HAL open science

Dynamic time series segmentation for health monitoring of hybrid systems

Léonie Hatte, Pauline Ribot, Elodie Chanthery

To cite this version:

Léonie Hatte, Pauline Ribot, Elodie Chanthery. Dynamic time series segmentation for health monitoring of hybrid systems. 2025 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Oct 2025, Vienne, Austria. pp.7269-7274, ⟨10.1109/SMC58881.2025.11342833⟩. ⟨hal-05319855⟩

HAL Id: hal-05319855

https://hal.science/hal-05319855v1

Submitted on 17 Oct 2025

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Copyright

Copyright - All rights reserved

Dynamic Time Series Segmentation for Health Monitoring of Hybrid Systems

Leonie Hatte*, Pauline Ribot* and Elodie Chanthery *

***Abstract*— Monitoring and diagnosing complex, real-world, industrial hybrid systems require accurate and up-to-date models that can adapt to evolving system behaviors. Such systems, characterized by both continuous and discrete dynamics, are best represented by hybrid models. In this article, we present the segmentation step of HyMED (Hybrid Model Enrichment for Diagnosis), a model-based health monitoring and diagnosis method that monitors hybrid systems and automatically updates the system model if necessary. HyMED uses noisy multivariate time series data to dynamically update models, addressing unanticipated degradations and faults. A key feature of HyMED is its online and passive segmentation step (ODS), which enables robust detection of system mode changes in complex, nonlinear time series. Unlike traditional segmentation methods, ODS dynamically determines its segmentation hyperparameters through an automatic parameter selection process. ODS guarantees adaptability without the need for manual adjustment. The effectiveness of HyMED's segmentation method is demonstrated through a case study on an engine timing system, where its performances are compared to the offline method depicted in the Ruptures library.**

INTRODUCTION

Real systems are difficult to characterize as they are inherently hybrid, combining both continuous and discrete dynamics. These systems are complex and multivariate: involving multiple interactions between variables, delays, and uncertainties that challenge mathematical modeling. The relationships between inputs and outputs cannot always be captured using simple equations or classical approaches, necessitating more sophisticated modeling frameworks. Additionally, sensor noise is an inherent challenge these kinds of systems, requiring careful consideration in all stages of monitoring, diagnosis, and maintenance.

In this sense, hybrid formalisms and more particularly Hybrid Automata (HA) [5] are becoming more popular in various industries including biology and medicine [11], robotics and industrial processes [10], cyber attacks [17], and automobiles [18]. This formalism is broadly discussed in the literature, making it well understood and widely adopted by the community.

Let define the *system* as the real cyber-physical system under study, and the *model* as the approximate representation of the system (represented by a hybrid automaton). While real systems undergo degradations and environmental changes overtime, their models – used for designing control or monitoring – often remain unchanged. As the system evolves, the initial model becomes outdated, leading to inaccurate health monitoring. The diagnostic process, in order to be efficient, must be performed on an updated model that reflects the current system dynamics.

However, identifying a complex system as a hybrid automaton is costly and time-consuming, especially for large-scale systems. To avoid repeatedly identifying the system model, our approach proposes to update parts of the model over time. This update should go beyond merely adjusting model parameters: it must also modify the model's structure by adding states or transitions if necessary. In the literature, the update process is referred to as *adaptive identification* [13] or *improvement* [18], but we prefer the term *enrichment*. Our goal is to make systems *resilient*, meaning they can maintain operations even when events that were not anticipated during the design of the system occur.

Through this work, we introduce HyMED, a model-based diagnosis method that automatically updates a given hybrid automaton, strengthening its applicability in online system monitoring and fault diagnosis. The paper specifically focuses on the segmentation step, which is an online, multivariate segmentation method, named ODS, designed to operate automatically. ODS only requires two hyperparameters, automatically determined for optimal adaptability and robustness. The hybrid automaton model used as input of HyMED comes from expert design or a prior identification phase.

This paper is organized as follows. Section I provides background on hybrid automata and an overview of existing segmentation methods. Section II presents the HyMED methodology, detailing its iterative approach to maintain an up-to-date model through dynamic segmentation. Section III presents ODS, the dynamic segmentation step, which constitutes the main contribution, highlighting its adaptability and automatic parameter selection. Section IV evaluates its effectiveness through a case study on a timing system for an engine, comparing it with existing offline segmentation approaches. Finally, the concluding section summarizes the key findings and discusses future research directions.

I. BACKGROUND

A. Hybrid Automata

The hybrid automata formalism [5] models hybrid systems that present discrete and continuous signals.

A hybrid automaton (HA) is formally defined as a tuple: $\mathcal{H} = (Q, X, E, F, T, Guard, Jump, q_0, x_0)$ where:

- Q is a finite set of discrete states, also called modes.
- $X \subseteq \mathbb{R}^n$ is the continuous state space, with $n \in \mathbb{N}$ representing the number of continuous variables.

* This work was not supported by any organization

* Authors are with LAAS-CNRS, Université de Toulouse, INSA, UPS, Toulouse, France firstname.name@laas.fr

- E is a finite set of discrete events.
- $F : Q \times X \rightarrow Q$ describes the flows or laws of continuous variables within each state.
- $T : Q \times X \times E \rightarrow Q$ defines the set of arcs – called transitions – connecting states.
- $Guard \subseteq Q \times Q \times X$ is the set of guard conditions associated with each transition. They specify when the transition between two states can be triggered.
- $Jump : Q \times Q \times X \times E \rightarrow X$ is the set of jump conditions. Each transition may include a jump condition that resets the value of a variable when crossing a transition.
- q_0 is the initial discrete state.
- x_0 is the initial continuous state.

Hybrid automata are represented by discrete states and transitions. Each state $(q, x) \in Q \times X$ is associated with a flow $f(q, x) \in F$, typically represented by a differential equation that describes how variables x evolve over time. The operating mode of the system at a given moment is called the *current state*. When the system is in the current state (q, x), all variables x follow the flow conditions $f(q, x)$.

A transition $t(q, x, e) = q' \in T$ allows the system to switch from state q to state q'. Each transition can be triggered by at least one of three conditions: a *guard condition*, an *event condition*, and a *jump condition*. The guard condition $guard(q, q', x)$ defines the permissible range of values for the continuous variables x. The event condition e represents the occurrence of a discrete event that enables a transition. Finally, the jump condition $jump(q, q', x)$ dictates how the continuous variables x change instantaneously when the transition from q to q' is fired; if no jump condition is specified, the continuous variables retain their values. Interested readers can refer to [5] for more details.

B. State-of-the-art in segmentation methods

Segmenting a noisy multivariate Time Series (TS) involves dividing it into segments, each exhibiting a stable or distinct behavior typically associated with a unique system mode. Identifying these modes enables the detection of changes in trends or patterns over time, thereby supporting improved analysis and prediction. Segmentation methods are broadly categorized into offline and online approaches. Offline methods process complete datasets, ensuring high accuracy but lacking real-time adaptability. Conversely, online methods incrementally process data for real-time use, though often at the cost of noise sensitivity and computational efficiency.

1) Offline segmentation methods: Offline approaches, such as regression-based segmentation and change-point detection techniques, provide high precision but assume access to the full dataset and are computationally demanding. For instance, the Linear Multistep Method (LMM) [8], [4] detects derivative discontinuities but relies on fixed thresholds, limiting adaptability. The Ruptures library [20], used in [18], provides accurate change-point detection through global optimization, but is unsuitable for streaming contexts or embedded systems with tight constraints.

2) Online segmentation methods: Four categories can be identified for online segmentation methods and their application.

Change-point detection methods such as CUSUM [7], ADWIN [1], and GLR [22] are designed to detect abrupt statistical changes, but often rely on static thresholds and exhibit a high sensitivity to noise. Their limited adaptability to slow transitions or hybrid behaviors restricts their usefulness for complex TS.

Clustering-based methods such as online K-Means, DBSCAN [3] or BIRCH [23], segment data by grouping similar points. While DBSCAN handles irregular shapes and BIRCH is efficient for large datasets, these methods typically assume static cluster structures and require careful tuning of parameters (e.g., radius, thresholds), limiting their performance on evolving signals.

Regression-based models use prediction to detect deviations that may indicate segmentation points. Linear regression is efficient, but not suitable for nonlinear dynamics [11]. Symbolic regression [16] adds flexibility and interpretability, but is computationally expensive. LSTM networks [12], although powerful for sequence modeling, require significant training data and resources, making them less appropriate for real-time segmentation.

Frequency-based methods like Fourier, Wavelet Transform [2], EMD [6], and Autoslex [15] analyze signal structures in the spectral domain. While effective for non-stationary signal analysis, they face challenges such as parameter sensitivity (e.g., wavelet selection), mode mixing (EMD), and loss of time-localized information (Fourier). Their computational cost also limits their use in real-time settings.

In summary, while online methods offer real-time capabilities, many existing techniques are limited by assumptions of stationarity, static parameters, or high complexity, making them less suited for hybrid or evolving systems.

II. HyMED method

This section presents HyMED (Hybrid Model Enrichment for Diagnosis), the proposed iterative method designed to maintain an accurate model even in presence of unexpected degradations or faults, ensuring the reliability of the model over time. Fig. 1 shows the steps of the method, but the contribution of this paper focuses only on the Segmentation step.

The method starts with a known model of the system, represented as a *HA*, which has been previously identified. HyMED's goal is to monitor and diagnose the system, while enriching *HA* from multivariate time series (TS) data coming from the real system, if necessary. In this article, time series are referred to as: $Z = \{z_1, \ldots, z_n\}$, where each observation $z_i \in \mathbb{R}^m$ represents a vector of m variables, and $n \in \mathbb{N}$ is the total number of observed values. The Diagnoser, built from *HA*, monitors the system and determines a set of possible states from the system's observed data. In HyMED, the diagnoser's generation follows the methodology proposed in

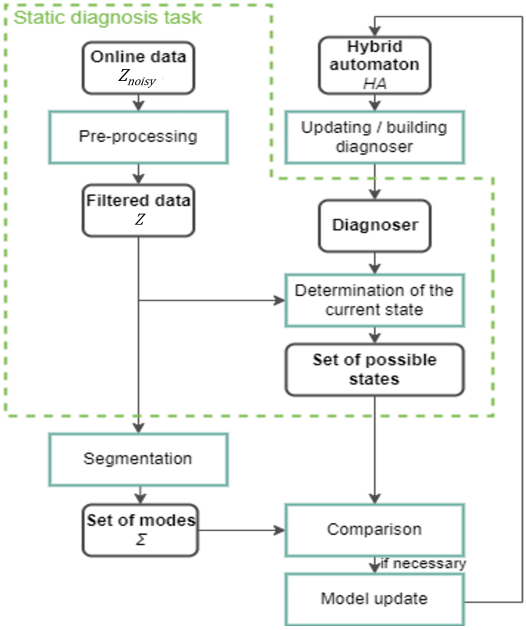

Fig. 1. HyMED overall diagram

[21], which is a previous contribution by some of the authors of this paper.

The Static diagnosis task (green box) begins with a data *Pre-processing* step: the system's observations (Online data) obtained from sensor measurements are inherently affected by noise, which can distort the data and impact subsequent analysis. To ensure a more accurate segmentation, a noise filtering step is performed to obtain filtered data Z. The Filtered data are used by the Diagnoser to determine the Set of possible states of the system, including anticipated faulty states.

The HyMED method adds to the Static diagnosis task a new enrichment functionality. The *Segmentation* step consists in cutting the Filtered data when a notable change occurs in at least one variable of the signal. The method is done online: as soon as a transition point is detected, the data is cut into a *segment* that is processed by the succeeding steps of the method. Each segment corresponds to an operating mode of the HA.

Once a segment representing a system operating mode and a set of possible current states have been identified, the next step is to compare them (Comparison box) to determine whether *HA* remains up-to-date, and what needs to be updated. If the real data match the simulated data generated from *HA*, the model accurately reflects the system behavior. Otherwise, the *Model update* step is performed, ensuring that *HA* is enriched in order to incorporate the newly observed system dynamics.

HyMED ensures frugal model updates, where only the necessary modifications are performed: a new state (q_i, x_i) is created, along with the required transitions – and their associated conditions – connecting it to both the preceding and succeeding states. Additionally, the continuous dynamics $f(q_i, x_i)$ corresponding to this new state must be determined. Since the diagnoser is derived from *HA*, it must be updated through the *Updating diagnoser* step whenever *HA* is enriched, following the same principles. Two variables are thus updated iteratively through HyMED: the hybrid automaton *HA* and its derived *Diagnoser*.

To our knowledge, HyMED is the first method to incorporate a diagnoser in the model updating process. In [18], the HA is updated every iteration of the method. With their method, a slow degradation could not be detected and would be integrated in the normal behavior of the model. Furthermore their segmentation is done offline and prior to the hybrid *Model update* step. Our method aims to detect those slow degradations and create a corresponding faulty or degraded mode when such behavior is identified. In [19] a method comparable to ours is presented, but the study is limited to linear hybrid automata, as in [14] where the study is offline.

III. Dynamic segmentation

This section focuses on the main contribution of this paper: the *Segmentation* step of HyMED method. An online dynamic segmentation method is specifically designed for multivariate, nonlinear time series.

A. Online Dynamic Segmentation (ODS) Algorithm

The goal of segmentation is to partition the signal into intervals where each segment corresponds to a consistent system behavior, thereby enabling effective tracking of mode transitions. The approach must be adaptive, robust to noise, and capable of operating incrementally as new data arrives.

The proposed segmentation procedure, named ODS (Online Dynamic Segmentation), is described by Algo. 1 that relies on the detection of significant changes in the dynamics of at least one variable within the multivariate signal. Let $Z = \{z_1, z_2, \ldots, z_n\} \subset \mathbb{R}^m$ be the input signal, where each observation $z_i \in \mathbb{R}^m$ is a vector of m variables. ODS produces a set $\boldsymbol{\kappa}$ of transition points and a set Σ of segments, which are updated at each iteration.

Algorithm 1 Online Dynamic Segmentation (ODS)

Input: Z, $\boldsymbol{\kappa}$, Σ, (W^*, F^*)
Output: Z, $\boldsymbol{\kappa}$, Σ

Let Z be the set of past observations
Let $\boldsymbol{\kappa}$ be the set of detected transition points
Let Σ be the set of cut segments
Let (W^*, F^*) be the set of optimal parameters for the m dimensions of signal Z
$Z \leftarrow \emptyset$, $\boldsymbol{\kappa} \leftarrow \emptyset$, $\Sigma \leftarrow \emptyset$
Let $\mathbf{z}_i$ the current observation
for each new observation $z_i \in Z$ **do**
    $(Z, \boldsymbol{\kappa}, \Sigma) \leftarrow Seg_track(z_i, Z, \boldsymbol{\kappa}, \Sigma, W^*, F^*)$
end for
    return Z, $\boldsymbol{\kappa}$, Σ

ODS is controlled by two sets of hyperparameters: $W = (w_1, \ldots, w_m)$ defining the window sizes for local derivative computations, and a vector $F = (f_1, \ldots, f_m)$ specifying

Algorithm 2 Seg_track

Input: z_i, Z, $\boldsymbol{\kappa}$, Σ, W^*, F^*
Output: Z, $\boldsymbol{\kappa}$, Σ

1: Let $Z = \{z_1, \ldots, z_{i-1}\}$ be the set of past observations
2: Let z_i be the current observation
3: Let $\boldsymbol{\kappa} = \{\kappa_1, \ldots, \kappa_{prev}\}$ be the set of transition points
4: Let Σ be the set of cut segments
5: Let $S = \{z_{prev}, \ldots, z_i\}$ be the current segment s.t $S \in \Sigma$
6: Let $W^* = (w_1^*, \ldots, w_m^*)$, $F = (f_1^*, \ldots, f_m^*)$ be the optimal parameters for the m dimension of signal Z
7: **for** each $z_{ij}, j \in (1, ..., m)$ in z_i **do**
8: Compute z''_{ij}, $\mathcal{N}_{z''_{ij}}(w_j^*)$, $\epsilon_{ij}(f_i^*, \mathcal{N}_{z''_{ij}})$ ▷ Eq.1, Eq.2
9: **if** $|z''_{i-1,j}| \leq |\epsilon_{i-1,j}|$ **and** $|z''_{ij}| > |\epsilon_{ij}|$ **then**
10: z_i is a potential transition point
11: **if** κ_{prev} is $None$ **then** $\kappa_{prev} \leftarrow z_i$
12: **end if**
13: **if** $|S = \{z_{prev}, \ldots, z_i\}| \geq 3$ **then**
14: z_i is a final transition point: $\kappa_i \leftarrow z_i$
15: **end if**
16: $\kappa_{prev} \leftarrow \kappa_i$
17: $\boldsymbol{\kappa} \leftarrow \boldsymbol{\kappa} \cup \{\kappa_{prev}\}$
18: $\Sigma \leftarrow \Sigma \cup S$ ▷ Last segment is stored
19: $S \leftarrow \{z_i\}$ ▷ Start of a new segment
20: **end if**
21: $S \leftarrow S \cup \{z_i\}$
22: $Z \leftarrow Z \cup \{z_i\}$
23: **end for**
return Z, $\boldsymbol{\kappa}$, Σ

the sensitivity factors used to construct dynamic thresholds for each variable of dimension m. We propose an automatic selection method for determining the optimal values W^* and F^* of these two hyperparameter sets in Section III-B. These values are computed beforehand and remain fixed throughout the segmentation step.

The segmentation algorithm, named Seg_track, processes each new observation z_i upon arrival. At each iteration, the sets of transition points $\boldsymbol{\kappa}$, past observations Z, and segments Σ are updated (Algo. 1, line 8).

In Seg_track (see Algo. 2), for every variable z_{ij}, the second derivative z''_{ij} is estimated based on the most recent windows of data. A local neighborhood $\mathcal{N}_{z''_{ij}}$, centered around the current point and of width w_j^*, is computed:

$$\mathcal{N}_{z''_{ij}} = \{z''_{lj} : |i - l| \leq w_j^*\} \tag{1}$$

with $w*_j$ the window size and $l \in \mathbb{N}$. This neighborhood is used to calculate the local mean and standard deviation of the second derivative. The dynamic threshold ϵ_{ij} (line 8) is then defined as the sum of this mean and the product of the standard deviation by the sensitivity factor:

$$\epsilon_{ij} = \overline{\mathcal{N}_{z''_{ij}}} + f_j^* \cdot \sigma_{\mathcal{N}_{z''_{ij}}} \tag{2}$$

where $\overline{\mathcal{N}_{z''_{ij}}}$ and $\sigma_{\mathcal{N}_{z''_{ij}}}$ are the mean and standard deviation of the neighborhood of z''_{ij} and f_j is the sensitivity factor. A data point is a possible transition point if the current second derivative exceeds this threshold while the previous value remained within bounds, indicating a meaningful change in signal dynamics (line 9). The set of *possible transition points* is formally defined as:

$$\{\forall i, z_i \in Z : \forall j, |z''_{i-1,j}| \leq |\epsilon_{i-1,j}| \wedge |z''_{ij}| > |\epsilon_{ij}|\} \tag{3}$$

To ensure the reliability of detected transitions, a refinement step is introduced. A possible transition point is confirmed as a final transition point in $\boldsymbol{\kappa}$ only if the current segment contains at least three observations, which guarantees that each segment captures a well-defined local behavior (line 13). If a possible transition point z_i appears too close to a previous one, it is treated as a regular data point that extends the current segment S.

This approach provides sufficient segment length for robust dynamic analysis while preventing over-segmentation due to closely spaced transition detections, and is inspired by the method proposed in [4]. This refinement step reduces false positives caused by minor oscillations and helps avoid over-segmentation. As a result, Seg_track adapts to both abrupt and gradual changes while maintaining temporal coherence and minimal latency. The effectiveness of Seg_track relies on the assumption that the signal is continuous and at least twice differentiable.

Throughout the procedure, the sets of transition points, segments and accumulated data are updated online. The segmentation thus remains continuously aligned with the system's evolving dynamics, allowing HyMED to track operational modes in real time.

B. Automatic Hyperparameter Selection

The performance and adaptability of ODS depend on the choice of two hyperparameters $W \in \mathbb{N}^m$ and $F \in \mathbb{R}^m$, which determine the temporal scale and sensitivity of change detection. Instead of relying on empirical manual tuning, we introduce an automatic optimal hyperparameter selection strategy based on simulated time series data D generated from the initial hybrid automaton model (see Algo. 3).

The key assumption is that the optimal segmentation parameters extracted from this synthetic reference data remain valid when applied to real data during system monitoring. The selection procedure evaluates a predefined grid of candidate parameter pairs for each variable independently (line 5). For every pair (w, f), the segmentation algorithm is executed, and the resulting number of detected transition points is recorded. The most frequently obtained number of detected transitions across all candidate pairs is considered indicative of stability. Among the parameter pairs yielding this number, the method computes the Kendall Tau coefficient between the resulting segmentations to assess their relative concordance (line 8).

The *Kendall Tau coefficient* (τ) [9] is a non-parametric measure of rank correlation that quantifies the degree of similarity between different segmentations and captures the consistency of detected transition points across parameter choices. Unlike other correlation coefficients (e.g., Pearson

Algorithm 3 Optimal hyperparameter selection for ODS

Input: D, $\boldsymbol{\kappa}$, Σ, $\mathcal{W}$, $\mathcal{F}$
Output: W, F

1: Let $D = \{d_1, \ldots, d_n\}$ be the set of n simulated data of dimension m
2: Let $\mathcal{W}, \mathcal{F}$ be the sets of candidate parameters
3: **for** each dimension $j \in (1, ..., m)$ of signal D **do**
4: **for** each possible couple of (w, f) s.t. $w \in \mathcal{W}, f \in \mathcal{F}$ **do**
5: $(D, \boldsymbol{\kappa}, \Sigma) \leftarrow Dynamic_seg(D, \boldsymbol{\kappa}, \Sigma, w, f)$
6: **end for**
7: **for** each pair (w, f) producing the most frequently obtained number of transition points $|\boldsymbol{\kappa}|$ **do**
8: Compute Kendall Tau coefficient $\tau(w, f)$
9: **end for**
10: $P = \{(w, f),\ \text{s.t. } \tau(w, f) = \max(\tau(w, f))\}$
11: **if** $|P| = 1$ **then** $(w_j^*, f_j^*) \leftarrow (w, f)$
12: **else**
13: **if** $|P| > 1$ **then**
 - Select (w, f) with the largest value of f
 - If several pairs have this value f, choose the one with the smallest w
14: $(w_j^*, f_j^*) \leftarrow (w, f)$
15: **end if**
16: **end if**
17: **end for**
 return $W^* = (w_1^*, ..., w_m^*), F^* = (f_1^*, ..., f_m^*)$

or Spearman), Kendall Tau does not assume linearity or specific data distributions, making it well-suited for comparing segmentation outcomes in complex, non-linear systems. The Kendall Tau coefficient is computed as:

$$\tau = \frac{\xi - \delta}{\frac{1}{2}\nu(\nu - 1)} \tag{4}$$

with ξ, the number of concordant pairs, δ, the number of discordant pairs, and ν, the number of elements in each ranking. A pair is considered concordant if the two corresponding points are positioned in the same relative manner with respect to the envelope $[-\epsilon_{ij}, \epsilon_{ij}]$ in both segmentations.

The optimal parameter pair $\{(w_j^*, f_j^*)\}$ for each variable is selected as the one that maximizes the Kendall Tau value (line 10). A coefficient close to 1 indicates high concordance, meaning that transition points detected across different configurations are consistently ranked, thereby signaling a stable and meaningful segmentation. If multiple pairs reach the same maximal correlation, the preference is given to the pair with the highest sensitivity factor, and if there are still several possible candidates, to the one with the smallest window size (line 14). This priority balances the accuracy of transition detection and responsiveness to signal variations.

By integrating this automatic hyperparameter tuning process relying on the Kendall Tau coefficient, HyMED ensures that its segmentation procedure is both autonomous and adapted to the specific dynamics of the monitored system. The resulting parameters enable stale and meaningful segmentation across signal dimensions, establishing a reliable foundation for subsequent diagnostic and model enrichment steps.

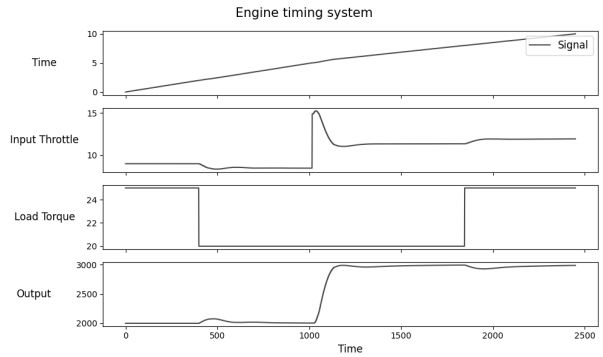

Fig. 2. Input and output signals of the engine timing system

IV. Case study

This section considers the data generated by the Simulink model of an engine timing system (Fig. 3). It has 3 inputs: the time as an independent input signal, the input throttle (the desired speed of the system in RPM), the load torque (in J/rad), and 1 output, the engine speed in RPM. The time, input throttle and output are continuous variables, whereas the load torque is a discrete variable.

Our goal is to prove the effectiveness of HyMED's segmentation with a multivariate, complex and hybrid time-series, in comparison with the offline segmentation proposed in [18] that uses the Ruptures library. Fig. 2 shows the time series to segment.

The segmentation process begins with the determinination of the optimal hyperparameters (W^*, F^*). Algo. 3 is applied to each dimension to determine the optimal pairs $\{(w_j^*, f_j^*)\}$, which are then used by OSD for segmenting all relevant dimensions. For example, while time is considered as an independent input signal, it is excluded from the segmentation process, as the segmentation focuses only on the other dimensions. This selection is guided by expert knowledge of the system.

Fig. 3 illustrates the results of HyMED's segmentation step applied to the case study. The method effectively detects

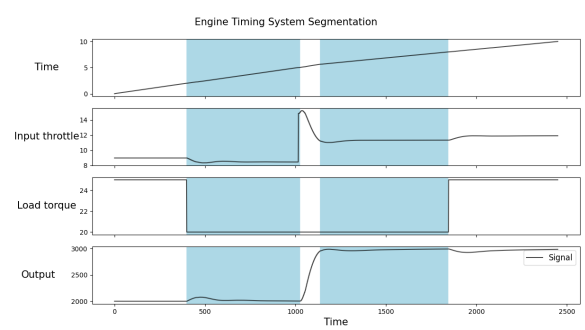

Fig. 3. Segmented TS from the engine timing system

transition points that represent mode changes, including subtle variations that Ruptures struggles to identify. Each detected transition point corresponds to a transition in at least one of the signal variables, reinforcing the validity of the segmentation. The background color changes highlight the identified modes. As observed, ODS successfully captures both major and minor shifts in system behavior.

The evaluation of segmentation quality is inherently subjective and depends on the requirements of subsequent analysis methods. In our case, we prioritize "simple segments" that showcase continuity in the derivatives, ensuring consistent dynamics within each segment. As no ground truth segmentation is available for direct comparison, performance is assessed based on the stability of the segmentation: our results show that the segmentation remains robust to reasonable changes in the hyperparameters - because it conducts to a slight change in the envelope width ϵ - consistently producing similar segmentations. This robustness demonstrates the reliability of our approach without excessive sensitivity to hyperparameter tuning. According to all those constraints, we consider that this segmentation is appropriate for HyMED.

Overall, HyMED demonstrates strong segmentation capabilities in this case study, showcasing its potential for real-time monitoring and system analysis.

Conclusion

This paper introduces HyMED (Hybrid Model Enrichment for Diagnosis), a model-based health monitoring and diagnosis method that monitors hybrid systems and automatically updates a hybrid automaton using online time series data from a real system. This paper focuses on the segmentation step of the method, which plays a crucial role in its ability to dynamically update the system model.

ODS, the proposed segmentation method, operates online and passively, making it well-suited for health monitoring applications. It is capable of handling multivariate, non-linear, and complex time series while requiring minimal manual intervention. A key feature is its automatic parameter selection, which ensures that segmentation is both adaptive and efficient without requiring extensive tuning.

By enabling online dynamic segmentation, HyMED lays the groundwork for adaptive fault diagnosis and prognosis. Experimental results show that ODS effectively detects mode transitions, including subtle changes often missed by other methods, demonstrating HyMED's potential for improving predictive maintenance and anomaly detection in hybrid systems.

As part of future work, we will focus on developing and improving the subsequent steps of HyMED, ensuring a more comprehensive and robust methodology for hybrid model updates.

References

[1] Albert Bifet and Ricard Gavalda. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM international conference on data mining*, pages 443–448. SIAM, 2007.

[2] Ingrid Daubechies. *Ten lectures on wavelets*. SIAM, 1992.

[3] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[4] Amit Gurung, Masaki Waga, and Kohei Suenaga. Learning nonlinear hybrid automata from input–output time-series data. In *International Symposium on Automated Technology for Verification and Analysis*, pages 33–52. Springer, 2023.

[5] Thomas A Henzinger. The theory of hybrid automata. In *Proceedings 11th Annual IEEE Symposium on Logic in Computer Science*, pages 278–292. IEEE, 1996.

[6] Norden E Huang, Zheng Shen, Steven R Long, Manli C Wu, Hsing H Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences*, 454(1971):903–995, 1998.

[7] Carla Inclan and George C Tiao. Use of cumulative sums of squares for retrospective detection of changes of variance. *Journal of the American Statistical Association*, 89(427):913–923, 1994.

[8] Xiangyu Jin, Jie An, Bohua Zhan, Naijun Zhan, and Miaomiao Zhang. Inferring switched nonlinear dynamical systems. *Formal Aspects of Computing*, 33(3):385–406, 2021.

[9] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[10] Charlotte Lacoquelle, Louise Travé-Massuyès, Xavier Pucel, Nathalie Barbosa Roa, and Christophe Merle. Deviation tracking with incomplete and distorted data-application to motion trajectories of industrial robots. In *33rd International Workshop on Principle of Diagnosis–DX 2022*, 2022.

[11] Imane Lamrani, Ayan Banerjee, and Sandeep KS Gupta. Hymn: Mining linear hybrid automata from input output traces of cyber-physical systems. In *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*, pages 264–269. IEEE, 2018.

[12] Benjamin Lindemann, Benjamin Maschler, Nada Sahlab, and Michael Weyrich. A survey on anomaly detection for technical systems using lstm networks. *Computers in Industry*, 131:103498, 2021.

[13] Yan Monier, Bruno Denis, Gregory Faraut, and Nabil Anwer. Identification de système à un automate hybride-revue des travaux récents. In *Modélisation des Systèmes Réactifs (MSR'23)*, 2023.

[14] Yan Monier, Bruno Denis, Gregory Faraut, and Nabil Anwer. A new global optimization for hybrid automaton identification. *IFAC-PapersOnLine*, 58(1):252–257, 2024.

[15] Hernando Ombao, Jonathan Raz, Rainer Von Sachs, and Wensheng Guo. The slex model of a non-stationary random process. *Annals of the Institute of Statistical Mathematics*, 54:171–200, 2002.

[16] Swantje Plambeck, Maximilian Schmidt, Goerschwin Fey, Audine Subias, and Louise Travé-Massuyès. Dynamics-based identification of hybrid systems using symbolic regression. In *2024 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 64–71. IEEE, 2024.

[17] Joseba Quevedo, Helem Sánchez, Damiano Rotondo, Teresa Escobet, and Vicenc Puig. A two-tank benchmark for detection and isolation of cyber attacks. *IFAC-PapersOnLine*, 51(24):770–775, 2018.

[18] Iman Saberi, Fathiyeh Faghih, and Farzad Sobhi Bavil. A passive online technique for learning hybrid automata from input/output traces. *ACM Transactions on Embedded Computing Systems*, 22(1):1–24, 2022.

[19] Miriam García Soto, Thomas A Henzinger, and Christian Schilling. Synthesis of hybrid automata with affine dynamics from time-series data. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, pages 1–11, 2021.

[20] Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

[21] Amaury Vignolles, Elodie Chanthery, and Pauline Ribot. A holistic advanced diagnosis approach for systems under uncertainty. In *32nd International Workshop on Principle of Diagnosis–DX 2021*, 2021.

[22] Alan S Willsky and Harold L Jones. A generalized likelihood ratio approach to state estimation in linear systems subjects to abrupt changes. In *IEEE conference on decision and control including the 13th symposium on adaptive processes*, pages 846–853, 1974.

[23] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. *ACM sigmod record*, 25(2):103–114, 1996.