Abstract

One of the pertinent problems in decision and control of dynamical systems is to identify the current operational regime of the physical process under consideration. To this end, there has been an upsurge in (data-driven) machine learning methods, such as symbolic time series analysis, hidden Markov modeling, and artificial neural networks, which often rely on some form of supervised learning based on pre-classified data to construct the classifier. However, this approach may not be adequate for dynamical systems with a variety of operational regimes and possible anomalous/failure conditions. To address this issue, the technical brief proposes a methodology, built upon the concept of symbolic time series analysis, wherein the classifier learns to discover patterns so that the algorithms can train themselves online while simultaneously functioning as a classifier. The efficacy of the methodology is demonstrated on time-series of: (i) synthetic data from an unforced Van der Pol equation, and (ii) pressure oscillation data from an experimental Rijke tube apparatus that emulates the thermoacoustics in real-life combustors where the process dynamics undergoes changes from the stable regime to an unstable regime and vice versa via transition to transient regimes. The underlying algorithms are capable of accurately learning and capturing the various regimes online in a (primarily) unsupervised manner.

Keywords: Pattern discovery and classification; Symbolic Dynamics; Anomaly detection.

1 Introduction

Large-scale dynamical systems, depending on their complexity, may exhibit a variety of operational regimes and a number of possible anomalous/failure conditions. Thus, when a control command is issued, it needs to be conditioned on both present and intended operational regimes. Accordingly, traditional classification techniques are often based on the concept of supervised learning, wherein the classifier is trained on labeled data corresponding to various regimes and anomalous characteristics of the underlying process.

Pattern recognition is a mature field of research, which has found its applications in many aspects of machine learning. Several methods [1, 2] exist for identification of various classes that the observed data emanate from. Many researchers have made use of several machine learning techniques for time-series classification in diverse applications. Bagnall et al. [3] have reported a comprehensive review of time-series analysis for online detection, classification and decision & control. Sugimura and Matsumoto [4] have reported feature extraction from time-series to generate a decision tree for pattern classification. The concepts of reduced-order Markov modeling and probabilistic finite state automata (PFSA) have been studied for analysis of combustion instabilities [5], failure prognosis of structural materials [6] and rolling-element bearings [7], as well as usage of sensor networks for detection of moving targets [8]. Several researchers have also used deep neural networks [9] as well as recurrent neural networks for time-series classification [10]. However, not all of these techniques are suitable for online detection, which requires real-time classification (e.g., by using as short a time-series window as possible). Online regime detection and classification have been demonstrated by Mondal et al [11], in the setting of hidden Markov models (HMM) and by Lim and Harrison [12] in the setting of neural networks, for example.

In a vast majority of the methods mentioned above, the models are trained off-line (i.e., no simultaneous learning and classification). Furthermore, all of the above algorithms belong to the category of supervised or semi-supervised learning in the sense that they need labeled data, where the labeling is done by (human) domain experts. Even when using an unsupervised technique such as K-means [2] for time series analysis [13], a human expert must decide the number of regimes (i.e., the parameter K). Other algorithms such as Gaussian mixture models are also used for learning subclasses in populations; however, for this approach to work, data from all possible classes need to be considered during offline training. These methods are flexible for learning multiple classes only if a human supervisor pre-informs the algorithm about the actual number of classes the given data needs to be divided into. In such a situation, if the algorithm comes across a
hitherto unfamiliar anomaly or signal during the online classification phase, it may mis-classify the operation as one of the trained regimes. Often, it is not possible to have a training data set containing all possible events, especially in systems that may be subjected to occurrence of rare events.

This technical brief proposes a low-computational-cost methodology that is built upon a concept that adaptively learns new patterns and signal forms in an online fashion (i.e., in real time) without the need to know the total number of classes a priori. The efficacy of the proposed methodology is demonstrated on both synthetic data from an unforced Van der Pol oscillator and experimental data of pressure time-series from an electrically heated Rijke tube apparatus [11, 14, 15] that emulates thermoacoustic instabilities (TAI) in real-life combustors, which can lead to failure of mechanical structures, loud noise, and flow-reversal in the combustion system. From these perspectives, major contributions of the technical brief are delineated below.

- **Novelty of the algorithms:** The classifier is trained offline only on a single ensemble of time series data corresponding to an a priori known regime. Subsequently, other regimes are autonomously identified and learned as they are encountered online, which enable the algorithms to decide whether or not a block of time series belongs to one of the trained regimes.
- **Experimental Validation:** The proposed algorithms have been tested and validated with time series of both synthetic data and experimental data to establish their feasibility for potential commercial applications.

## 2 Mathematical Theory

Before embarking on an explanation of the proposed algorithms, it is necessary to provide the background for construction of probabilistic finite state automata (PFSA) (see Subsection 2.1) [16, 17] and $D$-Markov machines (see Subsection 2.2) [18].

### 2.1 Probabilistic Finite State Automata

Time series of the measured signal is quantized and then symbolized as a symbol string. In this process, the signal space is partitioned into a finite number of cells, where the cardinality $|\Sigma|$ of the (symbol) alphabet $\Sigma$ is identically equal to the number of cells. A symbol from the alphabet $\Sigma$ is assigned to each (signal) value corresponding to the cell to which it belongs [19, 20]; the details are reported in [18]. Thus, a symbol is associated with a data point at a given instant of time when the value of that data point is located in the particular cell corresponding to that symbol. The following definitions, which are available in standard literature (e.g., [16, 18]), are recalled for completeness of the technical brief.

**Definition 1.** A finite state automaton (FSA) $G$, having a deterministic algebraic structure, is a triple $(\Sigma, Q, \delta)$ where:
- $\Sigma$ is a (nonempty) finite alphabet, i.e., its cardinality $|\Sigma|$ is a positive integer.
- $Q$ is a (nonempty) finite set of states, i.e., its cardinality $|Q|$ is a positive integer.
- $\delta : Q \times \Sigma \to Q$ is a state transition map.

**Definition 2.** A symbol block, also called a word, is a finite-length string of symbols belonging to the alphabet $\Sigma$, where the length of a word $w \triangleq s_1s_2 \cdots s_\ell$ with every $s_i \in \Sigma$ is $|w| = \ell$, and the length of the empty word $e$ is $|e| = 0$. The parameters of FSA are extended as:
- The set of all words, constructed from symbols in $\Sigma$ and including the empty word $e$, is denoted as $\Sigma^*$.
- The set of all words, whose suffix (respectively, prefix) is the word $w$, is denoted as $\Sigma^*w$ (respectively, $w\Sigma^*$).
- The set of all words of (finite) length $\ell$, where $\ell$ is a positive integer, is denoted as $\Sigma^\ell$.

**Remark 3.** A symbol string (or word) is generated from a (finite-length) time series by symbolization.

**Definition 4.** A probabilistic finite state automaton (PFSA) $K$ is a pair $(G, \pi)$, where:
- The deterministic FSA $G$ is called the underlying FSA of the PFSA $K$.
- The probability map $\pi : Q \times \Sigma \to [0, \mathbb{1}]$ is called the morph function (also known as symbol generation probability function) that satisfies the condition: $\sum_{\sigma \in \Sigma} \pi(q, \sigma) = 1$ for all $q \in Q$.
- The $(|Q| \times |\Sigma|)$ morph matrix $\Pi$, which is converted into the $(|Q| \times 1)$ morph vector $\nu$ to serve as a feature in the sequel, is generated by the morph function $\pi$.

Equivalently, a PFSA is a quadruple $K = (\Sigma, Q, \delta, \pi)$.

### 2.2 D-Markov Machines

The PFSA representation of a $D$-Markov machine generates symbol strings $\{s_1s_2 \cdots s_\ell : \ell \in \mathbb{N}^+ \text{ and } s_i \in \Sigma\}$ on the underlying Markov process. In the construction of a $D$-Markov machine, it is assumed that the generation of the next symbol depends only on a finite history of at most $D$ consecutive symbols, i.e., a symbol block of length not exceeding length $D$. A $D$-Markov machine [18] is defined as follows.

**Definition 5.** A $D$-Markov machine [16] is a PFSA in the sense of Definition 4 and it generates symbols that solely depend on the (most recent) history of at most $D$ consecutive symbols, where the positive integer $D$ is called the depth of the machine. Equivalently, a $D$-Markov machine is a statistically stationary stochastic process $S = \cdots s_{-1}s_0s_1 \cdots$, where the probability of occurrence of a new symbol depends only on the last consecutive (at most) $D$ symbols, i.e.,

$$P[s_n | \cdots s_{n-D} \cdots s_{n-1}] = P[s_n | s_{n-D} \cdots s_{n-1}]$$

(1)

Consequently, for $w \in \Sigma^D$ (see Definition 2), the equivalence class $\Sigma^*w$ of all (finite-length) words, whose suffix is $w$, is qualified to be a $D$-Markov state that is denoted as $w$. 

```
As the proposed $D$-Markov algorithms need a uniform dimension of the feature vectors, the concept of state-merging \cite{18} has not been included. There are four primary parameters in the algorithms as enumerated below:

1. **Alphabet size ($|\Sigma|$):** Larger is the alphabet size more distinct are the different regimes, but more training data would be needed. There are several algorithms for selection of the optimal alphabet size (e.g., \cite{21}); but, to demonstrate efficacy of the algorithms, an alphabet size $|\Sigma| = 16$ is chosen in the technical brief for the synthetic data set and an alphabet size $|\Sigma| = 6$ for the experimental data set. The choice of alphabet size is dependant primarily on how closely spaced the data are from each regime, with higher alphabet sizes typically yielding better regime separability (if needed).

2. **Partitioning Method:** While there are many data partitioning techniques, maximum entropy partitioning (MEP) \cite{18–20}, which is a commonly used partitioning technique, has been chosen in this technical brief.

3. **Depth ($D$) in the $D$-Markov machine:** Higher values of the positive integer $D$ may lead to better results at the expense of increased computational time due to larger dimension of the space and need for more training. In this technical brief, $D = 1$ has been chosen in order to keep lower word lengths and smaller PFSAs, which lead to faster training and testing.

4. **Choice of Feature:** The feature needs to be one that best captures the nature (e.g., texture) of the signal. The morph vector $v$ (see Definition 4) has been chosen in this technical brief as the feature, because it is not only easily computed but also captures pertinent dynamics embedded in the signal.

3 The Proposed Algorithms

To address the problem of online discovery and classification of operational regimes, the proposed algorithms are first executed upon a time-series at a given time epoch, not in its entirety, but in a windowed fashion with or without an overlap. The algorithms require only a single regime to be labeled, where a (human) expert is required to label a part of the ensemble of training time-series data, which corresponds to a specific regime, called the base regime. Typically this regime would represent the nominal operating condition of the physical process under consideration; however, it is not restricted to be so.

The algorithms have two learning phases; the first is the (offline) learning/training phase of the base regime, and the second is the (online) discovery-classification phase for learning and classifying the remaining regimes.

3.1 Learning the Base Regime

A windowed segment of time-series from the training data (e.g., corresponding to the base regime) is first analyzed. Each segment is symbolized by maximum entropy partitioning (MEP) \cite{19} with a preset alphabet size, followed by construction of a probabilistic finite state automaton (PFSA)\footnote{The PFSA codes used in this paper are developed in-house by the authors and are available at: https://github.com/Chandrachur92/PFSA.} as described in Section 2. The boundaries of MEP are computed for every window. Training is done for assigned time-series, where each time series possibly comprises of multiple windowed segments and the corresponding morph vector serves as the feature of each windowed time series, and is stored. After training, the mean of all morph vectors is taken as the centroid corresponding to the regime under consideration in the higher dimensional space, similar to what is done in K-means \cite{1}. The Euclidean distance $||v||_e$ of each of the training morph vectors from the base centroid is obtained as:

$$d_j^i = ||C_j - v_j^i||_e, \quad i = 1, \ldots, N_j$$

where $v_j^i$ is the morph vector of segment $i$ in regime $j$ ($1$ for this phase), and $C_j$ is the centroid of regime $j$ defined as:

$$C_j \equiv \frac{1}{N_j} \sum_{i=1}^{N_j} v_j^i.$$  

It is noted that the segment number $i$ runs from $1$ to $N_j$, which is the number of windowed segments used to train regime $j$. A neighborhood of the base regime (i.e., $j = 1$) is generated about its centroid with radius $\rho_1$ that is computed by setting $j = 1$ in the following equation:

$$\rho_j = \frac{1}{N_j} \sum_{i=1}^{N_j} d_j^i + \gamma \sqrt{\frac{\sum_{i=1}^{N_j} (d_j^i)^2}{N_j - 1}}$$

where the first term on the right hand of Equation (3) is the mean of all distances in regime $j$, and the second term is the standard deviation multiplied by a user-selected parameter $\gamma$. The rationale for choosing the specific structure in Equation (3) is that the distribution of $d_j^i$ is assumed to be nearly Gaussian, because the cumulative effects of many independent random variables in the construction of a PFSA tend to yield a Gaussian distribution. For a Gaussian distribution, the $4\sigma$ band contains almost 100% of the data and also ensures that obvious spurious outliers do not contaminate the estimate of the radius; the parameter $\gamma = 4$ is chosen in this technical brief. However, for a non-Gaussian (e.g., fat-tailed) distribution, $\gamma$ should be appropriately chosen.

It is noted that, at the end of this primary training, the algorithms are only aware of the feature centroid and neighborhood corresponding only to the base regime and are unaware of any other regimes (e.g., anomalies or other operational states). At this point, the number of trained regimes is $T = 1$.

3.2 Online Discovery and Classification

The algorithms executes on a window of data, hereafter referred to as a data segment, from an unknown time-series. The partitioning parameters of MEP are recomputed with the same alphabet size to construct a PFSA from the data segment and to compute the morph vector to serve as an extracted feature. If the morph vector is within the neighborhood of a known regime, the time-window is classified to belong to that regime; otherwise, the algorithms treat the data segment as belonging to a newly discovered regime.
The algorithms keep track of the number of segments that they receive corresponding to the new regime and store the feature vector for each data segment; it is noted that the segments need not be observed consecutively. As the algorithms discover a new regime, the feature vector \( (v_{\text{new}}^j) \) is taken to be the initial guess for the centroid of the new regime \( (C_{\text{new}}) \). However, there may not be enough data to estimate the radius of the neighborhood. To allow for larger initial search regions, the parameters, \( \alpha_j > 1 \), are defined as follows:

\[
\alpha_j \doteq \frac{1}{N_j} \sum_{i=1}^{N_j} d_j^i + \beta \sqrt{\frac{\sum_{i=1}^{N_j} (d_j^i)^2}{N_j}} \quad j = 1, 2, \ldots, T
\]  

where \( T \) is the number of trained regimes and the user-set parameter \( \beta \) determines how much of the symmetric tail of the probability density is removed. The value of the parameter \( \beta = 1.5 \) has been used for both the cases (i.e., synthetic data and Rijke tube data) in Section 4. For each trained regime, \( \alpha_j \) is computed, and a single parameter \( \alpha \) is obtained as the minimum of \( \alpha_1 \) (corresponding to the base regime \( j = 1 \)) and the average of \( \alpha_j \) for all \( T \) trained regimes:

\[
\alpha \doteq \min \left( \alpha_1, \frac{1}{T} \sum_{j=1}^{T} \alpha_j \right)
\]  

A restriction \( \alpha_1 \geq (\alpha_1)_{\text{min}} \) is imposed to ensure numerical stability, where the user-set lower bound is selected in this paper as \( (\alpha_1)_{\text{min}} = 1.5 \).

Now, the yet unknown radius, \( \rho_{\text{new}} \), of a newly discovered regime is initially assigned a value computed as the mean of \( \rho_j \)'s, over all \( T \) trained regimes, multiplied by the factor \( \alpha \) (see Equation (5)), as below:

\[
\rho_{\text{new}} = \frac{\alpha}{T} \sum_{j=1}^{T} \rho_j \quad \text{as the first (initial) guess}
\]  

For the subsequent \( (M_1 - 1) \) online training segments, (where \( M_1 \geq 1 \) is a user-defined parameter) the radius \( \rho_{\text{new}} \) is updated as:

\[
\rho_{\text{new}} \leftarrow \frac{\rho_{\text{new}} N_{\text{new}} + \alpha ||C_{\text{new}} - v_{\text{new}}^\text{obs}||_e}{N_{\text{new}} + 1}
\]  

where \( N_{\text{new}} \) is number of segments in the discovered regime; and \( v_{\text{new}}^\text{obs} \) is the observed morph vector from the \( N_{\text{new}} \)th segment that was classified to belong to the ‘new’ discovered regime.

The procedure in Equations (6) and (7) allows for larger initial search regions until the algorithms yield a better estimate of the radius of the new regime. This is necessary because each regime may have a different variance and would need a different neighborhood for accurate classification. The updating of \( \alpha \) in Equation (5) ensures that, as the algorithms become more confident, it attempts to make less conservative (i.e., tighter) estimates of the neighborhood radius.

The next \( M_2 \) online training segments, where \( M_2 \) is also a user-set parameter, the radius \( \rho_{\text{new}} \) is updated as:

\[
\rho_{\text{new}} \leftarrow \frac{1}{N_{\text{new}}} \sum_{i=1}^{N_{\text{new}}} d_{\text{new}}^i + \gamma \sqrt{\frac{\sum_{i=1}^{N_{\text{new}}} (d_{\text{new}}^i)^2}{N_{\text{new}} - 1}}
\]  

For each of the above \((M_1 + M_2)\) online training segments, the centroid of the regime is shifted as \( C_{\text{new}} \doteq \frac{1}{N_{\text{new}}} \sum_{i=1}^{N_{\text{new}}} v_{\text{new}}^i \) as the mean of all observed feature vectors of that regime.

Once the number of training segments in a given regime reaches the set number of \( M_1 + M_2 \), the training for that regime is deemed completed and no further parameter updating occurs for that regime. The values of \( M_1 \) and \( M_2 \) largely determine how long the algorithms will consider a given detected regime as untrained. Low values of \( M_1 \) and \( M_2 \) may degrade the quality of training. On the other hand, high values of \( M_1 \) and \( M_2 \) would enhance learning in general; however, it may cause poor learning in the presence of an unknown regime (e.g., a rare event) occurring for a short duration, because of insufficient data.

In the event that a feature vector of an observed segment \( (v_{\text{obs}}^i) \) lies in the intersection of two or more yet untrained regimes, the segment is assigned to the regime, in which it has a higher probability of belonging to, as described below.

\[
\text{Classified regime} = \arg \max_j \left( \frac{||C_j - v_{\text{obs}}^i||_e}{\rho_j} \right)
\]  

After the classifier has seen most of the typical regimes, it can decide on one of these regimes or detect a new regime. These algorithms are capable of learning and augmenting the regime library, and the operation can be executed in real time for simultaneous learning and classification.

The pseudo-code of Algorithm 1 describes the first phase of regime training, which is repeatedly followed by the second phase of discovery and classification for each new segment of time series as described in the pseudo-code of Algorithm 2.

Remark 6. The proposed method is also capable of merging regimes when they are too close to each other. This task of merging is achieved by checking if the new centroid of an untrained regime lies within the neighborhoods of any trained regime. In that case, all morph vectors corresponding to the untrained regime are assigned to the trained regime and the new values of centroid and radius of the amalgamated regime is re-computed using Equations (4) to (8).

4 Results and Discussion

This section addresses testing and validation of the proposed algorithms, where time-series data have been obtained from two sources. The first source is synthetically designed.
using the (unforced) Van der Pol equation [22], wherein the change-points are very firmly defined. The second source generates an ensemble of experimental data of pressure oscillations, collected from an electrically heated Rijke tube apparatus [11,23], where the operational regimes of the physical process are changed from stable to unstable and vice versa through various stages of transience.

4.1 Synthetic Data

The synthetic time-series consist of data-points of length 50,000 corresponding to an observation period of 10,000 seconds at a sampling frequency of 5Hz. This ensemble of data sets is constructed from the standard unforced Van der Pol oscillator whose governing equation is given below:

\[
\frac{d^2y}{dt^2} + \mu(y^2 - 1) \frac{dy}{dt} + y = 0 \quad \text{for } \mu > 0 \quad (10)
\]

The generated data set comprises of four (4) regimes, each corresponding to a distinct value of the damping parameter \(\mu\), where \(\mu\) takes the values of 0.1, 1.0, 1.5, and 2.5. The change-points have been randomly selected for each time series that has at most nine sections, where each section has any one of the above regimes randomly assigned to it. Sixty (60) such time series have been generated. This ensemble of time-series serves as the ground truth for testing the proposed algorithms, because it is generated by a known algorithm with specified regimes and the associated change-points. In order to verify whether an algorithm is working as expected, the ensemble of time series is split into three parts: (i) the first part (30\% of total data) for training the base regime, which is randomly chosen as any one of the above-mentioned four regimes, (ii) the second part (60\% of total data) to learn the other regimes

Algorithm 1 Base regime training from ensemble of data

**Input:** Time-series data set corresponding to the base regime

**Initialization:** (#segments in base regime data set) \(N_I\); (#untrained regimes) \(U = 0\); (#trained regimes) \(T = 1\)

**User-set parameters:** \(\beta, \gamma, \text{ and } (\alpha_1)_{\text{min}}\)

**Output:** Base regime information: Centroid \(C_1\); Neighborhood radius \(\rho_1\); (#Segments) \(N_I\); Parameters \(\alpha_1\) and \(\alpha\)

1: \(\text{for } i = 1 \text{ to } N_I\) \(\text{do}\)
2: Symbolize time-series segment \(i\) and generate PFSA
3: Store morph vector \(v_i^1\) as the extracted feature
4: \(\text{end for}\)
5: \(C_1 = \frac{1}{N_I} \sum_{i=1}^{N_I} v_i^1\)
6: \(\text{for } i = 1 \text{ to } N_I\) \(\text{do}\)
7: \(d_i^1 = ||C_1 - v_i^1||_e \% \text{ Euclidean norm}\)
8: \(\text{end for}\)
9: \(\rho_1 = \frac{1}{N_I} \sum_{i=1}^{N_I} d_i^1 + \gamma \sqrt{\frac{\sum_{i=1}^{N_I} (d_i^1)^2}{N_I-1}}\)
10: \(\alpha_1 = \max (\alpha_{\text{min}}, \frac{\sum_{i=1}^{N_I} d_i^1 + \beta \sqrt{\sum_{i=1}^{N_I} (d_i^1)^2}}{N_I-1})\)
11: \(\alpha = \alpha_1\)
12: \(\text{return } C_1, \rho_1, N_I, \alpha_1, \alpha\)

Algorithm 2 Regime identification from time-series window

**Input:** Time-series data segment windowed from the process

**Initialization:** (# Untrained regimes) \(U\); (# Trained regimes) \(T\); Parameter \(\alpha_1\) generated in Algorithm 1; Parameter \(\alpha\) generated in Algorithm 1 and possibly updated in a previous execution of Algorithm 2; (# Segments) \(N_j\), Centroid \(C_j\), and neighborhood radius \(\rho_j\), \(j = 1, \ldots, U + T\)

**User-set parameters:** \(\beta, \gamma, M_1\) and \(M_2\)

**Output:** Regime that the time series data belong to (i.e., the classified regime); (# Untrained regimes) \(U\); (# Trained regimes) \(T\); Centroid \(C_j\) and neighborhood radius \(\rho_j\), \(j = 1, \ldots, U + T\); Parameter \(\alpha\)

1: Symbolize time series and generate D-Markov Machine
2: Extract morph vector \(v^{\text{new}}\) to serve as a feature
3: \(\text{for } j = 1 \text{ to } (T + U)\) \(\text{do}\)
4: \(\text{if } ||C_j - v||_e \leq \rho_j \text{ then}\)
5: \(\text{if } N_j > (M_1 + M_2) \text{ then}\)
6: \(\text{Classified regime } = j \% \text{ as trained regime } j\)
7: \(\text{else}\)
8: \(C_j = \frac{1}{N_j + 1} \left( \sum_{i=1}^{N_j} v_i^j + v^{\text{new}} \right)\)
9: \(d_j^{N_j+1} = ||C_j - v^{\text{new}}||_e \) and \(v_j^{N_j+1} = v^{\text{new}}\)
10: \(\text{for } i = 1 \text{ to } N_j\) \(\text{do}\)
11: \(d_i^j = ||C_j - v_i^j||_e \% \text{ Euclidean norm}\)
12: \(\text{end for}\)
13: \(\text{if } N_j \leq M_1 \text{ then}\)
14: \(\rho_j = \frac{\rho_j N_j + \alpha d_j^{N_j+1}}{N_j + 1}\)
15: \(N_j = N_j + 1\)
16: \(\text{else if } M_1 < N_j < M_2 \text{ then}\)
17: \(N_j = N_j + 1\)
18: \(\rho_j = \frac{1}{N_j} \sum_{i=1}^{N_j} d_i^j + \gamma \sqrt{\frac{\sum_{i=1}^{N_j} (d_i^j)^2}{N_j-1}}\)
19: \(\text{end if}\)
20: \(\text{if } N_j \geq M_1 + M_2 \text{ then}\)
21: \(T \leftarrow T + 1 \% \text{#trained regime increased by 1}\)
22: \(U \leftarrow U - 1 \% \text{#untrained regime decreased by 1}\)
23: \(\text{for } j = 1 \text{ to } T \text{ do}\)
24: \(\alpha_j = \frac{1}{N_j} \sum_{i=1}^{N_j} d_i^j + \beta \sqrt{\frac{\sum_{i=1}^{N_j} (d_i^j)^2}{N_j-1}}\)
25: \(\text{end for}\)
26: \(\alpha = \min \left( \alpha_j, \frac{1}{T} \sum_{j=1}^{T} \alpha_j \right)\)
27: \(\text{end if}\)
28: \(\text{Classified regime } = j \% \text{ as untrained regime } j\)
29: \(\text{end if}\)
30: \(\text{else}\)
31: \(N_{T+U+1} = 1; C_{T+U+1} = v; \rho_{T+U+1} = \alpha \times \frac{1}{T} \sum_{j=1}^{T} \rho_j\)
32: \(\text{Classified regime } = T + U + 1 \% \text{ as a new regime}\)
33: \(U = U + 1 \% \text{#untrained regime increased by 1}\)
34: \(\text{end if}\)
35: \(\text{end for}\)
36: \(\text{return } \text{Classified regime}\)
online, and (iii) the third part (the remaining 10% of total data) to test whether the regimes have been learned correctly by verification against the ground truth.

For conversion of the time series into online data, individual data sets are windowed, with the windowing occurring every 100 data-points and each window having a size of 1000 data points. In this analysis the PFSA has an alphabet size $|\Sigma| = 16$ and depth $D = 1$. Values of the parameters $M_1$ and $M_2$ have been chosen as 50 and 250, respectively. The average error has been computed over 20 different runs, where each of these runs is executed on a newly generated data set, a new randomly selected base regime, and newly trained algorithm centroids and radii.

The results are presented in Table 1 as a confusion matrix, where the failure to correctly identify a classified regime (i.e., the algorithms erroneously deciding that the given data belongs to none of the four actual regimes) is called an error (Regime5), with the algorithms having an overall average error-rate below 10%. This is considered to be a low error because the assigned regimes are very close to each other in the feature space. While the algorithms could have identified more than four regimes in the first few trials, it has been capable of refining the results to the four true regimes by merging of regimes in subsequent runs.

<table>
<thead>
<tr>
<th>True Regime</th>
<th>Regime1</th>
<th>Regime2</th>
<th>Regime3</th>
<th>Regime4</th>
<th>Regime5 (Error)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regime1</td>
<td>89.14%</td>
<td>0.1%</td>
<td>3.66%</td>
<td>0.1%</td>
<td>7.00%</td>
</tr>
<tr>
<td>Regime2</td>
<td>0.19%</td>
<td>84.27%</td>
<td>7.18%</td>
<td>0.23%</td>
<td>8.13%</td>
</tr>
<tr>
<td>Regime3</td>
<td>0.06%</td>
<td>0.67%</td>
<td>93.11%</td>
<td>0.34%</td>
<td>5.82%</td>
</tr>
<tr>
<td>Regime4</td>
<td>0.11%</td>
<td>0.11%</td>
<td>2.75%</td>
<td>93.95%</td>
<td>3.08%</td>
</tr>
</tbody>
</table>

**4.2 Experimental Rijke Tube Data**

This subsection demonstrates the efficacy of the proposed methodology by showing its capability to perform on an emulated real-life problem. To this end, the underlying algorithms have been tested and validated on experimental data of pressure time-series from an electrically heated Rijke tube apparatus [11, 23]. The Rijke tube [14] is a commonly used experimental apparatus that emulates the phenomena of thermoacoustic instabilities (TAI) [15] encountered in real-life gas-turbine combustors. It is known that occurrence of TAI can be detrimental to the safe operation and health of a combustor. Thus, there is a need to be able to detect and identify the regime that the combustor is presently operating at and identify the transient regime, wherein the combustor is expected to move either from a stable regime to an unstable regime or vice versa. Regime identification is also the first step to perform monitoring and control on the combustor system. Several techniques have been suggested for detection of TAI using different algorithms (e.g., [5, 11, 24, 25] to name a few). However, as mentioned in Section 1, all these techniques need the assistance of a human expert for classifying the training samples and to create appropriate labels.

The Rijke tube apparatus, described by Mondal et al. [11], has been used to generate transient signals described above. In this work, the detection algorithms are trained for the nominal stable condition as the base regime. Using the methodology described above, other transient and unstable regimes are identified as shown in Figure 1.

![Fig. 1: Detection and Classification from Rijke tube data:](image)

(a) Time-series 1 with separate transient regimes
(b) Time-series 1 with amalgamated transient regimes
(c) Time-series 2 with separate transient regimes
(d) Time-series 2 with amalgamated transient regimes

The ensemble of data consists of a total 40 time series that were collected at a sampling rate of 8,192 Hz and subsequently filtered with a cut-off frequency of 400 Hz [11]. Then, data windowing has been done at ~10 Hz, i.e., the length of each data window is ~800. The time series is split into 3 groups, 20% of total data for training the base regime (i.e., Regime #1), 70% for online identification, and the remaining 10% for testing.

The $D$-Markov machines (see Subsection 2.2) have been constructed from the ensemble of above windowed data with an alphabet size $|\Sigma| = 6$ and depth $D = 1$. The parameters $M_1$ and $M_2$ (see Subsection 3.2) have been chosen as 20 and 180, respectively; the rationale for choosing these low values is that transience occurs over a very short time span and so the algorithms need to train quickly (e.g., within 36 data sets), and the reduced values of $M_1$ and $M_2$ achieve the goal. In this process, there is no available ground truth as to when the transient state begins and ends. Various researchers have used different approximations for deciding the change-points for similar events. For example, a cumulative sum (CUSUM) approach has been commonly used to approximately determine the change-points (e.g., Mondal2019), for which the regimes identified by the algorithm have been cross-checked.
The four plates in Figure 1 show regime predictions for two sets of time-series of experimental data. The two left-hand plates, (a) and (c), in Figure 1 show the actual regimes (i.e., the a priori estimated ground truth) for the time-series 1 and 2, respectively. The objective here is to identify the change of regime from stable to transient and then to unstable. In this study, the algorithms also identify a transient regime that is not apparent otherwise. In the right-hand images in the plates, (b) and (d), of Figure 1, the two predicted transient regimes have been amalgamated into a single transient regime for ease of comparison with the estimated ground truth, where it is seen that the ground truth and the predictions match closely, but not perfectly, because the ground truth itself may be flawed. However, the algorithms are capable of correctly detecting the changes and still have good accuracy, especially in identifying the learned unstable regime. This is seen in the confusion matrix (computed as an average over 4 test series across 20 repeated runs) in Table 2. Figure 1 also shows that the algorithms identify the most important change (i.e., from stable to transient states nearly perfectly). This capability is very useful for initiating a control action to prevent instability, because the transient regime is where the active controller should quench the pressure oscillations as early as possible.

Table 2: Confusion Matrix: Regime Classification Using Rijke Tube Data

<table>
<thead>
<tr>
<th>Truly</th>
<th>Classified Stable</th>
<th>Classified Transient</th>
<th>Classified Unstable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stable</td>
<td>99.73 %</td>
<td>0.27 %</td>
<td>0 %</td>
</tr>
<tr>
<td>Transient</td>
<td>11.21 %</td>
<td>57.76 %</td>
<td>31.03 %</td>
</tr>
<tr>
<td>Unstable</td>
<td>0 %</td>
<td>0 %</td>
<td>100 %</td>
</tr>
</tbody>
</table>

5 Summary, Conclusions, and Future Work

This technical brief has developed a (partially) unsupervised methodology for online identification and classification of operational regimes in dynamical systems. An objective here is to discover new regimes in an online fashion, based on the a priori knowledge of only a single selected regime. Although the proposed method constructs centroids and regime radii (in the feature space) representing each regime, it is not iterative like the standard K-means [1, 2] and thus can be used for online discovery. As a new time series from an unknown regime is observed, the algorithm is updated online (without the need to iterate) until the regime model converges. The underlying algorithms have been tested and validated with synthetic data from an unforced Van der Pol equation and also with experimental data from an electrically heated Rijke tube apparatus that emulates pressure oscillations in real-life combustors.

While there are many areas of theoretical and experimental research, which must be investigated before the proposed methodology can be implemented in real-life applications, the following topics of future research are suggested.

- Enhancement of the algorithms to accommodate smaller

data window lengths, i.e., faster detection & identification of regimes.
- Modification of the algorithms for guaranteed robustness to noise and uncertainties.
- Usage of other machine learning tools (e.g., hidden Markov models, Gaussian process modelling or neural networks [1, 2]) as alternatives to symbolic dynamics.
- Verification of the algorithms’ efficacy to be able to classify other standard dynamical systems, especially chaotic systems like the forced Van der Pol oscillator and the Rössler dynamical system.
- Autonomous learning of the algorithms’ hyper-parameters, namely, $\Sigma$, $\gamma$, $\beta$, $(\alpha_1)_{min}$, $M_1$, and $M_2$.
- Further investigation within a sensor fusion framework to improve identification of unforeseen regimes that may emerge under rare events.

Acknowledgements

The authors are grateful to Dr. Sudeepta Mondal at Penn State, who kindly provided the experimental data. The reported work has been supported in part by the U.S. Air Force Office of Scientific Research (AFOSR) under Grant Nos. FA9550-15-1-0400 and FA9550-18-1-0135 in the area of dynamic data-driven application systems (DDDAS). The first author is also thankful to Indo-US Science and Technology Forum (IUSSTF) for granting him the Research Internship for Science and Engineering (RISE) scholarship. Any opinions, findings and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

References


[23] Bhattacharya, C., Mondal, S., Ray, A., and Mukhopad-