa,\alpha,\beta]$ contains the initial conditions for the UPM and therefore serves as the algorithm's hyperparameters. The values of θ are assumed to be fixed and known beforehand, although methods exist to estimate them when they are

unknown. Because appropriate hyperparameter selection is crucial to BOCPD performance, estimating these values without compromising the online nature of the algorithm would be beneficial. The goal of BOCPD is to predict the most likely run length r at time t.

$$\begin{aligned}\pi(\mu,\tau) &= \text{NormalGamma}(\mu_0,\kappa_0,\alpha_0,\beta_0)\\ &\propto \tau^{\alpha_0-1/2}e^{-\beta_0\tau}e^{\frac{-\kappa_0\tau(\mu-\mu_0)^2}{2}}\end{aligned} \tag{1}$$

$$p(X|\mu,\tau) \propto \tau^{t/2}e^{\frac{-\tau(\sum(X_i-\bar{X})^2+t(\bar{X}-\mu)^2)}{2}} \tag{2}$$

$$p(\mu,\tau|X) \propto \tau^{t/2+\alpha_0-1/2}e^{\frac{-\tau(\sum(X_i-\bar{X})^2+2\beta_0+\frac{\kappa_0 t(\bar{X}-\mu)^2}{\kappa_0+t})}{2}} \tag{3}$$

Algorithm 1 assumes a Gaussian-Gamma prior as given in Eq. (1). Combining this prior with the likelihood function in Eq. (2) yields the Gaussian-Gamma posterior in Eq. (3).

$$\begin{aligned}\mu &= \frac{\kappa_0\mu_0+t\bar{X}}{\kappa_0+t}\\ \kappa &= \kappa_0+t\\ \alpha &= \alpha_0+t/2\\ \beta &= \beta_0+\sum(X_i-\bar{X})^2/2+\frac{\kappa_0 t(\bar{X}-\mu_0)^2}{2(\kappa_0+t)}\end{aligned} \tag{4}$$

$$p(X_t|X_{(t-i)}) = \int_\tau \Gamma(\tau|X)N(X_t|\tau,X)d\tau \tag{5}$$

Finally, the predictive distribution in Eq. (5) can be evaluated in closed form as a Student's t distribution, $ST_{2\alpha}(\mu,\frac{\kappa+1}{\kappa}\frac{\beta}{\alpha})$, as shown in Eq. (6), with online Bayesian parameter updates from Eq. (4). For completeness, the derivations can be verified from [36, 37].

$$\begin{aligned}p(X_t|X_{(t-i)}) = \int_\tau \Gamma(\alpha_0+\frac{t}{2},\beta_0+\frac{1}{2}(X_i-\bar{X})^2+\frac{t\kappa_0(\bar{X}-\mu_0)^2}{2(t+\kappa_0)}))N(X_t|\tau,X)d\tau\\ ST_{2\alpha}(\mu,\frac{\kappa+1}{\kappa}\frac{\beta}{\alpha})\end{aligned} \tag{6}$$

2.2.2. Cumulative Summation

Cumulative Summation (CUSUM) is an algorithm used to detect shifts in the mean level. Introduced by Page as a sequential procedure for detecting a change in the distribution of a monitored process [38], CUSUM is treated as a canonical online change detection method in the sequential-analysis literature [39]. The algorithm detects when a

Algorithm 1 BOCPD for detecting changes in a series of N observations

Require: Choose hyperparameters; $\mu = 0, \kappa = 2, \alpha = 1, \beta = 1, \lambda = 2$
  $prevMax = \emptyset$, $max = 0$, $rL = [0]$, $\alpha_{0:i} = [\alpha]$, $\beta_{0:i} = [\beta]$, $\mu_{0:i} = [\mu]$, $\kappa_{0:i} = [\kappa]$.
  for $Y_i \ni i \in 1 : N$ **do**
    $hazard = \frac{1}{\lambda}$, $priors_{0:i} = \mathrm{ST}(x, \alpha_{0:i}, \beta_{0:i}, \mu_{0:i}, \kappa_{0:i})$
    $probs_{0:i+1} = \emptyset$, $probs_{1:i} = (probs_{0:i} + priors_{0:i}) \cdot (1 - hazard)$, $probs_0 = \sum priors_{0:i} \cdot hazard$.
    if $prevMax < max$ **then**
      $probs = [\sum probs]$, $rL = [0]$, $\alpha_{0:i} = [\alpha]$, $\beta_{0:i} = [\beta]$, $\mu_{0:i} = [\mu]$, $\kappa_{0:i} = [\kappa]$.
    else
      $\alpha_{0:i+1} = [\alpha] + (\alpha_{0:i} + 0.5)$, $\beta_{0:i+1} = [\beta] + (\beta_{0:i} + \frac{\kappa_{0:i} \cdot (Y_i - \mu_{0:i})^2}{2 \cdot (\kappa_{0:i}+1)})$
      $\mu_{0:i+1} = [\mu] + (\frac{\kappa_{0:i} \cdot \mu_{0:i} + Y_i}{\kappa_{0:i}+1})$, $\kappa_{0:i+1} = [\kappa] + (\kappa_{0:i} + 1)$.
    end if
    $cpProb = \mathrm{ST}(x, \alpha_{0:i+1}, \beta_{0:i+1}, \mu_{0:i+1}, \kappa_{0:i+1})$
    if $cpProb < 0.05$ **then** Y_i is a change point
    end if
  end for

sample statistic of the data has begun to shift noticeably instead of when any specific point passes a threshold. Assume that the observations X_i typically follow a normal distribution. Before a shock occurs, $X_i \sim N(\mu_1, \sigma^2)$, where the mean μ and variance σ^2 are assumed known a priori. After a possible shock begins, a new state mean μ_2 is observed. Statistical hypothesis testing can then be used to determine whether the current state is typical or indicative of a potential shock. More specifically, one can compute $S_t = \sum_{i=1}^{t} s_i$ for some sample size t where $s(i)=\ln(p_{\mu_2}(X_i)/p_{\mu_1}(X_i))$ is the log-likelihood ratio. These calculations can then be used in Eq. (7) to decide if the latest observations suggest a state of shock. The algorithm concurrently tracks positive (C_t^+) and negative deviations (C_t^-) of the mean estimate from the expected mean [40]. A change point is expected to have occurred if either deviation exceeds a predefined threshold (H). The algorithm tracks the sample mean using a weighted average of the previous estimated sample mean and the current observation. The differences are used to estimate the shifts in mean level (Eq. (8) and (9)). This algorithm has the advantage that it is sensitive to mean level shifts, so it can detect subtle anomalies that may otherwise be missed.

$$d_t = \begin{cases} 0\ ,\ C_t < H\ ;\ H_0 \text{ no shock} \\ 1,\ C_t \geq H;\ H_1 \text{ shock} \end{cases} \tag{7}$$

where $C_t = S_t - m_t$ and $m_t = [S_i]_{1 \leq i \leq t}^{-}$.

$$\begin{aligned} C_t^+ &= \max\left\{0, C_{t-1}^+ + \frac{\alpha D_t}{\sigma^2}\left[X_t - D_t - \alpha D_t/2\right]\right\} \\ C_t^- &= \max\left\{0, C_{t-1}^- - \frac{\alpha D_t}{\sigma^2}\left[X_t + D_t + \alpha D_t/2\right]\right\} \end{aligned} \tag{8}$$

where $D_t = (\bar{\mu}_t - \mu_1)$ and $\bar{\mu}_t = \alpha\bar{\mu}_{t-1} + (1-\alpha)X_t$. This variant of the CUSUM algorithm is somewhat insensitive to $K = \delta\sigma/2$ and $\delta = 1.0$.

$$\mu_2 = \begin{cases} \mu_1 + K + \frac{C_t^+}{N^+}, C_t^+ > H \\ \mu_1 - K - \frac{C_t^-}{N^-}, C_t^- > H \end{cases} \tag{9}$$

The algorithm is defined by the values of K and H. The parameter K is a reference value determined by the shift in mean level that the CUSUM algorithm is designed to detect. The parameter H is a threshold selected to provide a long in-control average run length (ARL) while maintaining a short out-of-control ARL. The conventional CUSUM algorithm with $K = \delta\sigma/2$ and $H = 5\sigma$ is optimal since it provides a good balance between detecting a shift from the target mean and maximizing ARL. The algorithm can detect a shift of $\delta\sigma$ from the target mean when $K = \delta\sigma/2$. By setting $\delta = 1$, $K = \delta\sigma/2 = \sigma/2$. These values of K and H provide an algorithm with good ARL properties in the presence of small shifts in the process mean. In this study, a threshold of $H = 3\sigma$ was adopted for the main benchmark; the sensitivity of the results to this choice, over $H \in [1\sigma, 9\sigma]$, is quantified in Section 3.4. The CUSUM procedure used in this study is summarized in Algorithm 2.

Algorithm 2 CUSUM for detecting changes in a series of N observations

Require: Choose parameters: $\mu = \overline{Y}_{1:20}$, $\sigma = \sqrt{S^2_{Y_{1:20}}}$, $K = \delta\sigma/2$, $\delta = 1$, $\alpha = 0.025$

Choose threshold H, e.g., $H = 5\sigma$ or $H = 3\sigma$
for $i = 1, \ldots, N$ **do**
  $D_i = \overline{\mu}_{i-1} - \mu_i$
  $C_i^+ = \max\left\{0, C_{i-1}^+ + \frac{\alpha D_i}{\sigma^2}\left[Y_i - D_i - \frac{\alpha D_i}{2}\right]\right\}$
  $C_i^- = \max\left\{0, C_{i-1}^- - \frac{\alpha D_i}{\sigma^2}\left[Y_i + D_i + \frac{\alpha D_i}{2}\right]\right\}$
  $\overline{\mu}_i = \alpha\overline{\mu}_{i-1} + (1-\alpha)Y_i$
  if $C_i^+ > H \vee C_i^- > H$ **then**
    Y_i is detected as a change of state
    Set $C_i^+ = C_i^- = 0$
  end if
end for

2.2.3. Expectation Maximization-Based Gaussian Mixture Classification

Expectation Maximization (EM) is an iterative algorithm for maximum-likelihood estimation in models with latent variables [41]. Strictly speaking, EM is a maximum-likelihood estimation algorithm rather than a change point detector; in this work it is employed to fit a two-component Gaussian mixture whose posterior responsibilities provide the online normal/shock classification of each sample. The abbreviation EM is retained hereafter for brevity, with this distinction understood. The implementation assumes that the underlying data follow normal distributions and that the normal and abnormal states correspond to distinct, separable components, as represented in Eq. (10). Samples of data are provided

for each of the distributions with parameters θ=[$\theta_1 = (\mu_1, \sigma_1)$, $\theta_2 = (\mu_2, \sigma_2)$, π]. The algorithm repeatedly computes likelihoods (i.e., the responsibilities γ_i) for all of the points (the samples for the distributions along with the unknown), followed by updating the parameters for each distribution until either the parameters or predictions stabilize. Once stable, the likelihood for the given observation is used to determine if it is a change point or belongs to Y_2 (i.e., shock state).

First, assume that all of the samples can be classified into one of a finite set of classes. Then the likelihood of a sample in the sample vector Y can be represented as a mixture:

$$Y = (1 - \Delta)Y_1 + \Delta Y_2 \tag{10}$$

where $Y_1 \sim N(\mu_1, \sigma_1^2)$ represents the normal-state distribution, $Y_2 \sim N(\mu_2, \sigma_2^2)$ represents the abnormal-state distribution, and $\Delta \in [0, 1]$ indicates whether a random sample is abnormal. The samples are selected to be representative of the population, so $P(\Delta = 1) = \pi$, where π is the proportion of abnormal samples. The corresponding mixture probability density is given by Eq. (11):

$$g_Y(y) = (1 - \pi)\phi_{\theta_1}(y) + \pi\phi_{\theta_2}(y) \tag{11}$$

where $\phi_\theta(x)$ denotes the normal probability density function. For a data set of N points, the log-likelihood function can be written as follows:

$$l(\theta, Z) = \sum_{i=1}^{N} \ln[(1 - \pi)\phi_{\theta_1}(y_i) + \pi\phi_{\theta_2}(y_i)] \tag{12}$$

where $\theta = [\theta_1 = (\mu_1, \sigma_1), \theta_2 = (\mu_2, \sigma_2), \pi]$ denotes the model parameters and Z denotes the observed data. Direct maximization of Eq. (12) is difficult. Introducing the latent variable Δ_i, with $\Delta_i = 1$ for observations assigned to Y_2 and $\Delta_i = 0$ otherwise, yields the complete-data likelihood in Eq. (13); the corresponding posterior responsibility is estimated using Eq. (14).

$$l(\theta; \Delta, Z) = \sum_{i=1}^{N} [(1 - \Delta_i) \ln[(1 - \pi)\phi_{\theta_1}(y_i)] + \Delta_i \ln[\pi\phi_{\theta_2}(y_i)]] \tag{13}$$

$$\gamma_i(\theta) = E(\Delta_i \mid \theta, Z) = P(\Delta_i = 1 \mid \theta, Z) \tag{14}$$

The EM-based Gaussian mixture classification procedure used in this study is summarized in Algorithm 3.

Journal Pre-proof

Algorithm 3 EM for detecting two mixtures in a series of N observations

Require: Two classes of observations $X_{1:100} = [X_{1:70} \sim N(\overline{Y}_{1:70}, S^2_{Y_{1:70}}), X_{71:100} \sim N(100, 20^2), Y_i]$

Initialize parameters, $\Theta = (\mu_1, \mu_2, \sigma_1, \sigma_2, \pi)$ (alternatively from $X_{1:100}$)

for $Y_i \ni i \in 1 : N$ **do**

  for $j \in 1 : 11$ **do**

$$p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2)) = \frac{\hat{\pi}\phi(Y_i|\hat{\mu}_2, \hat{\sigma}_2^2)}{\hat{\pi}\phi(Y_i|\hat{\mu}_2, \hat{\sigma}_2^2) + (1-\hat{\pi})\phi(Y_i|\hat{\mu}_1, \hat{\sigma}_1^2)}$$

  end for

$$\hat{\mu}_1 = \frac{\sum_{i=1}^{N}(1-p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2))Y_i)}{\sum_{i=1}^{N}(1-p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2)))}, \hat{\mu}_2 = \frac{\sum_{i=1}^{N} p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2))Y_i}{\sum_{i=1}^{N} p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2))}$$

$$\hat{\sigma}_1^2 = \frac{\sum_{i=1}^{N}(1-p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2))(Y_i - \hat{\mu}_1)^2)}{\sum_{i=1}^{N}(1-p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2)))}, \hat{\sigma}_2^2 = \frac{\sum_{i=1}^{N} p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2))(Y_i - \hat{\mu}_2)^2}{\sum_{i=1}^{N} p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2))}, \hat{\pi} = \frac{\sum_{i=1}^{N} p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2))}{N}.$$

  if $p(Y_i \sim N(\hat{\mu}_2, \hat{\sigma}_2^2)) > 0.01$ **then** Y_i is a change of state

  end if

end for

2.2.4. Grey Systems Model

The Grey Systems Model (GM) used here is a clustering algorithm based on Grey theory [42]. Grey system theory was introduced in 1982 and has since become a common approach for studying and modeling systems whose structure or operating mechanisms are not completely known [43]. In Grey theory, grey information refers to incomplete information about a system. Unknown values can be represented by Grey numbers, denoted by the symbol $\otimes$, which describe ranges of possible values. Grey theory also defines properties and operations for these numbers, including the core (or kernel) of the number $\hat{\otimes}$, the degree of greyness g°, and the whitenization of a grey number. The latter operation represents a grey number as a finite interval describing its possible values [44]. For clustering problems, Grey indices are utilized with intuitive, equal-length vector comparisons using aggregate sum and absolute value difference sequences. A sequence of nonnegative observations $X^{(0)} = (x^{(0)}(1), x^{(0)}(2), \ldots, x^{(0)}(n))$ is collected from a stochastic process (original series). The sequence $X^{(1)} = (x^{(1)}(1), x^{(1)}(2), \ldots, x^{(1)}(n))$ is an accumulation sequence derived from the values of $X^{(0)}$ and computed as in Eq. (15). Subsequently, $Z^{(1)} = (z^{(1)}(2), z^{(1)}(3), \ldots, z^{(1)}(n))$ is the mean sequence of $X^{(1)}$ and is generated using Eq. (16).

$$x^{(1)}(k) = \sum_{i=1}^{k} x^{(0)}(i) \tag{15}$$

$$z^{(1)}(k) = \frac{x^{(1)}(k-1) + x^{(1)}(k)}{2}, \forall k = 2, 3, \cdots, n \tag{16}$$

The image sequences are calculated using Eq. (17), while Eq. (18) defines the absolute difference image sequence and Eq. (19) defines the relative difference image sequence. Finally, the Grey indices ϵ are obtained in Eq. (20) where the condition $\epsilon > p$ is checked for a change point. For a simpler approach, Algorithm 4 uses only absolute differences s and checks if $s > (1-p)/p$ for a change point.

Journal Pre-proof

$$|s_i| = |\sum_{k=2}^{n-1} z_i^0(k) + \frac{1}{2} z_i^0(n)| \tag{17}$$

$$s = \sum_{k=2}^{n-1} |z_i^0(k) - z_j^0(k-1)| + \frac{1}{2}|z_i^0(n) - z_j^0(n-1)| \tag{18}$$

$$s_r = \sum_{k=2}^{n-1} \frac{|z_i^0(k) - z_i^0(k-1)|}{z_i^0(0) + c} + \frac{1}{2}\frac{|z_i^0(n) - z_j^0(n-1)|}{z_i^0(0) + c},\ c = x^{(0)}(1) \tag{19}$$

$$\epsilon = \frac{1}{2(1+ps)} + \frac{1}{2(1+ps_r)} \tag{20}$$

Algorithm 4 GM based on Grey Indices for comparing windows in a series of N observations

$\alpha = 0.5$, $window = 4$, $p = 0.05$
for $i = 1; i < N$ **do**
  $X_{1:window}$=0, $Z_{1:window}$=0
  for $j = 1; j < window + 1$ **do**
    X_j=$Y_i + X_{j-1}$, Z_j=$\alpha X_j + (1-\alpha)X_{j-1}$
  end for
  s=0
  for $j = 1; j < window + 1$ **do**
    s+=$|Z_j - Z_{j-1}|$
  end for
  s+=$0.5|Z_{window} - Z_{window-1}|$
  if $[(1-p)/p] < s$ **then** Y_i is a change of state
  end if
end for

Each online detection method was run on the acceleration data from a test, and the predicted change points were recorded for evaluation. The methods were then compared with the offline baseline using the performance metrics defined below.

2.2.5. Isolation Forest

Isolation Forest was included as a machine-learning method for comparison with the statistical online detection approaches. Isolation Forest is a useful algorithm for identifying anomalies in a dataset. The algorithm works on the assumption that anomalies within a sequence are rare and different from normal observations [45]. The algorithm has two stages. In the first step, the algorithm takes a subset of normal and abnormal data, with more normal than abnormal samples, and inserts each element into a tree so that each item is its own leaf. Essentially, the samples are partitioned recursively until each sample is isolated from the others. It isolates observations by randomly selecting a feature and splitting its values so that items above and below the chosen splitting value are placed into different sub-trees. The

Journal Pre-proof

second stage is prediction. An input value is sent along the tree until it reaches a leaf node, marking the end of the path. Anomalies will tend to have shorter path lengths than normal values, so the path length is used for detecting whether or not an item is anomalous.

```
function IsolationForest(k, d)                    ▷ Number of starting trees, training data.
    t ← []
    for 0 … k do
        s ← sample from(d)
        t_new ← IsoTreeHelper(s)
        t.append(t_new)
    end for
    return t
end function
function IsoTreeHelper(s)                         ▷ Batch of samples
    f ← choose feature(s)
    if CanPartition(s.f) then
        p ← PartitionFeature(s.f)
        left ← s[s.f < p]
        right ← s[s.f >= p]
        tree_left ← IsoTreeHelper(left)
        tree_right ← IsoTreeHelper(right)
        return (tree_left, tree_right)
    else
        return ∅
    end if
end function
```

2.3. Data Preparation

The methods required different input formats, so the acceleration data were transformed accordingly. For example, BOCPD accepts raw data and processes one observation at a time, whereas GM uses a window of nonnegative input values. Each model required hyperparameters to be set before use. These hyperparameters affect model performance by encoding prior assumptions about the data. Generally, the hyperparameters describe the assumed mean and distribution of the data. EM and Isolation Forest draw random training samples at initialization, so their results vary between runs. For the clean-signal benchmark and parameter-sensitivity analyses, their reported metrics are averaged over three runs using recorded random seeds (1–3). The detection figures show the run whose F1 score is closest to the corresponding mean. In the noise-sensitivity analyses, each independently generated noise realization is evaluated once with a recorded seed, as described in Section 3.3. BOCPD, CUSUM, and GM are deterministic.

An example of the signals collected is shown in Figure 3. The plot shows the system response to the impact. The accelerometer recorded a peak acceleration of approximately 187 m/s^2. There is a transient response lasting approximately 100 ms. The peak acceleration is orders of magnitude higher than the steady state. Figure 3 illustrates this difference. The steady-state response has an average peak acceleration of 1.5 m/s^2. The difference is so stark that

Journal Pre-proof

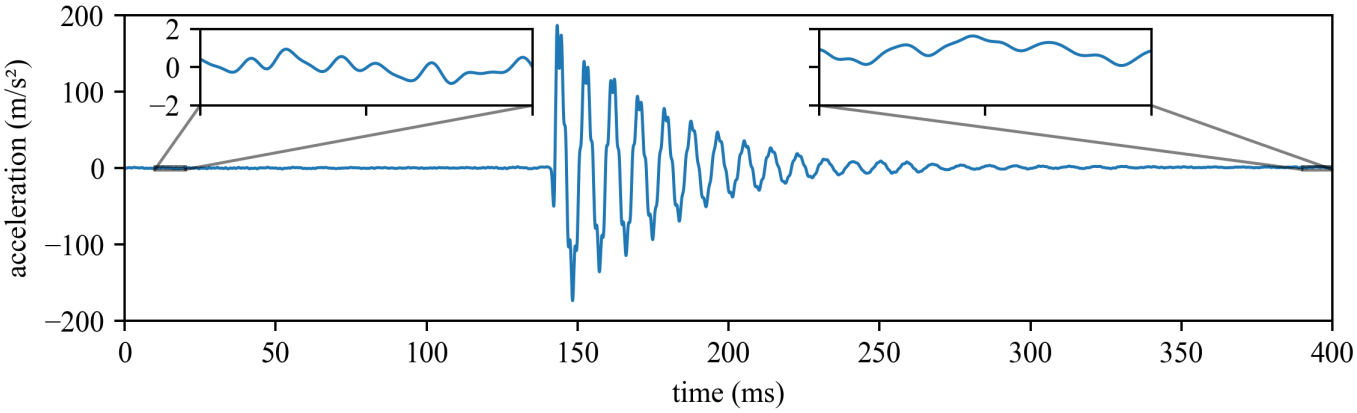


Figure 3: Acceleration signal with zoomed-in regions at the beginning and end of the record.

the steady-state response appears nonexistent. The structural responses at the beginning and end of the experiment run appear similar. This implies that the system was no longer in a state of shock by the end of the experiment.

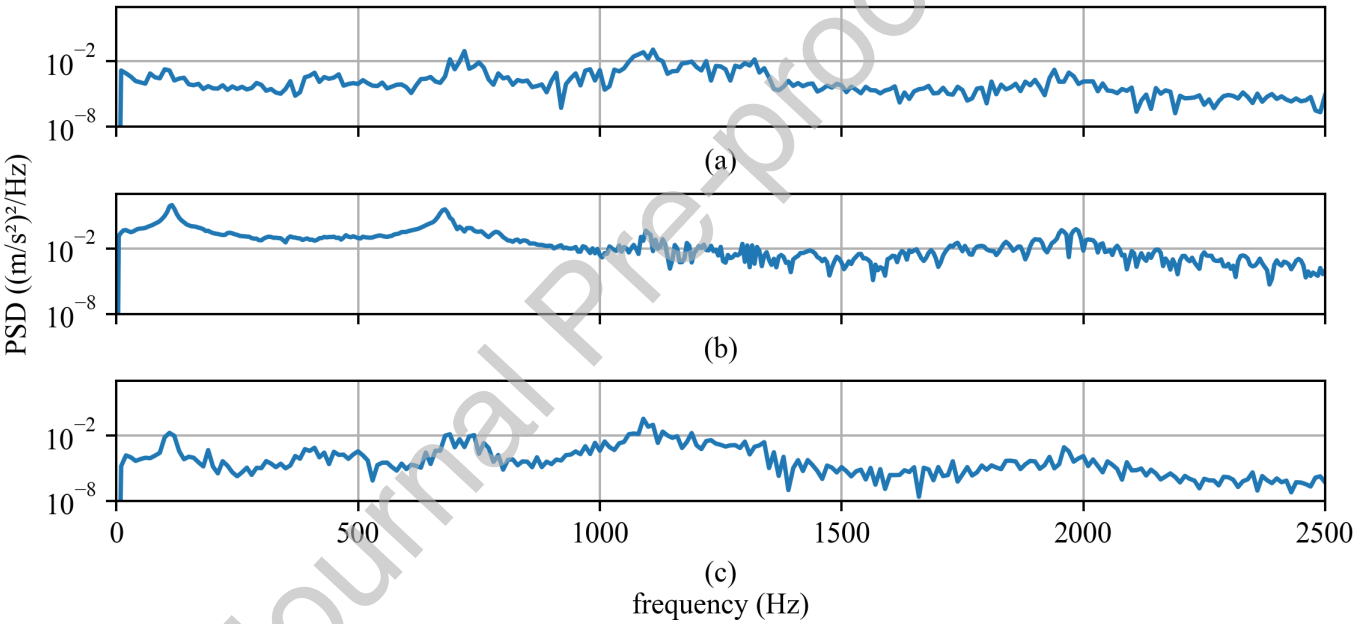


Figure 4: Power spectral density of separate regions of the acceleration data: (a) before the impact, (b) during the impact, and (c) after the impact.

Figure 4 shows the power spectral density (PSD) for different regions of the signal. Figure 4(a) shows the PSD for the region before the impact. Figure 4(b) shows the region encompassing the impact and initial response. Figure 4(c) shows the region after the impact towards the end of the experiment. The subplots show that there were three major modes for the structure at about 200, 750, and 1,900 Hz. The lower two modes are highly excited by the impact, as shown in Figure 4(b). The third mode shows some excitation, but the level is low compared to the initial energy level shown in Figure 4(a). A comparison of Figure 4(a) and (c) indicates that the modal response had returned to its baseline excitation level, suggesting that the test structure was not permanently damaged.

Journal Pre-proof

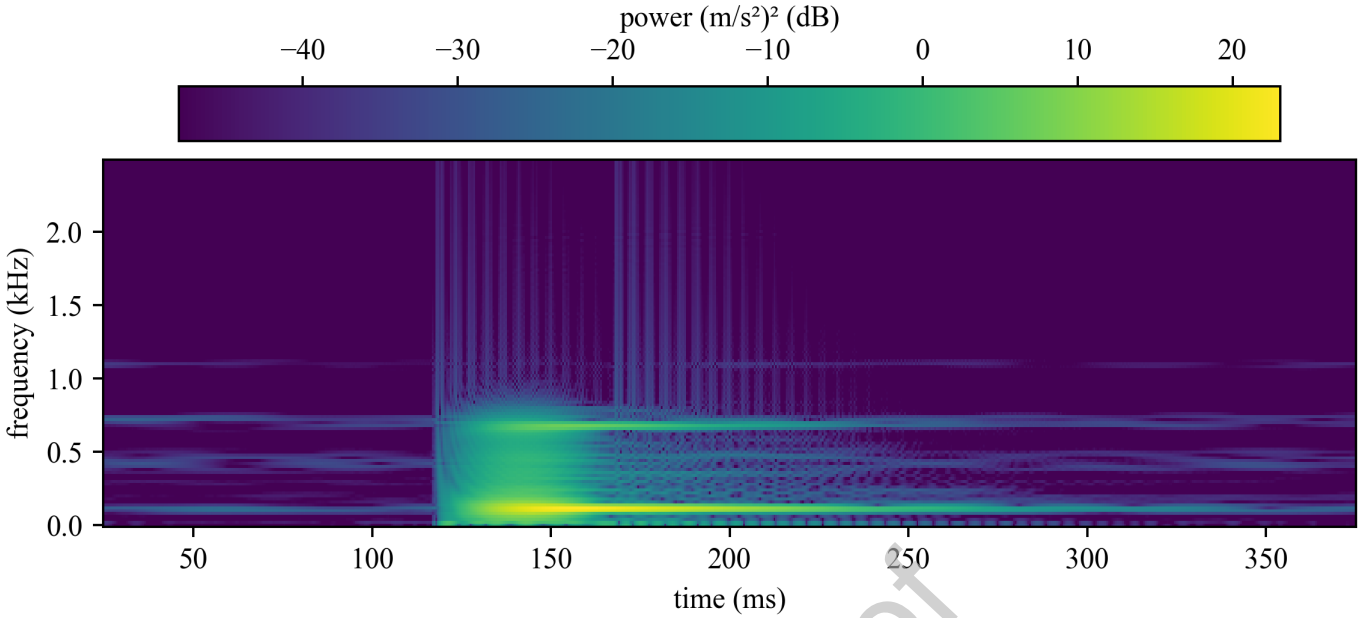


Figure 5: Spectrogram of lower frequencies of acceleration data in kHz.

Figure 5 presents the spectrogram of the signal shown in Figure 3. The spectrogram shows the initial excitation caused by the vibration actuator. The impact led to high excitation of two of the modes of the structure. The energy dissipates over time, returning to the steady excitation. The impact causes modal excitation without permanently affecting the modes of the system.

2.4. Ground Truth Establishment

A ground-truth reference was required to compare the performance of the online detection methods. An offline detection method was used to establish this reference because offline approaches can process the complete signal and the experiment contained a single shock event. The method was applied to the absolute value of the signal to identify two breakpoints. The absolute-value representation was selected because it improved the offline method's boundary detection. The online methods likewise required different input transformations and formats.

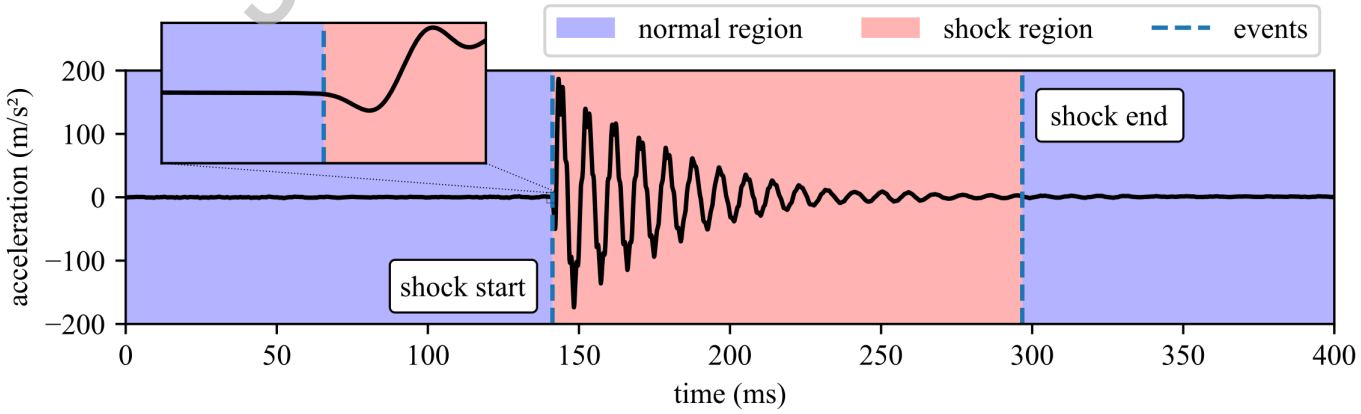


Figure 6: Ground-truth shock and non-shock regions.

Journal Pre-proof

Journal Pre-proof

The shaded regions in Figure 6 show the ground truth. The start of the shock was defined as the first point at which the signal envelope exceeded the baseline level of data known to be safe. The envelope was calculated as the maximum of absolute values for a window of recent data ($\max(|x|)$). For this study, the shock start was marked by the first time that the envelope for a window equaled or exceeded twice the value of the average envelope for data known to be safe. The end of the shock event was defined as the last observation that was greater than or equal to some percentage of the peak of the shock event. For this signal, the end of the shock event is marked as the last point that was greater than the baseline envelope value. The calculated start time of the shock was 141.242 ms, and the stop time for the shock event was 296.730 ms. The calculated times were used to define the normal and abnormal regions. The regions before and after the shock event were considered normal, whereas the shock interval was considered abnormal. These regions were used as a baseline to evaluate the aforementioned change point algorithms.

Because every reported metric is computed relative to this offline reference, the sensitivity of the evaluation to the choice of reference must be quantified. Three additional offline CPD methods from the ruptures library [46] were therefore applied to the same signal: binary segmentation, bottom-up segmentation, and dynamic programming, each with a rank cost function and two breakpoints. To make the dynamic-programming search tractable at 400,000 samples, the absolute signal was max-pooled by a factor of 100 before segmentation and the resulting breakpoints were mapped back to the original sample indices; max-pooling preserves the signal envelope on which the shock boundaries are defined. Table 2 lists the boundaries produced by all four methods: the estimated shock onset agrees to within approximately 4 ms and the shock end to within approximately 32 ms. Table 3 reports each online algorithm's metrics as the mean $\pm$ standard deviation across the four references. The absolute scores shift moderately, chiefly because the segmentation-based references place the shock end earlier and thereby shorten the abnormal region, but the ranking of the five methods by F1 score (BOCPD > Isolation Forest > GM > CUSUM > EM) is identical under every reference. The F1 ranking is unchanged across these tested reference definitions, although absolute scores vary; none constitutes instrument-independent physical ground truth. The envelope-based reference is retained for the remainder of the paper. The same invariance holds on each of Tests 2–5 (Section 3.5).

Table 2
Shock boundaries estimated by four alternative offline reference definitions from the same acceleration record (Test 1).

method	shock start (ms)	shock end (ms)
envelope (Sec. 2.4)	141.242	296.730
binary segmentation	137.500	269.500
bottom-up segmentation	141.500	265.000
dynamic programming	137.500	269.500

The delay standard deviations are dominated by the $\approx$4 ms offset in the onset estimated by the segmentation-based references; under any single reference the relative ordering of the delays is preserved.

Table 3
Online algorithm metrics as mean ± standard deviation across the four offline reference definitions of Table 2 (EM and Isolation Forest: mean of three runs).

algorithm	precision	recall	F1 score	detection delay (ms)
BOCPD	1.000 ± 0.000	0.496 ± 0.047	0.662 ± 0.043	2.08 ± 2.24
CUSUM	1.000 ± 0.000	0.287 ± 0.027	0.446 ± 0.033	2.38 ± 2.24
EM	1.000 ± 0.000	0.180 ± 0.017	0.305 ± 0.025	3.14 ± 2.24
GM	0.977 ± 0.001	0.336 ± 0.032	0.499 ± 0.035	1.71 ± 1.86
Isolation Forest	1.000 ± 0.000	0.398 ± 0.038	0.569 ± 0.039	2.23 ± 2.24

The offline reference and online predictions are derived from the same acceleration record, so an independent physical ground truth is unavailable. As an additional, reference-light check, we scored only samples more than 5 ms before the earliest estimated shock onset (normal) and the 10 ms following the latest estimated onset (shock). Table 4 reports results for all five tests. BOCPD, CUSUM, EM, and Isolation Forest produced no false alarms in the selected pre-impact regions, whereas GM flagged 0.6–5.7% of these samples. By hit rate minus false-alarm rate, BOCPD > Isolation Forest > CUSUM > EM on every test for the 10 ms window; shortening that window to 5 ms reverses the CUSUM/EM order on Tests 2–5. This check reduces dependence on the estimated shock-end boundary but does not remove all dependence on the acceleration-based onset estimates. The absence of independent impact timing remains a limitation (Section 3.7).

Table 4
Reference-light scores on separated pre-impact and early post-onset samples: false-alarm rate before impact and hit rate in the first 10 ms after estimated onset (in percent; EM and Isolation Forest: mean of three runs).

metric	algorithm	test 1	test 2	test 3	test 4	test 5
false-alarm rate (%)	BOCPD, CUSUM, EM, IF	0.0	0.0	0.0	0.0	0.0
	GM	0.6	0.6	1.9	3.5	5.7
hit rate (%)	BOCPD	97.2	92.1	89.5	87.5	88.1
	CUSUM	80.5	71.9	73.0	72.6	72.5
	EM	41.1	38.0	40.7	40.3	40.1
	GM	89.6	83.5	94.9	56.8	61.4
	Isolation Forest	85.9	81.5	77.2	76.6	76.6

Journal Pre-proof

3. Performance Analysis

This section first presents the per-algorithm results on Test 1 (Section 3.1) and their quantitative comparison (Section 3.2). It then examines noise robustness (Section 3.3), parameter sensitivity (Section 3.4), and consistency across five experimental runs (Section 3.5), followed by computational cost and runtime (Section 3.6) and practical deployment limitations (Section 3.7).

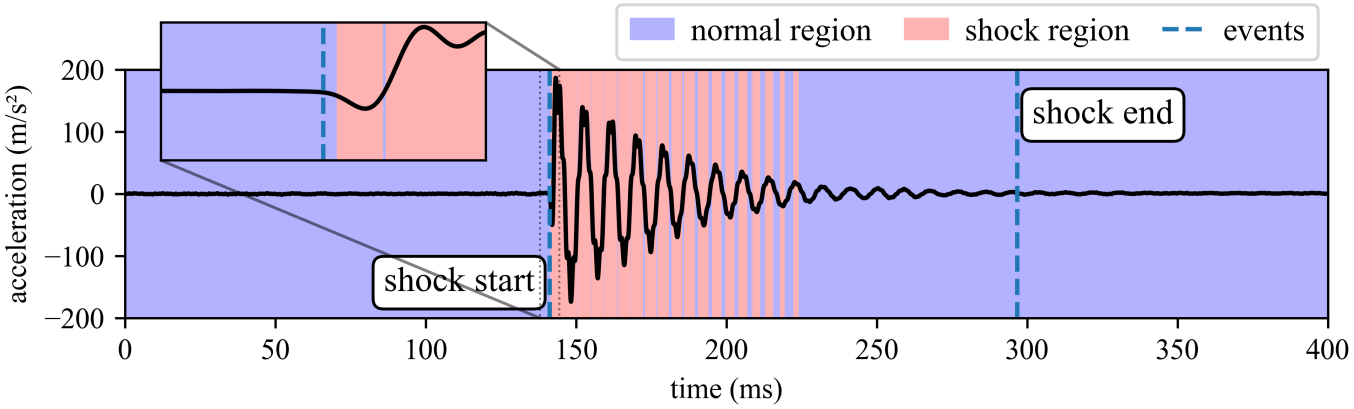


Figure 7: Detected regions from BOCPD.

3.1. Per-Algorithm Results

Figure 7 shows the predicted regions for BOCPD. The algorithm correctly identifies most of the high-magnitude sections of the shock region with minimal false positives. The shock is detected slightly late, and detection terminates prematurely. Late detection is a known limitation of online CPD methods, such as BOCPD, which require accumulated evidence before confirming a change point [47].

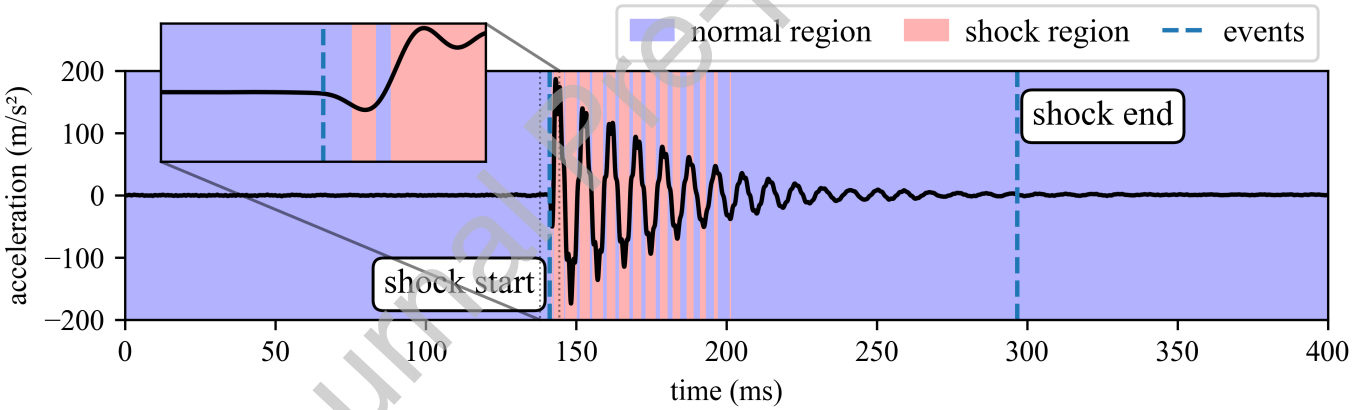


Figure 8: Detected regions from the CUSUM algorithm.

The predictions for CUSUM are shown in Figure 8. The CUSUM algorithm produced few false positives before and after the shock. The CUSUM algorithm identified the region of shock well, with some dispersed false negatives among the predictions. The algorithm also reported anomalies well into the later section of the shock region. The CUSUM algorithm is sensitive to shifts in the mean level, so that sensitivity is the likely source of the behavior of the algorithm with the lower magnitude acceleration.

Figure 9 shows the detected regions for the EM algorithm. The EM algorithm correctly identifies the high-amplitude sections of the shock region, but fails to recognize the adjacent, low-amplitude regions as part of the same region. The EM algorithm compares predicted Gaussian distributions to determine the correct classification. Because the algorithm

Journal Pre-proof

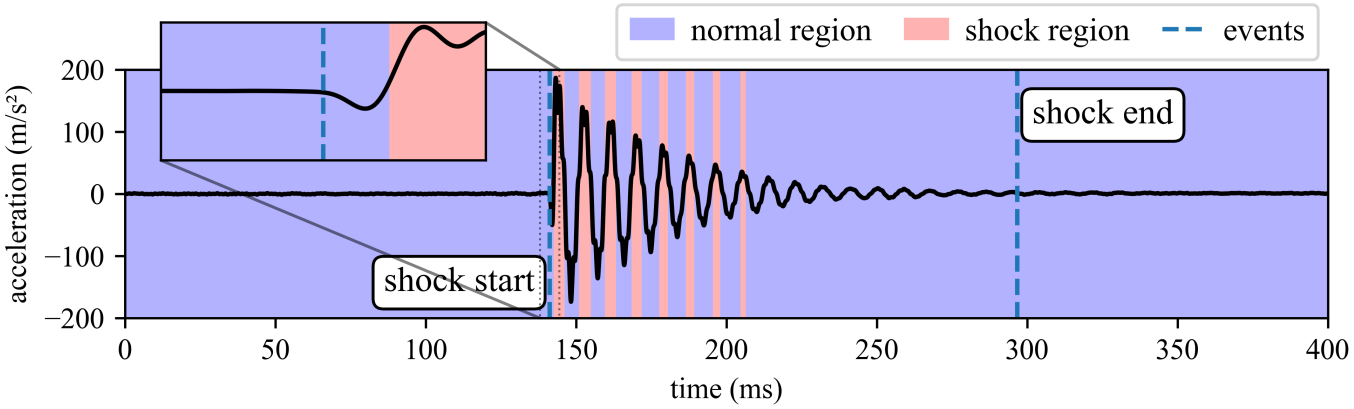


Figure 9: Detected regions from EM algorithm (representative run).

uses raw acceleration values for classification, each prediction depends primarily on the current value and does not incorporate neighboring observations.

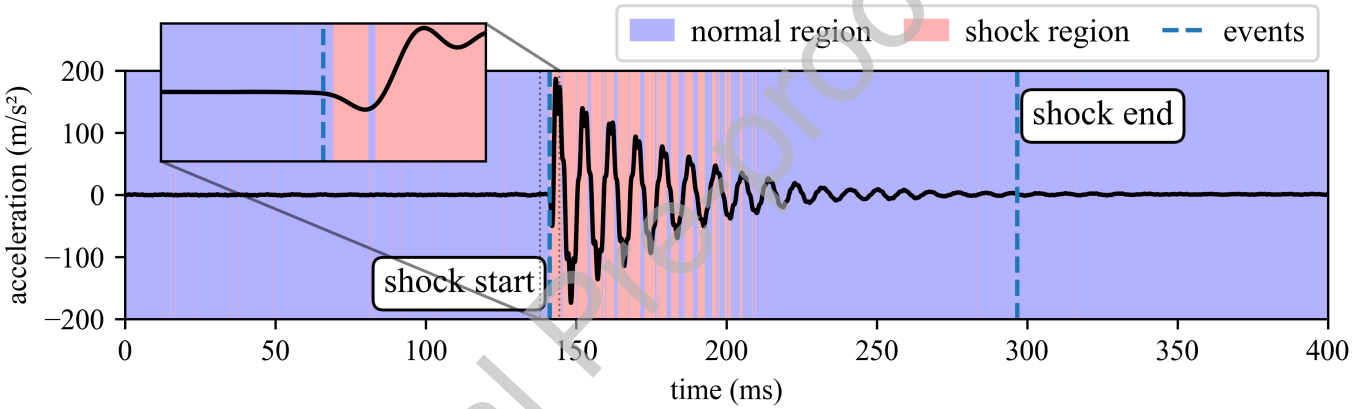


Figure 10: Detection regions from GM.

Figure 10 shows how the GM performed. The model predicted shock at about the same time as the other models. Once the shock occurred, there was little time during the shock event during which the model stopped detecting the shock event. This model continued to predict that a shock was in effect nearly 100 ms later. The model is likely responding to differences in amplitude and shape.

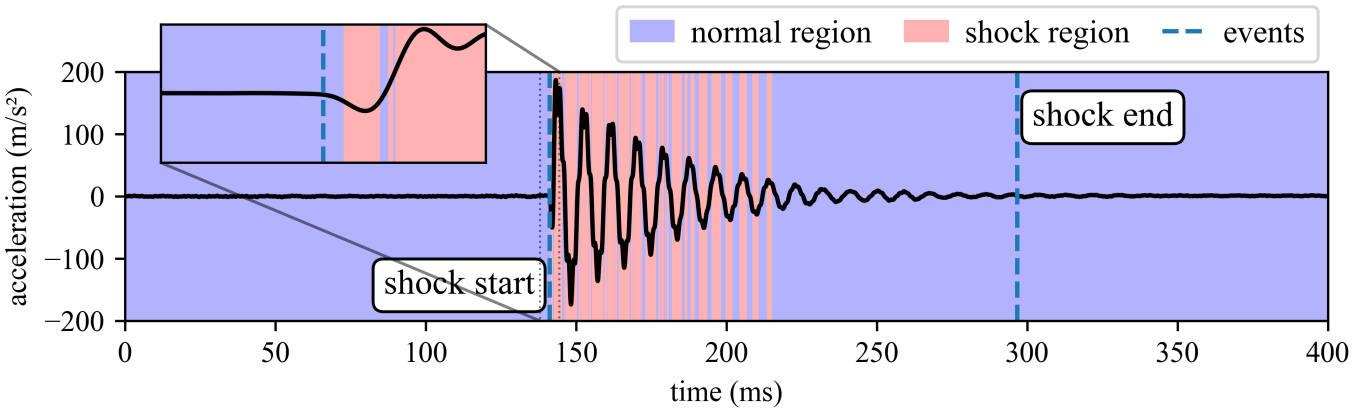


Figure 11: Detection regions from the Isolation Forest model (representative run).

Journal Pre-proof

Figure 11 shows the detection performance of the Isolation Forest model. The method identifies abnormal data effectively and performs comparably to BOCPD and CUSUM. Its measured runtime is closer to that of the lightweight statistical methods than to BOCPD. CUSUM requires only the mean and variance of the normal data, assuming that the process remains statistically consistent over time, while BOCPD can also operate with comparatively weak prior information.

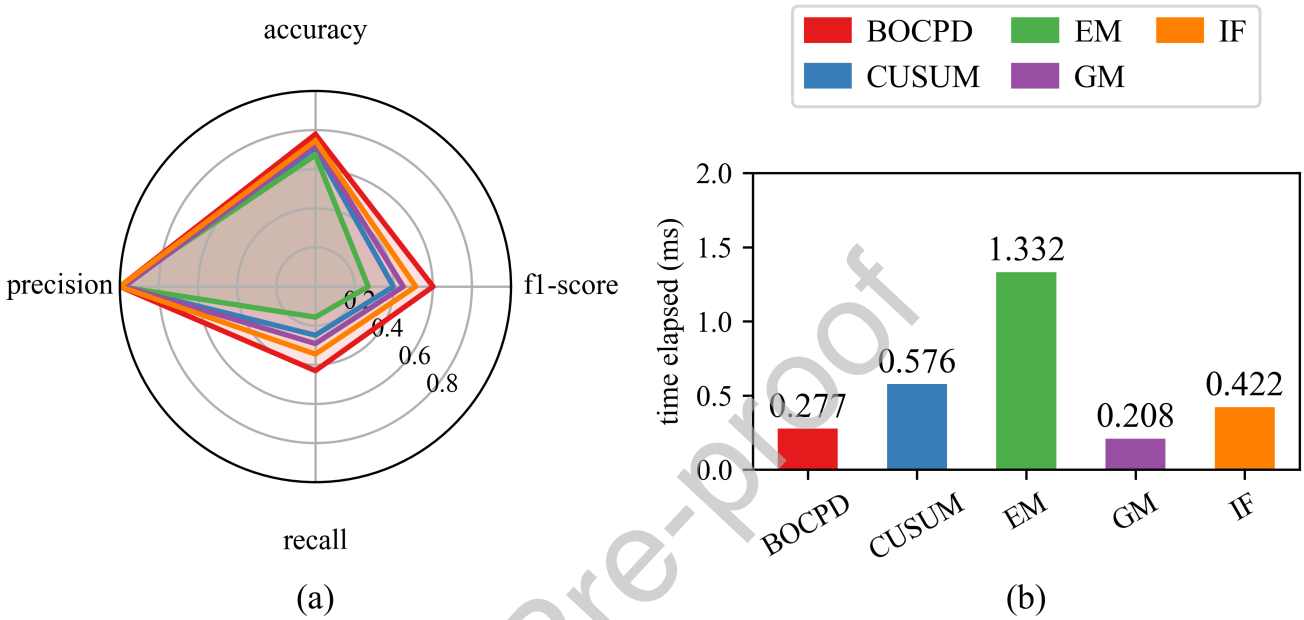


Figure 12: Radar and bar chart for metrics of algorithms on signal 1. (a) shows a radar chart comparing accuracy, precision, recall, and F1 score. (b) shows a bar chart comparing the detection delay between the start of the shock event and the first detection of shock after shock onset.

Journal Pre-proof

3.2. Algorithm Metric Comparison

The radar chart in Figure 12(a) illustrates the metric scores collected for the online change point detection algorithms. BOCPD, GM, and Isolation Forest have good F1 scores, with BOCPD having the best of the set. The strong standing of BOCPD is consistent with the benchmark study of van den Burg and Williams, which found BOCPD to be among the best-performing methods across 37 annotated real-world series [27]. This suggests that the three models achieved a comparatively good balance between detecting normal and abnormal states. All of the models had high precision values. This implies that the models were conservative in their detection of abnormal events. BOCPD, GM, and Isolation Forest had the shortest delay times between the start of the shock event and their first reported abnormal detection.

Table 5 summarizes the metrics collected for each detection method. These metrics provide complementary perspectives on performance by quantifying both how accurately each method identifies the shock state and how quickly it first detects the event. We include a metric called detection delay, which defines the time difference between

Table 5
Metrics for online change point detection algorithms on Test 1. EM and Isolation Forest values are the mean of three runs with recorded random seeds (Section 2.3).

Algorithm	Accuracy	Precision	Recall	F1 score	Detection Delay (ms)
BOCPD	0.778	1.000	0.430	0.602	0.277
CUSUM	0.708	1.000	0.249	0.399	0.576
EM	0.672	1.000	0.156	0.270	1.332
GM	0.722	0.978	0.292	0.449	0.208
Isolation Forest	0.746	1.000	0.345	0.513	0.422

the starting time defined by the ground truth and the first true positive detection. This metric is identical to the "time-elapsed" illustrated in Figure 12(b). This value represents the time delay between first encountering the shock event and detection. A small delay is only meaningful in combination with high precision: an algorithm that raises frequent false alarms trivially records a near-zero delay, since some alarm is almost surely active at the true onset. This caveat becomes relevant for GM under the noise conditions examined in Section 3.3.

3.3. Noise Sensitivity

Laboratory signals are substantially cleaner than measurements obtained on large full-scale structures, so the influence of measurement noise on the benchmark conclusions must be quantified. Zero-mean white Gaussian noise was added to the Test 1 signal at signal-to-noise ratios (SNRs) of 40, 30, 20, 10, 5, and 0 dB, where the SNR is referenced to the root-mean-square amplitude of the pre-shock steady-state response (0.434 m/s^2); at 0 dB the injected noise power therefore equals that of the monitored steady-state vibration itself. Five independent noise realizations were generated per level, all algorithms were re-run with unchanged hyperparameters (EM and Isolation Forest once per realization, each run with its own recorded seed, so that their spread combines noise and training-sample variability), and all metrics were evaluated against the ground truth of the clean signal, since the shock boundaries are a property of the event rather than of the measurement. Figure 13 presents the F1 score, precision, and detection delay as mean $\pm$ one standard deviation over the realizations.

Four of the five methods are remarkably insensitive to the injected noise. Across the entire range from 40 dB down to 0 dB, the mean F1 score of BOCPD varies only from 0.602 to 0.597, CUSUM is constant at 0.398–0.399, and EM (0.274–0.276) and Isolation Forest (0.512–0.516) remain within their run-to-run variability; the precision of all four is 1.000 at every level. The mechanism is straightforward: these methods employ fixed statistical thresholds calibrated far above the steady-state amplitude, while the shock response exceeds the steady state by more than two orders of magnitude, so even noise of steady-state magnitude does not bridge the margin. GM behaves differently: its precision degrades monotonically from 0.970 at 40 dB to 0.883, 0.718, 0.587, 0.561, and 0.539 at 30, 20, 10, 5, and 0 dB, respectively, accompanied by large realization-to-realization variability (F1 standard deviation up to 0.207 at 20 dB).

Journal Pre-proof

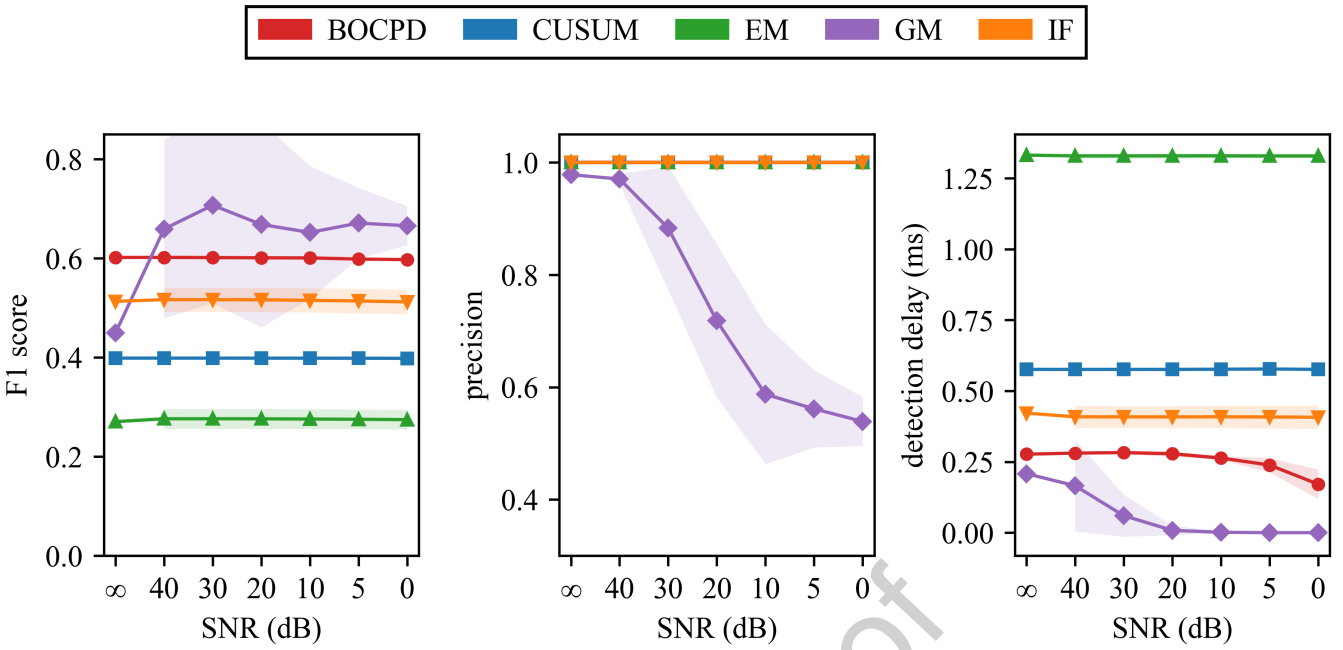


Figure 13: Noise sensitivity of the five methods on Test 1: F1 score, precision, and detection delay versus SNR, shown as mean ± one standard deviation over five noise realizations per level; the clean-signal values of Table 5 are included at the left of each axis (marked ∞) for reference. Noise severity increases to the right.

The F1 score of GM nevertheless remains nominally high across all levels, because the loss of precision is offset by a simultaneous inflation of recall (from 0.52 at 40 dB to 0.88 at 0 dB) as the detector fires increasingly indiscriminately; the precision panel of Figure 13 therefore carries the diagnostic information for this method. Even the mildest injected noise (40 dB, one percent of the steady-state amplitude) shifts GM away from its clean-signal operating point, raising its recall from 0.29 to 0.52, which explains the offset between its clean-signal F1 score of 0.449 and its nominal scores under noise. Because the grey-index statistic of GM responds to the local shape of short data windows, it reacts to the noise texture itself and increasingly raises false alarms as the SNR decreases; its apparent zero delay at low SNR is an artifact of this false-alarm behavior, as discussed in Section 3.2. The performance ordering of the remaining algorithms is unaffected by noise at every level examined. Section 3.5 shows that this behavior is reproduced on all five tests.

3.4. Parameter Sensitivity

Any benchmark of heterogeneous algorithms is potentially biased by implementation and parameter choices. To bound this effect, the principal detection parameter of each algorithm was swept on the Test 1 signal with all other settings unchanged: the BOCPD hazard timescale $\lambda \in \{2, 10, 100, 1000\}$, the CUSUM threshold $H \in \{1, 3, 5, 7, 9\}$ (in units of σ), the EM mixture prior $\pi \in \{0.5, \ldots, 0.9\}$, the GM window size $w \in \{4, 10, 50, 100, 500\}$, and the Isolation Forest contamination fraction $c \in \{0.01, 0.05, 0.1, 0.2, 0.3, 0.5\}$. Figure 14 summarizes the resulting F1 scores.

Four observations follow. First, the BOCPD configuration reported in the main benchmark is optimal within the swept range: $\lambda = 2$ attains F1 = 0.602, whereas $\lambda = 10$, 100, and 1000 attain 0.507, 0.513, and 0.535, respectively. Second, no parameter choice changes the standing of EM: for a given random draw of its training samples, EM yields

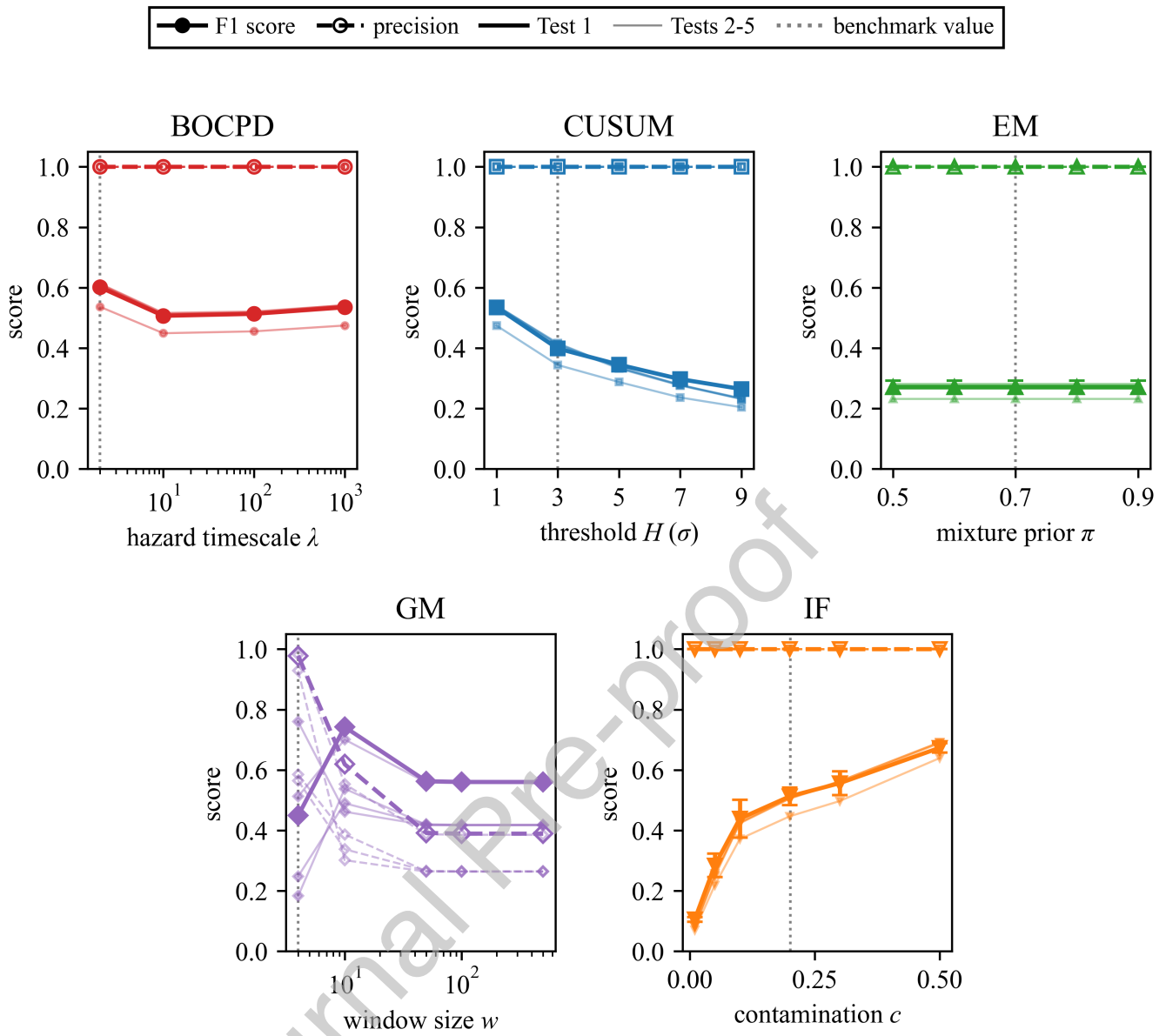


Figure 14: Sensitivity of the F1 score (solid, filled markers) and precision (dashed, open markers) to the principal detection parameter of each algorithm. Thick lines show Test 1, for which the error bars of EM and Isolation Forest give one standard deviation over three repeated runs per setting; thin lines show Tests 2–5 (EM and Isolation Forest: mean of three runs per setting).

Vertical dotted lines mark the values used in the main benchmark ($\lambda = 2$, $H = 3$, $\pi = 0.7$, $w = 4$, and $c = 0.201$).

identical scores at every swept value of π, because π only initializes the mixture weight, which the first EM update replaces with the weight estimated from the well-separated training samples. The F1 score of EM therefore varies only with the training draw (0.247–0.290 across the three seeds on Test 1, mean 0.270), indicating that the modest F1 of EM is a structural consequence of classifying raw acceleration values rather than an artifact of parameter selection. Third, the two remaining methods exhibit genuine trade-offs. For CUSUM, precision remains 1.000 across the entire sweep while the F1 score decreases monotonically from 0.535 at $H = 1$ to 0.264 at $H = 9$; the conventional setting $H = 3$ used in the main benchmark (F1 = 0.399) is therefore conservative, and a practitioner willing to tolerate a tighter threshold can obtain higher recall at no observed cost in precision on this data. For GM, the window $w = 4$

Journal Pre-proof

Journal Pre-proof

used in the main benchmark yields a balanced detector (precision 0.978), $w = 10$ trades precision for recall (0.619 and 0.927), yielding a higher F1 score of 0.742, but with a substantially higher false-alarm risk than at the nominal setting, and all windows $w \geq 50$ collapse to near-constant alarming (precision ≈ 0.39, recall ≈ 1.0), mirroring the fragility observed in the noise study. Fourth, Isolation Forest maintains unit precision across the entire contamination range; its mean F1 score over three runs per setting increases monotonically from 0.106 at $c = 0.01$ to 0.674 at $c = 0.5$, driven entirely by the fraction of the shock region that exceeds the anomaly-score threshold. The run-to-run variability from the random training samples and tree construction is small (standard deviation ≤ 0.040 at every setting except $c = 0.1$, where it reaches 0.063), and the swept value nearest the benchmark setting $c = 0.201$ attains F1 = 0.513, in the middle of the swept range, confirming that the algorithm was neither advantaged nor disadvantaged by this choice. Overall, the nominal-setting comparison is repeatable across experimental runs, but the cross-algorithm ranking is parameter dependent: GM and Isolation Forest can exceed the nominal BOCPD F1 score at other settings, with differing precision–recall trade-offs. The thin lines in Figure 14 show that each of these observations is reproduced on Tests 2–5 (Section 3.5).

3.5. Consistency Across Experimental Runs

To assess repeatability, the benchmark was applied to Tests 2–5 using the same preprocessing, envelope reference, parameters, and five algorithms as for Test 1. Across the five runs (Table 6), peak acceleration ranges from 134 to 187 m/s^2, pre-shock RMS vibration from 0.43 to 0.95 m/s^2, and shock duration from approximately 105 to 155 ms. EM and Isolation Forest results are averaged over three runs. For Tests 2–5, noise robustness was evaluated at the same SNR levels using three realizations per level, with one EM and one Isolation Forest run per realization.

Table 6
Operating conditions of the five experimental runs and spread of the four alternative offline reference definitions of Section 2.4. The last column gives the largest standard deviation of any algorithm's F1 score across the four references.

test	peak (m/s^2)	steady-state RMS (m/s^2)	peak/RMS	shock duration (ms)	onset spread (ms)	max F1 std
1	186.8	0.434	430	155.5	4.0	0.043
2	141.3	0.456	310	154.0	3.0	0.024
3	138.0	0.950	145	105.3	2.6	0.007
4	135.3	0.873	155	105.5	2.6	0.005
5	134.4	0.567	237	105.8	3.0	0.006

Table 7 and Figure 15(a,b) show that BOCPD achieved the highest F1 score on four of five tests (0.537–0.613), with detection delays of 0.28–0.37 ms. The nominal-setting order BOCPD > Isolation Forest > CUSUM > EM held on all tests, and these four methods maintained precision of 1.000. In contrast, GM varied substantially (F1: 0.183–0.760; precision: 0.564–0.981), ranking first on Test 3 and last on Tests 4 and 5.

The robustness analyses also show similar trends across runs. First, the four offline references locate shock onset within 4 ms on every test and preserve the per-test F1 ranking; the largest reference-dependent F1 standard deviation on

Table 7
Clean-signal metrics of the five algorithms on all five tests under the common protocol. EM and Isolation Forest values are the mean of three runs (run-to-run standard deviation of the F1 score at most 0.035). The Test 1 column repeats Table 5; Tables 9–12 in the Appendix give the same values for Tests 2–5 together with accuracy and recall.

metric	algorithm	test 1	test 2	test 3	test 4	test 5
F1 score	BOCPD	0.602	0.537	0.608	0.613	0.611
	CUSUM	0.399	0.344	0.410	0.416	0.413
	EM	0.270	0.232	0.277	0.282	0.277
	GM	0.449	0.512	0.760	0.183	0.248
	Isolation Forest	0.513	0.447	0.506	0.507	0.509
precision	BOCPD, CUSUM, EM, IF	1.000	1.000	1.000	1.000	1.000
	GM	0.978	0.981	0.929	0.585	0.564
detection delay (ms)	BOCPD	0.277	0.323	0.298	0.370	0.368
	CUSUM	0.576	1.273	1.083	1.141	1.138
	EM	1.332	1.236	1.043	1.099	1.096
	GM	0.208	0.603	0.133	0.792	0.801
	Isolation Forest	0.422	0.507	0.801	0.865	0.856

Tests 2–5 is 0.024. Second, under injected noise down to 0 dB (Figure 15(c,d)), BOCPD, CUSUM, EM, and Isolation Forest retain precision of at least 0.998, whereas GM precision decreases markedly on every test. Third, the parameter sweeps (Figure 14) identify $\lambda = 2$ as the best tested BOCPD setting on all five runs, show decreasing CUSUM F1 with increasing H, and reveal strong dependence of GM on its window length. Isolation Forest maintains unit precision but improves its F1 score as the contamination setting increases. These trends are repeatable across the available runs, although absolute scores and GM's ranking vary.

3.6. Computational Cost and Runtime

One metric not included in Table 5 is the time required for each algorithm to process the data. Runtime depends on algorithmic complexity, available computing resources, and implementation details, among other factors. To support a more general comparison, the theoretical time complexity of each method is also considered.

Table 8
Space and time complexity for different algorithms. n is sequence length, k is total size of samples, w is window size.

algorithm	time complexity	space complexity	time resources (s)	memory resources (MiB)
BOCPD	$O(n^2)\star$	$O(n)$	79.225	15.44
CUSUM	$O(n)$	$O(1)$	1.100	0.004917
EM	$O(kn)$	$O(k)$	6.394	10.73
GM	$O(wn)$	$O(w)$	1.860	3.887
Isolation Forest	$O(n \log_2 k)$	$O(k)$	23.213	33.62

Theoretical and experimental resource usage are shown in Table 8. The resource measurements were collected for 400,000 points of data. Figure 16 plots asymptotic running time versus sequence length n on log-log axes. The solid curves show BOCPD with $O(n^2)$ scaling and CUSUM with $O(n)$. Shaded bands depict parameter-dependent linear regimes: EM $O(kn)$ with $k \in [10^3, 10^5]$, GM $O(wn)$ with $w \in [10, 10^3]$, and Isolation Forest inference $O(n \log_2 k)$ with

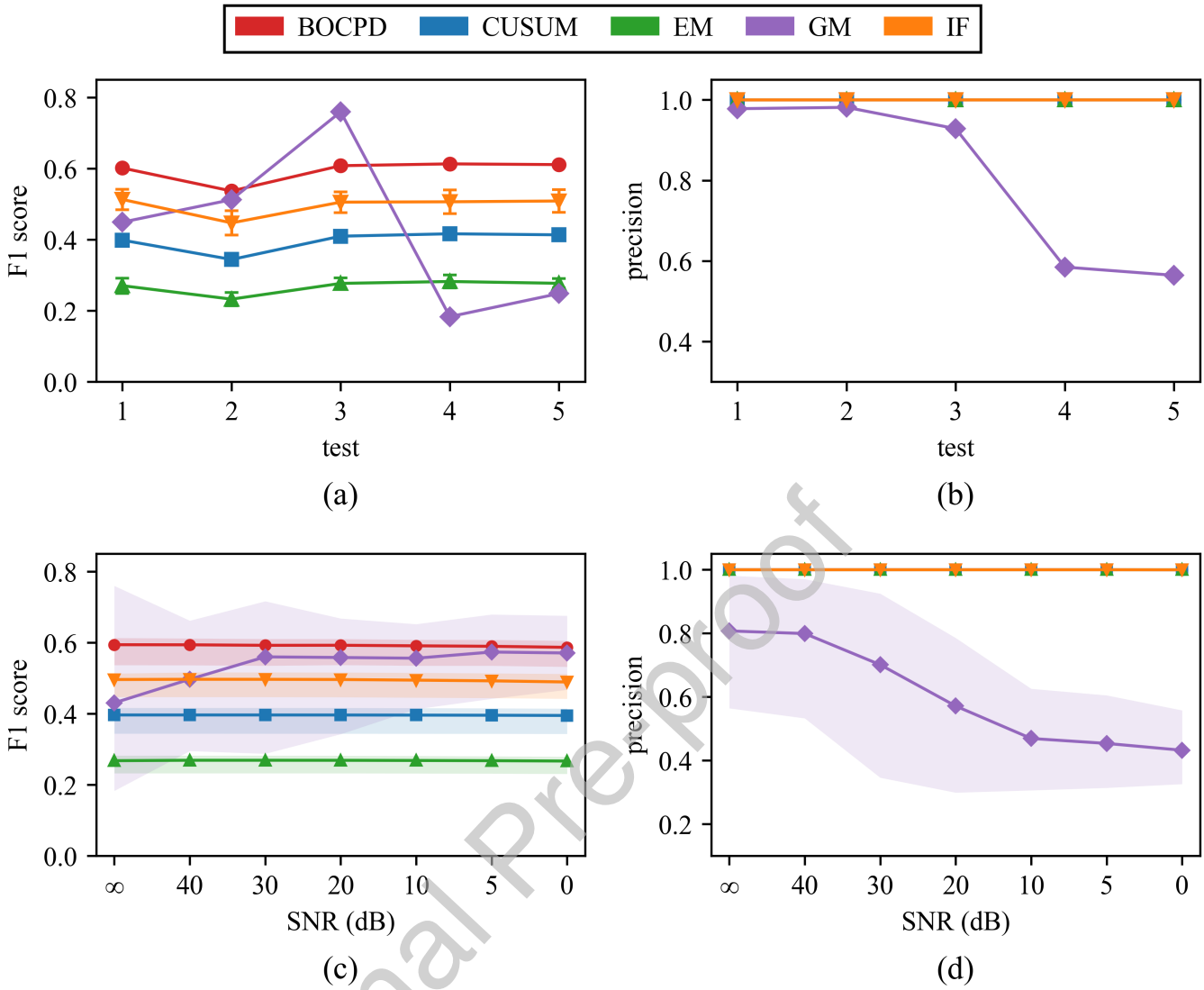


Figure 15: Results across the five experimental runs: (a) clean-signal F1 score and (b) precision for each test, with EM and Isolation Forest shown as mean ± one standard deviation over three runs; (c) F1 score and (d) precision versus SNR, where each line is the mean over Tests 1–5 of the per-test mean over noise realizations and the shaded band spans the minimum to maximum across tests. The clean-signal values are included at the left of the SNR axes (marked ∞).

Journal Pre-proof

$k \in [10^3, 10^5]$. Bandwidths indicate how constants grow with k and w, while slopes reflect the dominant dependence on n.

With fixed internal sample or window sizes, all methods except untruncated BOCPD scale linearly with sequence length. Each additional observation therefore adds approximately constant computational work, and the required memory can be bounded independently of the total sequence length. Untruncated BOCPD has higher computational complexity because the model retains additional run-length hypotheses as new observations arrive. However, the retained history can be periodically truncated so that the total number of model parameters remains below a prescribed upper bound [33], making bounded-memory online deployment feasible. The measured time and memory usage are broadly consistent with the theoretical complexity trends and provide a snapshot of the relative computational cost

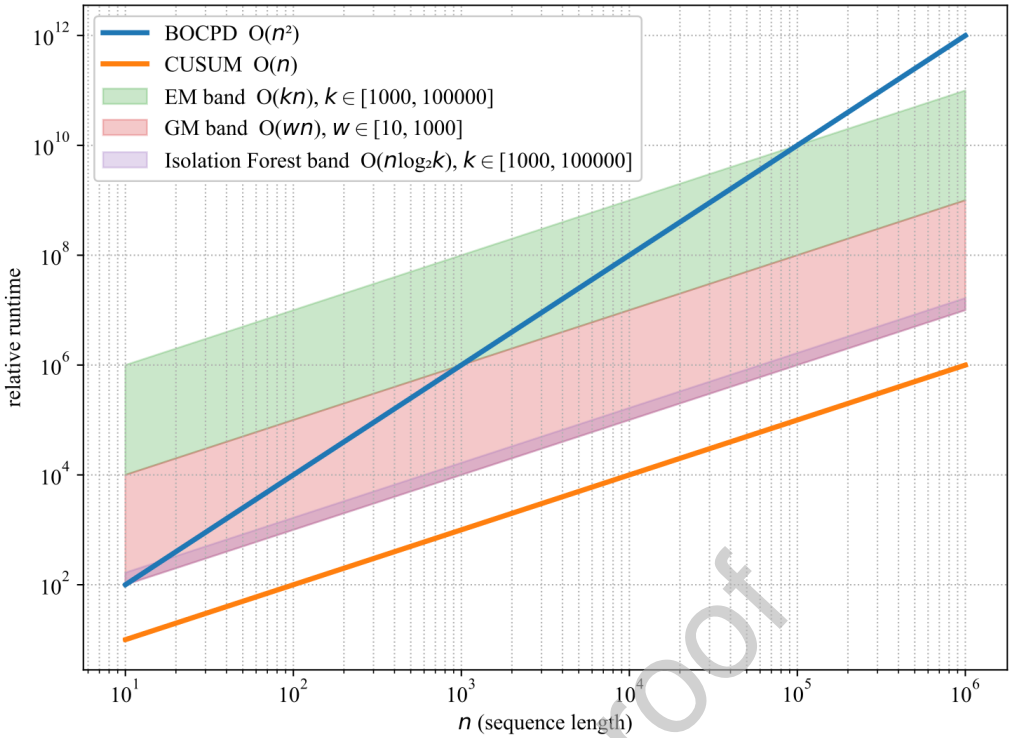


Figure 16: Scaling of computational cost with sequence length n on log–log axes. Here n is the sequence length, k is the total size of samples, w is the window size.

of the methods. Because the measurements were obtained from Python implementations on a single computer, they are most appropriate for relative comparison rather than as absolute deployment benchmarks. These desktop Python measurements are not an embedded-hardware validation. The observed mean processing times are approximately 2.8, 4.7, 16, 58, and 198 µs per sample for CUSUM, GM, EM, Isolation Forest, and untruncated BOCPD, respectively, exceeding the 1 µs interval at 1 MS/s. Thus, none of these implementations was demonstrated to operate at the acquisition rate. Fixed-size versions of CUSUM, GM, EM, and Isolation Forest have bounded per-sample state; BOCPD requires run-length truncation. Embedded latency, memory, and energy measurements remain future work. Isolation Forest, which is trained offline and then used for online inference, has an inference time complexity of $O(n \log_2 k)$ and a training time complexity of $O(k \log_2 k)$.

3.7. Applicability and Limitations in Real Deployments

The preceding analyses delimit the conditions under which the benchmarked methods can be expected to transfer to field deployments. First, all five methods require a modest amount of trusted statistical information at start-up: initialization of the mean and variance of the normal response, or, for EM and Isolation Forest, small samples representative of the normal and abnormal states. In practice this can be provided by a short guarded acquisition period after installation. Second, threshold-type parameters must be selected without labeled shock data; the parameter study of Section 3.4 shows method-dependent sensitivity, including substantial F1 changes for GM and Isolation Forest; conservative CUSUM thresholds preserve high precision but reduce recall on these tests. Third, the noise study of

Section 3.3 shows that threshold-based methods retain their performance even when broadband noise reaches the amplitude of the monitored steady-state response, whereas window-shape-based methods such as GM require noise-aware retuning of their window and critical values before field use. Fourth, the complexity analysis of Table 8 shows that the per-sample cost and memory of all methods except untruncated BOCPD are bounded independently of the record length, a necessary condition for deployment on microcontroller-class hardware; validation on such hardware was not part of this study. Several limitations remain. The injected noise was white and Gaussian, whereas operational noise on full-scale structures is typically colored and non-stationary; extending the benchmark to measured field-noise models is identified as future work. All analyses were repeated on the five available runs of a single test article (Section 3.5); the dataset provides no instrument-independent measurement of the impact time, which Section 2.4 addresses by an evaluation on well-separated pre-impact and early-impact samples; and the experiments consider a single shock event superimposed on stationary forced vibration, so multi-event scenarios and slowly evolving damage remain outside the present scope.

4. CONCLUSION

This study benchmarked five online detection methods: BOCPD, CUSUM, EM-based Gaussian mixture classification, GM, and Isolation Forest, for structural shock detection using 1 MS/s acceleration data from drop-tower experiments. On the representative Test 1 signal, BOCPD provided the best overall balance of detection accuracy and responsiveness, with an F1 score of 0.602, precision of 1.000, and a detection delay of 0.277 ms. Isolation Forest achieved the second-highest F1 score (0.513) with a delay of 0.422 ms (means of three runs). GM produced the shortest clean-signal delay (0.208 ms), but its lower precision and reduced robustness under noise limit the value of that speed advantage. CUSUM and EM were conservative, maintaining high precision but exhibiting lower recall.

The robustness analyses show that the nominal-setting comparison is repeatable across five experimental runs and four alternative offline reference definitions, and is stable to the tested white-noise perturbations; parameter sweeps reveal changes in the cross-algorithm ranking. The F1-score ranking remained unchanged across the four tested reference definitions, which are all derived from the measured acceleration. BOCPD, CUSUM, EM, and Isolation Forest maintained near-unit precision (at least 0.998 across the five tests) under additive white Gaussian noise down to 0 dB relative to the steady-state response, whereas GM precision decreased from 0.970 at 40 dB to 0.539 at 0 dB. The parameter sweeps identified the nominal BOCPD setting as its best tested configuration, while showing that CUSUM recall decreases with stricter thresholds, EM results do not change with its mixture prior, and Isolation Forest F1 increases with the contamination fraction while retaining unit precision. GM is particularly sensitive to its window size; parameter-dependent improvements in F1 do not guarantee comparable false-alarm behavior. Together, these results provide a more robust basis for comparing the methods than a clean-signal benchmark alone. Across the five

Journal Pre-proof

runs, BOCPD led on four, while the nominal-setting order BOCPD > Isolation Forest > CUSUM > EM remained consistent; GM's rank varied.

Overall, the results support the use of lightweight online detection methods for structural health monitoring and indicate their algorithmic suitability for edge deployment, but they do not identify a universally optimal detector. Practical implementation still requires representative initialization data and application-specific parameter selection. Untruncated BOCPD incurs increasing computational cost, EM can miss lower-amplitude portions of a shock because it classifies raw acceleration values, and GM requires noise-aware tuning. The present study is also limited to one structural configuration and shock type; the injected noise is white and Gaussian, all five experimental runs were obtained on a single test article, and the computational measurements were made on a desktop computer rather than on embedded hardware. Future work should therefore evaluate measured colored and non-stationary field noise, multiple or repeated shock events, additional structures and loading conditions, and embedded implementations with bounded computational resources.

CRediT AUTHOR STATEMENT

Zhymir Thompson: Conceptualization, Methodology, Software, Formal analysis, Validation, Visualization, Investigation, Writing – original draft, Writing – review and editing. **Yanzhou Fu**: Formal analysis, Validation, Visualization, Writing – original draft, Writing – review and editing. **Justice Adjei Owusu**: Methodology, Formal analysis, Visualization, Writing – original draft, Writing – review and editing. **Joud N. Satme**: Investigation, Data curation, Validation, Writing – review and editing. **Austin R.J. Downey**: Conceptualization, Methodology, Formal analysis, Resources, Supervision, Project administration, Funding acquisition, Writing – original draft, Writing – review and editing. **Gurcan Comert**: Conceptualization, Methodology, Formal analysis, Supervision, Writing – review and editing. **Jason D. Bakos**: Methodology, Software, Supervision, Formal analysis, Writing – review and editing.

DATA AVAILABILITY STATEMENT

The data, code, and analysis artifacts supporting this study are publicly available. The experimental acceleration dataset, including the raw and filtered acceleration data for the five drop-tower experimental runs, is available in the public dataset repository [31]. The benchmarking code and paper artifacts used to reproduce the analyses are available in the public paper repository [30]. The reusable online change-point algorithm implementation is available in the public software repository [32].

Journal Pre-proof

ACKNOWLEDGMENTS

We thank Dr. Jacob Dodson and Dr. Adriane Moura for early conversations that shaped the problem framing and requirements for developing change point detection algorithms for systems subjected to shock events superimposed on vibration; their high-level guidance helped steer this work. This research was supported by the United States' National Science Foundation (NSF), USA, Grants Nos. NSF-2131080, CNS-2200457, CCF-2234920, CCF-1956071, CCF-2503055, CCF-2234921, CPS-2237696, and through the NSF ASEE Fellowship. Additional support is provided by the National Heart, Lung, and Blood Institute of the National Institutes of Health under award number R01HL171295. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the National Institutes of Health.

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] V. Giurgiutiu, Stress, vibration, and wave analysis in aerospace composites: SHM and NDE applications, Academic Press, 2022.

[2] S. Al-Subaihawi, J. Ricles, S. Quiel, T. Marullo, Development of multi-directional real-time hybrid simulation for tall buildings subject to multi-natural hazards, Engineering Structures 315 (2024) 118348. doi:https://doi.org/10.1016/j.engstruct.2024.118348.

[3] J. N. Satme, D. Coble, A. R. Downey, Online health monitoring of electronic components subjected to repeated high-energy shock, in: Smart Materials, Adaptive Structures and Intelligent Systems, Vol. 88322, American Society of Mechanical Engineers, 2024, p. V001T05A002. doi:https://doi.org/10.1115/SMASIS2024-139950.

[4] A. Downey, L. Micheli, Vibration mechanics: A practical introduction for mechanical, civil, and aerospace engineers (Mar. 2025). doi:10.31219/osf.io/k95dn_v1.

[5] J. Dodson, A. Downey, S. Laflamme, M. D. Todd, A. G. Moura, Y. Wang, Z. Mao, P. Avitabile, E. Blasch, High-Rate Structural Health Monitoring and Prognostics: An Overview, Springer International Publishing, 2021, pp. 213–217. doi:10.1007/978-3-030-76004-5_23.

[6] Z. R. Wani, M. Tantray, E. Noroozinejad Farsangi, N. Nikitas, M. Noori, B. Samali, T. Yang, A critical review on control strategies for structural vibration control, Annual Reviews in Control 54 (2022) 103–124. doi:10.1016/j.arcontrol.2022.09.002.

[7] D. Wang, H. Nishimura, T. Shimogo, Active control of shock by gain scheduling, Journal of Sound and Vibration 308 (3–5) (2007) 647–659. doi:10.1016/j.jsv.2007.03.080.

[8] L. Cao, S. Laflamme, Real-time variable multidelay controller for multihazard mitigation, Journal of engineering mechanics 144 (2) (2018) 04017174. doi:https://doi.org/10.1061/(ASCE)EM.1943-7889.0001406.

[9] B. K. Ray, R. S. Tsay, Bayesian methods for change-point detection in long-range dependent processes, Journal of Time Series Analysis 23 (6) (2002) 687–705. doi:https://doi.org/10.1111/1467-9892.00286.

Journal Pre-proof

[10] V. M. Muggeo, G. Adelfio, Efficient change point detection for genomic sequences of continuous measurements, Bioinformatics 27 (2) (2011) 161–166. doi:https://doi.org/10.1093/bioinformatics/btq647.

[11] L. Horváth, M. Hušková, Change-point detection in panel data, Journal of Time Series Analysis 33 (4) (2012) 631–648. doi:https://doi.org/10.1111/j.1467-9892.2012.00796.x.

[12] S. Liu, M. Yamada, N. Collier, M. Sugiyama, Change-point detection in time-series data by relative density-ratio estimation, Neural Networks 43 (2013) 72–83. doi:10.1016/j.neunet.2013.01.012.

[13] N. Itoh, J. Kurths, Change-point detection of climate time series by nonparametric method, in: Proceedings of the world congress on engineering and computer science, Vol. 1, Citeseer, 2010, pp. 445–448.

[14] R. P. Adams, D. J. C. MacKay, Bayesian online changepoint detection (Oct. 2007). arXiv:0710.3742, doi:10.48550/ARXIV.0710.3742.

[15] Y. Kawahara, T. Yairi, K. Machida, Change-point detection in time-series data based on subspace identification, in: Seventh IEEE International Conference on Data Mining (ICDM 2007), IEEE, 2007, pp. 559–564. doi:10.1109/ICDM.2007.78.

[16] Z. Shi, A. Chehade, A dual-lstm framework combining change point detection and remaining useful life prediction, Reliability Engineering & System Safety 205 (2021) 107257. doi:10.1016/j.ress.2020.107257.

[17] M. Nigro, S. N. Pakzad, S. Dorvash, Localized structural damage detection: a change point analysis, Computer-Aided Civil and Infrastructure Engineering 29 (6) (2014) 416–432. doi:https://doi.org/10.1111/mice.12059.

[18] J. Wu, Y. Chen, S. Zhou, X. Li, Online steady-state detection for process control using multiple change-point models and particle filters, IEEE Transactions on Automation Science and Engineering 13 (2) (2015) 688–700. doi:10.1109/TASE.2014.2378150.

[19] A. Sharma, J. Zhang, G. Spagnoli, Z. Huang, M. Lu, H. Yao, S. Wu, Fast semiautomatic change point detection in deformation monitoring using kullback–leibler divergence, ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering 11 (4) (2025) 04025057. doi:https://doi.org/10.1061/AJRUA6.RUENG-1625.

[20] X. Ge, P. Smyth, Segmental semi-markov models for change-point detection with applications to semiconductor manufacturing, Tech. rep., Citeseer (2000).

[21] Y. Liao, A. S. Kiremidjian, R. Rajagopal, C.-H. Loh, Structural damage detection and localization with unknown postdamage feature distribution using sequential change-point detection method, Journal of Aerospace Engineering 32 (2) (2019) 04018149.

[22] S. Mariani, P. Cawley, Change detection using the generalized likelihood ratio method to improve the sensitivity of guided wave structural health monitoring systems, Structural Health Monitoring 20 (6) (2021) 3201–3226. doi:https://doi.org/10.1177/1475921720981831.

[23] Y. Li, T. Bao, X. Shu, Z. Gao, J. Gong, K. Zhang, Data-driven crack behavior anomaly identification method for concrete dams in long-term service using offline and online change point detection, Journal of Civil Structural Health Monitoring 11 (2021) 1449–1460.

[24] X. Lei, Z. Chen, H. Li, S. Wei, A change-point detection method for detecting and locating the abrupt changes in distributions of damage-sensitive features of SHM data, with application to structural condition assessment, Structural Health Monitoring 22 (2) (2023) 1161–1179. doi:10.1177/14759217221101320.

[25] P. B. Dao, Bridging data and diagnostics: a systematic review and case study on integrating trend monitoring and change point detection for wind turbines, Energies 18 (19) (2025) 5166. doi:10.3390/en18195166.

[26] M. Altamirano, F.-X. Briol, J. Knoblauch, Robust and scalable Bayesian online changepoint detection, in: Proceedings of the 40th International Conference on Machine Learning, Vol. 202 of Proceedings of Machine Learning Research, PMLR, 2023, pp. 642–663.

[27] G. J. J. van den Burg, C. K. I. Williams, An evaluation of change point detection algorithms, arXiv preprint arXiv:2003.06222 (2020). doi:10.48550/arXiv.2003.06222.

Journal Pre-proof

[28] B. Joyce, J. Dodson, S. Laflamme, J. Hong, An experimental test bed for developing high-rate structural health monitoring methods, Shock and Vibration 2018 (2018) 1–10. doi:10.1155/2018/3827463.

[29] Z. Thompson, G. Comert, J. N. Satme, A. R. J. Downey, J. D. Bakos, Real-time shock event classification from univariate structural response measurements, in: ASME 2024 Conference on Smart Materials, Adaptive Structures and Intelligent Systems, SMASIS2024, American Society of Mechanical Engineers, 2024. doi:10.1115/smasis2024-141124.

[30] Z. Thompson, A. R. Downey, Paper-2026-online-change-point-algorithms (2026).
URL https://github.com/ARTS-Laboratory/Paper-2026-Online-Change-Point-Algorithms

[31] D. Goshorn, J. Satme, A. R. Downey, Dataset-7 forced vibration and shock.
URL https://github.com/High-Rate-SHM-Working-Group/Dataset-7-forced-vibration-and-shock

[32] Z. Thompson, A. R. Downey, online-changepoint-algorithms (2025).
URL https://github.com/ARTS-Laboratory/online-changepoint-algorithms

[33] R. P. Adams, D. J. C. MacKay, Bayesian online changepoint detection (2007). arXiv:0710.3742.
URL https://arxiv.org/abs/0710.3742

[34] Y. Saatci, R. Turner, C. E. Rasmussen, Gaussian process change point models, Vol. 1, Haifa, Israel, 2010.

[35] R. Turner, Bayesian change point detection for satellite fault prediction, Diverse Engagement: Drawing in the Margins (2010) 213.

[36] J. M. Bernardo, A. F. Smith, Bayesian theory, Vol. 405, John Wiley & Sons, 2009.

[37] M. I. Jordan, The conjugate prior for the normal distribution (2010).

[38] E. S. Page, Continuous inspection schemes, Biometrika 41 (1/2) (1954) 100–115. doi:10.1093/biomet/41.1-2.100.

[39] M. Basseville, I. V. Nikiforov, Detection of Abrupt Changes: Theory and Application, Prentice Hall, Englewood Cliffs, NJ, 1993.

[40] D. C. Montgomery, Statistical quality control, Vol. 7, Wiley New York, 2009.

[41] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of Royal Statistics Society, Series B 39 (1977) 1–37. doi:https://doi.org/10.1111/j.2517-6161.1977.tb01600.x.

[42] D. Ju-Long, Control problems of grey systems, Systems & Control Letters 1 (5) (1982) 288–294. doi:10.1016/S0167-6911(82)80025-X.

[43] J. L. Deng, Introduction to grey system theory, J. Grey Syst. 1 (1) (1989) 1–24.
URL http://dl.acm.org/citation.cfm?id=90757.90758

[44] S. Liu, Y. Lin, Grey information: theory and practical applications, Springer Science & Business Media, 2006.

[45] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: 2008 Eighth IEEE International Conference on Data Mining, IEEE, 2008, pp. 413–422. doi:10.1109/icdm.2008.17.

[46] C. Truong, L. Oudre, N. Vayatis, Selective review of offline change point detection methods, Signal Processing 167 (2020) 107299. doi:10.1016/j.sigpro.2019.107299.

[47] S. Aminikhanghahi, D. J. Cook, A survey of methods for time series change point detection, Knowledge and Information Systems 51 (2) (2016) 339–367. doi:10.1007/s10115-016-0987-z.

Journal Pre-proof

Appendix

The dataset used in this study contains five tests in total [31]. The main text analyzes Test 1 (Figures 7–11). The following subsections provide per-signal detection outputs and summary metrics for Tests 2–5 to illustrate performance across varying impact amplitudes and base excitations. All results use the protocol of the main text: the same preprocessing (first 100 ms removed), the envelope reference of Section 2.4, and the nominal parameters ($\lambda = 2$, $H = 3\sigma$, $\pi = 0.7$, $w = 4$, and $c = 0.201$). Tables 9–12 therefore contain the values of Table 7, extended with accuracy and recall. EM and Isolation Forest values are means of three runs; their figures show the run closest to that mean (Section 2.3).

A.1. Test 2 Results and Discussion

Figures 17-21illustrate the algorithms' detection ability on signal 2. Table 9 shows the metric scores calculated for the same run of data.

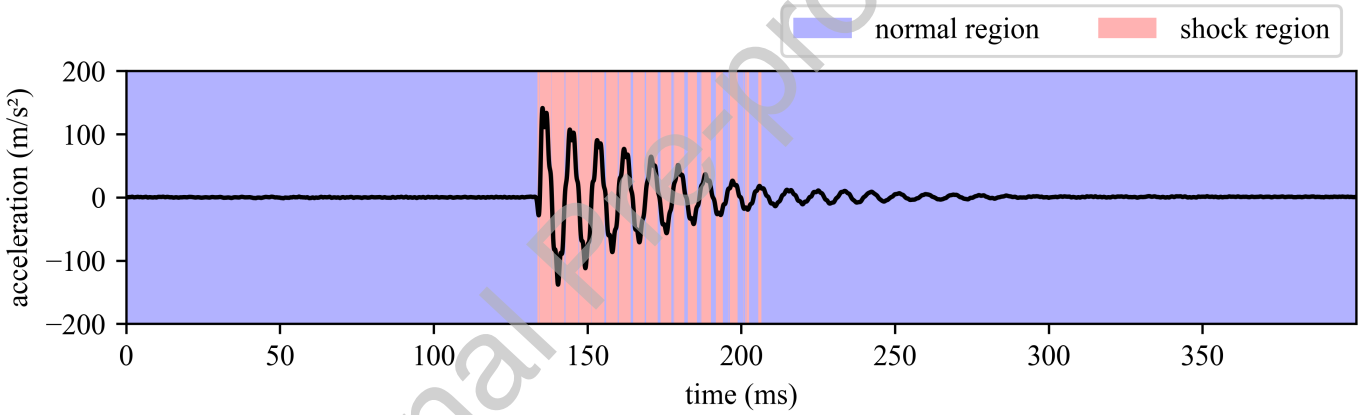


Figure 17: Detected regions from BOCPD algorithm.

The results in Figure 17 show strong detection performance and provide additional evidence that BOCPD remains consistent for a lower-amplitude signal.

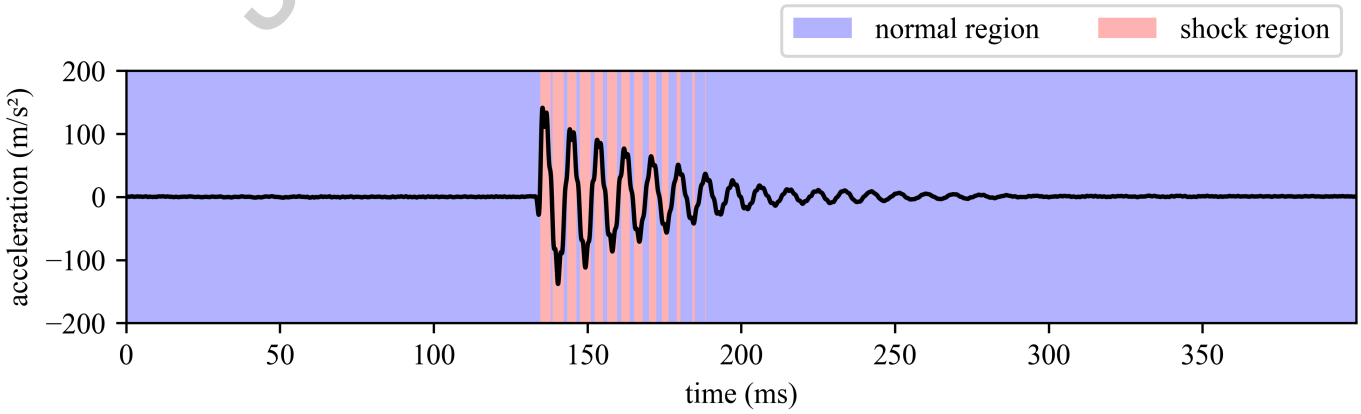


Figure 18: Detected regions from the CUSUM algorithm.

Journal Pre-proof

Journal Pre-proof

Table 9
Metrics for online change point detection algorithm performance on signal 2 (same protocol and values as Table 7; EM and Isolation Forest: mean of three runs).

algorithm	accuracy	precision	recall	f1 score	detection delay (ms)
BOCPD	0.756	1.000	0.367	0.537	0.323
CUSUM	0.695	1.000	0.208	0.344	1.273
EM	0.666	1.000	0.132	0.232	1.236
GM	0.746	0.981	0.347	0.512	0.603
Isolation Forest	0.726	1.000	0.288	0.447	0.507

Figure 18 shows the detections produced by CUSUM. The algorithm does not show a strong response to the signal after 200 ms.

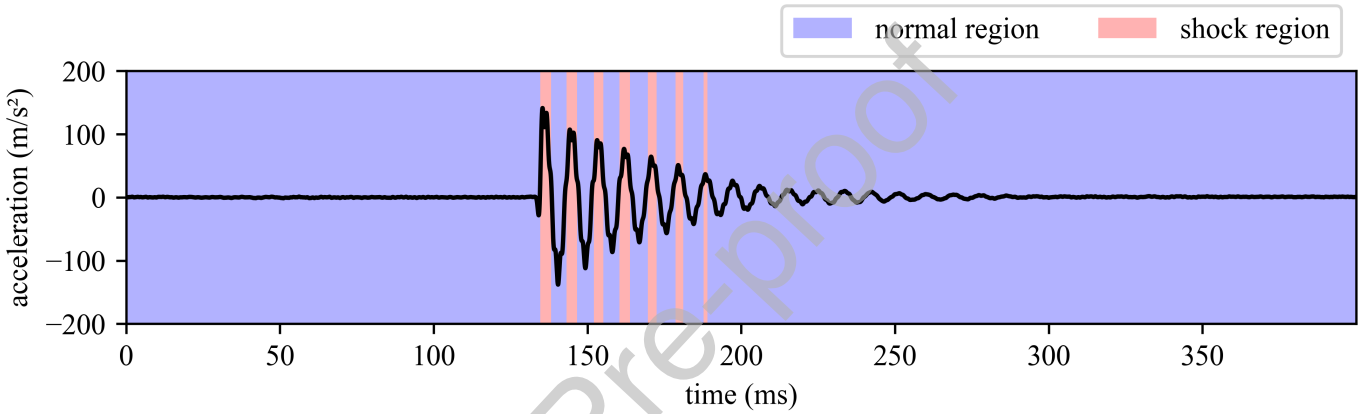


Figure 19: Detected regions from EM algorithm (representative run).

The detection regions in Figure 19 show that EM is still capturing high measurements exclusively.

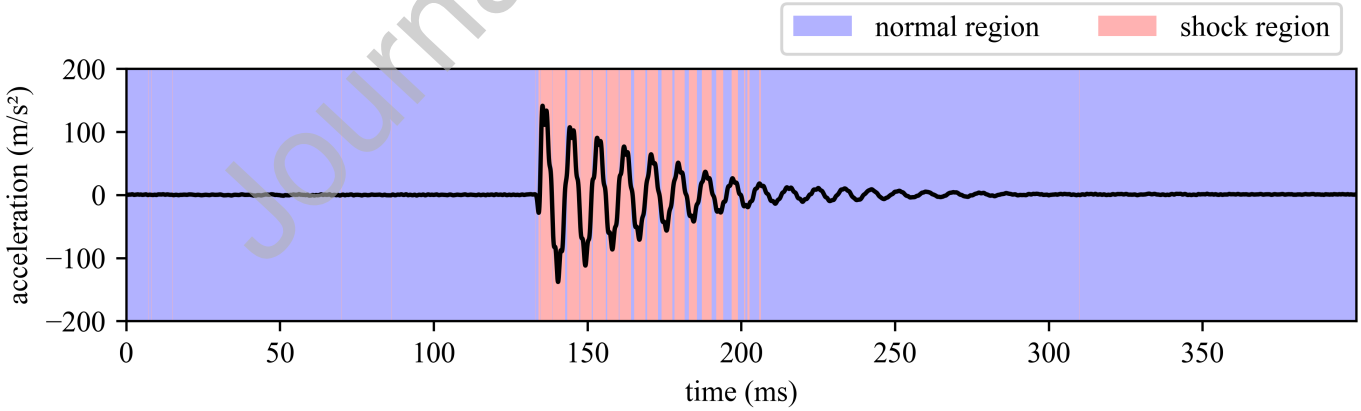


Figure 20: Detection regions from GM.

Figure 20 shows that GM detects the shock in this signal, as it did for Test 1, with a few isolated false alarms before and after the event.

Figure 21 shows the detections of Isolation Forest. Like CUSUM and EM, it flags the high-amplitude cycles of the response without false alarms and stops reporting shock as the oscillation decays.

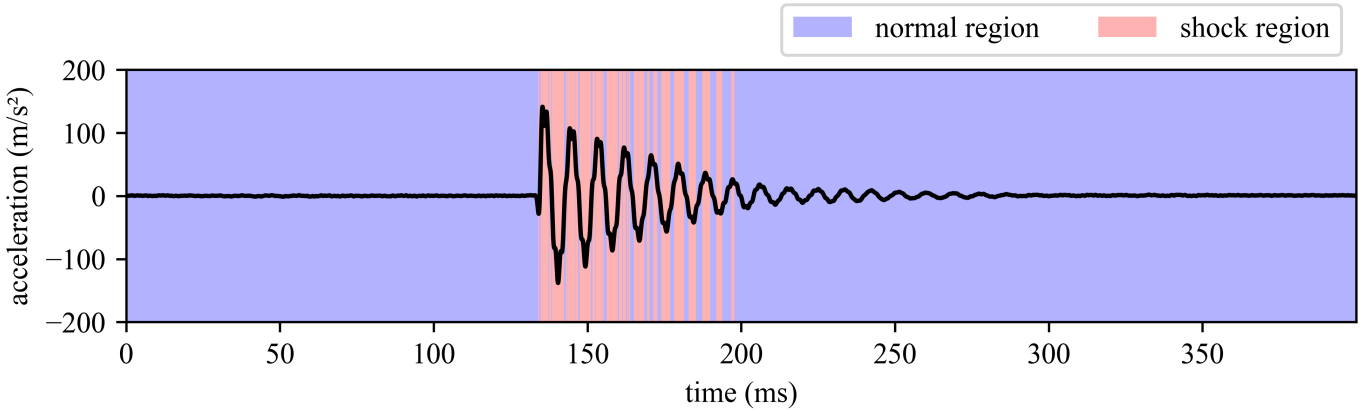


Figure 21: Detected regions from Isolation Forest (representative run).

Table 9 compares the performance of the different models on signal 2. BOCPD attained the highest F1 score (0.537), followed by GM (0.512) and Isolation Forest (0.447), with CUSUM (0.344) and EM (0.232) last. BOCPD, CUSUM, EM, and Isolation Forest kept a precision of 1.000, and GM reached a recall similar to that of BOCPD at a precision of 0.981. BOCPD detected the event 0.32 ms after the reference onset, Isolation Forest after 0.51 ms, GM after 0.60 ms, and CUSUM and EM after about 1.2–1.3 ms. Every method stops reporting the shock well before the reference end at 287 ms, which limits the recall of all five.

A.2. Test 3 Results and Discussion

Figures 22-26show the detection results for each algorithm on signal 3. Table 10 shows the metric scores for each algorithm for the same signal.

The detection regions in Figure 22 show that the model consistently identifies the shock region. The gaps between detections gradually widen as time from impact increases, suggesting that the decreasing response amplitude influences the model's decisions.

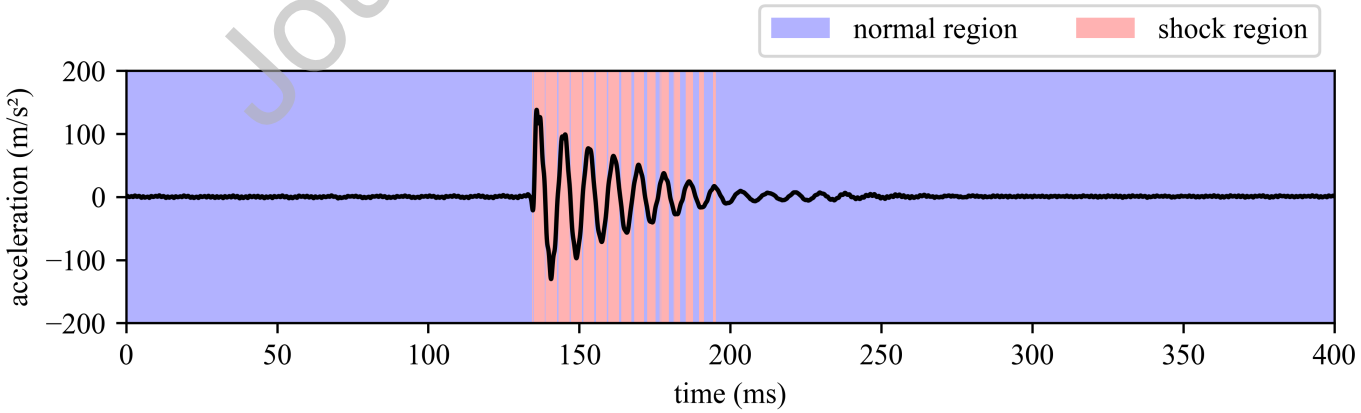


Figure 22: Detected regions from BOCPD algorithm.

Journal Pre-proof

Figure 23 shows the CUSUM performance on signal 3. The CUSUM model detected the initial impact well, but detections became sparse midway through the expected impact region. This behavior indicates that performance depends not only on the model structure but also on the selected averaging weight.

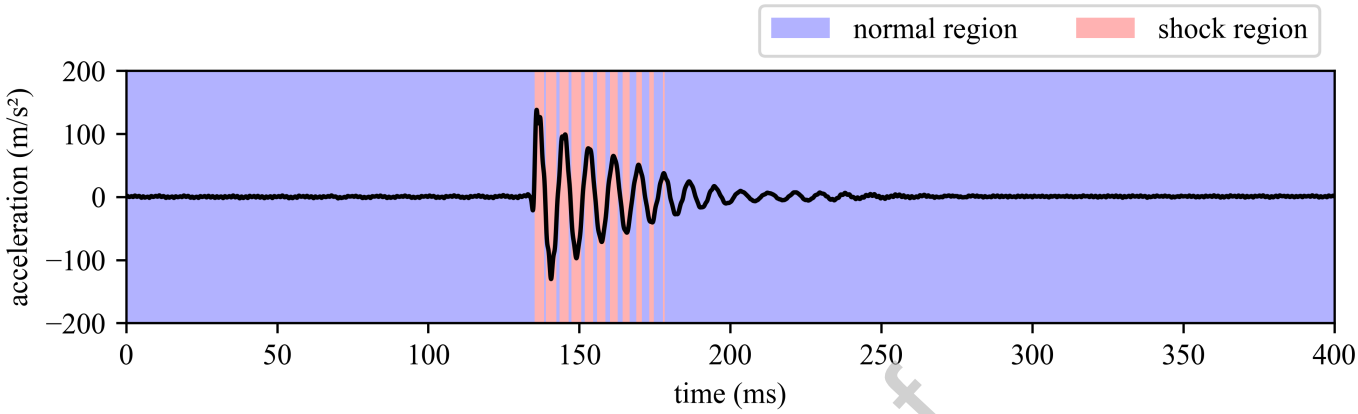


Figure 23: Detected regions from the CUSUM algorithm.

Figure 24 shows that the model still only captures the peaks. The model only captures values going beyond a threshold, so in the last peak, there is only the highest magnitude reported as shock.

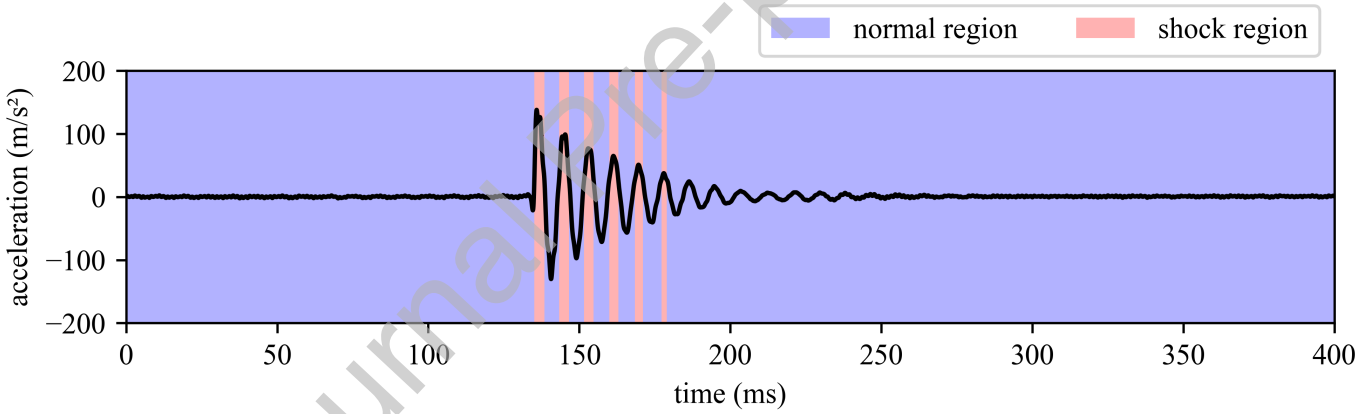


Figure 24: Detected regions from EM algorithm (representative run).

On signal 3, GM covers the shock region more densely than any other method (Figure 25), but it also raises scattered false alarms before and after the event.

Figure 26 shows that Isolation Forest flags the high-amplitude cycles without false alarms; as for BOCPD, the gaps between its detections widen as the response decays.

On signal 3, GM attained the highest F1 score (0.760) and the shortest delay (0.13 ms) because it covered most of the shock region (recall 0.643), at a precision of 0.929. BOCPD followed (F1 0.608, delay 0.30 ms), then Isolation Forest (0.506), CUSUM (0.410), and EM (0.277), all four with a precision of 1.000. Test 3 is the only run on which GM ranks first (Section 3.5).

Journal Pre-proof

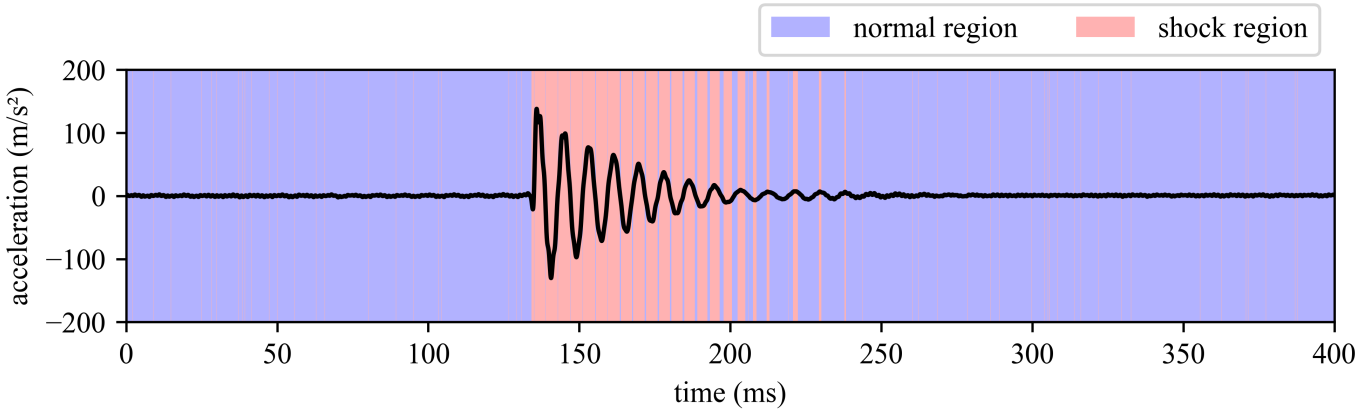


Figure 25: Detection regions from GM.

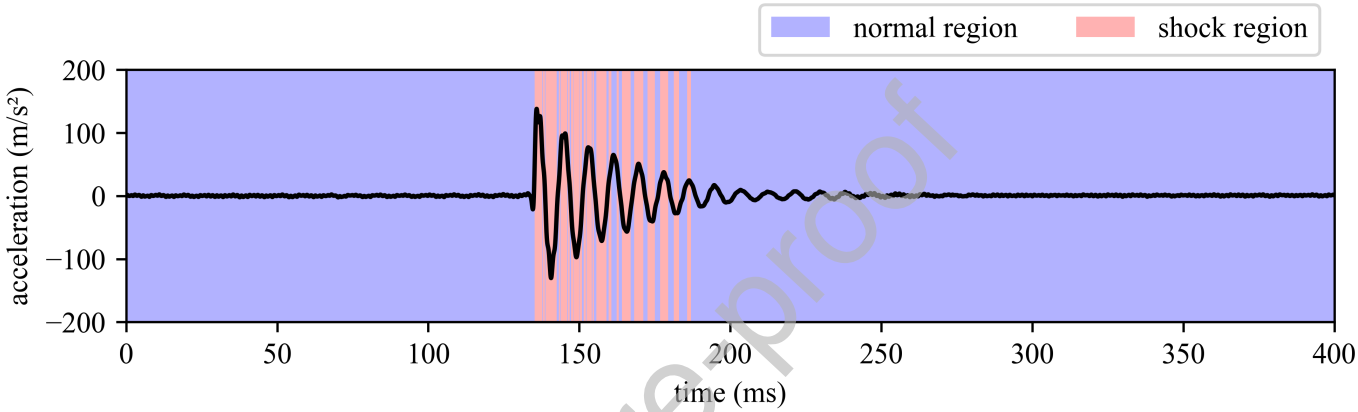


Figure 26: Detected regions from Isolation Forest (representative run).

Table 10
Metrics for online change point detection algorithm performance on signal 3 (same protocol and values as Table 7; EM and Isolation Forest: mean of three runs).

algorithm	accuracy	precision	recall	f1 score	detection delay (ms)
BOCPD	0.852	1.000	0.437	0.608	0.298
CUSUM	0.805	1.000	0.258	0.410	1.083
EM	0.779	1.000	0.161	0.277	1.043
GM	0.893	0.929	0.643	0.760	0.133
Isolation Forest	0.826	1.000	0.339	0.506	0.801

Journal Pre-proof

A.3. Test 4 Results and Discussion

Figures 27-31show the detected regions for Test 4. Table 11 shows the corresponding metrics for the algorithms.

Figure 27 shows the performance of BOCPD. The model correctly identifies the major part of the shock region. The model works despite the noise of the normal signal. This is further evidence of the robustness of the model. There are gaps between detected regions that appear to widen as the shock dissipates. The reason for this behavior is currently unknown. This could be caused by an interaction between the parameter resets and initial conditions.

The detected shock regions in Figure 28 are spread out. The detection is more concentrated around the initial impact. As the energy dissipates, the detection areas are fewer and farther apart before ultimately disappearing. The model performs well, but shows difficulty with capturing the end of the shock event.

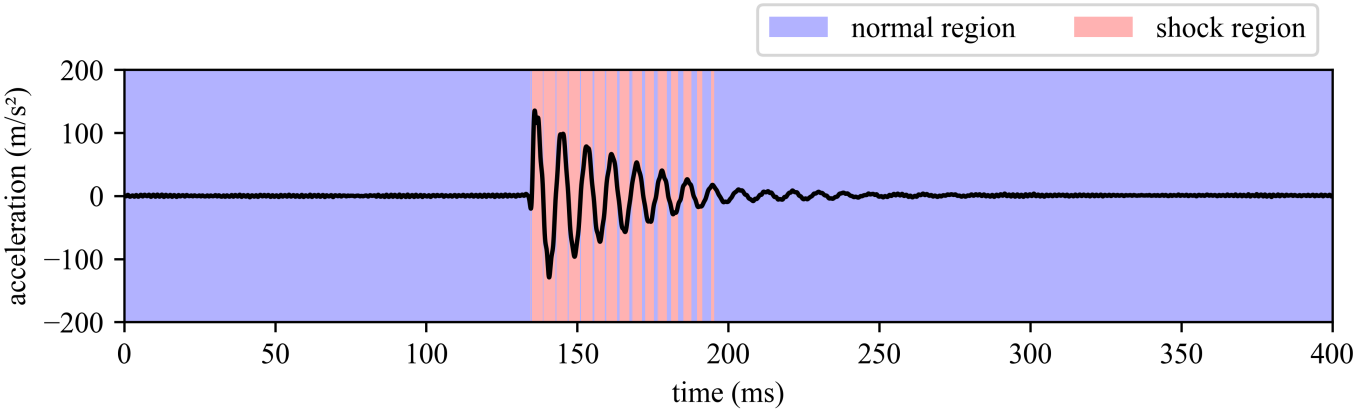


Figure 27: Detected regions from BOCPD.

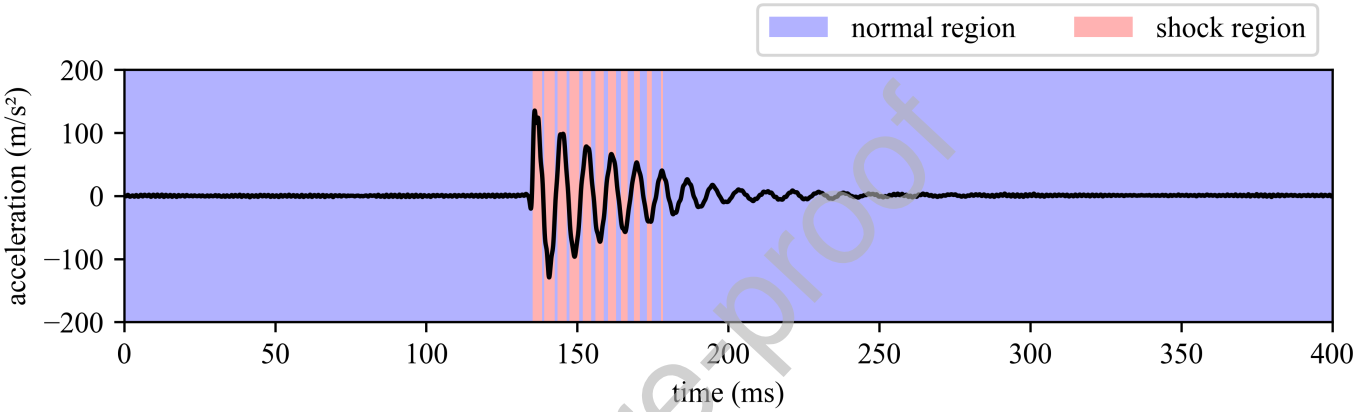


Figure 28: Detected regions from CUSUM algorithm.

Figure 29 shows the EM algorithm performance. The model shows similar behavior in that it only captures the peaks of the shock event. It detects the shock consistently and in a straightforward manner. The model does fail to capture the beginning of the shock event since the acceleration wasn't large enough.

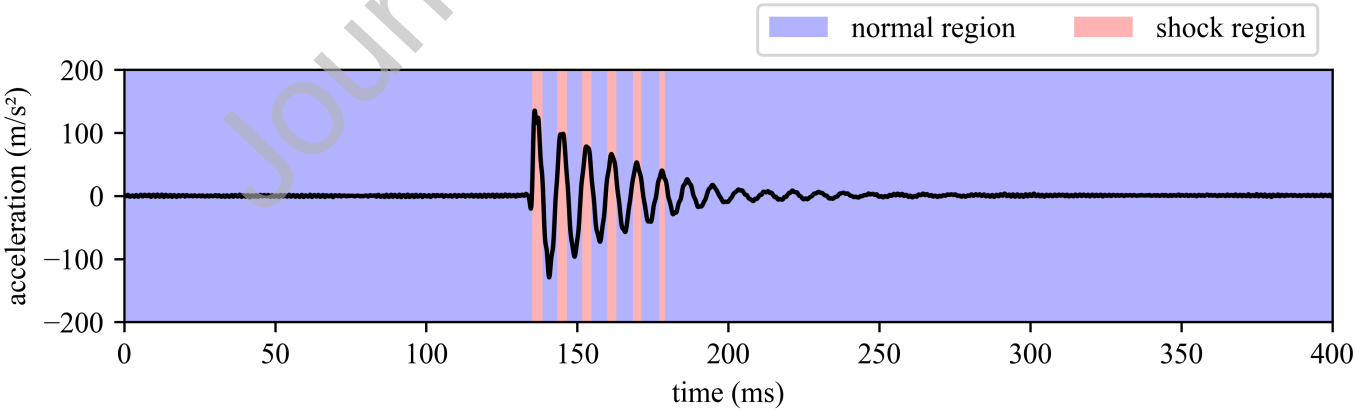


Figure 29: Detected regions from EM algorithm (representative run).

The GM results in Figure 30 differ from those on Tests 1–3. GM raises isolated alarms throughout the record, before, during, and after the shock. Its alarms are only moderately concentrated in the shock region: 58% of them fall inside it (precision 0.585), and they cover 11% of it (recall 0.109).

Journal Pre-proof

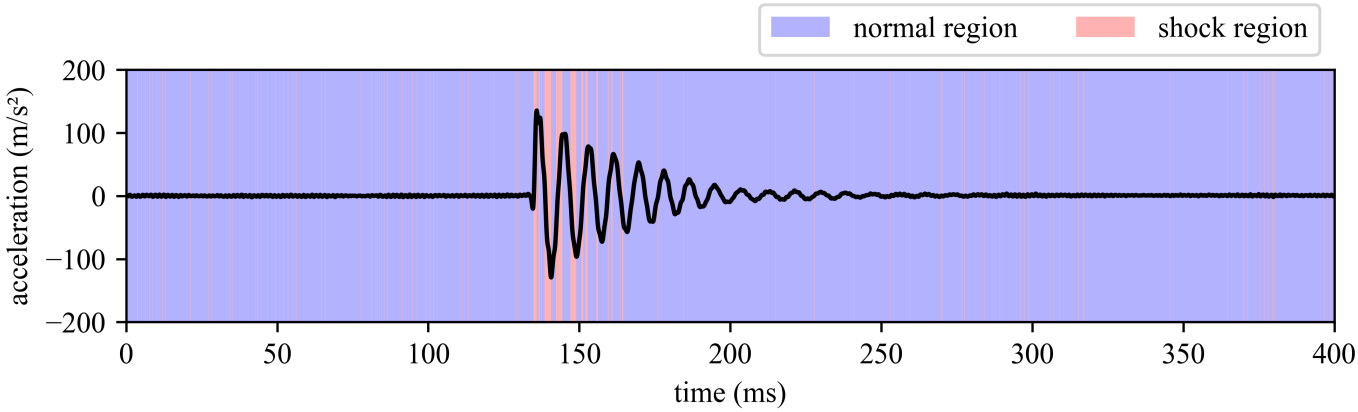


Figure 30: Detection regions from GM.

Figure 31 shows that Isolation Forest detects the shock from its onset without false alarms and, like CUSUM, reports it less frequently as the energy dissipates.

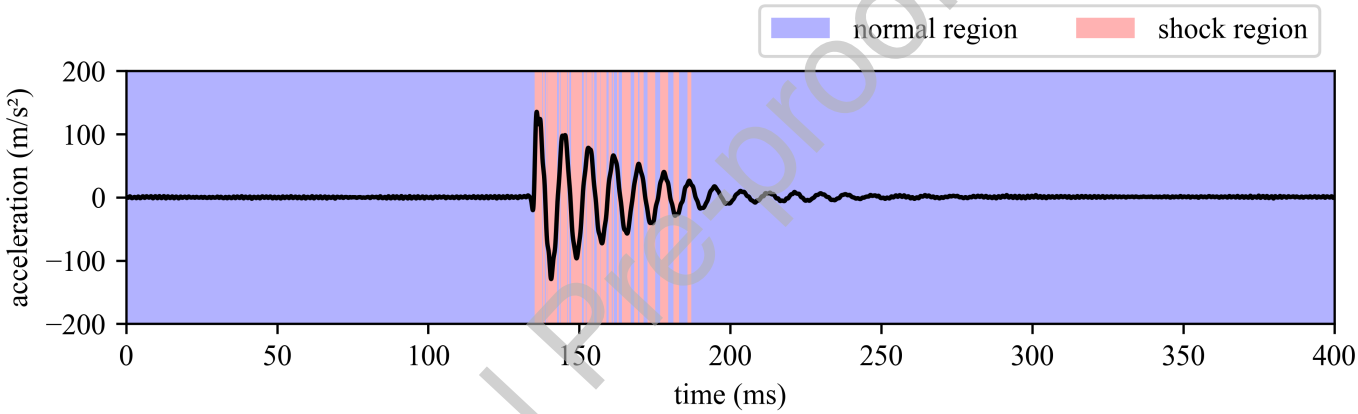


Figure 31: Detected regions from Isolation Forest (representative run).

Table 11
Metrics for online change point detection algorithm performance on signal 4 (same protocol and values as Table 7; EM and Isolation Forest: mean of three runs).

algorithm	accuracy	precision	recall	f1 score	detection delay (ms)
BOCPD	0.853	1.000	0.442	0.613	0.370
CUSUM	0.806	1.000	0.263	0.416	1.141
EM	0.780	1.000	0.164	0.282	1.099
GM	0.745	0.585	0.109	0.183	0.792
Isolation Forest	0.826	1.000	0.340	0.507	0.865

Table 11 shows the metrics for each of the models on signal 4. BOCPD performed best (F1 0.613, delay 0.37 ms), followed by Isolation Forest (0.507), CUSUM (0.416), and EM (0.282), all four with a precision of 1.000. GM ranked last (F1 0.183) because of the scattered alarms described above.

Journal Pre-proof

A.4. Test 5 Results and Discussion

Figures 32–36 show the detection performance of the various algorithms on Test 5. Table 12 shows the metric scores calculated for the respective algorithms.

Figure 32 shows performance similar to that observed for the other signals. The model detects the shock at the beginning and ends once enough energy has dissipated.

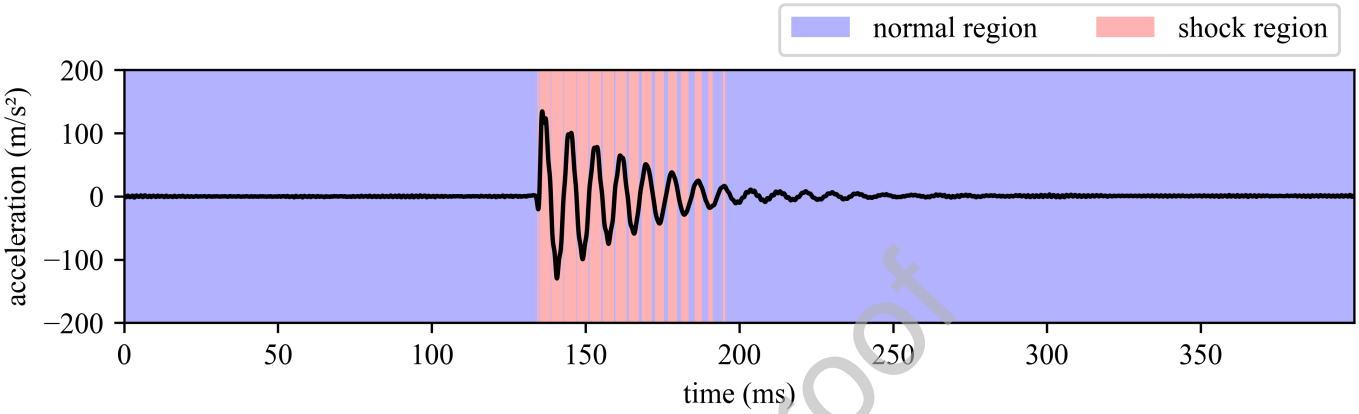


Figure 32: Detected regions from BOCPD algorithm.

Figure 33 shows the detection regions produced by CUSUM. The initial detection is a little bit delayed, but otherwise, the model detects most of the shock event.

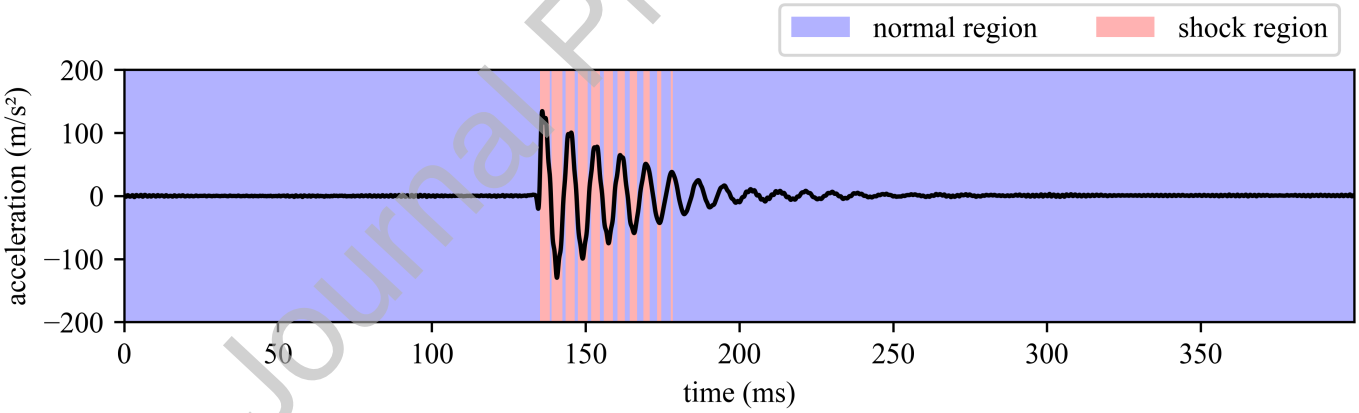


Figure 33: Detected regions from CUSUM algorithm.

Figure 34 shows that the EM method detects only the highest-amplitude portions of the shock response. The model does not show any false positives before or after the shock event. However, the model also fails to capture the lower acceleration areas where the effects of shock remain. This may work well in activating a mitigation response when an event occurs. This is similar to setting a predefined threshold, but the model requires only statistical parameters to operate rather than manual analysis.

Figure 35 shows how GM performed on signal 5. As on Test 4, GM raises scattered alarms over the whole record and covers only part of the shock region (recall 0.159, precision 0.564). The higher false-positive rate makes the model less reliable as a standalone shock detector because frequent false alarms could trigger unnecessary mitigation actions.

Journal Pre-proof

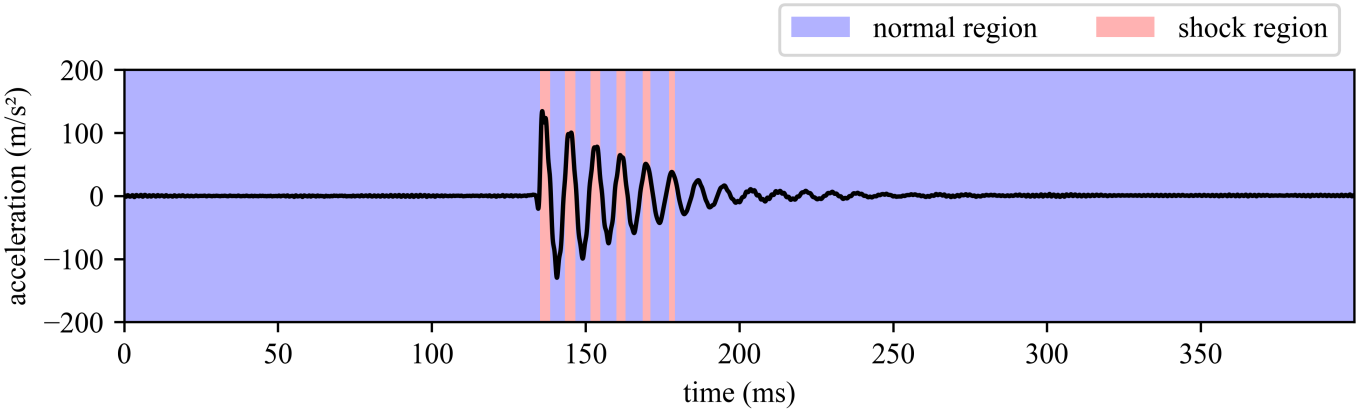


Figure 34: Detected regions from EM algorithm (representative run).

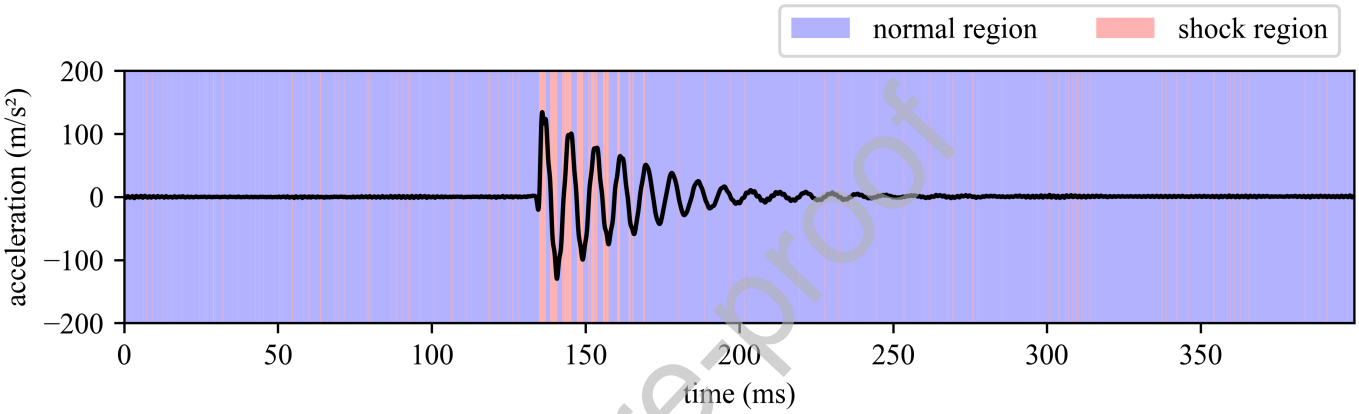


Figure 35: Detection regions from GM.

Figure 36 shows that Isolation Forest behaves as on the other signals: it detects the high-amplitude part of the shock without false positives before or after the event.

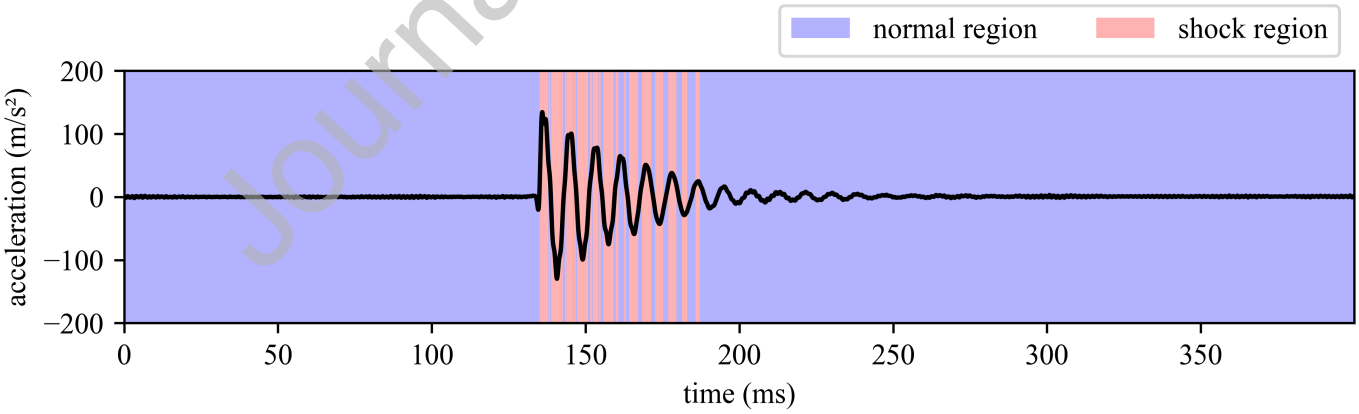


Figure 36: Detected regions from Isolation Forest (representative run).

Table 12 shows that BOCPD performed best on signal 5 (F1 0.611, delay 0.37 ms), followed by Isolation Forest (0.509), CUSUM (0.413), and EM (0.277), all with a precision of 1.000. The advantage of BOCPD over CUSUM and EM comes mainly from its higher recall. GM ranked last (F1 0.248, precision 0.564).

Journal Pre-proof

Table 12
Metrics for online change point detection algorithm performance on signal 5 (same protocol and values as Table 7; EM and Isolation Forest: mean of three runs).

algorithm	accuracy	precision	recall	f1 score	detection delay (ms)
BOCPD	0.852	1.000	0.440	0.611	0.368
CUSUM	0.804	1.000	0.261	0.413	1.138
EM	0.778	1.000	0.161	0.277	1.096
GM	0.745	0.564	0.159	0.248	0.801
Isolation Forest	0.826	1.000	0.342	0.509	0.856

Journal Pre-proof