CEFET/RJ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO

ONLINE EVENT DETECTION OVER NONSTATIONARY TIME SERIES

Rebecca Pontes Salles

Thesis submitted to the Postgraduate Program in Production Engineering and Systems of the Federal Center for Technological Education of Rio de Janeiro, CEFET/RJ, as partial fulfillment of the requirements for the degree of doctor.

Advisor: Eduardo Soares Ogasawara
Co-advisor: Fabio Andre Machado Porto

Rio de Janeiro,
September 2023

ONLINE EVENT DETECTION OVER NONSTATIONARY TIME SERIES

Thesis submitted to the Postgraduate Program in Production Engineering and Systems of the Federal Center for Technological Education of Rio de Janeiro, CEFET/RJ, as partial fulfillment of the requirements for the degree of doctor.

Rebecca Pontes Salles

Examining jury:

President, Professor D.Sc. Eduardo Soares Ogasawara (CEFET/RJ) (advisor)

Professor D.Sc. Fabio Andre Machado Porto (LNCC) (co-advisor)

Professor D.Sc. Diego Moreira de Araujo Carvalho (CEFET/RJ)

Professora Dr. Esther Pacitti (INRIA)

Professor D.Sc. José Antônio Fernandes de Macêdo (UFC)

Professora D.Sc. Marta Lima de Queirós Mattoso (COPPE/UFRJ)

Rio de Janeiro,

September 2023

Ficha catalográfica elaborada pela Biblioteca Central do CEFET/RJ

S168 Salles, Rebecca Pontes
Online event detection over nonstationary time series / Rebecca Pontes Salles. — 2023.
142f. + apêndice : il. color. , enc.

Tese (Doutorado) Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, 2023.
Bibliografia : f. 123-142
Orientador: Eduardo Soares Ogasawara
Coorientador: Fabio Andre Machado Porto

1. Análise de séries temporais. 2. Detecção de eventos. 3. Processamento de sinais I. Ogasawara, Eduardo Soares. (Orient.). II. Porto, Fabio Andre Machado (Coorient.). III.Título.

CDD 519.232

Elaborada pela bibliotecária Tania Mello – CRB/7 nº 5507/04

DEDICATION

To God and to my beloved family who have always helped, supported and guided me throughout my whole life.

ACKNOWLEDGMENTS

The present work was developed with the support of the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

The author thanks CNPq and FAPERJ for partially sponsoring this research.

The author also thanks the contributions of her Data Analytics Lab teammates Janio Lima, Luciana Escobar, Lais Baroni, Heraldo Borges, Arthur Ronald, Antonio Castro, Diego de Salles and Adalberto Andrade, and, most importantly, of her advisors, Eduardo Ogasawara and Fabio Porto. All the given support, advice, and observations were most appreciated and very important to the author's professional growth.

RESUMO

Detecção Online de Eventos sobre Séries Temporais Não-estacionárias

Os processos monitorados em quase todas as áreas produzem séries temporais. Dado o aumento no volume de dados gerados e na largura de banda observado nas últimas décadas, a maioria dessas séries temporais evolui em um cenário online, em tempo real ou de streaming. No entanto, a maioria dos métodos do estado-da-arte disponíveis, baseados em aprendizado de máquina, não estão adaptados à natureza dinâmica dessas séries temporais. Muitas vezes, é possível observar uma mudança significativa no comportamento dos dados em um determinado ponto ou intervalo de tempo que caracteriza a ocorrência de um evento de ponto de mudança. Nesse caso, a análise de séries temporais deve incluir a identificação e detecção de tais eventos. A detecção de pontos de mudança está relacionada a outras áreas proeminentes de monitoramento, como detecção de mudanças de conceito e rupturas estruturais em séries temporais, sendo crucial para aplicações práticas que afetam o dia a dia, como o monitoramento de condições médicas, detecção de mudanças climáticas e análise de atividades humanas. Como resultado, tem sido objeto de intensa pesquisa, especialmente nas áreas de estatística, econometria e mineração de dados. Algoritmos de detecção de pontos de mudança online ou em tempo real são executados em conjunto com o processo de monitoramento, processando cada ponto de dados à medida que se torna disponível. A detecção de pontos de mudança online é um problema fundamentalmente diferente e desafiador que aumenta a demanda por técnicas avançadas de aprendizado de máquina para mudanças de conceito em séries temporais com grandes fluxos de dados. Geralmente, as soluções baseadas em regressão exigem a comparação de modelos paramétricos fixos e observações suficientes nos dados de entrada. Em contraste, as soluções baseadas em desvios da previsão de séries temporais podem ser afetadas por propriedades não-estacionárias e conceitos que mudam gradualmente, o que pode aumentar os atrasos na detecção. Esta pesquisa aborda os desafios apresentados pelas abordagens baseadas em previsão, explorando a hipótese de que a detecção de pontos de mudança online sobre a previsão defasada da componente inercial de uma série temporal contribui para menores atrasos na detecção e robustez à não-estacionariedade em comparação com abordagens atuais. Para confirmar isso, esta pesquisa propõe o *Herald*, um novo método de detecção de pontos de mudança online baseado na previsão defasada da inércia intrínseca de uma série temporal. Os resultados experimentais indicam que o Herald pode detectar pontos de mudança com menores atrasos na detecção e maior sensibilidade do que os métodos de referência para detecção de pontos de mudança. Os efeitos são especialmente relevantes em séries temporais não-estacionárias com alta volatilidade.

Palavras-chave: séries temporais; detecção online de eventos; ponto de mudança; inércia; previsão; atraso de detecção; rupturas estruturais; desvio de conceito; avaliação suave

ABSTRACT

Online Event Detection over Nonstationary Time Series

Monitored processes of almost every field produce time series. Given the increasing volume of generated data and bandwidth observed in the last decades, most of these time series evolve in an online, real-time, or streaming scenario. However, most currently available methods and state-of-the-art machine-learning techniques are not adapted to the dynamic nature of these time series. It is often possible to observe a significant change in data behavior at a certain point or time interval that characterizes the occurrence of a change point event. In that case, time series analysis must include identifying and detecting such events. Change point detection is related to other prominent areas of monitoring of concept drifts and time series structural breaks, being crucial for practical applications affecting day-to-day life, such as medical condition monitoring, climate change detection, and human activity analysis. Consequently, it has been the subject of intensive research, especially in statistics, econometrics, and data mining. Online or real-time change point detection algorithms run concurrently with the monitoring process, processing each data point as it becomes available. Online change point detection is a fundamentally different and challenging problem that increases demand for advanced machine-learning techniques for concept drifts in large time series. Generally, regression-based solutions demand comparing fixed parametric models and enough observations in incoming data. In contrast, solutions based on deviations from time series prediction can be affected by nonstationary properties and gradually changing concepts, which may increase detection lags. This research addresses the challenges posed by prediction-based approaches, exploring the hypothesis that online change point detection over the lagged prediction of the inertial component of a time series contributes to smaller detection lags and robustness to nonstationarity compared to current approaches. To confirm it, this research proposes the *Herald*, a novel online change point detection method based on the lagged prediction of a time series intrinsic inertia. Experimental results indicate that Herald can detect change points with smaller detection lags and higher sensitivity than baseline methods for change point detection. Effects are especially relevant in nonstationary time series with high volatility.

Keywords: time series; online event detection; change point; inertia; prediction; detection lag; structural change; concept drift; soft evaluation

LIST OF FIGURES

Figure 1 – Approaches to online time series change point detection 20

Figure 2 – TSPred functionality modules 28
Figure 3 – General diagram of Harbinger framework 31
Figure 4 – Motivating example 34
Figure 5 – General idea of soft evaluation 36
Figure 6 – Research trajectory 38

Figure 7 – Sub-sequences and sliding windows of time series 40
Figure 8 – Taxonomy of event detection techniques 46
Figure 9 – SCP strategy illustration 51
Figure 10 – Illustration of algorithm memory approaches 57
Figure 11 – ROC curves for detection evaluation 62
Figure 12 – Event detection from time series prediction 64

Figure 13 – General steps to the Herald method 79
Figure 14 – Intuition behind Herald method 81

Figure 15 – General steps to the comparative analysis 92
Figure 16 – Nonstationary synthetic series in dataset CP1 95
Figure 17 – Nonstationary synthetic series in dataset CP2 96
Figure 18 – Example plots of detection produced by Herald, EDfP, and EDfR 101
Figure 19 – Statistical analysis of Herald against EDfR 103
Figure 20 – Statistical analysis of detection lags in observations and time 104
Figure 21 – Ablation study of methodology components 105
Figure 22 – Sensibility analysis of Herald parameters 107
Figure 23 – Sensibility analysis of Herald memory parameter 108
Figure 24 – Statistical analysis of SoftED metrics for Herald against EDfR 111
Figure 25 – Example plots of detection produced by Herald and EDfR 112

Figure 26 – Sensibility analysis of Herald parameters to SoftED metrics 113

Figure 27 – Auxiliary plots for comprehension of SoftED 145

LIST OF TABLES

Table 1 – Execution parameters and artifacts produced by *Harbinger* 32
Table 2 – Formulas of SoftED Metrics 36

Table 3 – Standard event detection performance metrics 60
Table 4 – Standard confusion matrix for event detection 60

Table 5 – Method input parameters 80
Table 6 – Artifacts produced by the method 81

Table 7 – Experimental parameterization 98

Table 8 – Definition of main variables for the formalization of SoftED 144
Table 9 – Formalization of SoftED Metrics 149

LIST OF ALGORITHMS

Algorithm 1 – Proposed Herald instantiation 87

Algorithm 2 – Computing soft scores for each detection 148

LIST OF ABBREVIATIONS

AIC	Akaike Information Criterion
AN	Adaptive Normalization
ARIMA	Autoregressive Integrated Moving Average
AUC	Area Under The Curve
AUTOML	Automated Machine Learning
BCT	Box-Cox Transform
BIC	Bayesian Information Criterion
CNN	Convolutional Neural Networks
CRAN	The Comprehensive R Archive Network
DIF	Simple Differencing
ELM	Extreme Learning Machines Network
EMD	Empirical Mode Decomposition
ETS	Exponential Smoothing State Space Model
FT	Fourier Transforms
GARCH	Generalized Autoregressive Conditional Heteroscedasticity Model
HHT	Hilbert-Huang Transform
HW	Holt-Winter's Exponential Smoothing
IMFS	Intrinsic Mode Functions
LMM	Linear Mixed Model
LSTM	Long Short-term Memory Neural Network
LT	Logarithmic Transform
MAPE	Mean Absolute Percentage Error
MAS	Moving Average Smoother
MLM	Machine Learning Methods
MLP	Multilayer Perceptron Network
MM	Min-max Normalization
MSE	Mean Square Error
NMSE	Normalized MSE
NNET	Feed-forward Neural Network

PCT	Percentage Change Transform
RBF	Radial Basis Function Network
RFRST	Random Forest Regression
ROC	Operating Characteristic Curve
SM	Structural Models
SMAPE	Symmetric MAPE
SVM	Support Vector Machine
TF	Theta Forecasting
THW	Taylor's Double Seasonal Holt-Winter's Model
WT	Wavelet Transform

CONTENTS

Introduction **17**

Online time series change point detection 19

Challenges to online change point detection 21

Trend analysis for online nonstationary time series change point detection 22

Outline 24

1 Preliminaries **26**

1.1 Preliminary scientific products 26

1.1.1 TSPred framework 26

1.1.2 Harbinger framework 30

1.1.3 SoftED metrics 33

1.2 Research trajectory 37

2 Time series event detection background **39**

2.1 Introduction to time series analysis 39

2.1.1 Nonstationarity 41

2.1.2 Time series prediction 44

2.2 Time series event detection 45

2.2.1 Time series events 46

2.2.2 Event detection 48

2.2.3 Online event detection 55

2.2.4 Event detection performance metrics 58

2.3 Time series event prediction 61

2.3.1 Event detection from time series prediction 63

2.3.2 Online event detection from time series prediction 65

3 Related works **66**

3.1 Event detection strategies 67
3.2 Advanced time series analysis 68
3.3 Types of detected events 69
3.4 Event detection from streaming data 71
3.5 Event detection performance metrics 72
3.6 Event detection from time series prediction 74
3.7 Related surveys 75
3.8 Considerations 77

4 Herald 78
4.1 Method intuition 79
4.2 Online detection process 82
4.2.1 Time series decomposition 82
4.2.2 Prediction of time series inertial component 83
4.2.3 Online event detection from time series prediction 84
4.2.4 On the use of frameworks 85
4.3 Proposed instantiation 86

5 Experimental evaluation 90
5.1 Comparative analysis 90
5.2 Experimental settings 93
5.2.1 Datasets 93
5.2.2 Implementation definitions and parameterization 96
5.2.3 Evaluation metrics 98
5.2.4 Significance tests 99
5.3 Results and discussion 100
5.3.1 Detection lag analysis 101
5.3.2 Detection performance analysis 109

Conclusion 114
Summary and reflection on main research findings 115
Future research strands and applications 117
Overview of scientific production 119

References 123

A SoftED metrics **143**
A.1 Metric formalization 143
A.1.1 Defining an event membership function 144
A.1.2 Maintaining integrity with hard metrics 146
A.1.3 Computing the SoftED metrics 149

Introduction

Monitored processes of almost every scientific field evolve over time, leading to a collection of sequential data referred as time series. In case a studied process demands continuous monitoring, the time series is collected uninterruptedly, in a continuous flow, characterizing a streaming of time series data. Given the increasing volume of generated data and bandwidth, most of future time series analysis works will most likely be performed over data streams. However, most of currently available methods are not particularly adapted to the dynamic nature of streaming time series [Esling and Agon, 2012].

The majority of real-world streaming time series are particularly susceptible to nonstationarity, where its statistical properties, such as mean, trend, variance, and even the frequency and distribution of the observations may vary over time [Shumway and Stoffer, 2017; Gujarati, 2021]. In fact, it is often possible to observe the occurrence of a significant change in the behavior of a time series at a certain point t or time interval $\{t_{beginning}, \ldots, t_{end}\}$. Such abnormal behavior change commonly characterizes the occurrence of an event at t or $\{t_{beginning}, \ldots, t_{end}\}$, respectively [Guralnik and Srivastava, 1999]. Unlike outliers or noisy data, this event represents a phenomenon with a specific meaning to a certain domain. For example, it can represent a breach of security, or a failure in an industrial process.

Generally, the occurrence of an event can be detected by the observation of anomalies and change points. An anomaly is a rare pattern or an observation that deviates from most other observations as if it was generated by a different phenomenon [Wang et al., 2019]. A change point, in the other hand, is a point in time, or a time interval, that marks a different state in the process that generates the time series [Aminikhanghahi and Cook, 2017]. Generally, the subsequences that precede and succeed a change point present different inherent functions (concepts) [Iwashita and Papa, 2019].

Event detection is the process of identifying events in time series data. With this process we may be interested in learning past events (offline) [Pimentel et al., 2014; Ding et al., 2014; Ahmed et al., 2016; Alevizos et al., 2017; Sebestyen et al., 2018; Zhou et al., 2019; Pang et al., 2021; Wang et al., 2019], identifying events in real-time (online) [Zhang et al., 2010; Ahmad et al., 2017; Ariyaluran Habeeb et al., 2019; Munir et al.,

2019], or even predicting future events before they happen [Yan et al., 2004; Salfner et al., 2010; Molaei and Keyvanpour, 2015; Gmati et al., 2019; Zhao, 2021]. In fact, it is recognized as a basic function in surveillance and monitoring systems and has gained much attention in research for application domains involving large datasets from critical systems [Pimentel et al., 2014]. In this context, the event detection problem becomes particularly relevant, especially for applications based on sensor data analysis. Examples of such applications can be observed in the domains of chemistry, reflection seismic, and oil drilling and exploration, where monitoring of operations is essential. Moreover, event detection is crucial for practical applications affecting day-to-day life such as medical diagnostic problems, fault detection in industrial systems, building structural damage identification, intrusion detection in electronic security systems, credit card fraud detection, sensor networks and astronomy catalogues, etc [Pimentel et al., 2014].

Generally, event detection methods found in the literature specialize in identifying a specific type of event. Although a considerable number of methods focus on identifying anomalies in time series [Provotar et al., 2019; Park et al., 2019; Zhao et al., 2020], change point detection has been the subject of intensive research in the fields of statistics, econometrics, and data mining [Aminikhanghahi and Cook, 2017]. Moreover, change point detection is related to other prominent areas of monitoring of concept drifts [Sayed-Mouchaweh and Lughofer, 2012; Gama et al., 2014; Ditzler et al., 2015] and time series structural breaks [Zeileis et al., 2010]. In fact, change points may also be described as the points in which we observe a drift in concept, or a structural break. The detection of change points can be particularly useful, not only as an end result, but also as a reference and insight to other tasks of time series analysis such as modeling and prediction. As a result, change point detection is used in many application areas such as medical condition monitoring, climate change detection, and human activity analysis [Aminikhanghahi and Cook, 2017]. A wide array of change point detection methods have been developed in the literature, while many of them focus on detecting breakpoints where there is a change in the time series inherent trend [Zeileis et al., 2010].

Online change point detection is the problem of detecting change points in streaming time series data. While offline methods consider the entire time series dataset to identify where change points occurred, online change point detection methods run concurrently with the monitored process. Each data observation is processed as it becomes available and it is detected as an event (or not) in real-time, or as soon as possible

[Aminikhanghahi and Cook, 2017]. The next subsection details the problem of online change point detection over streaming time series.

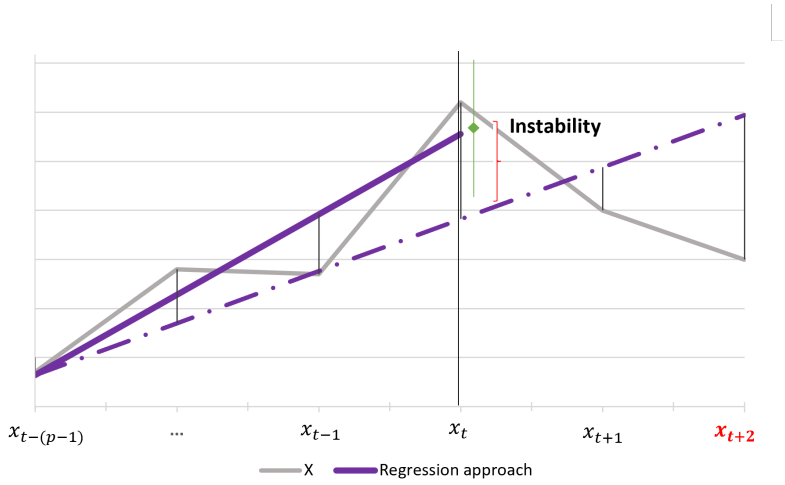
Online time series change point detection

The idealistic characteristics of an online change point detection method include: (i) observation x_t must be identified as normal or event before receiving the subsequent x_{t+1}; (ii) the method must learn continuously; (iii) it must not require access to the whole stream; (iv) it must be automated and run in unsupervised data; (v) it must adapt to dynamic and nonstationary environments with occasional concept drifts; (vi) it should make event detections as early as possible, while minimizing false positives and false negatives [Ahmad et al., 2017].

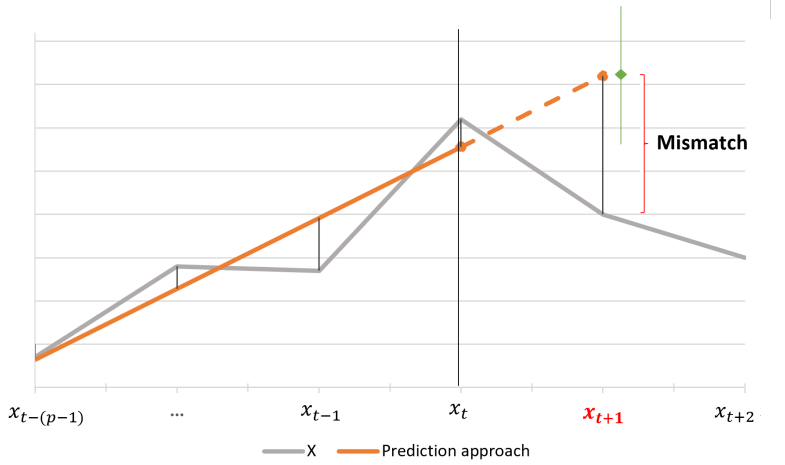
Online change point detection is a fundamentally different problem and the majority of available time series event detection methods do not conform to this characteristics [Gama et al., 2014; Ariyaluran Habeeb et al., 2019]. In fact, currently no change point detection methods run in perfect real-time, that is, without detection lags. This is due to the fact that they must inspect the new time series observation before determining if a change point occurred between the previously analyzed data points and the newly arrived [Aminikhanghahi and Cook, 2017].

Nonetheless, the developed online detection methods aspire to require increasingly less new observations (time) before a change point can be detected [Ariyaluran Habeeb et al., 2019]. Most frequently, online time series change point detection methods adopt one of two approaches: (i) monitoring parameter instabilities in parametric regression models [Zeileis et al., 2002], or (ii) predicting future states of the time series to identify mismatches from expected behavior [Salfner et al., 2010; Pimentel et al., 2014]. Figure 1 illustrates both approaches, respectively.

The first approach, illustrated in Figure 1a, is based on monitoring parameter instabilities in incoming data. As new observations arrive, parametric regression model estimates are computed from all available data and compared to the estimates based only on a historical sample [Chu et al., 1996]. A change point is detected if the difference between these estimates is above an acceptable threshold [Zeileis et al., 2010]. This



(a) Regression approach



(b) Prediction approach

Figure 1 – The main approaches adopted by online time series change point detection methods

regression approach is established and widely adopted in the context of econometrics and statistics [Zeileis et al., 2010].

The latter approach, illustrated in Figure 1b, has been mainly studied in the context of the very recent and prominent area of event prediction [Zhao, 2021]. It is based on event detection from time series prediction, which involves two steps. The first step regards time series prediction. Past values of the time series are used for deriving a predictive model, which can be statistical or machine learning-based. Next, the derived model is applied for time series prediction and future values of the time series are returned. The second step concerns online event detection per se. Change points are identified based on the deviations from the predicted time series above an acceptable threshold

[Mehrmolaei and Keyvanpourr, 2015]. The detection process in this *prediction approach* involves continually analyzing the current state to predict expectations for the next future streaming observations. In this case, the occurrence of a change point is detected by the mismatch of expected patterns.

Challenges to online change point detection

The detection methods that adopt a regression approach are designed to monitor particular properties of a time series, being more or less suitable for certain patterns of deviation from expected behavior [Zeileis, 2005]. In particular, it shares the issue with parametric methods regarding the sensitivity to the choice of regression model curve and initial parameter values [Aminikhanghahi and Cook, 2017]. Modeling may also be affected by high volatility levels and nonstationary properties such as heteroscedasticity, which are not addressed by most time series modeling methods [Aminikhanghahi and Cook, 2017; Salles et al., 2019]. Moreover, this approach demands a sufficient number of observations in incoming data in order to affect the differences in model estimates above the adopted threshold. Finally, in the presence of incremental concept drifts [Lu et al., 2018], model estimates may vary slowly making it difficult to quickly detect fluctuations, increasing detection lag [Zeileis, 2005]. If this lag does not conform to acceptable time within an application domain, a regression method is deemed inadequate [Molaei and Keyvanpour, 2015].

In this context, a prediction approach can be advantageous to online time series change point detection. The application of a detection method that adopts a prediction approach demands the availability of a time series that is just long enough to train a predictive model with the required accuracy, which in turn can be non-parametric and not make assumptions regarding time series characteristic curves. The trained model is then used to guide the identification of emerging abnormal observations, analyzing each new observation for deviations of expected behavior. However, real-world dynamic environments and complex processes can continually suffer from concept drifts and changes in distribution. Hence, there is a demand to periodically upgrade predictive models, based incremental/adaptive learning [Zhao, 2021].

Nevertheless, methods that adopt a prediction approach are still affected by several challenges. Since identification of change points is based on significant mismatches from expected behavior, it can be highly affected by the acceptable threshold, which must be carefully defined. Generally, the threshold is learned from data distribution or defined by expert domain knowledge. The predicted expectations for the time series are also highly dependent on the accuracy of the trained predictive model, which in turn is impacted by improper time series preprocessing. This becomes particularly challenging in the context of nonstationary time series with high volatility, which demand advanced time series analysis techniques. Furthermore, given an incremental learning process with high plasticity, the predictive model can continually absorb emerging abnormal behavior as part of the time series inherent function. Consequently, gradual changes in behavior can go unnoticed for a potentially long time. All these challenges ultimately lead to lags in change point detection, specially in the presence of incremental concept-drifts.

Assuming the scenario where a researcher is faced with an online change point detection application, where reaction response is critical, based on a streaming nonstationary time series dataset prone to the occurrence of incremental concept-drifts. Such online detection application demands (i) detection accuracy; and (ii) early detection of change points, that is, with small detection lags. This research aims to address the challenges to the adoption of a prediction approach as a solution to the problem of online change point detection described by this scenario.

Trend analysis for online nonstationary time series change point detection

Time series preprocessing can increase the prediction and subsequent detection accuracy of an adopted method [Han et al., 2022]. In particular, in the context of nonstationary time series, it is important to adopt advanced time series analysis techniques for data preprocessing prior to the steps of modeling and prediction. Among them, the splitting-based transformation methods [Salles et al., 2019] are drawing increasingly more attention [Chiroma et al., 2016]. However, their applications are still under-researched, specially for the problem of event detection [Nachane and Clavel, 2008; Yang and Zurbenko, 2010; Rocha et al., 2011].

Splitting-based transformation methods are based in signal processing techniques and perform the decomposition of a time series into components of the same length having different frequencies (scales). High-frequency components represent noise and volatility of the time series, seasonal components represent cyclic patterns, and low frequency components represent its inherent trend [Shumway and Stoffer, 2017]. Each component may be caused by different physical phenomena, and their collection can capture the intrinsic dynamics of the time series [Morana, 2007; Shu et al., 2014; Xi et al., 2014; Lahmiri, 2016].

Time series decomposition derives components that are more physically meaningful and generally simpler to explain. Each time series component can be more easily modeled, which can simplify the prediction problem and contribute to increased prediction accuracy [Dudek, 2016]. Particularly, the trend component of a time series serves as a proxy of the inertia intrinsic to its generating process. It maintains the memory of the undergoing behavior of the process until the occurrence of an event that causes enough impact to change its trend [Gujarati, 2021]. Therefore, it is less susceptible to volatility and incremental concept-drifts in the time series, serving as a simpler and more stable input for time series prediction methods.

The prediction of the trend component of a time series mitigates the effects of spurious values and, consequently, it can lower the demand for frequent model updates in incremental learning systems with high plasticity [Gama et al., 2014; Ditzler et al., 2015]. Ultimately, it may prevent the predictive models to absorb emerging abnormal behavior as part of the inherent function of the process. Moreover, the lagged analysis of the trend component may also contribute to the identification of incrementally changing concepts. The observation of previous behavior can enable a broader understanding of the current state of the time series compared to what is expected from its inertia [Gujarati, 2021]. In that sense, starting the prediction horizon of the trend component at a lagged time $t - h$ enables a more comprehensive analysis of time series observations surrounding and including current time t regarding their deviation from expected behavior. In turn, it may increase sensibility to incremental concept-drifts and help anticipate its identification.

Given the aforementioned discussion, this thesis hypothesizes that the analysis of the lagged prediction of time series inertia brings contribution to early online change point detection when compared to current methods based on either a prediction or regression approach. Hence, this research proposes *Herald*, a novel prediction approach method for

online change point detection based on the use of time series decomposition for deriving its characteristic trend component as a proxy of its intrinsic inertia. The derived trend component is given as input to a time series prediction process with a lagged prediction horizon. The predicted inertial component is expected to empower early detection of change points in streaming time series presenting nonstationary properties and concept drifts.

To confirm this hypothesis, Herald is expected to be able to detect change points in nonstationary time series, producing competitive results in detection accuracy and lower detection lags while being robust to incremental concept-drifts. For validating these expectations, this thesis experimentally evaluates the Herald approach against methods based on the traditional prediction approach, referenced as *EDfP*, and the regression approach, referenced as *EDfR*, adopted as a baseline. Herald is evaluated regarding its ability to outperform *EDfP* and *EDfR* methods regarding both early and accurate online detection of change points. The methods are applied to a simulated nonstationary times series dataset containing change point events to help further a more thorough discussion on the effects of the evaluated detection methods on a diverse range of representative time series.

Experimental results indicate that the Herald method contributes to earlier change point detection with statistical significance, especially in nonstationary time series with high volatility. Herald provides lower detection lags for 69% of the high volatility time series ($145/210$), corresponding to an effect size of 0.38 (large [Funder and Ozer, 2019]) compared to the baseline *EDfR*. On the other hand, the *EDfP* method had lower detection lags for 4.5% of the time series compared to the baseline. Ablation experiments also indicated the effect of both inertial analysis and lagged prediction horizons on lowering detection lags. Furthermore, for 89.5% of all time series (effect size 0.79) the Herald method had a higher recall metric, producing closer detections to change point events. In summary, Herald is able to detect change points earlier and with higher sensitivity, that is, demanding fewer observations of the time series stream and giving detections that are more representative (closer) of the events. Since the findings of the proposed experimental evaluation conform to expectations, the thesis hypothesis can be confirmed and the Herald method is to be made widely available for online time series prediction.

Outline

Besides this introduction, the remainder of this work is organized as follows. Chapter 1 describes preliminary results that are relevant to this thesis. They are detailed and formalized in the works of Salles et al. [2020], Escobar et al. [2021], and Salles et al. [2023b]. The research trajectory that culminated in this research is also presented.

Chapter 2 provides theoretical foundation on time series analysis, specially regarding the area of time series event detection. It presents (i) the definition of time series and their general concepts, including nonstationarity; (ii) an introduction to the problem of time series prediction; (iii) a definition of the problem of event detection and online event detection, as well as a taxonomy for event characteristics and detection techniques; (iv) a description of relevant event detection methods and its evaluation metrics; and (v) an introduction to the area of time series prediction and how it contextualizes with the proposed approach.

Chapter 3 discusses related works on time series event detection and detection performance metrics. It also contextualizes the proposed approach among current approaches to event detection from time series prediction. Finally, other related surveys are reviewed. Next, Chapter 4 formalizes the steps to the proposed Herald approach for online event detection. The plan for experimentally evaluating the Herald approach is presented in Chapter 5, including the adopted domains of application and experimental settings. Finally, Chapter 5.3.2 concludes by giving final considerations on the achieved results of this research. It also includes expected and published scientific papers.

1- Preliminaries

This chapter presents the main preliminary scientific products generated by this research that are relevant to this thesis and the development of the Herald method. Moreover, the research trajectory that culminated in this thesis is also presented. It provides a contextualization of the developed scientific products and their direct relation to the idealization, implementation, and evaluation of the Herald method.

1.1- Preliminary scientific products

The implementation and experimental analysis of Herald demand the application of established methods and techniques currently available in literature corresponding to the areas of (i) time series decomposition; (ii) time series prediction; and (iii) event detection. Moreover, online time series event detection demands specialized metrics to enable the comparison of algorithms. This research addressed these demands by providing time series prediction and event detection frameworks [Salles et al., 2022, 2020], as well as novel metrics for detection evaluation. These preliminary results are briefly described in this section and formalized in the works of Salles et al. [2017], Salles et al. [2019], Salles et al. [2022], Salles et al. [2020] and Salles et al. [2023b]. The next subsections are derived from relevant parts of these published works.

1.1.1- TSPred framework

The prediction of time series has gained more attention among researchers. Many time series prediction methods have been developed and can be found in the literature [Cheng et al., 2015]. Several state-of-the-art methods are based on machine learning, especially for long and potentially high-frequent time series. Unfortunately, most of these

methods tend to be optimistic regarding their assumptions over the time series.

A common assumption is the property of stationarity [Cheng et al., 2015]. In a stationary time series, statistical properties (mean, variance and covariance) do not change over time, which is not common for most real datasets. Generally, nonstationarity demands transformation methods to coerce the time series into stationarity [Salles et al., 2019]. Such methods can have different features depending on the implementation of the data properties they consider. Choosing an adequate method for a particular time series application is not a simple task.

Additionally, nonstationarity can lead to exploring many data transformation and prediction methods for building prediction models. Moreover, the choice of an appropriate transformation setup for a particular time series is not straightforward. The benchmarking of different candidate combinations helps select an appropriate setup for predicting a nonstationary time series.

In this context, this research presented *TSPred*, a framework for nonstationary time series prediction. The *TSPred* framework has been previously developed for nonstationary time series prediction and has recently been upgraded [Salles et al., 2022]. It specializes in integrating data transformation methods and machine learning methods (MLM) to predict long time series with potentially high frequency. *TSPred* is made available as an R-package [Salles and Ogasawara, 2021]. It is the first tool to seamlessly integrate a broad range of transformation methods [Salles et al., 2019] and state-of-the-art statistical and machine learning prediction methods for addressing nonstationary time series. The package automates the time series prediction process and parameterization while also enabling user-defined prediction methods and data transformations. The features provided by *TSPred* are shown to be competitive regarding time series prediction accuracy.

The methods regarding time series decomposition and time series prediction demanded by Herald can be defined from the available implementations in the TSPred framework. More details on the methods available in TSPred are given by Salles et al. [2017], Salles et al. [2019], and Salles et al. [2022]. The next Section 1.1.1 presents a general idea of the main functionality modules of *TSPred*.

Framework and Functionalities The main functionality modules representing the concept and structure of the framework are depicted in Figure 2. Together, they represent a particular time series prediction application and encapsulate the process described

in Section 2 and all the elements and steps necessary to fulfill it. *TSPred* has three main functionality modules: *Preprocessing & Postprocessing*, *Modeling & Predicting*, and *Evaluating*.

The first module is responsible for preprocessing/transforming and postprocessing/reverse transforming time series. It includes the implementation of the main nonstationary time series transformation methods [Salles et al., 2019], being either mapping-based, namely the logarithmic transform (LT), Box-Cox transform (BCT), percentage change transform (PCT), moving average smoother (MAS), and simple differencing (DIF), or splitting-based, such as empirical mode decomposition (EMD) and wavelet transform (WT). Furthermore, it implements other relevant preprocessing methods for time series prediction with MLM: partitioning time series data into training and evaluation datasets, subsetting the time series data into sliding windows, handling of missing values, and data normalization via Min-max normalization (MM) and Adaptive normalization (AN) methods [Ogasawara et al., 2010].

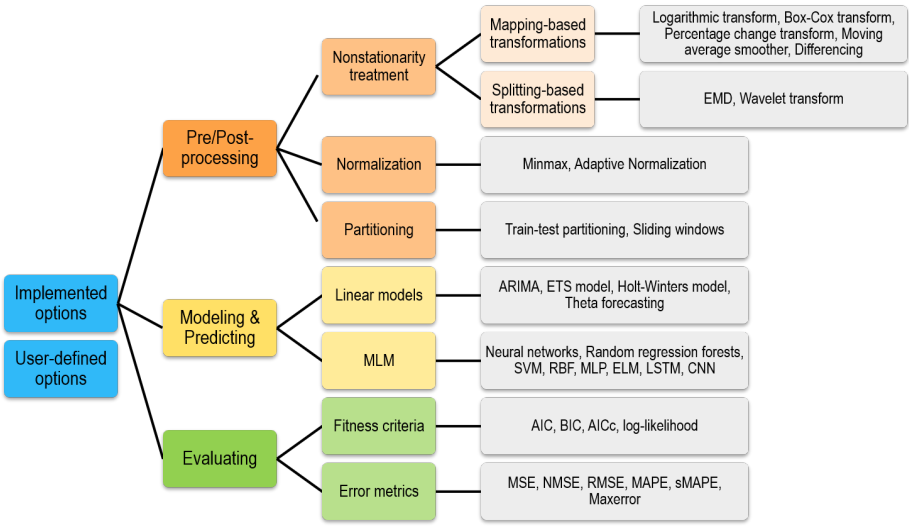


Figure 2 – TSPred functionality modules and pre-implemented algorithms

The *Modeling & Predicting* module is responsible for modeling and predicting a time series based on a particular time series model. These tasks are specialized for either statistical or machine learning models. For the latter, the framework is prepared to perform any necessary machine learning life-cycle tasks during the training and prediction steps, including coercing data into sliding windows and performing normalization and transformation of the input data. This module includes the implementation of the statistical models: autoregressive integrated moving average (ARIMA), Holt-Winter's exponential smoothing

(HW), theta forecasting (TF), and exponential smoothing state space model (ETS). *MLM* models include feed-forward neural network (NNET), random forest regression (RFrst), radial basis function network (RBF), support vector machine (SVM), multilayer perceptron network (MLP), and extreme learning machines network (ELM). Furthermore, the module provides deep learning models available in the *TensorFlow* library, namely Convolutional Neural Networks (CNN) and long short-term memory neural network (LSTM).

Finally, the *Evaluating* module is responsible for assessing the model fitness and quality of predictions for the given time series based on a particular metric. These tasks are specialized for computing either prediction accuracy (error) measures or model fitting criteria. Among the available prediction accuracy measures, it includes Mean Square Error (MSE), symmetric MAPE (sMAPE), and maximal error. It also includes model fitness criteria such as Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), and log-likelihood [Davydenko and Fildes, 2013].

TSPred can integrate the described modules in a *workflow*, encompassing the five steps described in Section 2. Ultimately, the package provides the means to perform the *benchmarking* of several prediction models. It is important to remark that although providing several pre-implemented options, *TSPred* design enables the user to define and apply customized time series prediction methods.

Moreover, the package provides several automatized features that can be useful for any time series prediction application. Among them, some of the main features are: (i) seamless recursive combination of two or more transformation methods; (ii) seamless integration of splitting-based transformation methods to the prediction process [Salles et al., 2019], which demands the combination of predictions for each component resulting from data decomposition (first package to include this approach); (iii) transformation/model parameter selection; (iv) multistep-ahead or one-step-ahead predictions; (v) rolling origin evaluation [Hyndman and Athanasopoulos, 2018] for both statistical and machine learning models; and (vi) machine-learning life-cycle tasks performed during training and prediction steps. It means that data normalization and transformation of sliding windows are seamlessly conducted during machine learning model training.

The framework is implemented in R using the S3 class system [Wickham, 2019]. *TSPred* is currently in version 5.1 and is available at The Comprehensive R Archive Network (CRAN). For more implementation details, please refer to the *TSPred* documentation [Salles and Ogasawara, 2021] and paper [Salles et al., 2022].

1.1.2- Harbinger framework

The literature presents many methods for event detection. Choosing an appropriate method for a time series is not a simple task. The selection and parameterization of a method are directly related to the initial assumptions about the data's behavior and statistical properties [Fontugne et al., 2010; Fanaee-T and Gama, 2014]. The complexity of this task is aggravated by the fact that the nature of events observed in a time series is often unknown. Events detected in time series commonly appear as anomalies, change points, or frequent patterns. In this case, methods specialized in detecting one type of event may overlook the occurrence of another or even identify them erroneously. Failures in event identification can affect decision-making or lead to false positives. This results in a loss of credibility in the use of control techniques and potential harm to applications.

In this context, this research produced *Harbinger*, a framework for integrating and analyzing event detection methods. To achieve this, some of the main event detection methods available in the literature are standardized and integrated, and users can also include their own methods. This integration enables a comparative qualitative and quantitative analysis of their detections, as well as the combined use of methods to characterize detected events more accurately. Harbinger can accommodate user-defined detection methods while also including the implementation of several of the main established event detection methods available in literature. Harbinger was evaluated based on synthetic and real data.

The related work and formalization of Harbinger, as well as its experimental evaluation based on synthetic and real data, are given in more detail by Salles et al. [2020] and Escobar et al. [2021]. It was found that its features help to guide the choice of event detection methods and its parameters, as well as understanding the nature of detected events. Furthermore, it can be used to benchmark different event detection methods based on their temporal bias, that is, their tendency to anticipate or delay an event detection. The *Harbinger* framework also has been recently upgraded and is currently available as an R-package [Salles et al., 2023a].

The event detection methods applied by Herald can be defined from the methods available in the Harbinger framework. It is important to remark that the range of methods implemented in both TSPred and Harbinger frameworks were extended on demand during

the development of this research. Moreover, the Herald approach itself is intended to be included in the Harbinger framework. The general structure of *Harbinger* is presented in the next subsection.

Framework structure The general structure of *Harbinger* is illustrated in Fig. 3, consisting of four main functionality modules: event detection (in blue), detection quality assessment (in green), combination of detection results (in orange), and comparison of detection performances (in purple). The description of the execution parameters of *Harbinger*, as well as of its produced artifacts is presented in Table 1.

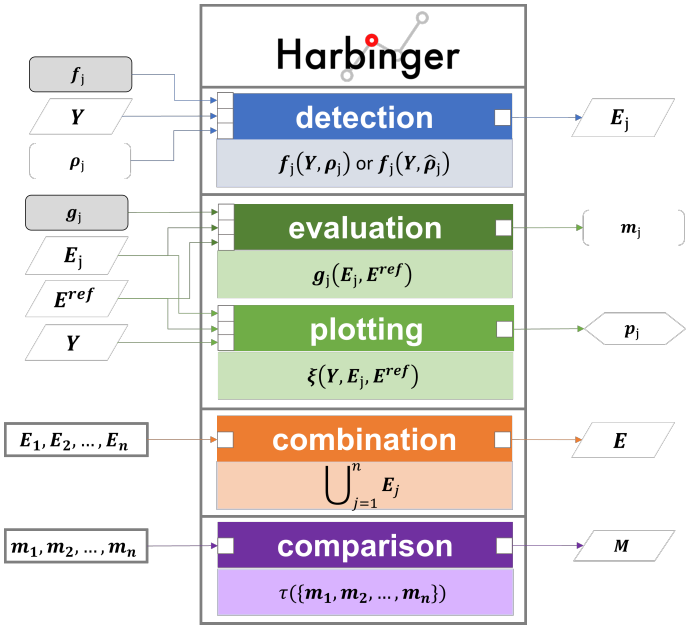


Figure 3 – General diagram of Harbinger framework modules

The process of using *Harbinger* initially includes the j-th instantiation of an event detection process, $O_j =< Y, E^{ref}, f_j, g_j, \rho_j, E_j, m_j, p_j >$. O_j is implemented by the detection and evaluation modules illustrated in the diagram in Fig. 3. For this instantiation, a set of time series Y is passed as input parameters, an optional list of their respective previously classified real events E^{ref}, a detection method f_j, your required parameters ρ_j and a selected metric for evaluating the quality of your detections, g_j (if E^{ref} has been reported).

As result of running a O_j instance, *Harbinger* returns the set of detected events $E_j = f_j(Y, \rho_j)$. The optimization of ρ_j by the ϕ function is also allowed. In this case, $E_j = f_j(Y, \hat{\rho}_j)$, with $\hat{\rho}_j = \phi(Y, f_j, \rho_j)$. E_j is presented to the user containing the time instants in which events were detected, the time series to which the events belong and

Table 1 – Execution parameters and artifacts produced by *Harbinger*

	Source	Class	Object	Description
Execution parameters	Input	Data	Y	Set of time series
			E^{ref}	Set of classified real events
		Functions	f_j	Function of detection method
			g_j	Function of evaluation metric
		Parameters	ρ_j	Set of parameters of f_j
	Pre-defined	Functions	ϕ	Optimization function
			ξ	Plot function
			τ	Ranking function
Produced artifacts	Instance	Data	E_j	Set of detected events
		Value	m_j	Metric value
		Graphic	p_j	Graphic of events
	Set of instances	Data	E	Combined detected events
			M	Ranked metric values

the types of events detected. Then, the quality of the detections in E_j is evaluated by g_j using E^{ref} as a basis for comparison. The computed metric value, $m_j = g_j(E_j, E^{ref})$, is returned. m_j is then used for the quantitative analysis of the quality of O_j detections. In addition, *Harbinger* generates a graphic p_j using the function ξ for visual analysis of E_j taking Y and E^{ref} as references. p_j can then be used for a qualitative analysis of O_j detections.

Finally, the process of using *Harbinger* allows the combination of the executing results of a set $O = \{O_1, \ldots, O_n\}$ of n instances of an event detection process defined in *Harbinger*. This task is illustrated in the diagram combining module of Fig. 3, where *Harbinger* joins the sets of detected events in each j instance, contained in E_j, with $j = 1, \ldots, n$. The E union set is returned containing all detected events in Y by different detection methods and/or different parameters. Consequently, E is capable of encompassing detected events of different types presenting different semantics that are properly identified according to the characteristic methodology of their respective detection methods.

Likewise, the diagram comparison module in Fig. 3 illustrates the comparison of the m_j quality metrics, with $j = 1, \ldots, n$, computed during the executions of the instances of O. The value set $\{m_1, m_2, \ldots, m_n\}$ is submitted to a τ ranking process. The M set of ranked values of quality metrics is returned. M can then be used as a basis for evaluating the n defined instances whose event detection performances are

ordered accordingly. This ordering guides the choice of detection methods and their most appropriate parameterizations for a given event detection application and can be registered in models life cycle management systems [da Silva et al., 2019].

1.1.3- SoftED metrics

To address the task of time series event detection, several methods have been developed and are surveyed in the literature [Hodge and Austin, 2004; Chandola et al., 2009; Gupta et al., 2014; Choudhary et al., 2017; Cook et al., 2020; Braei and Wagner, 2020; Blázquez-García et al., 2021; Schmidl et al., 2022]. Each detection method specializes in time series that present different characteristics or make assumptions about the data distribution. Therefore, the assessment of detection performance is important to infer their adequacy for a particular application [Fontugne et al., 2010]. In this case, detection performance refers to how accurate an event detection method is at identifying events in a time series. Detection performance is generally measured by classification metrics [Han et al., 2012].

Currently, standard classification metrics (around since the 1950s), including Recall, Precision, and F1, are usually adopted [Lavin and Ahmad, 2015; Tatbul et al., 2018]. Although Accuracy is a specific metric [Han et al., 2012], the expression detection accuracy is henceforth used to refer to the ability of a method to detect events correctly. Classification metrics focus mainly on an analysis of detection accuracy. On the other hand, inaccuracy in event detection does not always indicate a bad result, especially when detections are sufficiently close to events.

Motivating example and problem definition This section gives an example of the problem of evaluating inaccurate event detections and defines the problem with respect to the demand for adequate detection performance metrics. Consider, for example, a time series X containing an event at time t, represented in Figure 4. Given detection methods A and B applied to X, a user must select one of them as the most adequate for the underlying application. Method A detects an event at time $t + k_1$, while Method B detects an event at time $t - k_2$ ($k_2 > k_1$). As none of the methods could correctly detect

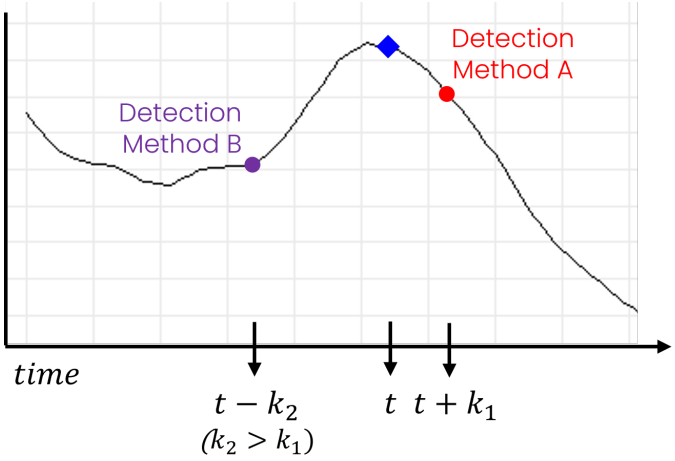


Figure 4 – Example regarding the problem of evaluating the detection of an event at time t. Method A detects an event at time $t + k_1$, while Method B detects an event at time $t - k_2$ $(k_2 > k_1)$.

the event at time t, based on the usual detection accuracy evaluation, the user would deem both as inaccurate and disposable.

However, inaccuracy in detecting an event can often result from its preceding or lingering effects. Take the adoption of a new policy in a business. While a domain specialist may consider the moment of policy enforcement as a company event, its effects on profit may only be detectable a few months later. On the other hand, preparations for policy adoption may be detectable in the antecedent months. Moreover, when accurate detections are not achievable, which is common, detection applications demand events to be identified as soon as possible [Lavin and Ahmad, 2015], or early enough to allow necessary actions to be taken, mitigating possible critical system failures or helping mitigate urban problems resulting from extreme weather events, for example. In this context, the results of Methods A and B would be valuable to the user. Note that while Method B seems to anticipate the event, its application to X and resulting detection is made after the event's occurrence. On the other hand, the detection of Method A came temporally closer to the event, possibly more representative of its effects.

In this context, evaluating event detection is particularly challenging, and the detection accuracy metrics usually adopted are insufficient and inadequate for the task [Singh and Olinsky, 2017]. Standard classification metrics do not consider the concept of time, which is fundamental in the context of time series analysis, and do not reward early detection [Ahmad et al., 2017], for example, or any relevant neighboring detections. For the remainder of this section, neighboring or close detections refer to detections whose

temporal distance to events is within a desired threshold. Current metrics only reward true positives (exact matches in event detection). All other results are “harshly” and equally discredited.

In this case, there is a demand to soften the usual concept of detection accuracy and evaluate the methods while considering neighboring detections. However, state-of-the-art metrics designed for scoring anomaly detection, such as the NAB score [Lavin and Ahmad, 2015], are still limited [Singh and Olinsky, 2017], while also being biased towards results preceding events. To the best of our knowledge, no metrics available in the literature consider both the concept of time and tolerance for detections that are sufficiently close to time series events. This research focuses on addressing this demand.

Contribution This research introduced the SoftED metrics, a new set of metrics for evaluating event detection methods regarding their detection accuracy and ability to produce neighboring detections to events. Inspired by fuzzy sets, SoftED metrics assess the degree to which a detection represents a particular event. It is an innovative approach since, while fuzzification is commonly applied to time series observations [Hüllermeier, 2011], in SoftED, the fuzzification occurs in the time dimension [Zadeh, 1965]. Hence, they incorporated time and temporal tolerance in evaluating the accuracy of event detection. In fact, these are the scenarios that domain specialists and users often face. And, up until now, there is neither a standard nor adequate evaluation metrics for event detection in such scenarios. SoftED metrics soften the standard classification metrics, which are considered in this research as hard metrics, to support the decision-making regarding the most appropriate method for a given application.

Figure 5 gives a general idea of the proposed approach, illustrating the key difference between the standard hard evaluation and the proposed soft evaluation. Blue rhombuses represent actual time series events. Circles correspond to detections produced by a particular detection method. The hard evaluation concerns a binary value regarding whether detection is a perfect match to the actual event. In this case, circles are green when they perfectly match the events and red when they do not. Conversely, soft evaluation assesses the degree to which detection relates to a particular event.

The scores computed for each detection d_i, $ds(d_i)$, are used to create soft versions of the hard metrics TP, FP, TN, and FN, as formalized in Table 2, where m and n represent the number of events and detections, respectively. Due to imposed constraints, the defined

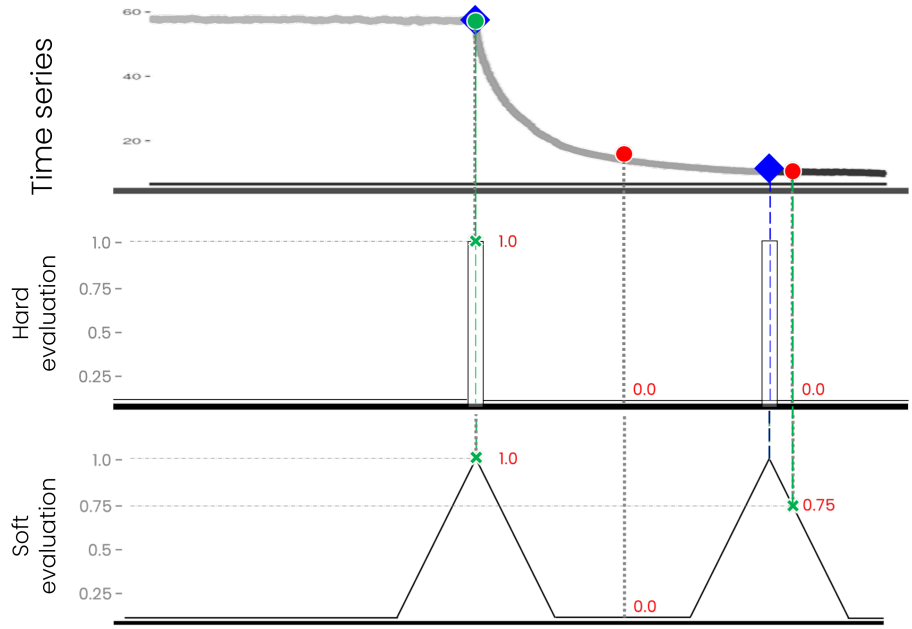


Figure 5 – The general idea behind the proposed approach comparing the standard “hard” evaluation and the “soft” evaluation of the event detection.

SoftED metrics TP_s, FP_s, TN_s, and FN_s, hold the same properties and the same scale as traditional hard metrics. Consequently, using their same characteristic formulas, they can derive soft versions of traditional scoring methods, such as Sensitivity, Specificity, Precision, Recall, and F1. Moreover, SoftED scoring methods still provide the same interpretation while including temporal tolerance for inaccuracy, which is pervasive in time series event detection applications.

Table 2 – Formulas of SoftED Metrics

$\text{TP}_s = \sum_{i=1}^{n} ds(d_i)$	$\text{FN}_s = m - \text{TP}_s$	$\text{Precision}_s = \frac{\text{TP}_s}{n}$	$\text{F1}_s = \frac{2\text{Precision}_s\text{Recall}_s}{\text{Precision}_s + \text{Recall}_s}$
$\text{FP}_s = \sum_{i=1}^{n} (1 - ds(d_i))$	$\text{TN}_s = (\lvert X \rvert - m) - \text{FP}_s$	$\text{Recall}_s = \frac{\text{TP}_s}{m}$	

This research also contributed by introducing a new general protocol for evaluating the performance of metrics in time series event detection. This protocol is designed to evaluate metrics that incorporate temporal tolerance into detection evaluation. It is inspired by a strategy commonly adopted in ontology design. It is based on using competency questions [Uschold and Gruninger, 1996; Noy and Hafner, 1997]. Following the proposed evaluation protocol, computational experiments were conducted to analyze the contribution

of the developed metrics against the usual hard and state-of-the-art metrics [Lavin and Ahmad, 2015]. We evaluated the SoftED metrics based on the devised competency questions. The results illustrate that SoftED can improve event detection evaluation by associating events and their representative (neighboring) detections, incorporating temporal tolerance in over 36% of the conducted experiments compared to usual hard metrics. More importantly, surveyed domain specialists evaluated the contribution of SoftED metrics to the problem of detection method evaluation.

The SoftED metrics and their evaluation protocol are formalized and quantitatively and qualitatively experimentally evaluated by Salles et al. [2023b]. Their implementation has been included in the latest version of the Harbinger R-package [Salles et al., 2023a]. Moreover, a self-contained idea of the SoftED metrics is discussed in Appendix A based on the work of Salles et al. [2023b].

1.2- Research trajectory

The development of the proposed Herald approach encompasses different scientific products generated during the last few years. In particular, time series decomposition is based on nonstationary transformation methods reviewed by Salles et al. [2019]. The derived time series trend component is submitted to a prediction process implemented by the TSPred framework developed by Salles et al. [2022]. In that case, both works are a direct result of a master's research focused mainly on the field of time series prediction, while TSPred has been recently upgraded.

Once predicted, time series observations are analyzed in order to identify mismatches from expected behavior. Relevant deviations from predicted trend are identified and recognized as events by methods implemented in the Harbinger framework [Salles et al., 2020]. Finally, the results obtained by Herald, as well as other state-of-the-art detection methods, can be evaluated and compared with basis on the developed SoftED classification metrics for event detection [Salles et al., 2023b]. Both scientific products are preliminary results of this research that focus on the particular time series analysis problem of event detection and online event detection [Ogasawara et al., 2021].

Figure 6 indicates the research trajectory that led to the proposal and enabled

the development of the Herald approach. It presents the four main scientific products previously generated, namely a review of nonstationary time series methods (P1), the TSPred framework (P2), the Harbinger framework (P3), and the SoftED metrics (P4). Moreover, it depicts how their composition is used in the development of the Herald approach to online time series event detection.

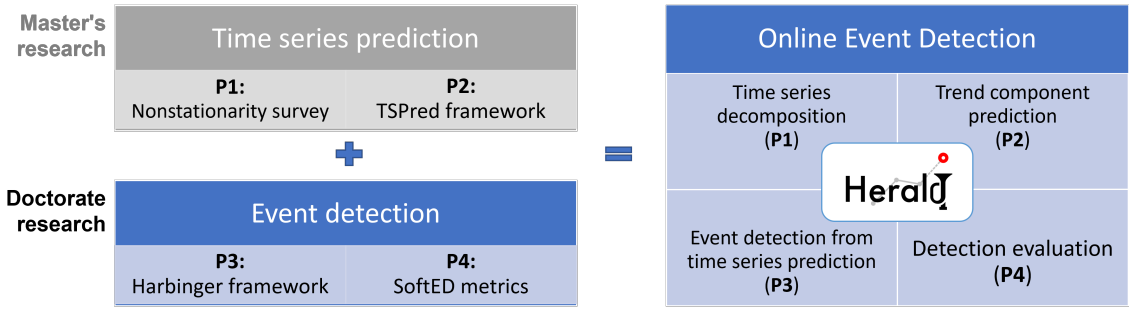


Figure 6 – Research trajectory and its generated products (P1 to P4). The Herald approach is presented as the next step to this research and it encompasses the use of all previously developed scientific products.

2- Time series event detection background

This chapter provides theoretical foundation on time series analysis and event detection. First, Section 2.1 defines time series and their general concepts, including the property of stationarity and how it can be violated, resulting in nonstationarity. Section 2.1 also gives an introduction to the problem of nonstationary time series transformation and decomposition, as well as the problem of time series prediction. Both subjects are thoroughly discussed by Salles et al. [2019] and Salles et al. [2017], respectively.

Subsequently, Section 2.2 provides background on the particular problem of time series event detection and contributes by providing a taxonomy for event characteristics and detection techniques. The different ways an event may present itself in a time series are presented in Subsection 2.2.1. Subsections 2.2.2 and 2.2.3 discuss the strategies generally adopted for event detection and the challenges posed by an online context, respectively. Relevant event detection methods currently available in literature are presented in Subsection 2.2.2 and Subsection 2.2.4 reviews the metrics generally used to evaluate their performance.

Finally, Section 2.3 introduces the broad and very recent area of time series event prediction. Although outside the scope of this thesis, the problem of event prediction helps contextualize the approach adopted by the proposed method. In particular, the adopted approach, based on event detection from time series prediction, has been recently and mainly studied in the context of event prediction. This work, however, employs this approach as a possible solution to online time series event detection.

2.1- Introduction to time series analysis

A time series is a sequence of observations of an object of interest collected over time. When observations are related to a single variable, a time series is referenced as a univariate one. Commonly, the behavior of a univariate time series is studied as a function of its past data [Hanssens et al., 2006]. Generally, one may consider

a univariate time series X as a stochastic process, that is, a sequence of n random variables, <$x_1, x_2, x_3, \ldots, x_n$>, where x_1 represents the value assumed by the series at the first (oldest) time point and x_n represents the value of the series at the newest time point [Esling and Agon, 2012; Shumway and Stoffer, 2017]. The length n of a time series X is represented as $|X|$ and a specific time series observation is referenced as, x_t, indexed in time by $t = 1, \ldots, n$, such that x_1 represents the first observation and x_n portrays the most recent one. A subsequence of size p obtained from X that ends at the i position can be represented by $seq_{i,p}(X)$, which is a continuous sequence of values <$x_{i-(p-1)}, x_{i-(p-2)}, \ldots, x_i$>, where $|seq_{i,p}(X)| = p$ and $p \leq i \leq |X|$. Subsequences enable the analysis of data samples to evaluate local properties [Chiu et al., 2003]. The moving average $\overline{X}_{i,p}$, at i position, of p terms in a time series X is calculated by the average of k observations from the subsequence $seq_{i,p}(X)$, as defined in Equation 1.

$$\overline{X}_{i,p} = p^{-1} \cdot \sum_{k=1}^{p} x_k \mid x_k \in seq_{i,p}(y), 1 \leq k \leq p \tag{1}$$

Sliding windows explore all possible subsequences of a time series [Keogh and Lin, 2005; Lampert et al., 2008]. They produce a set of subsequences of the same length. All sliding windows of size p for a time series X are a function $sw_p(X)$ that produces a matrix W of size ($|X| - p + 1$) by p. Each line w_i in W is the i-th subsequence of size p from X. Given $W = sw_p(X)$, $\forall w_i \in W$, $w_i = seq_{i,p}(X)$ [Borges et al., 2020]. This concept is widely used in time series analysis to make comparisons between subsequences to find their similarities [Van Hoan and Exbrayat, 2013]. Figure 7 depicts a time series X in blue. The red line represents a subsequence from X, and the green dashed lines are examples of subsequences extracted from X based on sliding windows.

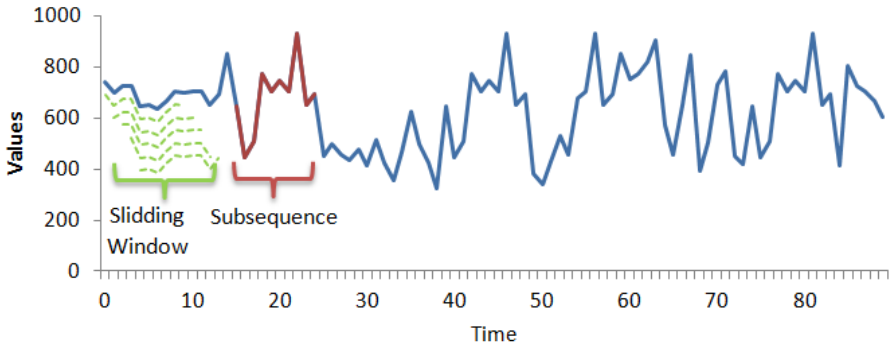


Figure 7 – Sub-sequences and sliding windows of time series [Borges et al., 2020]

2.1.1- Nonstationarity

Most methods applied for time series modeling assume that the behavior of a time series presents a level of regularity over time, which is generally approached with the study of the concept of stationarity [Gujarati, 2021; Shumway and Stoffer, 2017]. The more widely adopted definition of stationarity, namely weak stationarity, gives a milder definition of the property. A weakly stationary time series presents constant mean and variance, and its covariance function depends only on the time difference [Yang and Zurbenko, 2010]. These constraints are very important since they enable statistical inference to be drawn based on any sampled subset of a time series [Hanssens et al., 2006]. As in most works in literature, henceforth the term stationary refers to a weakly stationary process.

If a time series X violates any of the constraints imposed by a stationary process, it is considered a nonstationary time series. Nonstationarity may manifest in many different ways. Generally, it implies that the mean and/or variance functions of a time series are non-constant and vary over time, that is, they are dependent on time t. The changes in mean and/or variance in time series are often due to the occurrence of events, which can result in deterministic trends, structural breaks, level shifts or changing variances (a condition known as heteroscedasticity). They can also be due to the presence of unit roots [Hanssens et al., 2006].

Generally, any form of nonstationarity, if not adequately addressed, can have a relevant impact on time series prediction applications. Overlooking nonstationarity properties in a time series may lead to misleading statistical inferences and bad or unexpected prediction results. In this context, there exist several transformation methods in literature for coping with nonstationarity in times series. Transformation methods for handling nonstationarity in time series can be categorized as mapping-based or splitting-based [Salles et al., 2019].

Mapping-based transforms Mapping-based transformations correspond to methods that map the data of a time series to another representation. They return a new time series generated by passing each of its observations through a mathematical process and the transformed time series usually presents interesting properties, including stationarity. An example of such method is given by MAS [Marrocu, 2006; Ogasawara et al., 2010].

MAS is a mapping-based transformation widely used in finance and econometrics. It is useful for highlighting seasonality and long-term trends in a time series [Shumway and Stoffer, 2017; Ogasawara et al., 2010]. MAS can detect the evolving behavior of a time series by minimizing random noise [Ogasawara et al., 2010], and can also be used for seasonal adjustment of a time series [Marrocu, 2006]. One of the simplest forms of the MAS is represented by Equation 2, where o is the order of the moving average.

$$\hat{x}_t = \frac{1}{o} \sum_{i=t}^{t+o-1} x_i, \;\; 1 \leq t \leq n - o + 1 \tag{2}$$

Splitting-based transforms Splitting-based transformations, on the other hand, refer to methods that are also able to produce a new representation of time series data. Such transformations derive components of a time series which may be caused by different physical phenomena, to capture its intrinsic dynamics [Morana, 2007; Yang and Zurbenko, 2010; Shu et al., 2014; Xi et al., 2014; Lahmiri, 2016]. For example, a time series can be decomposed into short-term (high-frequency), seasonal, and long-term (low frequency) components. Each component series generally serve as simpler or even stationary inputs for time series prediction methods that can analyze and predict them separately [Dudek, 2016]. Most commonly, splitting-based transformations are nonparametric and have its origin in signal processing techniques such as spectral (frequency domain) analysis [Los, 2003; Nachane and Clavel, 2008]. Usually, decomposition may be achieved in a frequency only or a time-frequency domain. The latter enables the preservation of information regarding localized changes. Examples of most prominent splitting-based transformations methods are given by Fourier transforms (FT), WT [Joo and Kim, 2015], EMD [Lahmiri, 2016], and Hilbert-Huang transform (HHT) [Song et al., 2017]. Splitting-based transformation methods are drawing increasingly more attention during the last few decades, especially when associated with computational intelligence techniques [Chiroma et al., 2016]. However, they are still under-researched [Nachane and Clavel, 2008; Yang and Zurbenko, 2010].

Time series analysis in the frequency domain are often based on FT [Joo and Kim, 2015]. The FT creates a frequency-based representation (a spectrum) of a time series in terms of Fourier basis functions. The FT of a time series X, in this case represented as a function of time $x(t)$, can be formulated as in Equation 3, where $F(\xi)$ represents the Fourier spectrum, ξ is a frequency component and j is the imaginary number $j = \sqrt{-1}$ [Los, 2003]. As well as for the Fourier series, the drawback of FT is that it is a global

representation of a time series, and therefore information associated with time is lost [Los, 2003; Joo and Kim, 2015]. Recently, a method capable of efficiently analyzing localized changes in time series has been widely applied, which is based on finite wavelet bases [Los, 2003].

$$F(\xi) = \int_{-\infty}^{+\infty} x(t) e^{-j\xi t} dt, \quad e^{-j\xi t} = \cos \xi t - j \sin \xi t \tag{3}$$

Wavelets are finite basis functions localized in both time and frequency. The WT decomposes a time series (signal) by correlating it with a family of wavelets, providing an extremely flexible time-frequency representation [Los, 2003; Lahmiri, 2016]. The WT decomposes a time series X, which is again regarded as a function of time $x(t)$, into the wavelet series $\hat{x}(t)$ observed in Equation 4 [Joo and Kim, 2015]. Here the components $\zeta(t)$ and its coefficient b represent the scale part of the wavelet series (being responsible for modeling trends and seasonality), whereas the components $\psi_(t)$ and its coefficient c represent the detail part of the wavelet series (corresponding to noise or random deviations), at scale (decomposition level) l and position q. Also, L is the defined maximum decomposition level.

$$\hat{x}(t) = \sum_{q=1}^{n} b_{l,q} \zeta_{l,q}(t) + \sum_{l=1}^{L} \sum_{q=1}^{n} c_{l,q} \psi_{l,q}(t) \tag{4}$$

Since finite wavelets are usually irregular and asymmetric [Joo and Kim, 2015], the WT is particularly suited for analysis of nonlinear, noisy and nonstationary time series, with rapidly changing behavior [Los, 2003; An et al., 2011; Roshan et al., 2011; Lahmiri, 2016]. If the wavelet transform is applied to a nonstationary time series, the resulting decomposed series usually present a better behavior than the original series, being more easily and accurately predicted, even by simple linear models like ARIMA. Predictions of the original time series with increased accuracy can then be obtained by applying the inverse WT to the predictions of the decomposed series [Conejo et al., 2005; Joo and Kim, 2015].

Besides the WT, other technique that is very recent and prominent, is the EMD [Lahmiri, 2016]. Similarly to the WT, the EMD is an algorithm for nonlinear and nonstationary time series decomposition, generating a time-frequency representation of the series [Lahmiri, 2016; Wang et al., 2016]. EMD is also capable of decomposing a time series into more stable components (intrinsic mode functions (IMFs)), which can be more easily modeled, improving prediction accuracy [Sun et al., 2016]. However, the advantage of EMD is that it does not depend on predetermined (wavelet) functions, since its basis

functions are derived directly from the time series, therefore being adaptive and completely data-driven [Lahmiri, 2016; Wang et al., 2016].

Finally, another recent noteworthy approach is presented by the HHT, which combines the use of EMD with the application of the Hilbert transform for each of the resulting IMFs [Song et al., 2017]. The Hilbert transform is used for computing instantaneous frequencies, enabling a frequency-time-energy (or amplitude) representation of a time series. The HHT is capable of accurately representing local time and frequency characteristics of nonlinear and nonstationary time series [Song et al., 2017].

2.1.2- Time series prediction

One of the main problems of time series analysis is the prediction of time series observations [Gujarati, 2021]. Prediction is knowingly a crucial aspect of decision-making activities [Han et al., 2022; Esling and Agon, 2012]. The future states of information about a problem can massively impact the success or failure of its solution. Time series prediction is an object of interest for many researchers due to its increasing importance and applications in science, business, and government [Salles et al., 2015]. The prediction context encompasses both problems of classification (prediction of discrete data) and regression (prediction of continuous data) [Han et al., 2022; Esling and Agon, 2012; Buza, 2018]. However, this work only focuses on the problem of predicting numeric time series data through regression. For simplicity, this work may refer to prediction and regression interchangeably. Relevant models adopted for time series prediction generally fall into the categories of statistical or machine learning models.

Statistical models Statistical models are designed for modeling a time series by describing it in statistical terms. Some of these terms directly model nonstationary behavior, such as an inherent trend or seasonality. These models may also be combined with the application of transformations such as detrending and/or differencing to achieve stationarity. Examples of models falling into this category are set by the structural models (SM) [Clements and Hendry, 2008; Shumway and Stoffer, 2017], linear mixed model (LMM) [Fox, 2015], HW [Dudek, 2016], Taylor's double seasonal Holt-Winter's model

(THW), ETS, TF [Girish and Tiwari, 2016], and the generalized autoregressive conditional heteroscedasticity model (GARCH) [Nachane and Clavel, 2008].

Machine learning models MLM have been used for nonlinear time series prediction in areas such as manufacturing systems, finance, health informatics, and energy grids, among other fields [Cheng et al., 2015]. Since the models generated by MLM are universal approximators, that is, they can approximate any continuous function to an arbitrary precision, they can be beneficial to the problem of modeling nonstationary time series properties. Some of the most relevant MLM are the NNET [Haykin, 2011], MLP [Machado et al., 2016], ELM [Huang et al., 2006], RBF [Poggio et al., 1989], SVM [Cheng et al., 2015], RFrst [Breiman, 2001], and the deep learning models CNN and LSTM [Lim and Zohren, 2021].

Although a great variety of time series prediction methods exists in literature [Cheng et al., 2015], many of these methods and the majority of works that handle time series assume that the available time series is stationary [Gujarati, 2021]. However, in practice, it is observed that such properties are not constant in the majority of real-world time series, especially in socioeconomics [Tsay, 2010], where many of them are nonstationary. Thus, when observed the presence of nonstationarity in a time series, it is a usual approach to search for ways to transform them to achieve stationarity so that the known time series prediction methods can be applied. Moreover, observed changes in mean or variance in nonstationary time series are frequently related to the occurrence of environmental events [Hanssens et al., 2006]. In turn, the detection of such events can considerably contribute to the analysis of the time series.

2.2- Time series event detection

Event detection is the process of identifying the occurrence of events based on data analysis. In fact, it is recognized as a basic function in surveillance and monitoring systems. And it becomes even more relevant for applications based on time series and sensor data analysis [Pimentel et al., 2014]. With this process we may be interested in learning/identifying past events, or identifying events in real-time (online event detection),

or even predicting future events before they happen (event prediction).

In this context, this work contributes by providing a taxonomy for the detection of events in time series depicted in Figure 8. Inspired on the works of Ariyaluran Habeeb et al. [2019] and Chandola et al. [2009], it is divided into seven categories based on the set of parameters found in most literature reviews. The categories regard time series event characteristics such as event type and granularity, and event detection methodology aspects, encompassing learning modes, general strategies for detection, modeling approaches, and detection methods. Each category is detailed in the next sections.

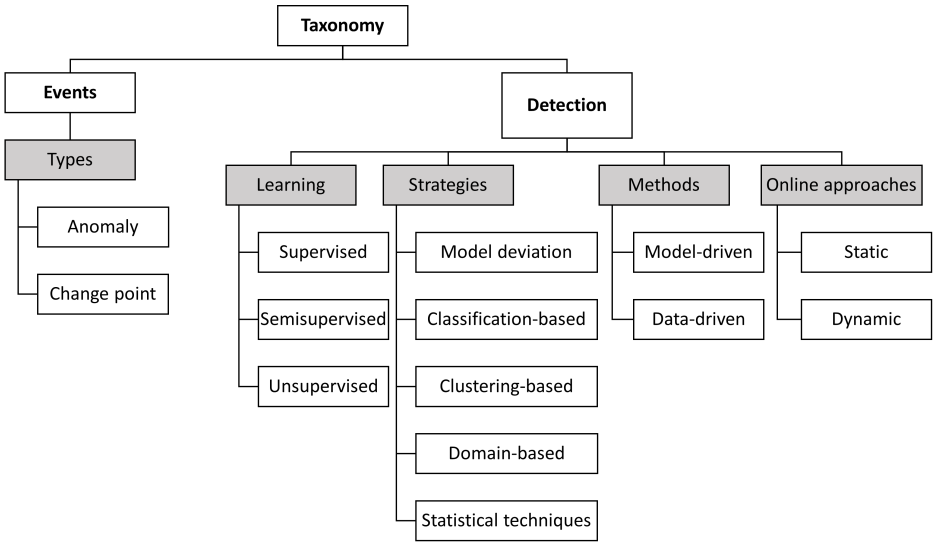


Figure 8 – Taxonomy of event detection techniques (adapted from Ogasawara et al. [2021])

2.2.1- Time series events

Events correspond to a phenomenon, generally pre-defined in a particular domain, with an inconstant or irregular occurrence relevant to an application. In the context of time series, events represent significant changes in expected behavior at a certain time or interval [Guralnik and Srivastava, 1999]. In general, punctual events of a given time series X = <$x_1, x_2, x_3, \cdots, x_n$>, can be identified in a simplified way by $e(X, k, \sigma)$ using the Equation 5, where k represents the length of nearby observations. A temporal component (TC) for an observation x_t is expressed as $tc(x_t)$. The TC can refer to the observation

itself, its instant trend, and its instant volatility, respectively represented as $tc_o(x_t)$, $tc_{tr}(x_t)$, and $tc_v(x_t)$ [Ogasawara et al., 2024].

Considering a typical autoregressive behavior [Gujarati, 2021], one can expect a TC for x_t to be related to previous observations. Let ep be the expected TC for x_t based on the previous k observations, where $ep(x_t, k) = E(tc(x_t) \mid tc(x_{t-k}), \ldots, tc(x_{t-1}))$. Analogously, one can also expect TC for x_t to be explained from the following observations [Lima et al., 2022]. So let ef be the expected TC for x_t based on the following k observations, where $ef(x_t, k) = E(tc(x_t) \mid tc(x_{t+1}), \ldots, tc(x_{t+k}))$. If a TC for x_t escapes the expected value above a threshold σ based on previous or following k observations [Gujarati, 2021; Lima et al., 2022], it can be considered an event. Equation 5 also considers an event if the expected TC from previous and following k observations are different above the threshold σ [Ogasawara et al., 2024].

$$\begin{aligned} e(X, k, \sigma) = \{t, |tc(x_t) - ep(x_t, k)| > \sigma \\ \vee |tc(x_t) - ef(x_t, k)| > \sigma \\ \vee |ep(x_t, k) - ef(x_t, k)| > \sigma\} \end{aligned} \tag{5}$$

Anomalies Most commonly, events detected in time series refer to anomalies. Anomalies appear not to be generated by the same process as most of the observations in the time series [Chandola et al., 2009]. Thus, anomalies can be modeled as isolated observations of the remaining nearby data. In this case, an event identified in x_t can be considered an anomaly if it escapes expected TC before and after time point t according to $a(X, k, \sigma)$ in Equation 6 [Ogasawara et al., 2024]. Generally, anomalies are identified by deviations from the time series inherent trend. However, anomalies may also present themselves as data volatility variations.

$$\begin{aligned} a(X, k, \sigma) = \{t, |tc(x_t) - ep(x_t, k)| > \sigma \\ \wedge |tc(x_t) - ef(x_t, k)| > \sigma\} \end{aligned} \tag{6}$$

Change points Change points in a time series are the points or intervals in time that represent a transition between different states in a process that generates the time series [Takeuchi and Yamanishi, 2006]. It is possible to define the change point detection as a hypothesis test problem, where the null hypothesis H_0 characterizes the absence of change points, and the alternative hypothesis H_A negates H_0. Let $seq_{i,p}(X)$ be a

subsequence of observations from a time series, $t, k \in \{i, \ldots, i+p\}$, and $t \leq k$. Formally, H_0 and H_A are as presented in Equation 7, where $\mathbb{P}_{x_i}$ is the probability density function of the subsequence and k is a change point [Chen and Zhang, 2015].

$$\begin{aligned} H_0 &: \forall\, t, k\ (t \neq k) \mid \mathbb{P}_{x_t} = \mathbb{P}_{x_k} \\ H_A &: \forall\, t\ \exists\, k\ (t \neq k) \mid \mathbb{P}_{x_t} \neq \mathbb{P}_{y_k} \end{aligned} \tag{7}$$

In this case, a change point event identified in time t follows the expected behavior observed before or after the time point t, but not both at the same time according to $cp(X, k, \sigma)$ in Equation 8. It can also refer to a significant difference between the expected trend before or after the time point t [Ogasawara et al., 2024].

$$\begin{aligned} cp(X, k, \sigma) = \{t,\ (|tc(x_t) - ep(x_t, k)| > \sigma \\ \veebar\ |tc(x_t) - ef(x_t, k)| > \sigma) \\ \vee\ (|ep_{tr}(x_t, k) - ef_{tr}(x_t, k)| > \sigma)\} \end{aligned} \tag{8}$$

As this thesis focuses on the problem of online detection of change points, henceforth the next subsections present the main concepts, strategies and methods generally adopted for event detection that are particularly relevant to identifying time series change points. The common challenges posed by online detection are another point of focus for this work. Furthermore, the metrics generally used to evaluate detection performance are discussed.

2.2.2- Event detection

Learning modes Based on the availability of the *normal* or *abnormal* labels, event detection techniques can be trained in three modes. In supervised mode the techniques assume the availability of a training data set that has labeled instances. In semisupervised mode, they assume that the training data has labeled instances only for the normal class. On the other hand, techniques that operate in unsupervised mode do not require labels in training data, and thus are most widely applicable. The techniques in this category implicitly assume that normal instances are more frequent than abnormal [Chandola et al.,

2009].

Detection strategies Event detection methods found in literature are usually based on five general strategies: model deviation analysis, classification-based analysis, clustering-based analysis, domain-based analysis or statistical techniques [Chandola et al., 2009; Pimentel et al., 2014]. The first strategy, model deviation analysis, is one of the most commonly adopted and is based on analysis of model deviation. That is, first a statistical or a machine learning model is fitted to the available data. Then events are identified as the observations that most deviate from the fitted model [Han et al., 2022].

Classification-based analysis and clustering-based analysis represent event detection as a problem of classification or clusterization, respectively [Han et al., 2022]. The former performs a binary classification of time series observations as falling into a normal or abnormal class based on the previously learned patterns. The latter focus on identifying abnormalities based on their alienation from the learned clusters of normal observations. Analogously, domain-based strategies compare new data observations against what is expected based on expert knowledge of the domain.

Finally, we have statistical techniques that generally identify deviations from the data distribution. Other probabilistic approaches involve a density estimation of the normal class [Alevizos et al., 2017].

Event detection methods Due to the increasing demand for event detection in real-world applications, a wide number of event detection methods have been developed and are currently available in literature. Several of such methods, from all techniques presented in Section 2.2.2, have been described and compared by Chandola et al. [2009]. Moreover, Ariyaluran Habeeb et al. [2019] give an overview of the state-of-the-art machine learning-based big data processing technologies for event detection. This section presents a selected subset of some of the most relevant event detection methods. The methods are divided into model-driven or data-driven according to the approach adopted. While model-driven techniques are more traditional and statistics-based, data-driven techniques are enabled by machine learning.

Model-driven methods

In event detection, model-driven techniques are frequently used due to their lower computational costs and interpretability [Ahmad et al., 2017; Ariyaluran Habeeb et al., 2019]. Moreover, change point detection methods have been the subject of intensive research investigations. Thus, there has been a wide array of model-driven change point detection methods in the literature, many of which focus on detecting breakpoints where the predefined features (e.g., mean, variance and any statistics calculated from the given time series) change. For example, one proposal developed for detecting change points in time series is based on the Exponentially Weighted Moving Average (EWMA) [Raza et al., 2015]. There are also approaches such as: the cumulative sum (CUSUM) which searches for breakpoints when the means of the subsequences shift; the BIFAST method which detects a change point when the coefficients of subsequence models change; and the GPchange method which considers variation in auto-covariance of a time series as a change point [Chen et al., 2014].

Furthermore, due to their sequential nature and their adequacy to address online problems, the sequential probability ratio test (SPRT), the CUSUM, and the generalized likelihood ratio (GLR) are among the most popular change-point detection methods. What these widely used algorithms have in common is that the logarithm of the likelihood ratio between two consecutive intervals (for the same time series) is monitored and a change point is declared whenever these intervals possess different statistical properties [Ba and McKenna, 2015].

However, the seminal method for detecting change points (SCP) has become a reference in the literature [Guralnik and Srivastava, 1999]. It adopts the following strategy: for any time point in a data window, models are adjusted to data segments before and after the point, and then it is determined that there is a change point if the total fitting errors are significantly reduced compared to the case where there is no change point. In its simplest version, the method adopts a polynomial function, such as linear regression, as a model for adjusting the data. The adjustment error is measured in terms of square errors [Takeuchi and Yamanishi, 2006]. Figure 9 illustrates the strategy adopted by the method.

SCP promoted the development of many approaches, such as the *ChangeFinder* (CF) [Takeuchi and Yamanishi, 2006], for example, which is composed of two phases. In

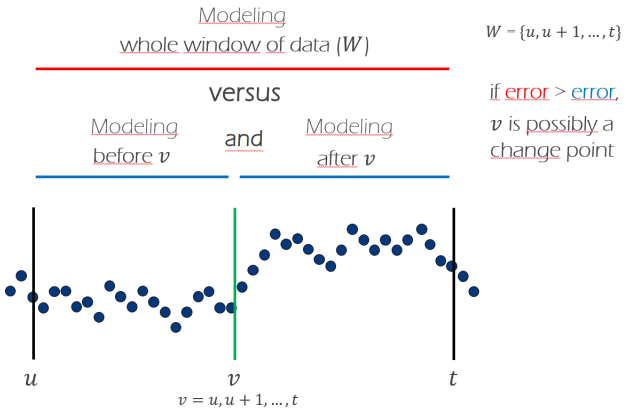


Figure 9 – Illustration of the strategy adopted by the method proposed by Guralnik and Srivastava [1999].

the first phase, given a time series X, a model α is adjusted, resulting in $\hat{X}$ and identifying its anomalies from the residuals in the series s defined in the Equation 9. In the second stage, a new series $\overline{s}_p$ is produced, which is composed of the moving averages of s with p terms. The change point detection is reduced to anomalies detection in $\overline{s}_p$.

$$s_i = (\hat{x}_i - x_i)^2, \ \hat{x}_i = \alpha(x)_i \tag{9}$$

Data-driven methods

In contrast to the methods presented previously, the data-driven methods based on machine learning are not necessarily restricted to certain kinds of problems, and they apply to time series event detection. Moreover, they are not restricted to detecting a specific type of event. Some of the main data-driven methods used for event detection are K-MEANS, NNET, SVM, ELM, CNN, and LSTM. K-Means is applied to identify clusters in a dataset. The k clusters are identified based on the similarity of the Euclidean distance of the observations relative to the centroids of each cluster. The value k is the arbitrary number of clusters to be identified. The centroid of each cluster is defined by the algorithm iteratively, by averaging the values of the observations within each cluster [Muniyandi et al., 2012]. Event detection is based on the degree of dissimilarity between the clusters.

The remainder of the data-driven methods here described are generally used as basis to event detection based on model deviation. In that case, machine-learning or deep-

learning models are applied in order to estimate a time series inherent function. Significant deviations from the time series model serve as an indication of events. Particularly, NNET, ELM, SVM, CNN, and LSTM were also presented as methods for time series prediction in Section 2.1.2.

According to Haykin [2011], NNET can be seen as a set of mathematical tools used to learning the relationship between input and output variables. One of the advantages of its use resides in the flexibility of the distributed model defined by the weights of the network. Thus, the linear and non-linear decisions can be defined by adjusting the neural network configuration. Each node in a layer is an artificial neuron, where the input is modified by weight and added to all other inputs, and the resulting value is passed through a transfer function to one or more neurons in the next layer [Riese and Keller, 2020].

SVM are supervised learning models used in classification tasks to analyze data and recognize patterns. This technique consists of mapping input data points to a high-dimensional resource space using kernel transformations and aiming for pattern recognition in the data. SVM seeks to locate classifiers with a greater distance between the support vectors from an infinite number of classifiers [Chauhan et al., 2019].

SVM-based classifier is suitable for handling large sets of attributes during the event detection and for quadratic optimization seeking to obtain a stable solution [Rahul and Choudhary, 2021]. SVM algorithms generally perform well on classification problems. Consider $\{a_1, b_1\}, \cdots, \{a_i, b_i\}$, such that $a \in R^n$ and $b \in \{-1, 1\}$, $i = 1, \cdots, N$, where N is the number of training instances, a is the input vector and b is the desired classification. The objective is to estimate a function $F : R^n \rightarrow \{-1$ or $1\}$, using the training examples and applying it in the test examples, in order to classify them correctly.

ELM is a learning algorithm for hidden layer feed-forward neural networks. ELM has a lower training error and a lower weight standard compared to other machine learning models. In this structure, parameters of the hidden nodes are generated randomly, and the weights of the output are calculated analytically [Tang et al., 2016]. Among the advantages compared to conventional gradient-based learning methods, [Ismaeel et al., 2015] highlight that ELM is suitable for most nonlinear activation functions and can achieve a better-generalized performance compared to backpropagation. Huang et al. [2006] stand out some points about the performance of ELM: *(i)* extremely fast learning speed; *(ii)* better generalization performance, compared to gradient-based learning algorithms; and *(iii)* ability to work with differentiable and non-differentiable activation functions.

One of the most well-known deep learning methods is CNN. It is a network formed basically by modules superimposed on convulsion and grouping layers. CNN considers the identification of the local relations between the analyzed data [Lim and Zohren, 2021]. The process of a CNN comprises applying the convolution layers, composed of several neurons, to the input data. Combining the inputs of a neuron with the respective weights of each connection produces an output for the next layer. A matrix that contains the weights assigned to a neuron's connections represents the convolution filter.

The idea behind deep learning is to discover multiple levels of representation expecting high-level resources may represent a more abstract semantics of the data [Guo et al., 2017]. A CNN is an architecture composed of three distinct layers: an input layer, a convolutional layer, and a pooling layer, which reduces the size of the input data. The convolution layers in the datasets can be applied as an extractor of characteristics implicit in the data. The Equation 10 presents a convolution process, where g is the input layer, h is one of the k filters that a CNN will optimize for an aim function during the learning process at time t, $*$ is the convolution operator, and n is a hidden layer in the neural network.

$$(g * h)[n] \equiv \sum_{t=0}^{k} g[k-t]h[t] \tag{10}$$

LSTM networks have the same properties as conventional recurring networks. However, they can store information for long periods when processing a time sequence. The memory points of an LSTM network are called cells. Cells can carry information until the end of a sequence or identify information that the network should forget after some processing step [Greff et al., 2017].

Structural break detection

This subsection brings attention to a special application of model-driven detection methods, which is directed to the problem of time series structural break detection. Change points mark the occurrence of structural changes in a time series. Methods for detecting structural changes as parameter instabilities in parametric models, typically (linear) re-

gression models, have been receiving much attention in both econometrics and statistics communities. Take the standard linear regression model $y_t = x_t\beta_t + u_t$ ($t = 1, \ldots, n$), where y_t is the observation of the dependent variable, x_t is the vector of observations of the independent variables, u_t are iid(0, σ^2), and β_t is the vector of regression coefficients. Tests on structural change are concerned with testing the null hypothesis of "no structural change", that is, $H0 : \beta_t = \beta_0$ against the alternative that the coefficient vector β_t varies over time [Zeileis et al., 2002, 2010].

Various classes of tests were developed focusing on different properties, being more or less suitable for certain patterns of deviation from the null hypothesis. One of the main classes is based on F statistics [Zeileis, 2005]. The class of tests based on F statistics has been developed for the alternative hypothesis of a single break at an unknown time. As the statistics are easy to interpret, these tests enjoy great popularity and are probably the most used in practice. The most popular tests based on F statistics include the Chow test and the $supF$. Chow [1960] was the first to suggest a test on structural change for the case where the (potential) change point t_{cp} is known. He proposed to fit two separate regressions for the two subsamples defined by t_{cp} and to reject whenever the F statistic is too large [Zeileis et al., 2002].

Assume n is the number of time series observations and k the number of regressors in the linear model. For computing the F statistic an OLS model is fitted for the observations before and after the potential change point t_{cp} ($2k$ parameters are estimated), and the error sum of squares is computed (ESS). Another OLS model for all observations with a restricted sum of squares (RSS) is computed (k parameters are estimated). The F statistic is given by Equation 11 [Zeileis et al., 2002].

$$F = \frac{(RSS - ESS)}{ESS/(n - 2k)} \tag{11}$$

The drawback of the Chow test is that the change point has to be known in advance, and it is often not the case. On the other hand, the $supF$ test, also based on F statistics (Chow test statistics), does not require a specification of a particular change point. It is based on the extension of the Chow test, where F statistics F_t are calculated for all potential change points in an interval, for $t_i < t_a \leq t \leq t_b < t_f$, where t_i and t_f represent the first and last available time points, and t_a and t_b define the evaluated interval, respectively. Next, the series of F statistics can be aggregated into a single test statistic

using the $supF$ statistic suggested by Andrews [1993]. In that case, the null hypothesis is rejected when the maximal F statistic gets too large. F statistics boundaries can be computed such that the asymptotic probability that the supremum of the statistics F_t (for $t_a \leq t \leq t_b$) exceeds this boundary is α. As the F statistics cross their boundary, there is evidence for a structural change (at the level $\alpha = 0.05$). The actual breakpoint (change point event) estimate is given by minimizing the residual sum of squares (RSS). Alternatively, the point that maximizes the sequence of F statistics can also serve as an estimate for the breakpoint, as it is equivalent to the minimum OLS estimator of the breakpoint in a 2-segment partition [Zeileis et al., 2002].

Zeileis [2005] also proposed an extension of the $supF$ test to a monitoring scenario. Online monitoring of structural changes is concerned with detecting parameter instabilities in incoming data, rather than detecting changes in historical samples. The basic idea for extending the $supF$ test to the monitoring setup: when new observations arrive, estimates are computed sequentially from all available data (historical sample plus newly arrived data) and compared to the estimate based only on the historical sample [Chu et al., 1996]. As in the retrospective case, the hypothesis of no structural change is rejected if the difference between these two estimates gets too large [Zeileis et al., 2010]. Zeileis [2005] indicate that monitoring with $supF$ tests is able to detect structural breaks quickly after only a few observations in the monitoring period.

2.2.3- Online event detection

Online event detection is the process of identifying the occurrence of events in streaming data. Streaming applications impose particular challenges for time series modeling. These applications involve analyzing a continuous sequence of data occurring in real-time. In contrast to offline event detection (batch processing), the full dataset is not available. The system needs to observe each data record in sequential order as they arrive and any processing or learning must be done in an online fashion. Let X be a streaming time series composed of a continuous stream of inputs: <$\ldots, x_{t-2}, x_{t-1}, x_t, x_{t+1}, x_{t+2}, \ldots$>. At each point in time t, a model trained on previous observations <$\ldots, x_{t-2}, x_{t-1}$>is used to determine whether the current behavior of the system is unusual. This determination

must be made in real-time by the arrival of the next input x_{t+1}. For this, the time series model must be continuously updated [Ahmad et al., 2017].

Unlike offline event detection, data is not split into static train/test sets, and algorithms cannot look ahead. Practical applications impose additional challenges. Typically, the sensor streams are large in number and at high velocity, leaving little opportunity for expert intervention; manual parameter tweaking and data labeling are usually not viable. Thus, operating in an unsupervised, automated fashion is a necessity [Ahmad et al., 2017]. Furthermore, in streaming applications, early detection of events is often important, where events are identified as soon as possible. Detection of events can give information that is crucial to decision-making. And this information must be given early enough that it is actionable, preventing possible critical system failures. However, there is still a tradeoff between early detections and false positives. An algorithm that often makes inaccurate detections is likely to be ignored.

Applications aiming the uninterrupted monitoring of events in a network infrastructure usually generate fast data. In fast data, data dynamically changes in certain time intervals that can be seconds and milliseconds. The huge quantity of data arrive continuously to the pipeline. Streamed data is considered in the big data analytics, so online big data processing is critical for event analysis and decision-making [Ariyaluran Habeeb et al., 2019].

Static and dynamic modeling Despite the detection of events in real-time data having practical and substantial applications across several industries, reliable solutions do not yet exist [Ariyaluran Habeeb et al., 2019]. Current approaches for online event detection can be categorized into two major categories regarding their modeling approaches. In the first category a static model is trained on large data samples, and then deployed on data streams. In the second category the event detection model is initialized on a data sample and then learned incrementally as new data arrives, adopting a dynamic strategy.

Indeed, online detection is a much more complicated problem that presents a number of challenges. Accommodating large volumes of streaming data in the main memory of a machine is often infeasible. In that case, full-memory algorithm approaches, depicted in Figure 10 (left), that consider all observations (batches) seen so far, are impractical [Iwashita and Papa, 2019]. Hence, an online processing is required. In this case, data models can be retrained using recent batches of data, or they can be trained

incrementally by continuous update. Incremental algorithms process input examples one by one (or batch by batch) and update the data model after receiving each example. Typically, for any new batch of data, the update operation of the model is based on the previous one (Figure 10 (middle)). However, incremental algorithms may also have random access to previous examples or the most representative ones. In this case, these algorithms are said to have partial (windowed) memory Figure 10 (right) [Gama et al., 2014].

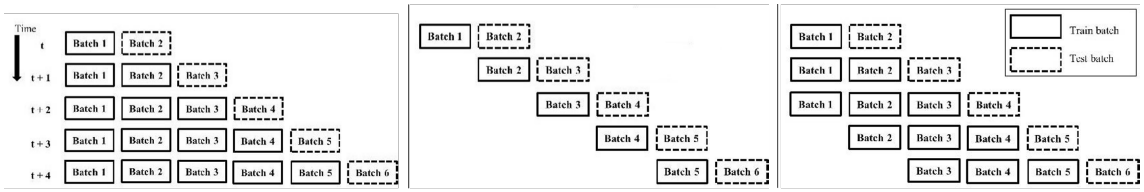


Figure 10 – Illustration of algorithm approaches based on full-memory, no-memory or windowed memory, respectively (adapted from Iwashita and Papa [2019])

Learning algorithms often need to operate in environments that are dynamic and can change unexpectedly. A desirable property of these algorithms is their ability to incorporate new data. Which, in turn, brings us to the stability-plasticity dilemma [Ditzler et al., 2015]. That is, how can a learning system be designed to remain adaptive in response to significant changes and yet remain stable in response to irrelevant changes?

If the data-generating process is not strictly stationary, the underlying concept, which is being predicted, may be changing over time. Learning in nonstationary environments requires adaptive or evolving approaches that can monitor the underlying changes, and adapt a model to accommodate these changes. Given that it is the case for most real-world applications, there is a strong demand for effective and efficient algorithms for learning from (and adapting to) evolving or drifting environments [Ditzler et al., 2015]. The ability to adapt to such changes can be seen as a natural extension for the incremental learning systems that learn predictive models example by example. Adaptive learning algorithms can be seen as the next step to incremental learning ones that are able to adapt to the evolution of the data-generating process over time [Sayed-Mouchaweh and Lughofer, 2012].

In nonstationary scenarios, data is expected to evolve/change over time, and this includes the underlying data distribution. When it happens, we have a phenomenon known as concept drift. Formally, concept drift between time point t_0 and time point t_1 can be defined as $\exists X : p_{t_0}(X, y) = p_{t_1}(X, y)$, where p_{t_0} denotes the joint distribution at time t_0

between the input time series X and the target variable y (class normal or event) [Gama et al., 2014].

The general assumption is that concept drifts happen unexpectedly and unpredictably, although there are cases in which they can be known beforehand based on the occurrence of particular environmental events. Moreover, the change may take different forms, that is, it can happen suddenly/abruptly, incrementally, gradually or even previously seen concepts may reoccur after some time [Iwashita and Papa, 2019]. In such cases models must adapt to a new definition of "normal" in an unsupervised, automated fashion [Ahmad et al., 2017]. In that case, adaptive learning refers to updating predictive models online during their operation to react to concept drifts [Gama et al., 2014].

Adaptive learning algorithms are primarily based on either an active or passive approach. Algorithms following the active approach specifically aim at detecting concept drift. It is actually related to the detection of structural breaks and change point events, which may be described as the points in which we observe a drift in concept. Algorithms following the passive approach continuously update the model every time new data is given as input, regardless of whether there is drift or not. Both active and passive approaches intend to provide an up-to-date model [Ditzler et al., 2015].

Overall, the requirements described in this section suggest that event detection for online applications is a fundamentally different problem than offline event detection. Furthermore, the majority of existing event detection algorithms are not applicable to streaming applications [Ahmad et al., 2017]. Thus, there is currently a demand for developing reliable methods particularly designed for online event detection.

2.2.4- Event detection performance metrics

An important aspect of any event detection method is the way in which the events are reported. Typically, the outputs produced by event detection methods are either scores or labels. Scoring detection methods assign an event score to each instance in the data depending on the degree to which that instance is considered an event. Thus the output of such methods is a ranked list of events. In that case, one can choose the top n events or use a threshold (may be domain-specific) to select the events to be analyzed. Labeling

detection methods, on the other hand, assign a label (normal or event) to each data instance [Chandola et al., 2009].

A labeling detection method can learn from the input data and distinguish between normal and event classes. Based on the labels available for the training phase, classification-based event detection methods can be grouped into multi-class or one-class. Multi-class event detection methods assume the data contains labeled instances of multiple normal classes. A data instance is considered an event if it is not classified as any of the known normal classes. On the other hand, one-class event detection methods assume that all data instances have only one class label. Such methods are the most commonly found in the literature. They learn a boundary to discriminate the normal instances. Any data instance that does not fall within the learned boundary is declared as event [Chandola et al., 2009].

However, there is no single best method when it comes to event detection methods and the assessment of their detection performance is often important in order to infer their adequability to a particular application. In this case, detection performance refers to how good or how "accurate" a labeling event detection method is at predicting the class label of data instances. For this purpose, the standard metrics for evaluating time series classification algorithms (around since the 1950s) are generally used [Salfner et al., 2010; Tatbul et al., 2018]. The standard classification evaluation metrics are summarized in Table 3 including accuracy, recall, specificity, precision, F1, and F_β. Some of these metrics are also known by other names. Although accuracy is a specific measure, the word "accuracy" is generally used to refer to the predictive abilities of a method [Han et al., 2022].

The metric formulas in Table 3, depend on the measures P and N. Respectively, they refer to the number of positive instances (data instances of the main class of interest) and negative instances (all others). In the context of event detection, the positive instances indicate the occurrence of an event ($event = yes$) while the negative instances indicate normal behavior ($event = no$). Moreover, Table 3 presents other four important measures used to compare the class labels yielded by the detection method with the known class labels of the data instances. They are TP, TN, FP, and FN [Han et al., 2022].

The TP measure is the number of true positives, which refer to the positive instances that were correctly labeled by the method. Analogously, TN is the number of true negatives, negative data instances that were correctly labeled by the method. On

Table 3 – Standard event detection performance metrics [Han et al., 2022]

Metric	Formula
accuracy, recognition rate	$\frac{TP+TN}{P+N}$
error rate, misclassification rate	$\frac{FP+FN}{P+N}$
sensitivity, true positive rate, recall	$\frac{TP}{P}$
specificity, true negative rate	$\frac{TN}{N}$
false positive rate, fallout	$\frac{FP}{N}$
precision	$\frac{TP}{TP+FP}$
F, F1, F-score, harmonic mean of precision and recall	$\frac{2 \times precision \times recall}{precision + recall}$
F_β , where β is a non-negative real number	$\frac{(1+\beta^2) \times precision \times recall}{beta^2 \times precision + recall}$

the other hand, the measure FP is the number of false positives, that is, the negative data instances that were incorrectly labeled as positive. Finally, FN is the number of false negatives, referring to the positive data instances that were mislabeled as negative [Salfner et al., 2010; Han et al., 2022]. The four measures are summarized in a confusion matrix as presented in Table 4, which is a useful tool for analyzing how well a detection method can recognize the positive (event) class. TP and TN refer to correct labeling, while FP and FN refer to mislabels. Totals are also provided. Besides P and N, P' is the number of data instances labeled as positive and N' is the number of instances labeled as negative. Given a time series X, the total number of instances is $TP + TN + FP + TN = P + N = P' + N' = |X|$.

Table 4 – Standard confusion matrix for event detection

		Detected event		
		yes	no	Total
Known event	yes	TP	FN	P
	no	FP	TN	N
	Total	P'	N'	$P+N$

The standard (classification-based) event detection performance metrics listed in Table 3 allow specific analysis regarding the the performance of the detection methods.

The accuracy metric, reflects how well the method recognizes data instances of both classes (positive and negative). The error rate is the inverse of the accuracy. The sensitivity and specificity metrics reflect, respectively, how well the method can recognize the positive instances ($event = yes$) and the negative instances ($event = mo$). The fallout is the inverse of the specificity. The precision and recall metrics are widely adopted. Precision reflects exactness (the percentage of instances labeled as positive that are actually so), whereas recall reflects completeness (the percentage of positive instances labeled as such). A perfect precision score (1.0) means that every instance the method labeled as event is indeed a known event. A perfect recall score (1.0) means that every known event was labeled as such. Precision and recall are combined in the F metric (or F1 score) and the F_β metric. The F metric is the harmonic mean of precision and recall, giving them equal weight. The F_β metric, on the other hand, assigns β times as much weight to recall as to precision [Han et al., 2022]. Both F and F_β metrics help gauge the quality of event detection [Tatbul et al., 2018].

Event detection methods frequently use the Operating Characteristic Curve (ROC), illustrated in Figure 11, to visualize the ability of a detection method to discriminate between positive instances (events) and negative instances (nonevents) based on the true positive rate (tpr) versus false positive rate (fpr). Detection methods whose curves come closer to the upper left corner of the ROC space are the most accurate [Han et al., 2022]. Likewise, the accuracy of the methods can be compared by the Area Under the Curve (AUC) defined in Equation 12, that summarizes the ROC curve into a metric [Salfner et al., 2010].

$$AUC = \int_0^1 tpr(fpr)\,\mathrm{d}fpr \ \in [0,1], \tag{12}$$

2.3- Time series event prediction

Event detection and online event detection are applied to identify historical or ongoing events. Event prediction, on the other hand, corresponds to the problem of predicting events in the future [Zhao, 2021]. The anticipation of the occurrence of events can help mitigate potential damages and allow educated decision-making. Event prediction



Figure 11 – Illustration of ROC curves for assessing the ability of a detection method to discriminate between positive and negative instances (adapted from Salfner et al. [2010])

is similar to the problem of time series prediction. However, instead of predicting the next n observations from analyzing past time series data, it focus on correctly predicting whether a target event will occur within a future window of time [Yan et al., 2004].

Event prediction methods take as input large amounts of time series data containing historical events, build a predictive model, and output event predictions. To build a predictive model on the input data, generally, such methods apply techniques from machine learning, data mining, pattern recognition, or statistics [Zhao, 2021]. The most common time series event prediction methods are based on decision trees, artificial neural networks and support vector machines [Molaei and Keyvanpour, 2015]. In recent years, there has been a considerable amount of research on new event prediction methods. Nevertheless, most of them are designed for specific application domains, although their approaches can usually be generalized. Overall, event prediction methods are considered to be still in their infancy [Zhao, 2021].

Generally, time series event prediction methods adopt one of two main strategies: (i) event prediction from time series analysis; or (ii) event detection from time series prediction. Event prediction from time series analysis is most commonly adopted. In this case, variations preceding the target time series event are analyzed to deduce temporal patterns that can be used for event anticipation [Gmati et al., 2019]. Then, the prediction process involves matching the learned features against the time series observations, commonly by the use of classification methods. If the match succeeds the occurrence of a target event is predicted [Yan et al., 2004].

Once prohibitively challenging, the area of time series event prediction is growing rapidly in the big data era [Zhao, 2021]. However, methods that are adherent of this first strategy are still particularly affected by a number of common issues that demand special attention from researchers [Molaei and Keyvanpour, 2015]:

- Data complexity: time series data commonly present complex behavior, demanding advanced analysis techniques.
- Noise: omnipresent in real-world time series, noise tend to decrease event prediction accuracy, and demands noise reduction techniques to be applied before data analysis.
- Unlabeled or unbalanced data: given a large amount of unlabeled instances or an unbalanced dataset, the use of supervised learning methods demand time and cost to gather labels (computationally or by experts) and instances of the rare event class.
- Parameterization: some methods need to set initial values and thresholds that usually demand trialing and optimization.
- Detection time: methods must perform the prediction process in an acceptable time to allow sufficient anticipation of events.
- Evaluation: comprehensive evaluation and comparison of time series mining methods for significant event prediction is a challenge.

The aforementioned issues and challenging data properties are very frequently observed, specially in online time series applications, focus of this study. Therefore, this strategy and the exploration of its methods lie outside the scope of this thesis. However, its introduction is used to contextualize and clarify the proposed approach, based on the second strategy, namely event detection from time series prediction.

2.3.1- Event detection from time series prediction

Event detection from time series prediction is based on a two-step approach. The first step involves performing time series prediction. For that, autoregressive techniques

are generally adopted. In this case, the historical time series $X = x_1, \ldots, x_t$ is used to obtain the future time series $\hat{X} = x_{t+1}, \ldots, x_{t+h}$. The second step consists of identifying events in the predicted time series $\hat{X}$ using unsupervised or supervised event detection methods [Zhao, 2021].

In this context, event prediction on the basis of time series analysis is different from time series prediction alone. The latter outputs future values of the time series, while the former a predicted event, that is, the future time of occurrence of an event [Gmati et al., 2019]. Figure 12 presents the general tasks to the process of event detection from time series prediction for event prediction systems in time series.

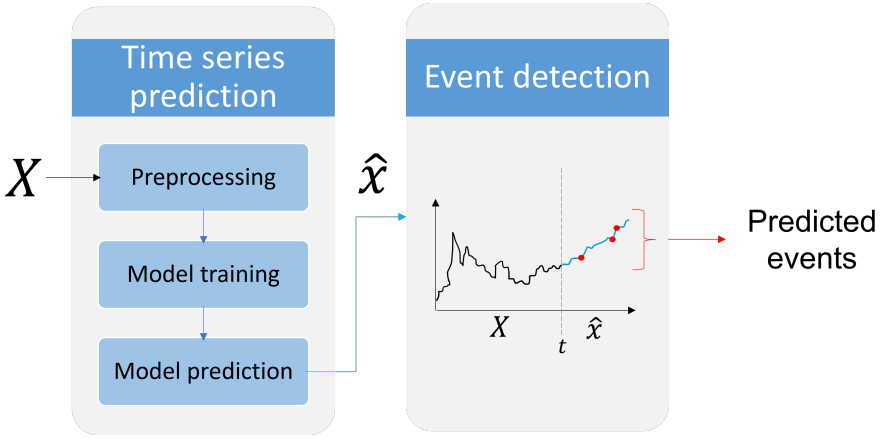


Figure 12 – General architecture for the process of event detection from time series prediction

The first step to time series prediction takes X as input and involves data preprocessing. Preprocessing is crutial to increase prediction accuracy. It includes performing feature selection, class labeling and also increasing data quality in general, as by the treatment of nonstationarity with data transformation methods. Next, past values of the preprocessed time series are used for model derivation. The derived model is then applied for event prediction and future values of the time series ($\hat{X}$) are outputted. Finally, in the last step, events are detected based on the predicted time series, $\hat{X}$, and available patterns [Mehrmolaei and Keyvanpourr, 2015].

2.3.2- Online event detection from time series prediction

Online event detection is based on runtime monitoring and methods that use both the current state of a system and past experience. Frequently, the analysis of the streaming time series involves (i) computing a residual value on which the current state is judged to be an event or not, or (ii) future states of the time series are predicted to estimate, for example, time until event [Salfner et al., 2010]. In this context, an approach based on event detection from time series prediction, mainly studied for event prediction, combines both strategies and can also be exploited for the problem of online event detection. In that case, the current state of streaming observations can be identified as event based on what is expected by the prediction of the current time series model.

Generally, solutions for online event detection demand the availability of sufficient data to allow the identification of detectable patterns. However, the time and effort necessary to gather such data can potentially make an application unfeasible. In turn, gathered datasets are usually unbalanced. In this case, abnormal or irregular patterns are rare, and their learning and identification become challenging due to insufficient data. Also, in the presence of nonstationarity, the learned time series patterns may change gradually or even abruptly, invalidating the acquired knowledge on the data. In this context, an approach based on event detection from time series prediction can be advantageous to online time series event detection since its application only demands the availability of a time series long enough to train a prediction model with the required accuracy. Such model is then used to guide the identification of event observations. However, as real world events continually change dynamically, drifts in concepts and distribution are inevitable. Hence, it is imperative to periodically upgrade the employed prediction models [Zhao, 2021].

3- Related works

Event detection has been extensively explored over the years and it has been attracting increasingly more interest over the last decade [Blázquez-García et al., 2021; Zhao, 2021]. The available literature was analyzed through systematic mapping studies. Firstly, works on event detection for time series were searched using a query string involving the keywords “event detection” and “time series”. The query was executed on the Scopus database and returned 454 references. Approximately 58 publications were selected to base the discussed review due to their relevance to the researched subject and impact (citation count). Moreover, survey papers on event detection were gathered based on snowballing.

A second search was performed to find works on the specific area of event prediction using the keywords “event prediction” and “time series”. The query was also run on the Scopus database in January 2022. It returned 100 references, from which 19 were selected as relevant for contextualizing this research and further discussion. The selection was restricted to works that adopted approaches based on event detection from time series prediction. From snowballing, extra survey papers were included.

Finally, additional snowball search was performed mainly on the subject of detection performance metrics, due to its increasing importance to the current research scenario and our research proposal. The next sections discuss the selected related works on time series event detection, event prediction and event detection performance metrics.

The Sections 3.1 to 3.4 discuss related works on time series event detection based on (i) their adopted detection strategies, (ii) advanced approaches for time series analysis, (iii) specialization to detect certain types of events, and (iv) applicability to an online context, respectively. Section 3.5 presents works that address event detection performance metrics. Section 3.6 contextualizes this research among current approaches to event detection from time series prediction. Moreover, related surveys are reviewed in Section 3.7.

3.1- Event detection strategies

Event detection works found in literature generally adopt one of five general strategies. They were discussed in Section 2.2.2. This section reviews related works on time series event detection methods based on their adopted detection strategy.

Model deviation analysis One of the most commonly adopted strategy is based in model deviation analysis. This strategy was adopted by Liu et al. [2015] that present a multivariate event detection method based on the deviation from an ARIMA model above a threshold. An ensemble for all variables is computed and compared with Multivariate Euclidean distance (MED) and Linear prediction filters (LPF). Meenakshi and Raghavan [2015] also performs event detection in internet resource dynamics using time series models and Inter Annual Absolute Variations. Kattan et al. [2015] use an unsupervised learning framework based on genetic programming, and Laptev et al. [2017] propose a novel end-to-end recurrent neural network architecture for event detection. It aims at predicting completed trips based on Uber data.

Classification-based detection Other works adopt a classification-based strategy where a model is trained to enable the classification of normal or abnormal data samples. For example, Gensler and Sick [2018] perform event detection in time series using SwiftEvent, an algorithm with supervised learning of detection criteria, while Provotar et al. [2019] perform unsupervised anomaly detection in time series using LSTM-based autoencoders. Park et al. [2019] also combined autoencoder and LSTM networks for fault detection and system diagnosis. Polz et al. [2020] and Kiskin et al. [2020], on the other hand, use convolutional neural networks. Polz et al. [2020] detect rain events in commercial microwave link attenuation data, and Kiskin et al. [2020] conducts bioacoustic detection with wavelet-conditioned convolutional neural networks.

Clustering-based detection Abnormal data samples can also be identified through a clustering-based strategy by comparing them against normal sample clusters. Afzalan et al. [2019] adopted this strategy, identifying events through automated clustering. They propose self-configuring event detection in electricity monitoring for human-building interaction. Zhao et al. [2020], on the other hand, perform spectral analysis for identifying

anomalies in time series of network traffic.

Domain-based detection Domain-based strategies use expert knowledge or established models of the domain problem to identify new data samples as abnormal in case they differ enough from expected. The literature presents several works that employ this strategy. Pelech-Pilichowski and Duda [2006] applies general structure of T-lymphocyte to event detection in financial time series. Gabarda and Cristóbal [2010] present a domain specific technique for finding events in a seismic dataset using time-frequency methods. Guanche García et al. [2019] perform extreme anomaly event detection in biosphere using both linear regression and a MRF model. They distinguish abnormal events based on Mahalanobis distance values surpassing a domain-specific threshold. Le Page et al. [2020] also use thresholds to calibrate their method to assess irrigation events on maize plots using Sentinel-1 soil moisture data and a water balance model. Zhang et al. [2020] identify human fall events based on a model of body posture evolution.

Statistical techniques Finally, there are works that adopt statistical techniques for detecting events based on deviations from data distribution. As an example, the method adopted by Ihler et al. [2006] involves the analysis of divergence in the distribution of time-varying poison processes. Other probabilistic approaches also fall into this category, such as the one adopted by McKenna et al. [2007]. They present a method that check if outliers can be marked as events in a water quality time series, if their probability in a data window is higher than expected (binomial distribution). Statistical approaches are also reviewed by Souto and Liebig [2016] in the context of anomaly detection for urban traffic applications. They also present frameworks for anomaly detection on streaming data.

3.2- Advanced time series analysis

Spatial-time series analysis Some time series can be specially challenging to analyze, generally demanding more advanced approaches. This is the case for spatial-time series, that encompass data that evolve in both space and time. Andrienko et al. [2010] and Souto and Liebig [2016] designed their methods to extract anomaly events from spatial time series. Guanche García et al. [2019] opted to use a spatiotemporal MRF model, and

Zhang et al. [2020] derived a model for body posture spatio-temporal evolution.

Nonstationary time series analysis As discussed in Section 2.1.1, the property of nonstationarity is pervasive in most real-world time series. It demands particular attention and specialized techniques for time series analysis applications. In this context, Hunter and McIntosh [1999] and DiCecco et al. [2006] introduced algorithms for nonstationary time series event detection. Hunter and McIntosh [1999] perform knowledge-based event detection, while DiCecco et al. [2006] perform biological event detection using statistical nonstationarity analysis. Marik and Bohac [2019] also perform nonstationary events detection based on Extrema Value Theory estimators. Kiskin et al. [2020], on the other hand, use wavelet and Fourier transforms to deal with data with low signal-to-noise ratios. Also, a framework based on extreme value theory that can work well in the presence of noisy nonstationary data is proposed by Talagala et al. [2020].

3.3- Types of detected events

Anomalies Commonly, event detection methods found in literature specialize in identifying a specific type of event among the listed in Section 2.2.1. Most methods focus on detecting anomalies in time series [Provotar et al., 2019; Park et al., 2019; Zhao et al., 2020]. Memarzadeh et al. [2020] perform anomaly detection in flight data using convolutional variational auto-encoder. The works of Perelman et al. [2012] and Loureiro et al. [2016] monitor water quality and distribution flow, respectively. Both works of Lawhern et al. [2013] and Eichmann et al. [2019] contribute by presenting tools for identification of anomalies. Efros et al. [2017] discuss the impact of temporal aggregation on anomaly detection. Results show that too high or too low aggregations may lead to improper event detection. Flach et al. [2017] performs an ensemble of different algorithms for multivariate anomaly detection in Earth observations.

The Matrix Profile technique is part of the state-of-the-art for multivariate time series anomaly detection [Yeh et al., 2016]. Multiple algorithms and adaptations have been proposed for efficient computation of the Matrix Profile, such as STAMP, STOMP [Yeh et al., 2018], SCRIMP++ [Zhu et al., 2018], and kNN Matrix Profile [Mondal et al.,

2023]. The development of new techniques and insights based on the Matrix Profile opens avenues for advancements in multivariate anomaly detection in time series data [De Paepe et al., 2020].

Change points Among the works that focus on identifying change point events or drifts in time series is the one by Guralnik and Srivastava [1999] that gives one of the seminal papers on change-point discovery. DiCecco et al. [2006] also presents an algorithm for biological event detection using drift analysis. Moreover, Aminikhanghahi and Cook [2017] review different methods for time series change point detection. Although the application of Matrix Profile to concept drift detection has begun to be explored [Madani and Vlajic, 2022], works that approach this solution are still scarce.

Motifs There are works that approach event detection by identifying motifs in time series. They generally apply sequence mining techniques. An example is given by Batal et al. [2012] that perform mining of recent temporal patterns for event detection in multivariate time series data. They consider temporal relations between events and compute frequent patterns among them. Bai et al. [2013] also perform event detection in multiple temperature series by computing correlation among motifs. They also indicate related work on finding patterns and motifs in multiple time series. The work of Batal et al. [2016] presents a pattern mining approach for event detection in multivariate time series based on Allen's temporal relations. After identifying events for all-time series, it maps segments as items and performs sequence mining for sliding windows. Afzalan et al. [2019] identify recurring motif events through automated clustering.

Detection of multiple event types It can also be found in literature some methods that are able to detect multiple types of events in time series. Takeuchi and Yamanishi [2006] present a unifying framework for detecting both anomalies and change points from time series. Similarly, the method of Andrienko et al. [2010] applies both anomaly and drift detection. If both occur, a change point event is identified.

3.4- Event detection from streaming data

Online detection Online event detection involves the identification of events in streaming data, occurring in real-time. Such online detection applications are particularly challenging and have been gaining more attention during the last decade due to the increasing demand for IoT sensor data analysis and automatic system monitoring [Ariyaluran Habeeb et al., 2019]. Falling into this category, is the work of Chen et al. [2014] that study online detection of group level events in time series. They work with the concept of a contextual change, referring to a deviation in the behavior of one or a group of time series with respect to its context, where the context is a collection of time series that behaves similar to the target group of time series over a time window. Ba and McKenna [2015] apply online change-point detection methods for water quality monitoring, while Alkhamees and Fasli [2017] present a domain specific (financial) method for event detection in streaming time series that uses directional change and dynamic thresholds. Ahmad et al. [2017] propose a novel anomaly detection algorithm based on an online sequence memory algorithm called Hierarchical Temporal Memory (HTM). Yadav et al. [2019] study events that are not isolated, but superimposed. They perform real-time multiple event detection and classification in power system using signal energy transformations. Zhang et al. [2019] introduces a multi-dimensional joint prediction model for IoT sensor data event search.

Fast event detection In addition to an online approach to event detection on streaming data, some works set out to do it fast and enabling real-time reaction. This problem is know as fast event detection. Yu et al. [2015] introduces an algorithm for performing fast event detection on big time series. In most cases, this algorithm presents complexity of $O(\log N)$ for detecting the most significance event, while previous algorithms present at least $O(N^2)$. Also, Lesti and Spiegel [2017] present a sliding window filter of variance of differentiation for enabling fast event detection for streaming time series.

3.5- Event detection performance metrics

Traditional metrics The problem of benchmarking and comparing event detection performance is notoriously difficult [Chandola et al., 2009; Lavin and Ahmad, 2016]. Standard classification metrics are generally used for evaluating the performance of an algorithm to distinguish normal from abnormal data samples [Chandola et al., 2009; Tatbul et al., 2018]. Aminikhanghahi and Cook [2017] review traditional metrics for change point detection performance evaluation, such as Accuracy, Sensitivity (recall), G-mean, F-Measure, ROC, PR-Curve, and MSE. Detection evaluation measures have also been investigated in the areas of sequence data anomaly detection [Chandola et al., 2008], time series mining and representation [Ding et al., 2008], sensor-based human activity learning [Cook and Krishnan, 2015; Ward et al., 2011]. Metrics found in literature are mainly designed to evaluate the detection of punctual anomalies, however, many real-world event occurrences extend over a period of time (range-based). Motivated by this, Tatbul et al. [2018] expand the well-known Precision and Recall metrics to measure accuracy of detection algorithms over event ranges.

Detection delay evaluation Not many works opt to evaluate event detection algorithms based on other than traditional metrics. Nonetheless, analysis of delays are found in studies involving Wireless Sensory Networks (WSNs). For example, Wang et al. [2011] calculate the delay until an event is detected by an individual node and the delivery delay in a transmission network. The work presents a framework for capturing delays in detecting events in large-scale WSN networks with a time-space simulation. Conversely, Tatbul et al. [2018] also observe the neighborhood of event detections, not to calculate detection delays, but to evaluate positional bias in anomaly ranges. Ultimately, no references were found that use the delay measure to evaluate the ability of algorithms to detect real events in time series. This research contributes by introducing new metrics based on the time distance between event detections and identified events. These metrics further a qualitative analysis of the temporal bias of detection algorithms, that is, their tendency to anticipate or delay event detections.

Online detection evaluation Online time series event detection impose unique challenges such as the demand to detect events as soon as possible based on streaming time series. On the other hand, traditional metrics do not incorporate the concept of time and are not designed to reward early detection [Lavin and Ahmad, 2016]. Therefore, standard classification measures (TP, FP, TN, and FN) are not applicable for evaluating the online event detection methods that address the demands described in Section 2.2.3.

Lavin and Ahmad [2016] addressed the lack of proper time series measurement techniques for online anomaly detection in their Numenta Anomaly Benchmark (NAB) work. The NAB scoring system combines anomaly windows, a smooth temporal scoring function (sigmoidal), and application profiles to allow researchers to evaluate online anomaly detector implementations [Ahmad et al., 2017]. It rewards earlier detections and penalizes FPs. However, Singh and Olinsky [2017] extended the discussion providing insights about the challenges and gaps in the current NAB framework which makes it difficult to use in real-world industry applications. For example, determining anomaly window size is based on the number of instances and anomalies in the given dataset. However, especially in streaming environments, these numbers are generally not known in advance. Also, the scoring function equation is not well-defined, presenting arbitrary and unclear choices. Moreover, every user can tweak with the weights in application profiles, which makes it difficult to analyze and benchmark results obtained by different researchers or generated by different experimental setups.

Under these circumstances, currently there is still a demand of metrics for scoring the performance of online event detection methods that incorporate both the concept of time and tolerance to detections that are sufficiently close to time series events. This research contributes by introducing new distance-based metrics inspired by soft computing designed to evaluate the performance of methods for detecting events in streaming time series. They soften the standard classification metrics to incorporate time tolerance or imprecision in event detection. Such metrics support detection method selection based not only on accuracy but also ability to produce detections that are close enough to events.

3.6- Event detection from time series prediction

Among the proposed approaches for addressing events, is a combination of two time series analysis problems: event detection and time series prediction. In this case, events are detected over predicted time series observations. Event detection from time series prediction has been mainly studied in the very recent and prominent area of event prediction [Zhao, 2021]. This approach has been employed for predicting flood and traffic events [Damle and Yalcin, 2007; Wang et al., 2018], cardiac and healthcare events [Henriques et al., 2015; Mehrmolaei, 2021], and wireless sensor networks and internet resource monitoring [Ali et al., 2010; Meenakshi and Raghavan, 2015], respectively. Other examples are set by Amodeo et al. [2011] that use hybrid models for future event prediction based on newspaper publications, and Gholami and Borujerdi [2008] that combines the approach with techniques for fuzzy knowledge discovery from time series data.

Online event detection An approach based on event detection from time series prediction can be exploited for the problem of online event detection. In that case, streaming observations are identified as anomalies based on what is expected by the time series model. A similar approach has been adopted for aerology event detection using both decision trees and neural networks clustering models [Lajevardi and Minaei-Bidgoli, 2008b,a]. Tan and Darken [2015] introduce a technique for learning and predicting a stream of relational time series data for event detection. Ali et al. [2012] focus on efficient predictive monitoring of wireless sensor networks. Both Rahman et al. [2015] and Seth and Chaudhary [2020] provide frameworks for predictive analysis of sensor data streams. Rahman et al. [2015] considers the uncertainty on predicting closure events in shellfish farms, while Seth and Chaudhary [2020] focus on predicting insider trading events.

Nonstationary time series event detection Among the works that perform event detection from time series prediction, not many are designed to deal with the property of nonstationarity which can gravely impact prediction accuracy and the subsequent event detection. In particular, for predicting community evolution, Ilhan and Oguducu [2015] perform time series modeling based on ARIMA, a model that includes a mapping-based nonstationary time series transform based on differencing. Conversely, Rocha et al. [2011]

detects time series patterns using a wavelet-based approach. It adopts a splitting-based nonstationary time series transform based on WT. Detected patterns are used for event detection on telemonitoring data.

3.7- Related surveys

Anomaly detection Due to the importance of this subject and its numerous applications, there have been published several surveys of research on event detection, especially in the last decade [Zhao, 2021]. Among them, most focus on reviewing and comparing methods for detecting anomalies in time series [Hodge and Austin, 2004; Chandola et al., 2009; Gupta et al., 2014; Pimentel et al., 2014; Ding et al., 2014]. Some works present their analysis in a particular context, such as Ahmed et al. [2016] that surveyed anomaly detection techniques in the financial domain, and Aggarwal [2013] that provide comprehensive coverage of the field of outlier analysis from a computer science point of view. Other works focus on methods that can be applied to data presenting specific characteristics. For example, Taha and Hadi [2019] focus on categorical data, Thudumu et al. [2020] focus on high dimensional big data, and Chandola et al. [2008] focus on sequence data events. Alevizos et al. [2017] and Pang et al. [2021], on the other hand, concentrate their reviews on probabilistic methods and deep learning methods, respectively. Wang et al. [2019] chose to analyze the progress of anomaly detection methods from 2000 to 2019.

Several anomaly detection survey papers available in literature also provide taxonomies that introduce a categorization of detection techniques [Chandola et al., 2009; Zhang et al., 2010; Ariyaluran Habeeb et al., 2019; Blázquez-García et al., 2021; Pang et al., 2021]. For example, Zhou et al. [2019] study nonparametric event detection in multiple time series for power distribution networks. They categorize event detection methods as: Domain-based; Signal Processing and Information Theory Based; and Machine Learning Based. Besides the taxonomy, Sebestyen et al. [2018] also propose a platform that allows a developer to find and tune a given anomaly detection method that is optimal for an application.

Regarding the context of online event detection, a smaller number of survey papers

focus attention on reviewing methods that can be applied to anomaly detection over wireless sensor networks data [Zhang et al., 2010]. Ariyaluran Habeeb et al. [2019] survey real-time big data processing methods for anomaly detection. Munir et al. [2019] also provide a comparative analysis of traditional and deep learning-based anomaly detection methods for streaming data.

Change point detection Conversely, not all surveys found in literature focus on anomaly events. Aminikhanghahi and Cook [2017] present a survey of methods for time series change point detection. Also, Truong et al. [2020] review algorithms for the offline detection of multiple change points in multivariate time series. Change point detection is related to the monitoring of concept drifts as both problems focus on the observation of considerable changes in time series behavior. In particular, concept drifts may happen due to the occurrence of environmental events such as change points. In this context, Gama et al. [2014] reviews existing strategies for handling concept drift using adaptive learning processes, while Pratama et al. [2017] also presents incremental learning techniques using recurrent fuzzy neural networks. Other surveys approach concept drifts in the context of nonstationarity by presenting efficient algorithms and applications for learning and adapting to evolving environments [Sayed-Mouchaweh and Lughofer, 2012; Ditzler et al., 2015].

Event detection from time series prediction In recent years, a considerable amount of research has been dedicated to the monitoring of events regarding both method development and applications. Among them, many approach the problem by performing event detection from time series prediction. Unfortunately, it is difficult to assess these works across different domains [Zhao, 2021]. To address this gap, survey papers have been published presenting analytical reviews of research and methods for event monitoring and anticipation in time series data [Yan et al., 2004; Molaei and Keyvanpour, 2015; Mehrmolaei and Keyvanpourr, 2015; Gmati et al., 2019]. Particularly, some works focus on methods for streaming time series. Salfner et al. [2010] provide a survey of online failure prediction methods, while Zhao [2021] present a systematic survey of methods for time series events in the Big Data era. As methods for event detection from time series prediction are typically motivated by specific application domains, Zhao [2021] list a number of other surveys in domains such as flood events, social unrest, wind power ramp forecasting, tornado events, temporal events without location information, online failures,

and business failures. Taxonomies are derived by Salfner et al. [2010], Gmati et al. [2019], and Zhao [2021].

3.8- Considerations

Generally, the solutions to an online event detection problem demand a potentially long time to gather sufficient data to allow the identification of detectable patterns. In turn, the gathered datasets are usually unbalanced, which makes it difficult to learn rare abnormal patterns. Also, in the presence of nonstationarity, the learned time series patterns can change gradually or even abruptly.

In this context, an approach based on event detection from time series prediction is advantageous since its application only demands the availability of a time series long enough to train a prediction model with the required accuracy. Such model is then used to guide the identification of abnormal observations. Nonetheless, the methods currently available for online event detection from time series prediction still do not take into consideration the nonstationarity in time series.

This research contributes by introducing a novel method that uses inertial analysis and lagged time series prediction for online event detection in nonstationary time series. The proposed method aims for the early detection of change points in streaming time series.

4- Herald

Henceforth, this thesis explores the adoption of event detection from time series prediction as a solution to the problem of online event detection. In particular, the focus of the proposed method lies on the use of time series decomposition for deriving its characteristic trend component. The trend component is used for inertial analysis and given as input to a time series prediction process. Moreover, prediction is performed with a lagged horizon, that is, starting before the current time. A lagged prediction horizon allows an analysis of expectations regarding observations surrounding and including current time. Predicted observations serve as reference for online nonstationary time series event detection.

The main use-case scenario of this approach encompasses the availability of a time series dataset (possibly presenting nonstationarity) (i) with streaming acquisition; (ii) prone to the occurrence of rare abnormalous events and concept-drifts (possibly containing limited amount of identifiable abnormalous patterns). Such dataset is to serve as input to an online event detection application, where reaction response is critical, demanding of (i) detection accuracy; (ii) robustness to unbalanced data and concept-drifts; and (iii) fast and early detection of events.

The proposed approach for this scenario generates the *Herald* detection method, depicted in Figure 13. The detection process conducted by Herald is composed of three main steps: time series decomposition; prediction of time series inertial component; and online event detection from predicted observations. The specific input parameters of the approach and its produced artifacts are detailed in Table 5 and Table 6, respectively. The Herald method is iterative and is meant to run on each current time t, having the most recent subsequence $W = seq_{t,p}(X)$ of the streaming time series X as input. The last step of the method, regarding event detection takes the next observation of X, x_{t+1}, to evaluate deviation from the generated predictive model. The next sections give an intuition behind the Herald method and describe each step of the Herald method.

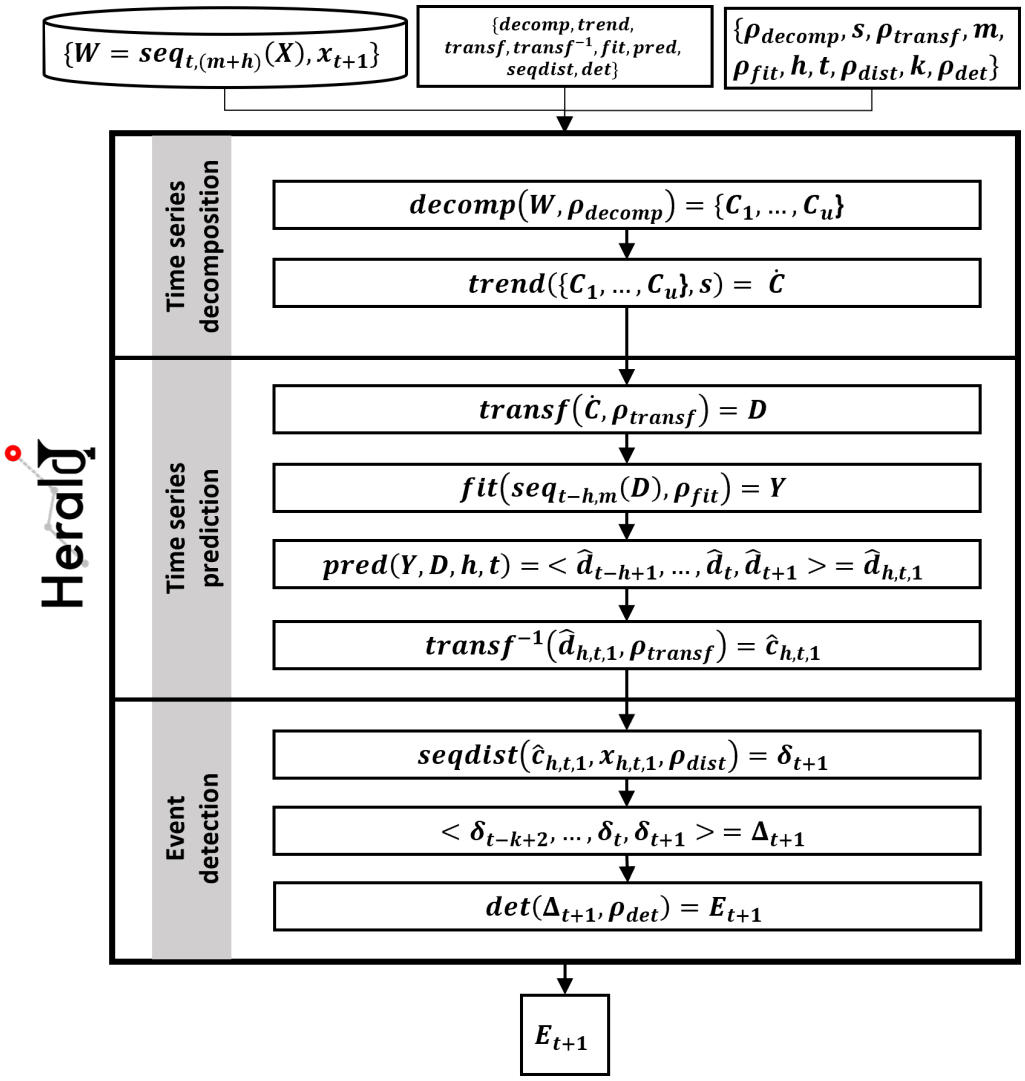


Figure 13 – General steps to the Herald method for online time series event detection

4.1- Method intuition

Figure 14 gives an intuition behind the general behavior of the Herald method for online time series event detection in comparison to the most common approaches to event detection from time series prediction. It is depicted a specific window $W = seq_{t,p}(X)$ of time series X and its next observation x_{t+1}. The usual approach fit a model to the whole window W (solid orange line) and predict the next observation x_{t+1} (dashed orange line) ($h = 0$). On the other hand, the Herald method defines $h = 1$, fitting the model Y based on the trend component of the subsequence <$x_{t-(p-1)}, \dots, x_{t-1}$> (solid blue line). It duplicates the prediction horizon and produces a prediction even to the previously known

Table 5 – Method input parameters

Class	Object	Description
Data	$W = seq_{t,(m+h)}(X)$	Subsequence of size $(m+h)$ of X ending in current time t
	x_{t+1}	Observation of X in time $t+1$
Functions	$decomp(X, \rho_{decomp})$	Function for decomposing the time series X into u components of length $\|X\|$ having different frequencies considering the parameters ρ_{decomp}
	$trend(\{C_1, \ldots, C_u\}, s)$	Function for selecting the s-th ($s = 1, \ldots, u$) component of the decomposed time series X, $\{C_1, \ldots, C_u\}$, as representative of its inherent trend and inertial component
	$transf(X, \rho_{transf})$	Function for transforming the time series X into another representation considering the parameters ρ_{transf}
	$transf^{-1}(\hat{X}, \rho_{transf})$	Function for reverse transforming the time series $\hat{X} = transf(X, \rho_{transf})$ back into X considering the same parameters ρ_{transf}
	$fit(X, \rho_{fit})$	Function for training a model Y based on the time series X considering the parameters ρ_{fit}
	$pred(Y, X, h, t)$	Function for predicting the next $h+1$ observations of the time series X from x_{t-h} (<$x_{t-h+1}, \ldots, x_t, x_{t+1}$>) based on the model Y
	$seqdist(X, \hat{X}, \rho_{dist})$	Function for computing the sequence distance between the time series X and $\hat{x}$ considering the parameters ρ_{dist}
	$det(X, \rho_{det})$	Function for detecting events in the time series X considering the parameters ρ_{det}
Parameters	ρ_{decomp}	Parameters for function $decomp$
	ρ_{transf}	Parameters for function $transf$ and $transf^{-1}$
	ρ_{fit}	Parameters for function fit
	ρ_{dist}	Parameters for function $seqdist$
	ρ_{det}	Parameters for function det
	s	Index of the component of the decomposed time series X to be selected
	m	Size of model input data passed to fit
	h	Prediction horizon preceding time t
	t	Current time t
	k	Memory size, that is, the size of subsequences to be inspected for events

observation x_t from model Y (dashed blue line).

In the scenario of Figure 14, X is at the early stages of a incremental concept-drift that initiated at x_t. However, the predictive model of the usual approach fits to x_t as a part of the characteristic behavior of the time series. Consequently, its prediction $\hat{x}_{t+1}$ deviates from x_{t+1} by δ'_{t+1} which does not surpass the acceptable threshold (green range). The Herald method, conversely, analyzes the neighborhood of x_t, including earlier deviations from the time series inertia since before the current time t. Thus, it is possible to observe that x_{t+1} deviates from the expected $\hat{c}_{t+1}$, based on the series inertia, by a difference of δ_{t+1}, which is greater than the threshold and requires an event signalization. While the

Table 6 – Artifacts produced by the method

Object	Value	Description
$\{C_1, \ldots, C_u\}$	$decomp(W, \rho_{decomp})$	Set of u components of W of length $\|W\|$ having different frequencies
$\dot{C}$	$trend(\{C_1, \ldots, C_u\}, s)$	The s-th ($s = 1, \ldots, u$) component of the decomposed time series X, $\{C_1, \ldots, C_u\}$, selected as representative of its inherent trend and inertial component
D	$transf(\dot{C}, \rho_{transf})$	Transformed representation of $\dot{C}$
Y	$fit(seq_{t-h,m}(D), \rho_{fit})$	Fitted model of $seq_{t-h,m}(D)$, the subsequence of D ending in d_{t-h} (<$d_{t-(m+h-1)}, \ldots, d_{t-h}$>)
$\hat{d}_{h,t,1}$	$pred(Y, D, h, t)$ = <$\hat{d}_{t-h+1}, \ldots, \hat{d}_t, \hat{d}_{t+1}$>	Set of the next $h+1$ observations of D from d_{t-h} based on Y
$\hat{c}_{h,t,1}$	$transf^{-1}(\hat{d}_{h,t,1}, \rho_{transf})$	Reverse transformed representation of $\hat{d}_{h,t,1}$ back into $\dot{C}$ scale
δ_{t+1}	$seqdist(\hat{c}_{h,t,1}, x_{h,t,1}, \rho_{dist})$	Sequence distance between $\hat{c}_{h,t,1}$ and the corresponding observations from X, $x_{h,t,1}$
Δ_{t+1}	<$\delta_{t-k+2}, \ldots, \delta_t, \delta_{t+1}$>	Subsequence of size k of the latest computed sequence distances including δ_{t+1}
E_{t+1}	$det(\Delta_{t+1}, \rho_{det})$ = <$e_{t-k+2}, \ldots, e_t, e_{t+1}$>	Logical sequence indicating a possible event identified in Δ_{t+1} at time $\hat{t}_{cp}$ ($t-k+2 < \hat{t}_{cp} < t+1$)

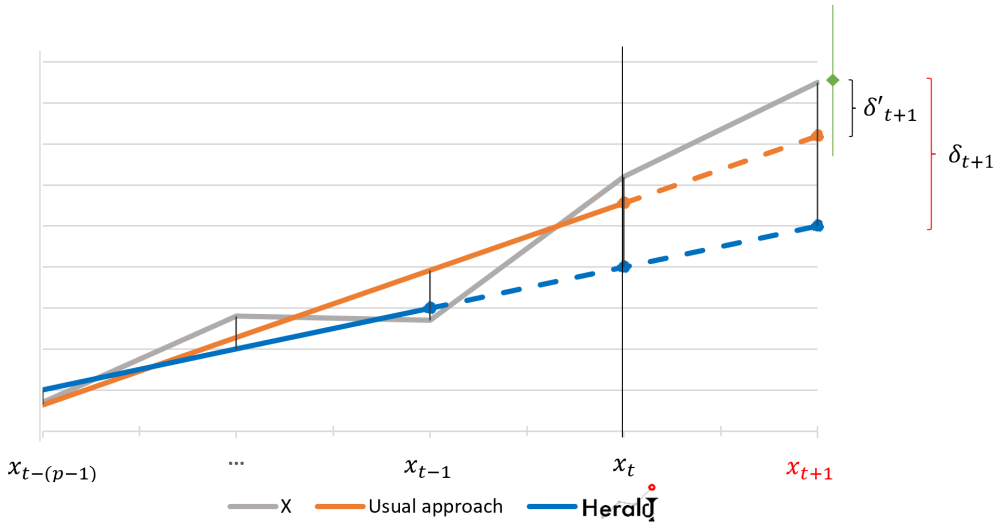


Figure 14 – Intuition behind the general behavior of the Herald method for online time series event detection in comparison to the most common approaches to event detection from time series prediction

usual approach would not indicate the occurrence of an event at time $t+1$, demanding a longer detection lag, Herald is capable of enabling the earlier observation of incremental drifts in the last identified concept. The Herald method is detailed in the next sections.

4.2- Online detection process

4.2.1- Time series decomposition

The first step to Herald encompasses the decomposition of the most recent input subsequence W into u components, $\{C_1, \ldots, C_u\}$, of length $|W| = m + h$ having different frequencies (scales). Commonly, these components are obtained by splitting-based time series transformations (Section 2.1.1), given by function $decomp$. In this case, $decomp$ may represent for example, the FT frequency transformation, the WT or EMD time-frequency transformations, or the HHT time-frequency-energy transformation.

An advantage provided by time series decomposition is that the derived components are more physically meaningful and generally simpler to explain, besides being more easily modeled, which can simplify the time series prediction problem [Dudek, 2016]. High-frequency (short-term) components may represent noise and volatility of the process that generates the time series, seasonal components represent its cyclic patterns, and low frequency (long-term) components represent its inherent trend [Shumway and Stoffer, 2017]. Particularly, the trend component of a time series is a proxy of the inertia intrinsic to its generating process. Therefore, is less susceptible to volatility and incremental concept-drifts in time series observations, maintaining the memory of the undergoing behavior of the process until the occurrence of an event that causes enough impact to its trend [Gujarati, 2021]. The trend component of the time series $\dot{C}$, is selected from $\{C_1, \ldots, C_u\}$ based on the function $trend$. It is the s-th ($s = 1, \ldots, u$) low frequency component of W.

Alternatively, the low-frequency trend component of W can also be obtained by specialized mapping-based transformations. An example is given by MAS. Given as input to $decomp$, MAS returns the unitary set $\{C_1\}$ ($u = 1$), ultimately selected as the trend component of the time series $\dot{C}$.

4.2.2- Prediction of time series inertial component

The trend component series $\dot{C}$ is used for inertial analysis of the monitored process and is given as input for time series prediction. It can serve as a simpler and more stable input for time series prediction methods [Dudek, 2016], mitigating the effects of spurious values and the demand for frequent model updates in incremental learning systems. The $\dot{C}$ is submitted to a systematic process of time series prediction based on a particular setup of preprocessing method and prediction model [Salles and Ogasawara, 2021]. Refining such preprocessing-model setup and/or its parameters is generally crucial to obtain a more accurate model and lower prediction errors. This process may be repeated if the evaluated time series prediction setup does not reach the desired performance.

The first activity of the time series prediction process refers to performing data preprocessing by applying a time series transformation method defined by function $transf$. The function $transf$ represents a mapping-based time series transformation method such as LT, BCT, PCT and DIF, described by Salles et al. [2019]. The transformed time series D is returned.

Next, the subsequence of D, $seq_{t-h,m}(D)$ =<$d_{t-(m+h-1)}, \ldots, d_{t-h}$>, is given as input for function fit responsible for training the time series prediction model returned as Y. In this context, the model Y can be derived based on (i) retraining at each new input data, or (ii) being the subject of an incremental learning process. The latter involves the computation of a loss function based on the adjustment of Y. The model Y corresponds either to (i) a statistical model, such as ARIMA, SM, ETS, TF, and GARCH; or (ii) a machine learning model, such as NNET, MLP, ELM, RBF, SVM, RFrst, and the deep learning models CNN and LSTM. The trained predictive model Y is then used for performing the third activity of the process, namely the prediction of the next time series observations.

The predictive model training activity is affected by the input data $seq_{t-h,m}(D)$, which in turn depends on the definition of the size m. The setting of m defines the memory of the prediction process and can influence the ability of the methods to accurately predict X. The model input size m, which helps define the size $m + h$ of $W = seq_{t,(m+h)}(X)$, can be fixed (*batch memory*) or it can increase with the data stream $m = |t| - h$, where $|t|$ is the number of observations of X until the considered current time t) (*full memory*). In that case, Herald takes as input the initial value of the parameter m as the minimum model

size, which also defines the minimum size of the first input data window W, henceforth referenced as the *warm-up* data [Alexopoulos, 2007; Law, 2020].

The function $pred$ applies the previously fitted model Y to an evaluation set derived from D to obtain $\hat{d}_{h,t,1}$, the set of the next $h+1$ observations of D from d_{t-h}, that is, the subsequence <$\hat{d}_{t-h+1}, \ldots, \hat{d}_t, \hat{d}_{t+1}$>. Given the current time t, h gives the prediction horizon preceding t. In this context, the Herald method differs from the mainstream since the general approaches to event detection from time series prediction usually fit a model to the whole window W, and use it to predict the next observation w_{t+1} ($h = 0$). Starting the prediction horizon at $t - h$ enables a broader analysis of observations surrounding and including current time t compared to what is expected from the previously observed time series inertia.

Finally, the set $\hat{d}_{h,t,1}$ is submitted to the last activity of the process corresponding to the postprocessing of predictions, reversing transformations applied to the time series data by $transf$ in the first activity. The reverse transformation is done by the function $transf^{-1}$ that receives the same parameters as $transf$. The returned set $\hat{c}_{h,t,1}$ represents the predicted observations $\hat{d}_{h,t,1}$ back into the original scale of the trend component $\dot{C}$.

4.2.3- Online event detection from time series prediction

The predictions in the set $\hat{c}_{h,t,1}$ serve as expectations for the observations $x_{h,t,1} =$ <$x_{t-h+1}, \ldots, x_t, x_{t+1}$>, according to the inherent inertia of the observed time series. A measure of deviation from expected behavior is computed by the function $seqdist$, which returns δ_{t+1}, the sequence distance between $\hat{c}_{h,t,1}$ and $x_{h,t,1}$. The function $seqdist$ may represent typical distance measures including Euclidean distance (EDist), correlation coefficient, and dynamic time warping (DTW) [Mueen, 2014; Ding et al., 2008]. The k latest computed sequence distances (<$\delta_{t-k+2}, \ldots, \delta_t, \delta_{t+1}$>) are maintained and represented by Δ_{t+1}. In this case, k gives the memory size of deviations contained in Δ_{t+1}. In the presence of a incremental concept drift, the time series increasingly deviates from expected behavior, as a consequence the latest sequence distances in Δ_{t+1} tend to increase accordingly, possibly characterizing a structural break.

The function det implements a change point or structural break detection method

such as the ones discussed in Section 2.2.2. It is responsible for evaluating Δ_{t+1} and determining the relevance of the deviations from the time series inertia. It is based on a threshold, which is given as a parameter to the method in ρ_{det}, or automatically computed depending on the implemented detection method (see Section 2.2.2). Generally, the threshold is a domain-based acceptable value range, or it can be defined in terms of prediction error metrics such as MSE, Normalized MSE (NMSE), Mean Absolute Percentage Error (MAPE), sMAPE, and maximal error [Davydenko and Fildes, 2013]. If deviations from expected behavior contained in Δ_{t+1} surpass a threshold, it may indicate the occurrence of an event in the time window $\{t-k+2,\ldots,t+1\}$ or the imminence of an event in the near future. In that case, det returns at time $t+1$ a set E_{t+1}, a logical sequence <$e_{t-k+2},\ldots,e_t,e_{t+1}$> indicating the occurrence of a possible change point event (breakpoint) at time $\hat{t}_{cp}$ ($t-k+2<\hat{t}_{cp}<t+1$). The set E_{t+1} serves then as the herald of a concept drift.

The parameterization of the functions $decomp$, $transf$, fit, and det by the definition of ρ_{decomp}, ρ_{transf}, ρ_{fit}, and ρ_{det}, respectively, can be selected based on four different strategies, namely by (i) statistical tests, (ii) heuristics, (iii) hyperparameter optimization, and (iv) domain expertise. For example, autocorrelation tests can be used for indicating adequate input sizes for neural networks and for indicating the nonstationarity in time series; the central limit theorem can be used as an heuristic for the minimum definition of m and k; methods such as grid search and random search can be used for exploring the search space for each parameter combination and optimize their definition; finally, the parameters can be defined based on domain expert knowledge.

4.2.4- On the use of frameworks

Possible input functions of Herald methods are implemented and currently available in the frameworks *TSPred* [Salles et al., 2022] and *Harbinger* [Salles et al., 2020]. Both frameworks are part of the preliminary results of this research. TSPred is a framework for nonstationary time series prediction. It establishes a prediction process that seamlessly integrates nonstationary time series transformation and state-of-the-art statistical and machine learning methods. It is made available as an R-package [Salles and Ogasawara,

2021], which provides functions for defining and conducting time series prediction, including data pre(post) processing, decomposition, modeling, prediction, and accuracy assessment. It also enables user-defined methods, expanding its applicability.

On the other hand, Harbinger is a framework for the integration and analysis of event detection methods. Such integration enables a qualitative and quantitative comparison of their detections, as well as the combined use of the methods to more assertively characterize the detected events. It was designed to foster the selection of event detection methods and their parameters, as well as understanding the nature of detected events. The framework includes the implementation of several of the event detection methods discussed in Section 2.2.2. Similarly to TSPred, it also enables user-defined detection methods. More details on Harbinger are presented by Salles et al. [2020].

In particular, possible input functions of the proposed approach and comparative analysis regarding time series prediction, namely $transf$, fit, $pred$, and, $transf_{-1}$ can be defined from the available implementations in the TSPred framework. The decomposition function $decomp$ can also be implemented as a splitting-based time series transformation in TSPred. As for the event detection function det, it can be defined from the methods available in the Harbinger framework. Ultimately, this research extended the range of methods implemented in both frameworks on demand during the development and evaluation of Herald. Moreover, the Herald method itself is to be included in the Harbinger framework.

4.3- Proposed instantiation

This section presents the proposed definitions for the particular instantiation of the Herald method produced for this thesis. The formalized Herald method can be implemented in different versions. As stated in Section 4.2, the functions given as input to Herald can be set as a number of different possible methods. Which, in turn, have their own characteristic parameters to be set accordingly. However, it is not in the scope of this research to evaluate the possible instantiations of Herald and the scenarios in which they bring the most benefit to event detection. The proposed Herald instantiation described in this section focuses on creating a basic version of Herald, using simple and established

methods, therefore minimizing demands of parameter optimization and biases regarding different modeling and detection method performances. The focus lies on exploring the contribution of an inertial analysis and lagged prediction of time series to event detection. The instantiation of Herald described in this section is performed in R and is made publicly available at Github[1]. The pseudocode for the proposed Herald instantiation is given by Algorithm 1, and it is discussed in detail in the remainder of this section.

Algorithm 1 – Proposed Herald instantiation

Input: $W = seq_{t,(m+h)}(X) =$ vector of time series data;
$t =$ current time step;
$m = |t| - h$ minimum model input size, defining warm-up size;
$o =$ order of the moving average process;
$\rho_{fit} =$ parameters for AR(1) model;
$h =$ prediction horizon preceding time t;
$k =$ memory size parameter;
$\rho_{det} =$ parameters for supF tests;
Output: $E_{t+1} =$ possible events identified in Δ_{t+1};

```
begin
    foreach t do
        Ċ ← MAS(W, o);
        Y ← AR(seq_{t−h,m}(Ċ), ρ_fit = {φ_1 = 1, ε_t = 0});
        ĉ_{h,t,1} ← pred(Y, Ċ, h, t);
        δ_{t+1} ← EuclideanDist(ĉ_{h,t,1}, x_{h,t,1});
        Δ_{t+1} ←<δ_{t−k+2}, ..., δ_t, δ_{t+1}>;
        E_{t+1} ← supF(Δ_{t+1}, ρ_det);
    return E_{t+1};
```

Data input The proposed instantiation of Herald assumes an online event detection application in which the time series data stream receives the addition of new observations one by one, that is, in batches of one observation. New observations are included in the data window W given as input to the method. Every newly included observation is considered in time $t+1$. Herald is then applied iteratively for each t and $W = seq_{t,(m+h)}(X)$. Particularly, a full memory approach is adopted (model input size $m = |t| - h$). It is a simple solution as it does not demand the choice of a fixed parameter for m whose arbitrary definition can influence the prediction accuracy. In that case, m is set as an initial value, which is used to define the warm-up data that is sufficient for producing the first model Y. This time series data stream and online event detection application is simulated and

[1]Herald implementation, datasets, and experiment codes: `https://github.com/cefet-rj-dal/herald`

managed by the Nexus framework implemented in R and currently available in GitHub[2].

Decomposition Given W, it is submitted to the first step of time series decomposition performed by $decomp$. The method MAS is given as input to $decomp$, as it is a simple and well-known method for highlighting evolving behavior and long-term trends. Accordingly, ρ_{decomp} includes o, the order of the moving average process. It returns the unitary set $\{C_1\}$ ($u = 1$) containing the low-frequency trend component of W selected by $trend$ as the inertial component of X, $\dot{C}$ (line 3 of Algorithm 1). The MAS method is available in the TSPred R-package [Salles and Ogasawara, 2021].

Prediction As $\dot{C}$ is already the result of a mapping-based transformation of W, $transf$ and $transf^{-1}$ are set as identity functions ($\rho_{transf} = \{\}$). Although it is possible and at times beneficial to combine and perform sequential data transformations [Salles et al., 2019], the exploration of preprocessing methods is not the focus of this instantiation. Next, the function fit takes $seq_{t-h,m}(D)$ ($D = \dot{C}$) to generate model Y. As discussed in Section 2.1.2, there are several methods available in the literature for time series modeling and prediction. However, they can vary in modeling accuracy, demanding further exploration, and state-of-the-art methods commonly require the optimization of several parameters.

In this context, this instantiation adopts causal regression models, which are established and often preferred when projections of related variables are available [Verma and Pearl, 1991], which is the case for time series that are usually autoregressive. This instantiation employs a simple autoregressive model AR(1) defined by $y_t = \varphi_1 x_{t-1} + \varepsilon_t$ (with parameters $\rho_{fit} = \{\varphi_1 = 1, \varepsilon_t = 0\}$), that uses the historical time series data to obtain the future time series and is retrained at each new input data (line 4 of Algorithm 1). The function $pred$ takes Y and D as input and returns $\hat{d}_{h,t,1}$, the $h + 1$ predictions for <$\hat{d}_{t-h+1}, \ldots, \hat{d}_t, \hat{d}_{t+1}$>, by extrapolating Y given the formula $\hat{d}_{t+1} = y_t$ (line 5 of Algorithm 1).

Detection The predicted subsequence $\hat{c}_{h,t,1} = \hat{d}_{h,t,1}$ is then compared to the corresponding subsequence $x_{h,t,1}$ based on a Euclidean distance measure ($\rho_{dist} = \{\}$). The returned δ_{t+1} distance (line 6 of Algorithm 1) is included in Δ_{t+1} =<$\delta_{t-k+2}, \ldots, \delta_t, \delta_{t+1}$>. For that, the memory size parameter k is given as input (line 7 of Algorithm 1). The k

[2] `https://github.com/cefet-rj-dal-dev/nexus`

latest computed sequence distances in Δ_{t+1} are submitted to a detection process given by the function det.

As Herald specializes in detecting concept drifts regarding time series inherent long-term trends, this instantiation focuses on the detection of change points that mark the occurrence of structural breaks in trend. Such change points can be identified in Δ_{t+1}, since deviations from expected behavior and corresponding sequence distances tend to increase in the occurrence of a incremental concept drift. In this context, the det function is defined as the $supF$ test for structural breaks as described in Section 2.2.2 (line 8 of Algorithm 1). It is an established, easy-to-interpret, and widely adopted test, being designed for cases in which the (potential) change point is unknown. Based on Δ_{t+1}, the det function computes F statistics F_i for $t - k + 2 < t_a \leq i \leq t_b < t + 1$, where t_a and t_b define the interval evaluated for change points, that is, $\hat{t}_{cp}$ lies in the interval $[t_a, t_b]$.

The det function instantiation uses the *Fstats* function from the *strucchange* R-package [Zeileis et al., 2002]. The *strucchange* R-package provides tools to display information about structural changes in regression relationships and to assess their significance. All procedures in this package are concerned with testing or assessing deviations from stability in the classical linear regression model. By default, *Fstats* sets the interval $[t_a, t_b]$ corresponding to 70% of the sample size, that is, both $[t - k + 2, t_a)$ and $(t_b, t + 1]$ correspond to 15% of the size of $|\Delta_{t+1}|$. Based on the computed F statistics, the $supF$ statistic is calculated by the *sctest* function. The null hypothesis of no structural break in the interval $[t_a, t_b]$ is rejected when there is evidence for a structural change at the level $\alpha = 0.05$. The actual change point event $\hat{t}_{cp}$ is estimated by maximizing F_i [Zeileis et al., 2002]. Finally, E_{t+1} is returned (line 9 of Algorithm 1) with the possible events identified in Δ_{t+1}.

5- Experimental evaluation

This research includes the experimental evaluation of the Herald approach as a solution to the problem of online event detection. In particular, Herald is evaluated regarding its ability to outperform current methods regarding fast, early and accurate online detection of change points, while also providing robustness to unbalanced data and incremental concept drifts. For that, we devise an experimental and comparative analysis of the detection results of Herald against traditional event detection from time series prediction (prediction approach) as described in Section 2.3.1, henceforth referenced as *EDfP*, and a baseline event detection from the monitoring of a parametric linear regression model (regression approach) as described in Section 2.2.2, henceforth referenced as *EDfR*.

All methods are applied to streaming time series datasets, presenting nonstationarity and containing concept drifts. The adopted datasets correspond to simulated nonstationary times series containing change point events that mark a change in the inherent trend. The simulated time series trends follow different characteristic curves and have different levels of volatility to help further a more thorough discussion on the effects of the evaluated detection methods on a diverse range of representative time series. Herald is expected to contribute by reducing event detection lags and increasing sensitivity to incremental concept drifts. The next section describes the devised experimental comparative analysis of Herald against methods EDfP and EDfR.

5.1- Comparative analysis

Experiments conducted for the application of Herald to the adopted datasets follow the general steps described in Chapter 4 and summarized in Figure 13. The evaluated instantiation of Herald is detailed in Section 4.3. It uses simple and established methods, minimizing the need for parameter optimization and biases regarding different modeling and detection method performances. Thus, the focus lies on studying whether the inertial

analysis and lagged prediction of time series performed by Herald can contribute to the change point detection problem.

In this context, a similar approach is adopted for the application of the EDfP method. However, the main difference lies in the fact that EDfP methods (i) do not include the first step regarding time series decomposition, taking the original data window W as input for time series prediction, and (ii) do not include the definition of the parameter h for the lagged prediction horizon preceding time t, being fixed as 0. On the other hand, for the application of the EDfR method, focusing on the deviation from a fitted model, besides (i) and (ii) it also differs by (iii) not performing any time series prediction, and consequently no sequence differences are calculated from predicted observations. Figure 15 presents the general steps to the application of Herald, EDfP, and EDfR methods for comparative analysis in online time series event detection.

Given a streaming time series X of an adopted dataset, it is iteratively consumed by Herald, EDfP, and EDfR methods in terms of its most recent subsequence W of size $m + h$. Herald performs time series decomposition by MAS given as input to function $decomp$. The order of the moving average o is included in ρ_{decomp}. Next, the resulting inertial trend component of W, $\dot{C} = C_1$ is given as input to time series prediction. EDfP and EDfR methods take W for model fitting directly.

Next, the transformed time series D, and D'/D'' are derived from $\dot{C}$ or W for Herald and EDfP/EDfR methods, respectively. In this case, D and D'/D'' are the result of the identity function given as input to $transf$. Herald takes the subsequence of D, $seq_{t-h,m}(D)$ to train the prediction model Y based on function fit, while EDfP trains Y' using the whole series D'. For both, the models Y and Y' correspond to an AR(1) model retrained for each new input data.

Herald uses the trained model Y to predict the set $\hat{d}_{h,t,1}$, the next $h+1$ observations of D from d_{t-h}. EDfP sets $h = 0$ and predicts the set $\hat{d'}_{t,1}$ =<$\hat{d'}_{t+1}$> based on Y'. Next, the sets $\hat{d}_{h,t,1}$ and $\hat{d'}_{t,1}$ are postprocessed. As $transf/transf^{-1}$ are identity functions, the returned sets $\hat{c}_{h,t,1}$ and $\hat{w}_{t,1}$ are kept in the original scales of the trend component $\dot{C}$ and the data window W, respectively.

The predictions in $\hat{c}_{h,t,1}$ and $\hat{w}_{t,1}$ are compared to the corresponding subsequence $x_{h,t,1}$ and x_{t+1} based on a Euclidean distance measure, returning distances δ_{t+1} and δ'_{t+1}, which are included in the sets Δ_{t+1} and Δ'_{t+1}, respectively. For both Herald and EDfP, a detection process given by the function det is performed based on the k latest computed

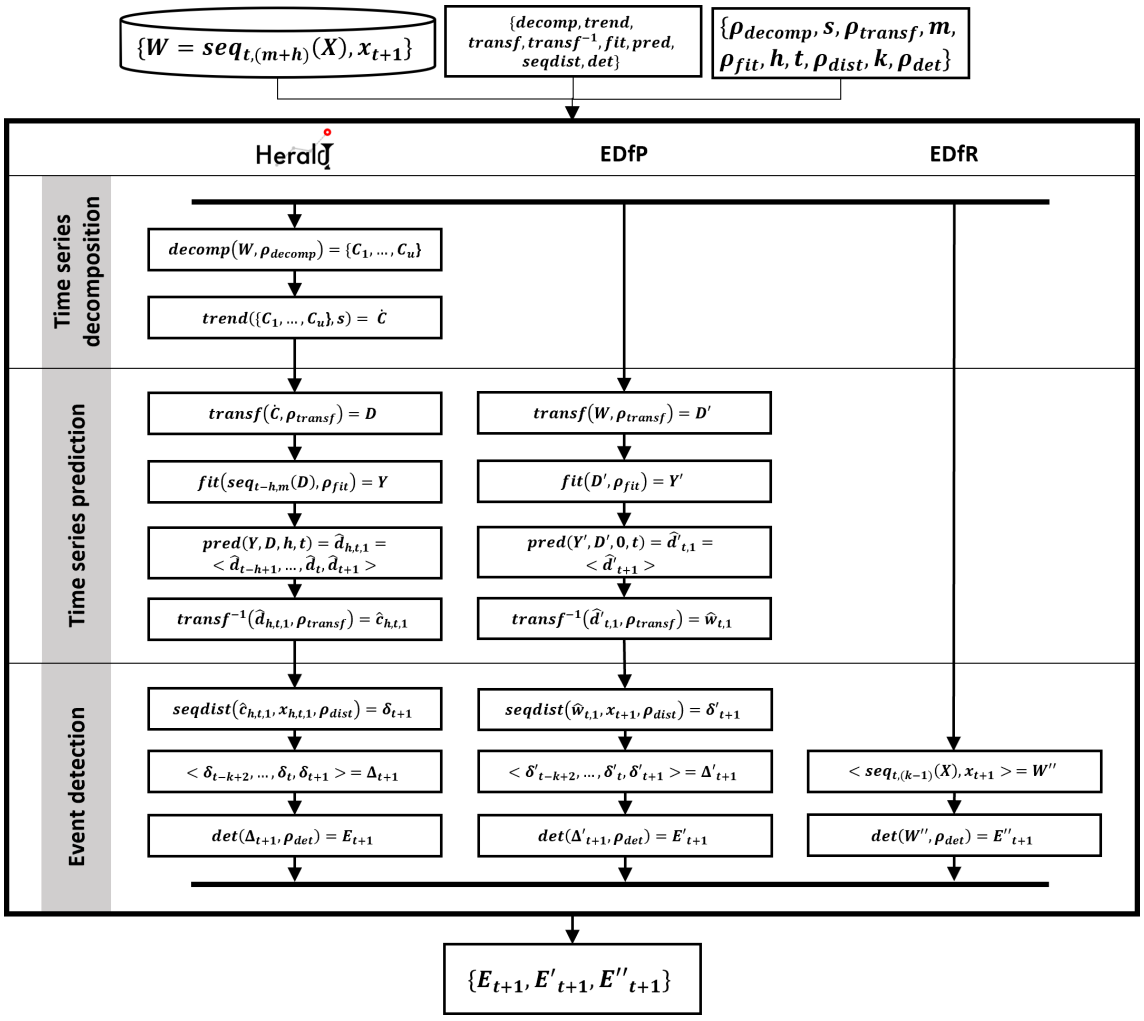


Figure 15 – General steps to the comparative analysis of Herald against methods based on event detection from time series prediction (EDfP) and methods based on model deviation (EDfR) for online time series event detection

sequence distances contained in Δ_{t+1} and Δ'_{t+1}, respectively. Gradual concept drifts lead to an increasing deviation from expected behavior which is represented in the latest sequence distances in Δ_{t+1}, which tend to increase accordingly, possibly characterizing a structural break. For the method EDfR, since it does not include the prediction of the time series, det performs event detection based on W'', the k latest observations from X including x_{t+1}.

As this research focuses on the problem of detecting change points regarding time series inherent long-term trends, the det function corresponds to the $supF$ test for structural breaks as described in Section 2.2.2. The $supF$ test is easy-to-interpret and one of the most widely adopted tests for structural breaks currently available in the literature. Based on Δ_{t+1}, Δ'_{t+1}, and W'', respectively, the det function fits classical linear regression

models and computes F statistics that are designed to assess deviations from model stability. The $supF$ test evaluates the F statistics and provides evidence of a structural change at level α ($\rho_{det} = \{\alpha\}$). In that case, the point $\hat{t}_{cp}$ that maximizes the F statistics is considered a change point event and is indicated as a positive signal in the sets E_{t+1}, E'_{t+1}, and E''_{t+1} at time $t+1$, respectively.

The det function is the only one which is common for all three methods. In that case, event detection for the EDfR method is solely based on deviations from the linear regression model fitted by the $supF$ test. Henceforth, EDfR is used as a baseline for both Herald and EDfP [Zeileis, 2005]. At this point, the results from Herald, EDfP, and EDfR methods (E_{t+1}, E'_{t+1}, and E''_{t+1}) can be used to compare the methods regarding their ability to produce both early and accurate online detection of change points. In particular, Herald is expected to be able to detect change points in nonstationary time series, producing competitive results in detection accuracy and lower detection lags while being robust to incremental concept-drifts.

5.2- Experimental settings

This section describes the experimental settings adopted for the devised experimental comparative analysis of Herald against methods EDfP and EDfR. The characteristic time series contained in the adopted simulated datasets are presented, as well as their construction strategy. Furthermore, relevant definitions regarding experimental implementation and parameterization are enumerated. Finally, this section describes the evaluation metrics and statistical significance tests adopted in order to compare and discuss the results of Herald against the other methods.

5.2.1- Datasets

The literature presents benchmark datasets designed for evaluating event detection methods for both static and streaming time series. These datasets are widely available

and are composed of simulated and real-world time series regarding several different domain applications such as water quality monitoring (GECCO) [Rehbach et al., 2018], network service traffic (Yahoo) [Webscope, 2015], oil well exploration (3W), social media (NAB) [Ahmad et al., 2017], network data traffic (RARE) [Lomio et al., 2020], and medicine (UCR) [Wu and Keogh, 2021], among others. The datasets present a reasonable number of representative time series presenting different types of events, as well as different types of nonstationarity and statistical properties.

This research studied the available benchmark datasets in order to analyze the time series that most approximate to the main use-case scenario of Herald, described in Chapter 4. In particular, further attention was given to the 3W dataset collected and provided by Petrobras. It contains real events in oil well-exploring processes, where reaction response is critical. Each series in the 3W dataset contains a single change point that marks an undesirable event, which causes a concept-drift in the time series trend.

Inspired by the 3W dataset, this section presents the simulated datasets produced for evaluating the Herald and EDfP against EDfR methods. The datasets were developed for an online application, being passed to the methods as streams of data. They consist of nonstationary times series containing change point events that mark a concept drift in the inherent trend. The simulated trends follow different characteristic and traditional curve functions. In addition, unlike 3W, all time series were reproduced with different levels of volatility [Gujarati, 2021] to help further a more thorough discussion on the effects of the evaluated detection methods on a diverse range of representative time series.

For the first dataset, henceforth called *CP1*, was developed time series containing a sequence of 200 observations. The first 100 observations of the time series follow a stationary process with a constant mean. On the other hand, the last 100 observations follow a nonstationary process presenting a non-constant trend corresponding to a characteristic curve function namely: linear, logarithmic, square root, second order polynomial, third order polynomial, exponential, and sigmoid. In that case, the ground truth regarding the time point where the change point in fact occurs, t_{cp} corresponds to the point 100. Additionally, different levels of volatility are introduced, generating time series with low-level volatility ($\sigma = 0.025$), medium-level volatility ($\sigma = 0.05$), or high-level volatility ($\sigma = 0.1$). Finally, for obtaining independent replications of the simulation, different random numbers are used [Alexopoulos, 2007; Law, 2020]. Each replication of a time series uses the same initial conditions, however, the random noises introduced at each level of volatility are

generated with 30 different seeds $(123, \ldots, 152)$. In total, the dataset CP1 contains 630 time series (7 curves $\times 3$ volatility levels $\times 30$ replications). Figure 16 depicts examples of the time series of the CP1 dataset for the linear and sigmoid curves.

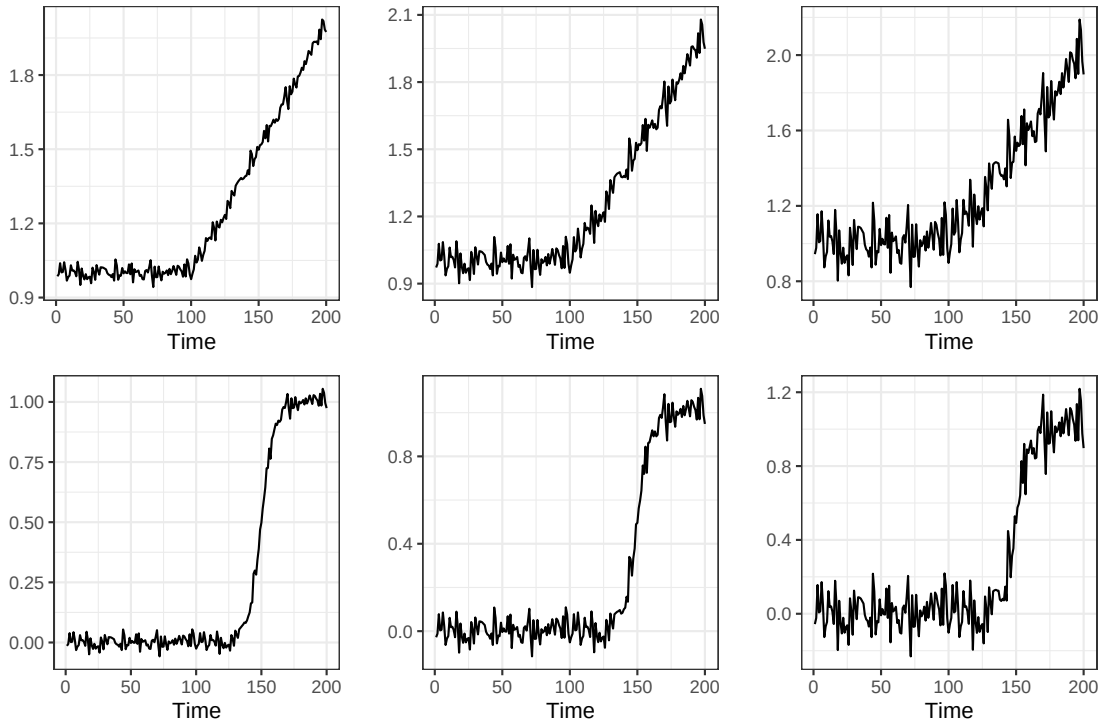


Figure 16 – Examples of nonstationary synthetic series in dataset CP1. The first three plots represent the time series with a linear trend and low, medium and high volatility levels, respectively. The last three plots represent the time series with a sigmoid trend and low, medium and high volatility levels, respectively.

A second dataset, henceforth called *CP2*, was developed by sequentially concatenating the time series in CP1 and their mirroring version. The resulting time series contain double the observations (400) and a total of 3 change points at times 100, 200 and 300, respectively. CP2 was developed in order to evaluate the methods for time series containing a larger number of change points. Figure 17 depicts examples of the time series of the CP2 dataset for the linear and sigmoid curves.

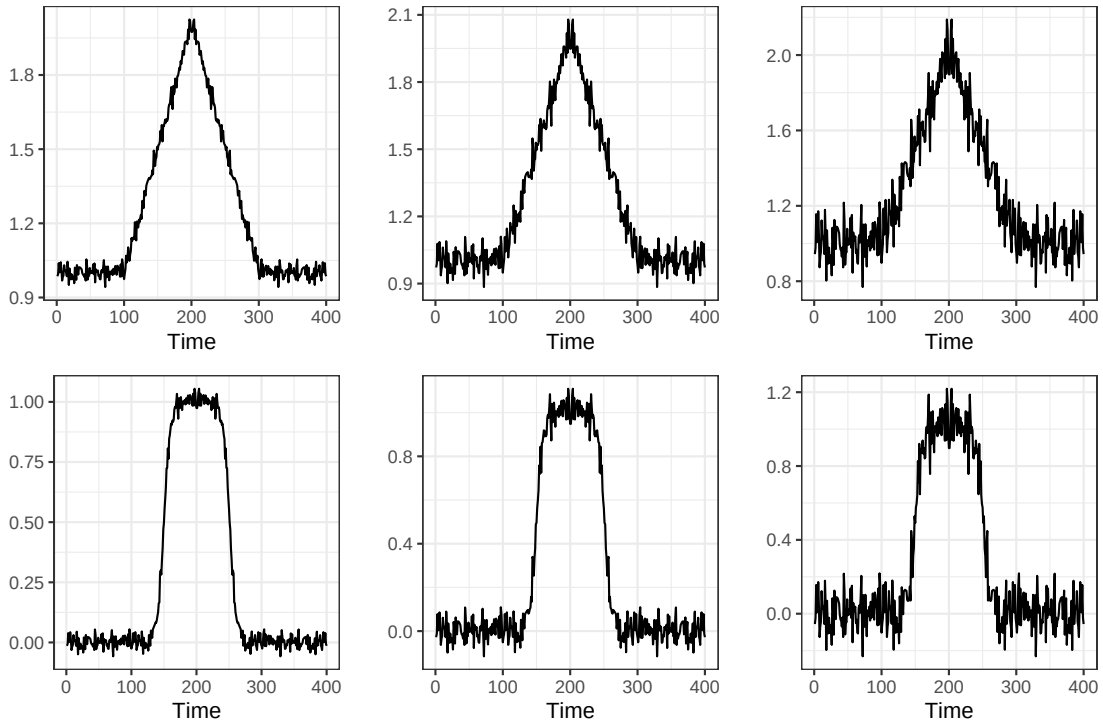


Figure 17 – Examples of nonstationary synthetic series in dataset CP2. The first three plots represent the time series with a linear trend and low, medium and high volatility levels, respectively. The last three plots represent the time series with a sigmoid trend and low, medium and high volatility levels, respectively.

5.2.2- Implementation definitions and parameterization

Online simulation For conducting the experimental comparative analysis using Herald, EDfP, and EDfR methods, we devise a strategy for simulating an online event detection application using the Nexus framework[2]. In that case, the time series contained in the adopted datasets are given as a data stream in W with batches of size 1, that is, W receives the addition of new observations $t+1$ one by one. So the process described in Section 5.1 is conducted iteratively for each t and $W = seq_{t,(m+h)}(X)$. A full memory approach is adopted (model input size $m = |t| - h$), so m is set as an initial value, which is used to define the warm-up data [Alexopoulos, 2007; Law, 2020].

Frameworks and R-packages The Herald, EDfP, and EDfR methods are implemented in R as an object of the *Harbinger* R-package [Salles et al., 2023a]. The MAS transformation passed to $decomp$, and the fit and $pred$ functions regarding the AR(1) model are

available in the *TSPred* R-package [Salles and Ogasawara, 2021]. The det function is implemented with functions from the *strucchange* R-package [Zeileis et al., 2002] for detecting structural changes in regression relationships and assessing their significance. The *Fstats* functions computes F_i statistics for the interval $[t_a, t_b]$ $(t-k+2 < t_a \leq i \leq t_b < t+1)$. The $supF$ statistic is calculated by the *sctest* function and the actual change point event $\hat{t}_{cp}$ is estimated by the *breakpoints* function.

Parameterization The main parameters given as input to Herald, EDfP, and EDfR are listed and described in Table 7. Function specific parameterization is given by ρ_{decomp}, ρ_{transf}, ρ_{fit}, ρ_{dist}, and ρ_{det}, respectively. The order of the moving average is given by parameter o included in ρ_{decomp}. The identity function and the Euclidean distance function do not take parameters, so $\rho_{transf} = \rho_{dist} = \{\}$. The ρ_{fit} takes parameters for obtaining the AR(1) model in its simplest form, setting $\varphi_1 = 1$ and $\varepsilon_t = 0$. The ρ_{det} set includes the parameters α, t_a and t_b. The null hypothesis of no structural break in the interval $[t_a, t_b]$ is rejected when there is evidence for a structural change at the level $\alpha = 0.05$. Also, by default, *Fstats* sets the evaluated interval $[t_a, t_b]$ corresponding to 70% of the sample size, that is, $t_a = \lceil 0.15 * k \rceil$ and $t_b = \lceil 0.75 * k \rceil\}$ [Zeileis et al., 2002].

The remaining parameters of the methods, namely the initial value of model input data passed to fit in the warm-up data, m, the order of the moving average transformation o, the lagged prediction horizon preceding, h, and the memory size, k, can also have a relevant impact on their results. For this experimental evaluation, these parameters are set from the geometric progression of ratio 3 defined as 3^p, with $p = 1 \ldots, 4$. The maximum value of p is given based on the size of the smaller time series in the datasets ($\lfloor log_3(|X| = 200) \rfloor$). However, the proposed evaluation of Herald, EDfP, and EDfR is based on the definition of m, o, h, and k as $3^3 = 27$, as it is closest to the minimum value of p that ensures a sample that is enough to hold the central limit theorem (CLT) used as a heuristic.

However, for sensibility analysis, the order of the moving average o, as well as the lagged prediction horizon h, are also defined from the set $\{3, 9, 27\}$. In that case, the maximum value of p was defined as 3 to avoid over-aggregation and minimize multistep ahead prediction error accumulation. On the other hand, the impact of the memory k in the detection activity is one of the main points to be analyzed in a sensibility analysis. Thus, k is defined from the set $\{9, 27, 81\}$. As 3 points are insufficient for testing structural

breaks, the minimum value of p was defined as 2.

Table 7 – Experimental parameterization. The top part of the table presents fixed parameters. The bottom part presents parameters that were submitted to a sensibility analysis.

Parameters	Values	Description
ρ_{decomp}	$\{o\}$	Parameters for function $decomp$
ρ_{transf}	$\{\}$	Parameters for function $transf$ and $transf^{-1}$
ρ_{fit}	$\{\varphi_1 = 1, \varepsilon_t = 0\}$	Parameters for function fit
ρ_{dist}	$\{\}$	Parameters for function $seqdist$
ρ_{det}	$\{\alpha = 0.05, t_a = \lceil 0.15 * k \rceil, t_b = \lceil 0.75 * k \rceil\}$	Parameters for function det
m	$\{27\}$	Initial value of model input data passed to fit in the warm-up data
o	$\{3, 9, 27\}$	Order of the moving average transformation
h	$\{3, 9, 27\}$	Lagged prediction horizon preceding time t
k	$\{9, 27, 81\}$	Memory size, that is, the size of subsequences to be inspected for events

5.2.3- Evaluation metrics

Finally, the results from Herald, EDfP, and EDfR methods are compared regarding their ability to produce fast, early, and accurate online detection of change points. For that, we evaluate the methods based on the computation of their detection lags. The detection lag corresponding to the result of a given method can be calculated regarding three different units: (i) streaming batches, (ii) time series observations, and (iii) running time. Detection lags measured in batches compute how many batches of new data were streamed since the actual change point was available to the method. With batches of size 1, which is the case for this experimental evaluation, the detection lag represents how many new observations were necessary before the method was able to detect a change point has occurred. The detection lag measured in batches gives an idea of how early a method is able to detect a change point event.

On the other hand, a detection lag measured in observations computes the difference between $\hat{t}_{cp}$ and the ground truth regarding the time point where the change point in fact occurred, t_{cp}. The detection lag in observations gives an idea of how close a method came to detecting the time of the actual change point event. Finally, a detection

lag measured in running time computes how much time (in seconds) passed before the method was able to detect the occurrence of the event. The detection lag in running time of a method is influenced by its detection lag in batches and the complexity of its corresponding algorithm.

Generally, the performance of detection methods is measured regarding their detection accuracy. That is the case for most works in the literature, where detection accuracy is evaluated by standard classification metrics. However, evaluation based on these metrics is limited as they only reward exact true positives. This research adopts the developed Soft Classification Metrics for Event Detection (*SoftED*) [Salles et al., 2023b]. The SoftED metrics soften the standard classification metrics to incorporate time tolerance to imprecision in event detection. They provide the means to evaluate detection methods not only on detection accuracy but also on their ability to produce close detections that are most representative of time series events.

5.2.4- Significance tests

The evaluation metrics computed for both Herald and EDfP are compared against the baseline EDfR method. The statistical significance of their differences is tested by the Wilcoxon signed rank test on paired samples with a confidence level of 95% ($\alpha = 0.05$). It is a widely adopted non-parametric alternative to the paired samples t-test for comparing data that are not normally distributed [Hollander et al., 2013]. The Wilcoxon test is available from the *rstatix* R-package [Kassambara, 2023]. Since multiple tests are performed for each volatility level and parameter setting (in sensibility analysis) analyzing the same hypotheses, the target α needs to be adjusted accordingly. For that, the Bonferroni method is adopted for adjusting the resulting p values for multiple comparisons [Strassburger and Bretz, 2008]. Compared to other adjustment methods, the Bonferroni produces more conservative significance confirmations based on a small adjusted α.

Besides the significance, the effect size measure is calculated as the matched-pairs rank-biserial correlation, available from the *effectsize* R-package [Ben-Shachar et al., 2020]. It is a non-parametric measure that indicates the magnitude of the difference between two groups that are not normally distributed. The correlation represents the

difference between the proportion of favorable and unfavorable pairs, defined by positive and negative differences (indicated by Wilcoxon signed ranks) [Kerby, 2014]. In other words, the effect size corresponds to the difference $p_f - p_u$, where p_f is the proportion of paired differences with a favorable result and p_u is the proportion of paired differences with an unfavorable result. In this case, a favorable result is a paired difference that conforms to the expected goal, for example, a positive difference between the detection lags of a baseline method and the evaluated detection method, meaning the evaluated method produced a smaller detection lag. Values range from +1 meaning complete dominance of the positive differences to -1, complete dominance of the negative differences. However, the effect size measure can be abstract. In order to facilitate the understanding of the results, many works have standardized, validated, and published a set of rules of thumb as guidelines. This experimental evaluation adopts the rules proposed by Funder and Ozer [2019] (default in package effectsize), which interprets the correlation coefficient r defining: $r < 0.05$ as a Tiny difference, $0.05 \leq r < 0.1$ as a Very small difference, $0.1 \leq r < 0.2$ as a Small difference, $0.2 \leq r < 0.3$ as a Medium difference, $0.3 \leq r < 0.4$ as a Large difference, and $r \geq 0.4$ as a Very large difference.

5.3- Results and discussion

This section presents and discusses the results obtained by Herald, EDfP, and EDfR methods regarding the metrics described in Section 5.2.3, including detection lags (in the units of batches, observations and running time), and SoftED classification metrics for event detection. The computed metrics are used as a base for the comparison of Herald and EDfP against the baseline EDfR method. Moreover, the comparison of their results is based on statistical analysis as described in Section 5.2.4. First, Section 5.3.1 presents an analysis based on detection lags, which henceforth correspond to the metric of detection lag in units of batches, unless stated otherwise. Finally, Section 5.3.2 analyses detection performance based on the SoftED metrics. For the purpose of discussion, all results regarding Herald correspond to parameterization of m, o, h, and k set as $3^3 = 27$ with basis on the CLT, as described in Section 5.2.2, unless stated otherwise. Accordingly, this is also the case for the parameters m and k of EDfP, and k of EDfR.

5.3.1- Detection lag analysis

A detection lag analysis of the evaluated methods is related to the assessment of their detection speed, in the sense of how early a method is able to detect a change point event after its occurrence. In that case, this analysis mainly focuses on assessing whether the Herald method is able to produce earlier detections compared to the baseline EDfR. The results of EDfP against EDfR are also analyzed for comparison. The plots in Figure 18 exemplify the results for each method regarding the detection of the change point contained in the time series from de CP1 dataset, corresponding to a linear curve trend with high volatility (seed 123). Figure 18 presents snapshots of the data stream at the moment EDfP, EDfR, and Herald, first detected the change point, respectively. The ground truth regarding the occurred change point is marked by blue dots, while red dots (and dashed vertical lines) mark event detections produced by the methods. The fitted models Y and Y' are also depicted in orange lines, while orange dots represent their $h+1$ predictions which are the base for computing $\delta_{t+1}/\delta'_{t+1}$. It is also presented the corresponding detection lag for each method result.

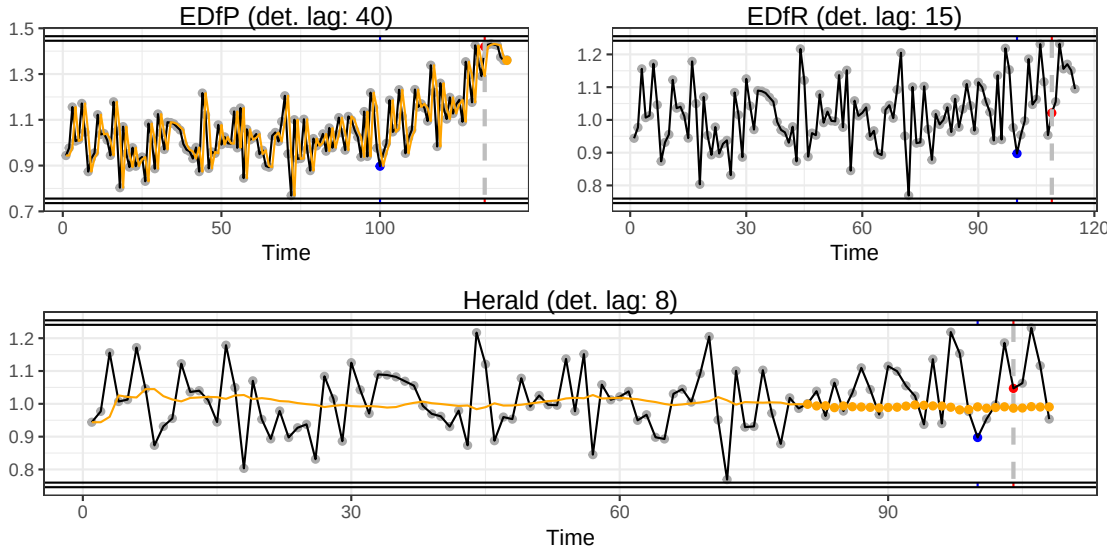


Figure 18 – Example plots of event detection produced by EDfP, EDfR, and Herald, respectively. Blue dots mark the actual change points, while red dots and dashed vertical lines mark event detections. Their coincidence is marked with green dots. Orange lines represent the fitted model, and orange dots represent their predictions. The detection lags for each method are also presented.

It is possible to observe the practical differences between each approach, where

the model Y produced by Herald is more representative of the time series inertia and long-term trend when compared to model Y' produced by EDfP. Furthermore, while EDfP produces only one prediction for $t+1$, Herald produces $h+1=28$ predictions which are used for computing the sequence distance δ_{t+1}. Although EDfR does not produce any time series prediction, it is able to detect a structural break in the last k known observations of the time series after a detection lag of 15, that is, almost three times earlier than EDfP which had detection lag of 40. However, for this time series, the analysis of the k last sequence distances in Δ_{t+1} helped Herald to detect the change point after a detection lag of 8 batches. The next subsections analyze the statistical significance of the differences between the detection lag results of Herald and EDfP, against EDfR, for all time series in the datasets CP1 and CP2. For that, it is adopted the alternative hypothesis of smaller lags from Herald and EDfP. Additionally, an ablation and parameter sensibility analysis is provided.

Statistical analysis

A statistical analysis of the detection lags produced by Herald against the EDfR baseline is presented in Figure 19. The left plot presents the boxplots of detection lags produced by both methods for all time series in CP1, while the plots on the right present the results grouped by volatility level. The statistical significance of the differences between the detection lags of Herald and EDfR are given on top of each plot, as well as the corresponding effect size. Overall, detection lag differences are statistically significant for dataset CP1 (p-value$=2.76e-03$), for which Herald produces smaller detection lags with effect size 0.13 (interpreted as small). In addition, by simple algebraic manipulation ($p_f - p_u = 0.13$), we can obtain the proportion of favorable results (p_f), that is, positive differences between the detection lags of EDfR and Herald, which produced smaller detection lags corresponding to 56.5% of the cases for all time series in CP1.

However, a more thorough analysis is given by the plots on the right of Figure 19, where one can evaluate the lag differences grouped by the level of volatility of the adopted time series. The last two plots show that although there was no statistical significance to the Wilcoxon tests for low or medium volatility levels, there was also no clear dominance

of unfavorable lag difference results indicated by effect sizes 0 and -0.04, interpreted as tiny. On the other hand, Herald was able to produce smaller detection lags especially for time series of high volatility levels, with statistical significance represented by a p-value of $4.17e-06$. Moreover, a large effect size of 0.38 is obtained, meaning that Herald provides lower detection lags for 69% of the high volatility time series in CP1 $(145/210)$.

This result can can be explained by the smoothing nature of the adopted inertial component, which minimizes the influence of the high volatility and highlights the long-term trend of the time series, whereas this effect is not observed for low/medium volatility levels with the same relevance. Due to the relevance of Herald's effect on high-volatility time series, the remainder of this section further analyses the subset of dataset CP1 corresponding to such time series.

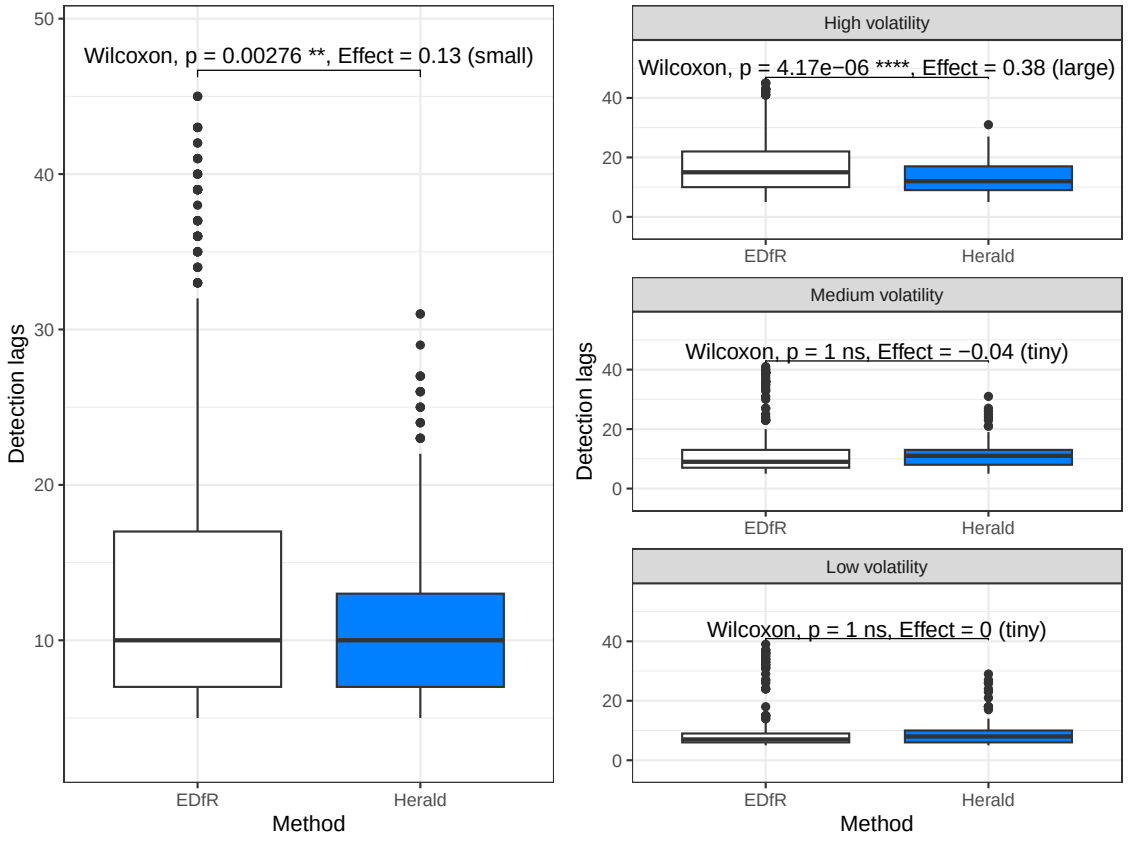


Figure 19 – Statistical analysis of the detection lags produced by Herald against the EDfR baseline. The left plot presents values for all time series in CP1, while the plots on the right present the results grouped by volatility level.

Looking further into the results for the time series with high volatility, an analogous statistical analysis is performed to evaluate Herald regarding the detection lags computed in the units of observations and running time. The left plot of Figure 20 presents detection

lags measured in number of observations between t_{cp} and the detected time point $\hat{t}_{cp}$ (represented as blue and red dots in Figure 18, respectively). The paired difference between the results of EDfR and Herald is statistically significant, meaning that Herald is able to produce detections at time $\hat{t}_{cp}$ that come closer to t_{cp}, being more representative of the actual change point for 57.5% of the time series (effect size of 0.15). The average running time (in seconds) necessary for the methods to detect the change points after their occurrence is presented in the right plot of Figure 20, as well as standard deviations. In this case, the Wilcoxon test indicated that despite the baseline EDfR producing greater detection lags (in batches and observations), it demands less running time to give its outputs. This result is expected given the increased complexity of Herald, which includes the steps of time series decomposition and prediction. Nonetheless, Herald was still able to output its detections with competitive running times for 42.5% of the time series, as indicated by an effect size of -0.15.

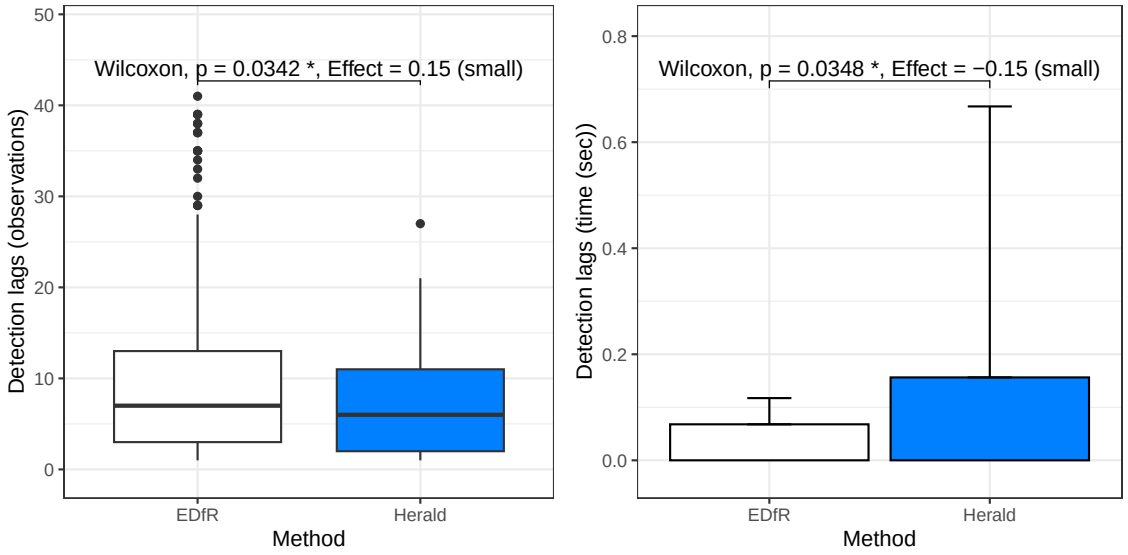


Figure 20 – Statistical analysis of detection lags in units of observations and running time, produced by Herald against the EDfR baseline for time series with high volatility. The left plot presents detection lags in number of observations, while the right plot presents the mean and superior standard deviation of running times.

Ablation analysis

In order to gain more insight into Herald's detection results, an ablation analysis is performed [Vishnusai et al., 2020]. In this ablation study, we investigate the performance of Herald by removing key components of its methodology to understand their contribution to the overall task of change point detection. In particular, the evaluated methodology components of Herald correspond to the use of the inertial component (IC) of the time series, obtained by the step of time series decomposition, and the lagged prediction (LP) performed in the step of time series prediction. Figure 21 depicts the results of this study over the time series of high-volatility of CP1. The right plot presents the boxplots for the detection lag differences between the baseline EDfR and Herald for each ablation setting, while the left plot presents their respective effect sizes. Wilcoxon statistical significance is also given on the top of boxplots/dots.

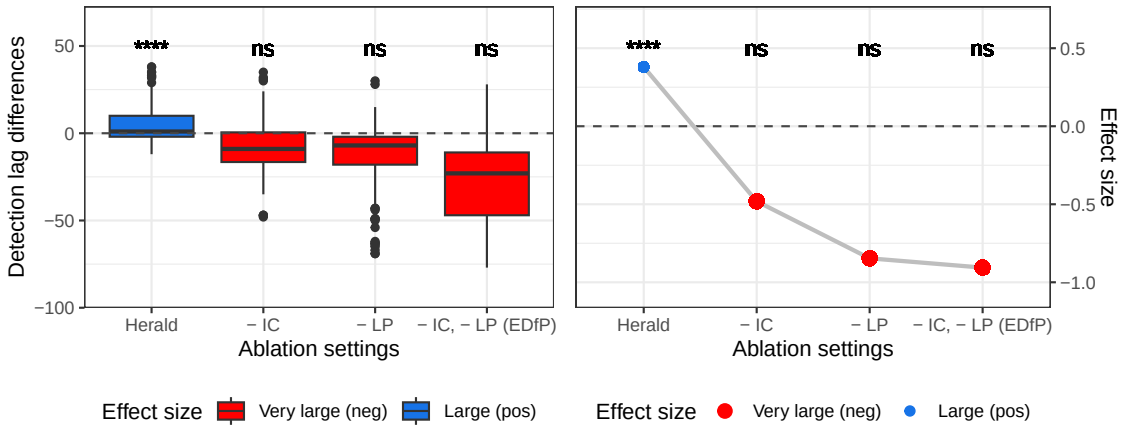


Figure 21 – Ablation study of Herald key methodology components based on boxplots for the detection lag differences between EDfR and Herald. Ablation settings correspond to the removal of the use of the inertial component (IC), the lagged prediction horizon (LP), or both. The last setting corresponds to the EDfP method. The left plot presents respective effect sizes. Wilcoxon statistical significance is given on the top of boxplots/dots. A greater number of "*" symbols indicate higher significance, while "ns" indicates "no significance".

The first ablation setting corresponds to the removal of IC, in that case, Herald performs change point detection over the lagged prediction of the original time series. The second ablation setting corresponds to the removal of LP, so h is set to 0 and the prediction of the inertial component of the time series is only produced for point $t + 1$. Finally, the third ablation setting corresponds to the removal of both IC and LP. With this

setting, h is again set to 0 and event detection is performed over the last k deviations from the prediction of the original time series for point $t + 1$, which in fact characterizes the method EDfP described in Section 5.1.

It is possible to observe that the EDfP method presented the highest detection lags compared to the baseline, presenting negative differences for 95.5% of the time series which resulted in the very large effect size of -0.91. On the other hand, the isolated adoption of either an inertial analysis of the time series or a lagged prediction horizon helped decrease detection lags as illustrated by the boxplots of lag differences (in the middle of the left plot). Nonetheless, applied separately, none of these methodology components were enough to provide smaller detection lags than the baseline EDfR with statistical significance. In fact, they still presented very large negative effect sizes of -0.85 and -0.48, respectively. However, among both methodology components, the lagged prediction had the most impact, reducing the negative effect size of EDfP to almost half (-0.48). Overall, ablation results indicate the effect of the combined use of both inertial analysis and lagged prediction horizons, characterizing the Herald method, for lowering detection lags and producing smaller detection lags than the baseline EDfR method with statistical significance and a large positive effect size.

Sensibility analysis

This subsection presents additional analysis regarding the sensibility to the setting of relevant parameters such as the order of the moving average (MA) for obtaining the inertial component (o), and the size of the lagged prediction horizon (h), and the memory size (k). While varying each evaluated parameter, all others are fixed as the default value of 27, as described in Section 5.2.2. Figure 22 presents the results of the sensibility analysis for each setting of parameters o and h, respectively. The plots are analogous to those from Figure 21, presenting the lag differences between EDfR and Herald for each parameter setting and respective effect sizes, as well as Wilcoxon statistical significances.

Results indicate that the smallest orders of moving averages and lagged prediction horizons were not sufficient to produce smaller detection lags than the baseline with statistical significance. This is due to the fact that small orders of moving average can

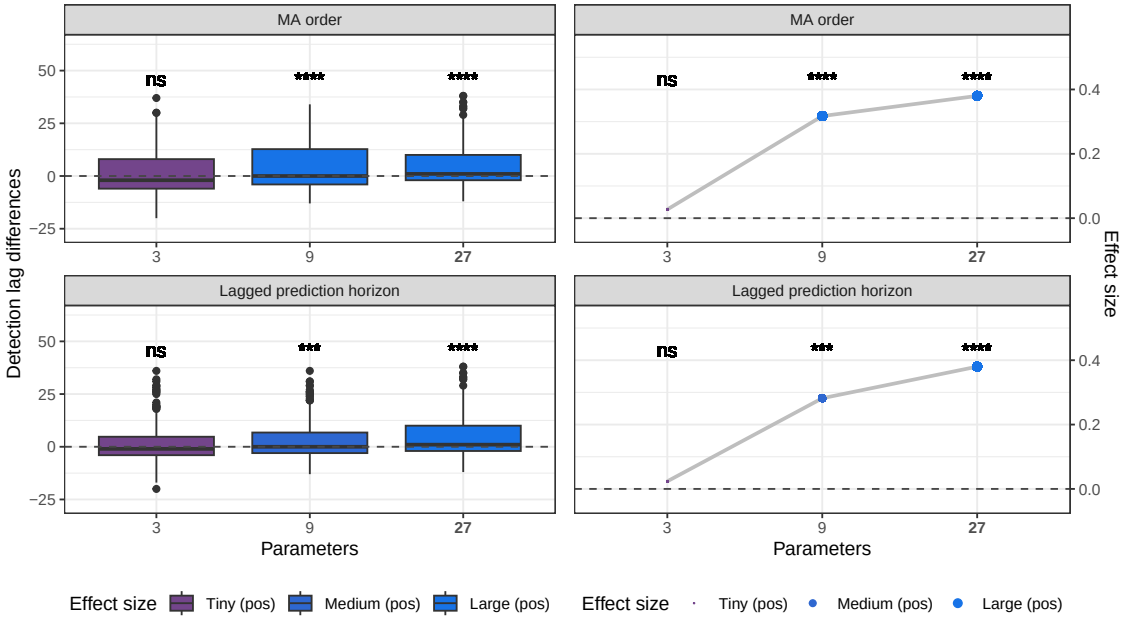


Figure 22 – Sensibility analysis of Herald parameters based on boxplots for the detection lag differences between EDfR and Herald for each parameter setting. The left plot presents respective effect sizes. Wilcoxon statistical significance is given on the top of boxplots/dots. A greater number of "*" symbols indicate higher significance, while "ns" indicates "no significance".

quickly absorb recent changes in behavior, not providing enough information from the time series past inertia for Herald to appraise. This is also the case for small lagged prediction horizons, which does not enable the evaluation of current behavior against what is expected based on the long-term trend, instead, expectations are generated based on relatively recent behavior. On the other hand, greater values for the parameters allow for smaller detection lags for the same reasons. In that case, the maximum value of 27 was adopted as default, since given the size of the time series (200), the next value in the geometric progression (81) could result in over-aggregation and high multistep ahead prediction error accumulation.

The k parameter, which defines the memory of the detection process was also submitted to a sensibility analysis. It sets the size of the sequences Δ_{t+1}, Δ'_{t+1} and W'' that are given as input to structural break detection, so it can have a relevant influence in the ability of the methods to accurately detect the change point events. For this reason, there is a demand for a sensibility analysis in order to adequately define the memory of Herald and EDfR methods for empowering early event detection. In particular, this sensibility analysis was performed based on the time series of the CP2 dataset, since its

time series contain three change points in total, allowing a more complete overview of the memory effects on a higher number of events. Figure 23 presents the results for each memory setting, grouped by the number of sequential events analyzed, going from one to a total of three.

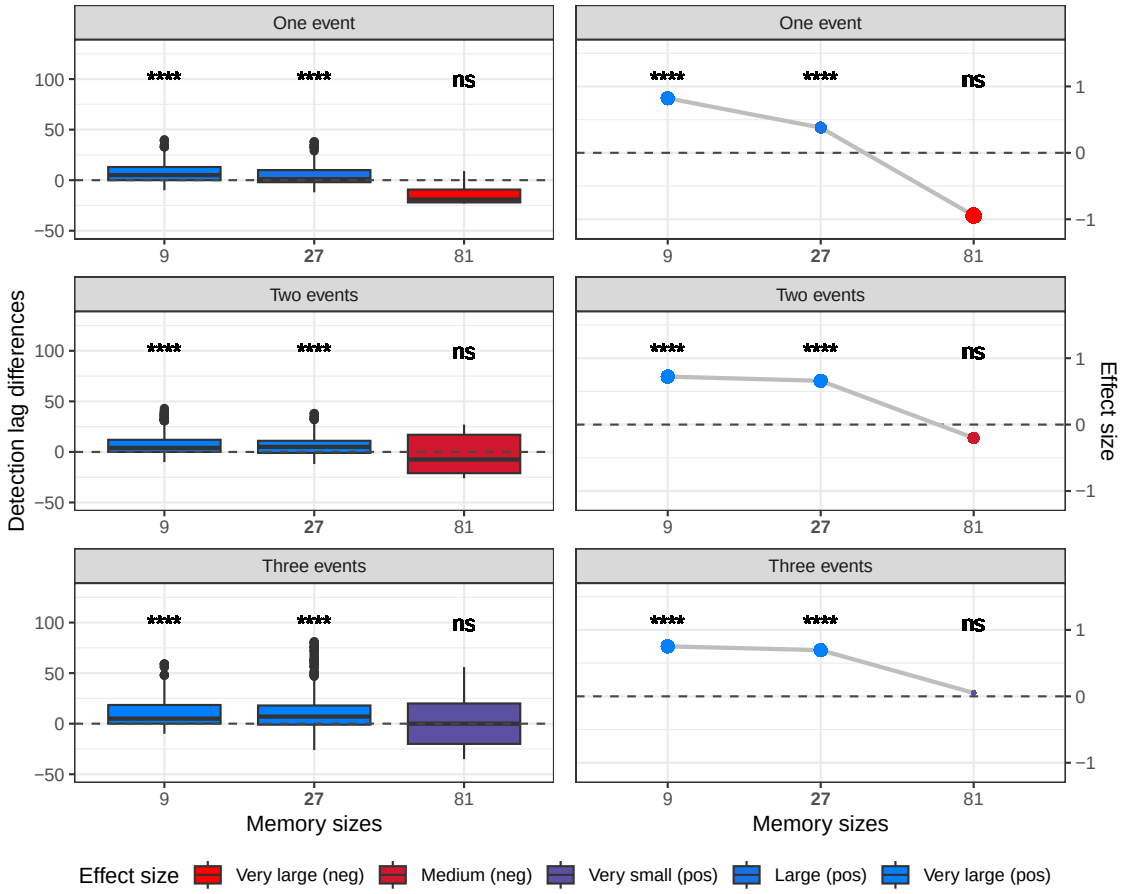


Figure 23 – Sensibility analysis of Herald memory parameter based on boxplots for the detection lag differences between EDfR and Herald for each parameter setting, grouped by the number of events analyzed. The left plot presents respective effect sizes. Wilcoxon statistical significance is given on the top of boxplots/dots. A greater number of "*" symbols indicate higher significance, while "ns" indicates "no significance".

It can be argued that intuitively, a higher memory size, providing a greater amount of information regarding past data behavior could help the identification of change points. However, the higher setting for parameter k (81) did not produce the best results regarding detection lags. We observe that this result is due to the size of the time series and the closeness between the events they contain. For example, given the default values for m and h, the warm-up data in W consists of $m + h = 27 + 27 = 54$ observations and the first iteration of Herald produces $\delta_{t+1} = \delta_{55}$. In that case, given $k = 81$ the memory in

Δ_{t+1} is not full until the computation of $\delta_{55+k} = \delta_{136}$, that is, there is a minimum lag of 36 observations before the change point occurred in $t_{cp} = 100$ can be detected. For this reason, detection lags for $k = 81$ when looking solely to the first event are mostly higher than the baseline. The last two events do not suffer from this problem as indicated by the rise of boxplots and increase of effect sizes in the two bottom plots.

The same influence can seen for $k = 27$, which presents a smaller positive effect size than $k = 9$ based on the first event. Overall, there is stability in results for memories of size 9 and 27. Nonetheless, $k = 9$ defines a relatively small sequence for structural break analysis, which can make the method overly sensitive to random noise. Hence, the default memory $k = 27$, is still preferable with basis on the CLT and the discussed results.

5.3.2- Detection performance analysis

A detection performance of the evaluated methods is related to the assessment of their detection accuracy. In particular, we adopt a soft evaluation based on the SoftED classification metrics for event detection. They allow an evaluation of how close, that is, how representative of the actual events the detections are, as opposed to traditional classification metrics, which only focus on assessing the cases where $t_{cp} = \hat{t}_{cp}$. In this context, traditional classification metrics are inadequate for assessing the earliness of a method's detections. In that case, this analysis mainly focuses on assessing whether the Herald method is able to produce detections that are more representative of change point events compared to the baseline EDfR. This analysis is mainly focused on the SoftED metrics of recall, precision and their harmonic mean, F1, which are computed for the detection of events of the time series in the CP2 dataset.

Statistical analysis

A statistical analysis of the SoftED metrics computed for detections of Herald against the EDfR baseline is presented in Figure 24, which is analogous to the analysis

presented in Figure 19. The top plot presents the boxplots of SoftED metrics produced by both methods for all time series in CP2, while the plots on the bottom present the results grouped by volatility level. The statistical significance of the differences between the metrics of Herald and EDfR are given on top of each plot, as well as corresponding effect sizes. The results indicate that SoftED recall metric differences are statistically significant for dataset CP2 (p-value$= 6.86e-6$). It means that, overall, Herald produces time series detections that are closer and more representative of the three change point events in the time series of CP2 with a very large effect size of 0.79. In that case, positive differences between the metrics of Herald and EDfR correspond to 89.5% of all time series in CP2, for which the Herald method produced a higher recall metric.

The same is observed for all levels of volatility, as illustrated in the bottom plots, however, the effect of Herald is maximized for the time series with high volatility. In fact, this result was also previously indicated by the analysis of the detection lags measured in observations presented in Figure 20, where we observed that Herald produced closer detections to the single event contained in CP1. For this reason, the remainder of this section further analyses the subset of dataset CP2 corresponding to high-volatility time series.

However, Figure 19 shows that the SoftED precision metrics calculated for Herald are considerably smaller than the ones calculated for the baseline EDfR over all time series in CP2, and consequently, the F1 metrics are directly affected. To help explain this result, the Figure 25 provides examples of detection results for each method regarding the detection of the change points contained in the time series from de CP2 dataset, corresponding to a linear (top plots) and a sigmoid (bottom plots) curve trend with high volatility (seed 123). Again the ground truth regarding the occurred change point is marked by blue dots, red dots (and dashed vertical lines) mark event detections produced by the methods and the fitted models Y is also depicted in orange lines, while orange dots represent their $h+1$ predictions.

The SoftED recall, precision and F1 metrics are presented for each method result. At this point, it is important to mention that the comparison of these results via traditional classification metrics would be unfeasible since recall and precision would be computed as 0 (F1=NaN) for all examples in the first three plots of Figure 19. Only the last example would have an F1 metric, since there was two true positives in points 200 and 300.

It is possible to observe that, although Herald is able to produce detections that

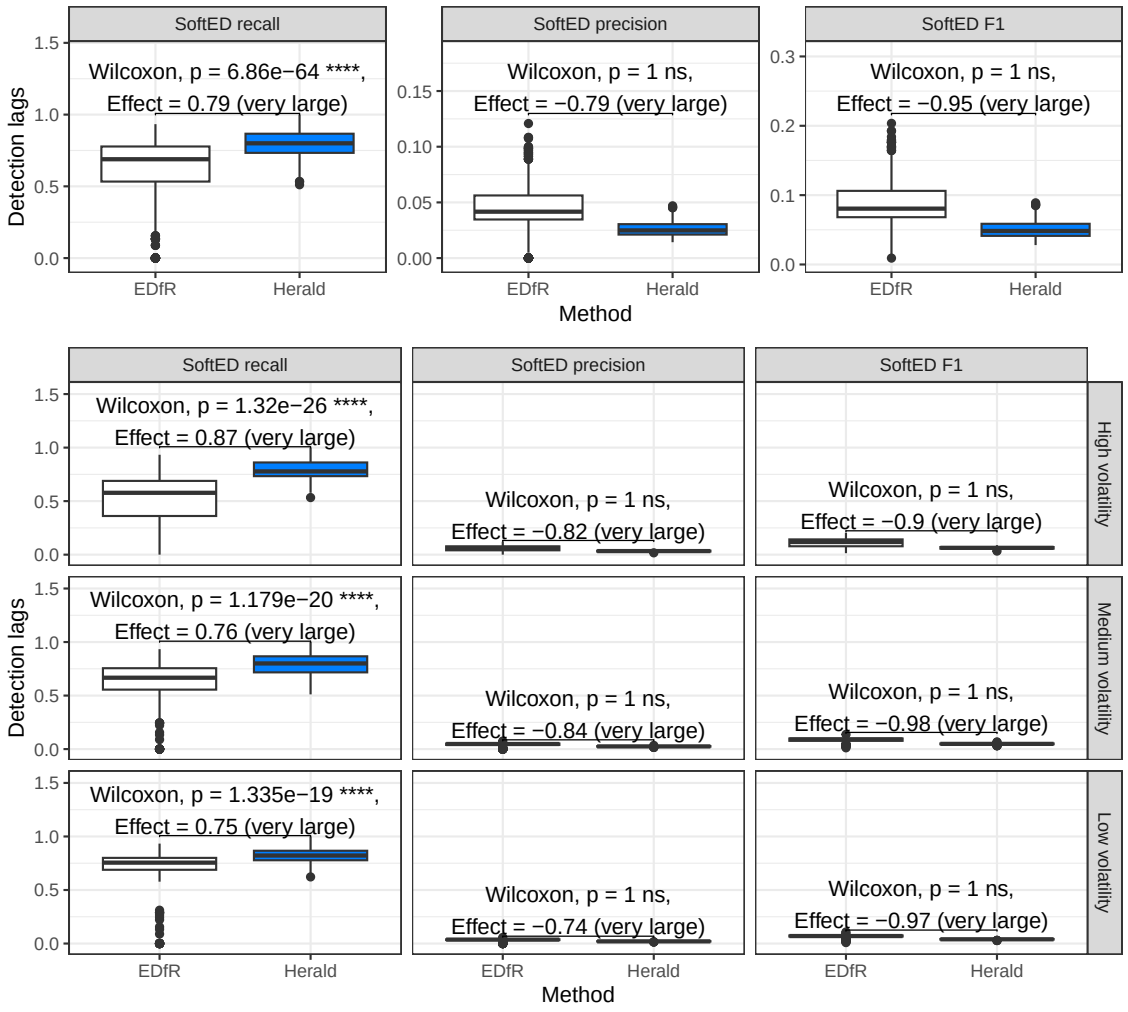


Figure 24 – Statistical analysis of the SoftED metrics computed for the detections of Herald against the EDfR baseline. The top plot presents values for all time series in CP2, while the plots on the bottom present the results grouped by volatility level.

come closer to the change point events in the time series (located at time point 100, 200 and 300), it emits a greater number of positive signals, which increases the number of false positives considered for the metrics computation. The reason for the rate of false positives produced by Herald can be understood by analyzing the model Y (in orange). As it represents the inertia of the time series, it takes time for it to conform to new arriving concepts. In that case, it maintains an increased deviation from the series observations during the effect of the trends between time points 100 and 300, as a consequence, signals indicating the occurrence of the change point tend to be propagated and reinforced for the subsequent observations in the data stream. This is an observed limitation of the current instantiation of the Herald method. Thus, it is suggested the exploration of alternative models for representing the time series inertia and the adoption of techniques

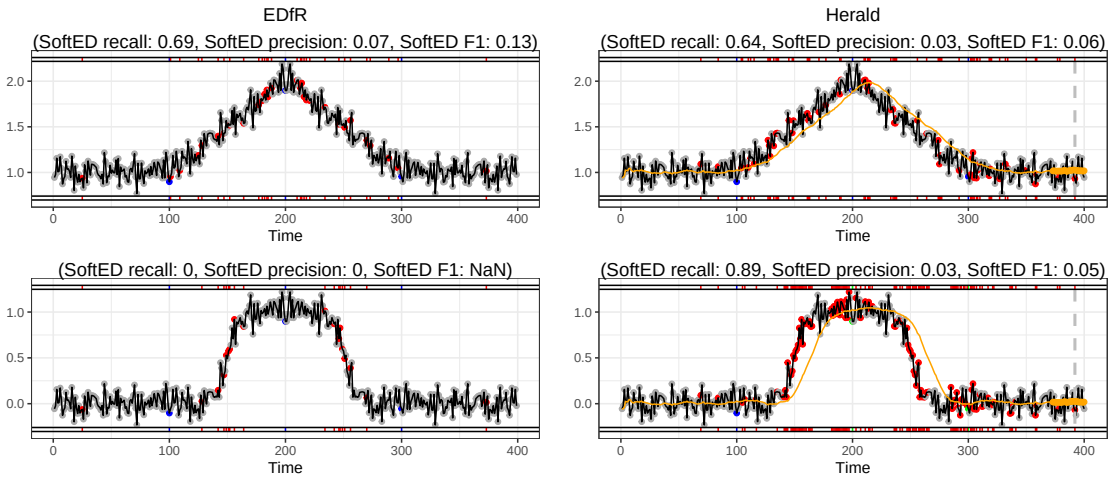


Figure 25 – Example plots of event detection produced by EDfR and Herald, respectively. Blue dots mark the actual change points, while red dots and dashed vertical lines mark event detections. Their coincidence is marked with green dots. Orange lines represent the fitted model, and orange dots represent their predictions. The SoftED metrics calculated for each method are also presented.

for aggregating detections that correspond to a particular change point event.

Sensibility analysis

A sensibility analysis of the effects of different parameter settings on the evaluated SoftED metrics is provided in this subsection. Similarly to the sensibility analysis presented in Subsection 5.3.1, we evaluate the parameters of the order of the moving average (MA) (o), the lagged prediction horizon (h), and the memory size (k), that are set as described in Section 5.2.2. Figure 26 presents the results of the sensibility analysis for each setting of parameters o and h, respectively, for the metrics of SoftED recall (top plots), precision (middle plots), and F1 (bottom plots). The plots present the metric differences between Herald and EDfR for each parameter setting, and their respective effect sizes (on the left), as well as Wilcoxon statistical significances.

Higher orders of moving averages and higher lagged prediction horizons allow for an expanded analysis of the time series inertia and past behavior expectations, which in turn contributes to earlier and closer detections produced by Herald, represented by the positive differences in the SoftED recall metric values, presenting very large effect sizes.

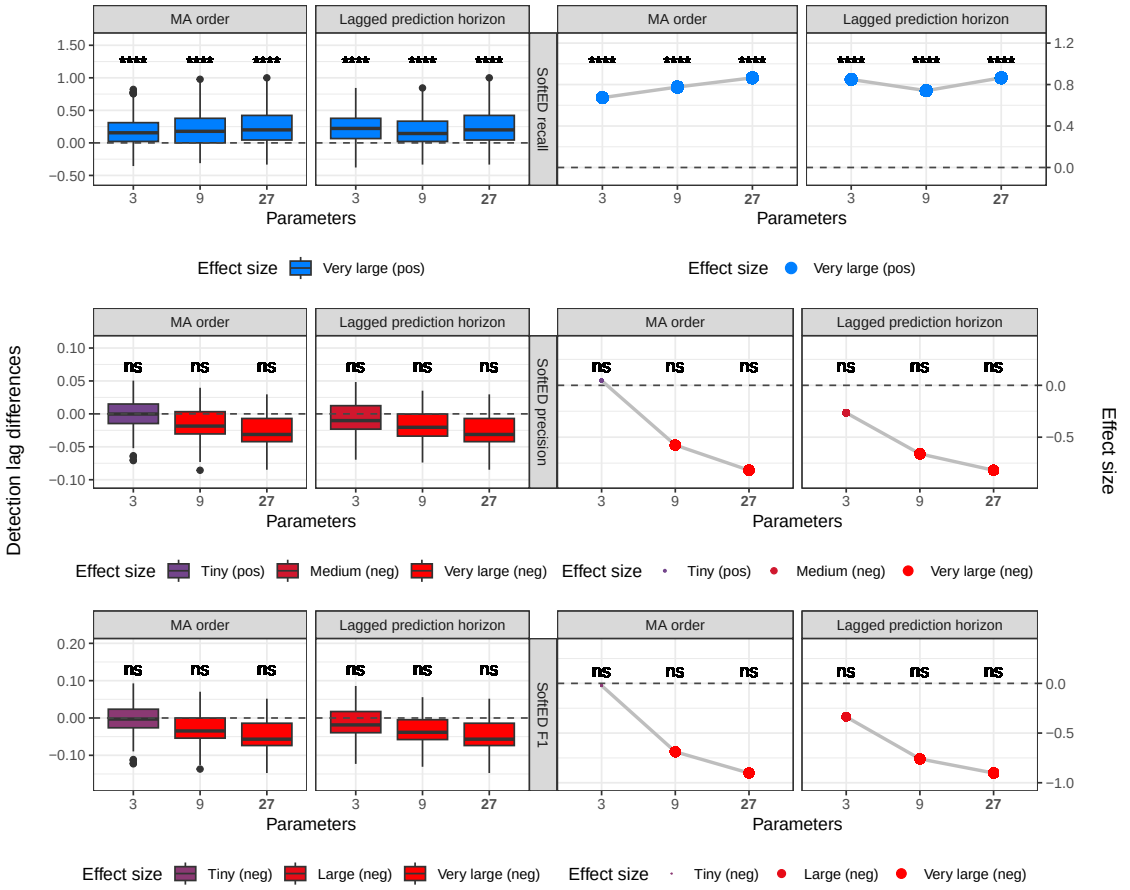


Figure 26 – Sensibility analysis of Herald parameters based on boxplots for the SoftED metrics differences between Herald and EDfR for each parameter setting. The left plots present respective effect sizes. Wilcoxon statistical significance is given on the top of boxplots/dots. A greater number of "*" symbols indicate higher significance, while "ns" indicates "no significance".

However, large settings for parameters o and h also tend to increase the rate of false positives, decreasing SoftED precision and F1, as the inertial component further deviates from the current behavior, as illustrated in Figure 25 and discussed in subsection 5.3.2. Conversely, when o and h are set to small values, such as 3, precision and F1 differences increase, as well as respective effect sizes, while the inverse results are observed for the recall metric. In this context, finding the balance between keeping high SoftED recall metrics while increasing SoftED precision, and consequently, their harmonic mean, F1, is currently an observed demand for further improving Herald evaluation.

In summary, the results discussed in this section indicate that Herald is able to detect change points earlier and with higher sensitivity, presenting smaller detection lags both in batches and observations, and higher SoftED recall metrics when compared to

the EDfR baseline. Meaning that, overall, it demands fewer observations to be streamed before it can detect and point to an event at time $\hat{t}_{cp}$, which in turn is temporally closer to the time point t_{cp} when the actual change point event occurred. In addition, the ablation analysis indicates the contribution of the combined use of both inertial analysis and lagged prediction horizons to the observed results, which characterize the methodology of the Herald method. Parameter sensibility analysis also gives insights into opportunities for improvement regarding the balance between SoftED recall and precision metrics.

Conclusion

This research aimed to answer the question of whether the analysis of the lagged prediction of time series inherent inertia brings contribution to online change point detection regarding lower detection lags when compared to current methods based on either a prediction or regression approach. To answer this question, it was proposed and implemented a novel method that combines the use of both inertial analysis and lagged prediction horizons for online change point detection. The method was named Herald, after the medieval official messengers that carried announcements of important events.

Based on experimental results it can be concluded that Herald is able to produce lower detection lags, detecting change points earlier than methods based on either a prediction or regression approach. Moreover, the contribution of the combined adoption of both inertial analysis and lagged prediction horizons to lowering change point detection lags was indicated by statistical analysis. Hence, the findings of the proposed experimental evaluation conform to expectations, and the research question posed by this thesis is answered positively, confirming the proposed hypothesis.

The method proposed by this thesis contributes to the progress toward achieving the idealistic characteristics of an online change point detection method [Ahmad et al., 2017], including (i) unrequired access to the whole stream; (ii) automated analysis over unsupervised data; (iii) adaptability to dynamic and nonstationary environments with concept drifts; and most importantly, (iv) the detection of events as early as possible, or with increasingly smaller detection lags.

Summary and reflection on main research findings

Current approaches for time series change point detection pose a number of challenges in order to ensure accurate and early detection of events. Regarding most established regression approaches, it was observed the need for adopting initial assumptions regarding the fitted parametric models, which ultimately may not conform to actual

time series properties. Also, they demand a sufficient number of observations in incoming data to enable the comparison of model estimates to historical data. On the other hand, approaches based on the analysis of deviations from time series prediction can suffer from nonstationary properties, which can affect the accuracy of predictive models. Moreover, incrementally changing concepts can be absorbed as part of the inherent function of the time series process, which in turn may increase the time required for the new concept to be noticed by the methods.

Herald addressed the aforementioned challenges based on the derivation of a time series trend component as a proxy of its intrinsic inertia. It is then used for inertial analysis and given as input to a time series prediction process with a lagged prediction horizon. In this context, the inertial analysis of a time series can help regarding robustness to nonstationarity and the minimization of the effects of emerging behavior. The inertial component of a time series is a low-frequency representation of the data serving as more stable input for time series prediction, while also holding information on its long-term trend. Furthermore, the lagged prediction of the inertial component enables the evaluation of the current and most recent states of the time series compared to what is expected from its long-term inertia. Ultimately, as we look further into the past inertia of the time series, instead of the recent memory of historical behavior, we increase sensibility to incremental concept drifts, minimizing the detection lags of their corresponding change points.

To confirm this expectations, Herald was expected to be able to detect change points in nonstationary time series, producing lower detection lags and competitive results in detection accuracy while being robust to incremental concept-drifts. For validating these expectations, this thesis experimentally evaluated Herald in applications based on a streaming simulated nonstationary time series prone to the occurrence of concept drifts containing up to three change point events. The Herald approach was evaluated against methods based on the usual prediction approach, referenced as *EDfP*, and the regression approach, referenced as *EDfR*, which was adopted as a baseline.

Experimental results attested to these expectations, where Herald outperforms *EDfP* and *EDfR* methods regarding both early and representative online detection of change points. In particular, Herald was most effective for time series with high volatility, which is explained by the smoothing nature of the adopted inertial component, which mitigates the influence of spurious values and highlights the long-term trends. Herald also produced a higher sensitivity to change point events represented by higher SoftED recall

metrics over all time series of the adopted datasets. The interpretation of these results is that Herald demands fewer observations to be streamed before it can detect a change point, and when it does, its detection is more representative, being temporally closer to the time point when the actual change point event occurred.

Ablation experiments also indicated the effect of both inertial analysis and lagged prediction horizons on change point detection lags. Both components characterize the methodology of the Herald method. Although individually their adoption is able to outperfom the usual *EDfP* approach, only their combination allowed Herald to outperform the baseline *EDfR* regarding smaller detection lags.

Another unexpected result regards the running time necessary for the methods to output their change point detections. Given the increased complexity of Herald, which includes the steps of time series decomposition and prediction, it was still able to output its detections with competitive running times having a small effect size on their increase. Parameter sensibility analysis also provided insights into opportunities for improving the balance between SoftED recall and precision metrics.

Future research strands and applications

Once successfully evaluated, the Herald method is to be made widely available for online time series change point detection. For that, it will compose and extend the detection method catalog of the Harbinger framework [Salles et al., 2023a]. In particular, it should be included in Harbinger as a more general implementation of the Herald methodology, allowing user definition of characteristic methodology functions such as $decomp$, fit, $pred$ and det. In that case, the Harbinger framework should enable not only the reproduction of the Herald instantiation proposed and evaluated in this thesis, but also the exploration of different instantiation versions and parameter settings.

In fact, the functions given as input to Herald can be set as a number of different possible methods as stated in Section 4.2 and so the formalized Herald method can be instantiated in different versions. Accordingly, each possible instantiation has its own characteristic parameters to be set. Although the key components of the Herald methodology were evaluated with a more basic instantiation, further research can explore

possible input parameter combinations and techniques for hyperparameter optimization and the scenarios in which they bring the most benefit to change point detection. This exploration is facilitated by the Harbinger framework.

Furthermore, to help achieving the goal of automatizing nonstationary time series prediction and event detection, it is a worthwhile approach to make both processes more systematic with the use of Automated Machine Learning (AutoML) techniques. These artificial intelligence-based techniques have been gaining increasingly more attention over the last years [Feurer et al., 2015; Kotthoff et al., 2017; Jin et al., 2018]. They can help provide automatic recommendations for prediction and detection methods and parameters for defining a particular instantiation of Herald, reducing the need for human interaction. This approach can produce models that outperform others designed by hand, possibly improving the detection results of Herald.

Based on these conclusions, researchers should consider evaluating the Herald method with their datasets possibly containing larger time series with high-level volatility and a greater number of change point events. Moreover, the experimental analysis presented in this thesis can be extended to benchmark streaming time series datasets composed of simulated and real-world data on domain problems where reaction response to events is critical. Among them are three of the most prominent benchmarks for online time series event detection algorithms currently available in the literature, produced by Webscope [2015]; Ahmad et al. [2017]; Wu and Keogh [2021]. These and many other datasets used for developing and evaluating methods for time series events detection are included in the R-package *dalevents* available on GitHub[1]. In particular, the provided time series datasets present a considerable amount of representative time series containing different types of events, as well as different types of nonstationarity and statistical properties. It helps further a more thorough discussion on the effects of event detection methods on a diverse range of time series. The application of Herald to the simulated and real-world datasets on *dalevents* could also enable a better understanding of its implications for the detection of other types of structural changes in behavior, such as in volatility instead of the inherent trend.

Experimental results indicate that further research is needed to determine adequate strategies for obtaining a better balance between the SoftED recall and precision metrics calculated for Herald resulting change point detections. In that case, it is suggested the

[1] `https://github.com/cefet-rj-dal/event_datasets`

exploration of alternative models for representing the time series inertia such that signals indicating the occurrence of a change point are not overly propagated and reinforced for the subsequent observations in the data stream. Techniques for aggregating detections that correspond to a particular change point event can also be studied.

Finally, a qualitative analysis of the contribution of Herald to the problem of change point detection in different domains can be a complementary study to the evaluation presented in this thesis. In particular, it can be based on domain specialists' knowledge acquired by a systematic survey. It should provide a qualitative evaluation of whether Herald can produce detections that enable acceptable reaction response time to change points in different domains.

Overview of scientific production

This section provides an overview of scientific research products resulting from this research. In particular, the main preliminary scientific products generated by this research that are relevant to this thesis and the development of the Herald method are presented in Chapter 1. They include the most recent version of the TSPred R-package [Salles et al., 2022], the Harbinger framework [Salles et al., 2020], which has also been made available as an R-package [Salles et al., 2023a], and the SoftED metrics which were formalized [Salles et al., 2023b], and included in the Harbinger R-package.

The TSPred R-package implements a systematic framework for nonstationary time series prediction. It is the first tool to seamlessly integrate a broad range of transformation methods [Salles et al., 2019] and state-of-the-art MLM prediction methods for addressing nonstationary time series. It has been downloaded almost $80,000$ times being in the 69th percentile of impact on CRAN (according to `depsy.org`). The earliest versions of *TSPred* were published in the International Joint Conference on Neural Networks (IJCNN) [Salles et al., 2017], while the current version of *TSPred* package was formally presented in the Neurocomputing journal [Salles et al., 2022].

The R-package for the Harbinger framework [Salles et al., 2023a] focuses on the integration and analysis of several time series event detection methods enabling a comparative analysis of their detection performance, as well as the combined use of

methods. Although recently published, the Harbinger R-package counts with almost 700 downloads. It was presented in a paper published in the XXXV Brazilian Database Symposium (SBBD) [Salles et al., 2020].

Moreover, this research on time series event detection resulted in a set of novel classification metrics, called *SoftED* [Salles et al., 2023b], for soft evaluation of event detection performance, being designed for online applications. The *SoftED* metrics are in the process of publication and have been introduced in preprint [Salles et al., 2023b], which is about to be submitted to a journal in the area of computer science.

In addition, this section presents other publications produced during the period of this doctorate either as a consequence of this thesis, or that contributed to its development. Among them, we mention the main and most directly related to this research, which includes:

- Best paper (category: short, vision, industry) presenting a novel model for nonstationary time series (Autoregressive Adaptive Integration Model (ARAI)) published in the proceedings of the XXXIV Brazilian Database Symposium (SBBD) [Ronald et al., 2019].
- TSPred was applied in the development of a paper on the use of data science to predict fertilizer consumption in Brazil published in the proceedings of the XIV Brazilian e-Science Workshop [Andrade et al., 2020].
- Reviewed nonstationary preprocessing techniques and implemented TSPred framework were applied to estimate COVID-19 under-reporting in the Brazilian States through SARI in a paper published in the New Generation Computing journal [Paixão et al., 2021].
- A detection delay measure was used to evaluate the ability of algorithms to detect real events in time series in a paper published in the Journal of Information and Data Management [Escobar et al., 2021]. This paper studies the time distance between event detections and events and furthers a qualitative analysis of the tendency of algorithms to detect before or after the occurrence of an event.
- A taxonomy for online detection of events in time series, including incremental and adaptive learning and some of the main methods addressed in the literature was published in the Proceedings of the XLII Ibero-Latin-American Congress on Computational Methods in Engineering [Ogasawara et al., 2021].

- Challenges of machine learning approaches to extreme weather event prediction in urban areas, in particular in Rio de Janeiro, were presented in a paper published in the Supercomputing Frontiers and Innovations journal [Porto et al., 2022]. Developing techniques to improve rainfall prediction and extreme rainfall forecast were presented.

- A novel clustering-based approach to hyperparameter selection, called CHyper was proposed and evaluated in the context of time series prediction of annual fertilizer consumption in a paper published in the XXXVII Brazilian Database Symposium (SBBD) [Izaú et al., 2022].

- A novel method for anomaly detection called Forward and Backward Inertial Anomaly Detector (FBIAD) was published in the International Joint Conference on Neural Networks (IJCNN) [Lima et al., 2022]. The method analyzes surrounding inertia around observations in order to identify anomalies, being able to outperform reference methods both in accuracy and elapsed time.

- The TSPredIT framework that seamlessly integrates the tuning of data preprocessing hyperparameters and model building for time series prediction was published in the International Journal on Transactions on Large-Scale Data- and Knowledge-Centered Systems (TLDKS) [Salles et al., 2023c]. TSPredIT is currently available on GitHub[2]. It is an ongoing evolution of TSPred, built on top of the DAL Toolbox[3].

Overall, there were 22 papers published during the period of this doctorate. However, in an aggregated overview, the main scientific papers of this research, described in this section, sum to a total of 11 publications, plus a preprint of the formalization of the SoftED metrics. Among these publications, 6 were published in computer science conference proceedings, while 5 corresponded to journal papers, which were regularly published over the duration period of this research. Also, it is important to mention that 4 of the publications were produced with international collaboration, namely the works of Lima et al. [2022], Salles et al. [2022], Salles et al. [2023c], and Salles et al. [2023b].

In summary, this thesis aims to contribute to the progress of the field of online change point event detection in time series. Gaps in the literature were observed regarding early and accurate event detection in the presence of nonstationary time series behavior. This thesis addressed these gaps by providing research on time series event detection,

[2] https://github.com/cefet-rj-dal/tspredit
[3] https://cran.r-project.org/web/packages/daltoolbox/index.html

and how it can benefit from studies of the areas of nonstationary time series transformations and time series prediction. Results were published in papers and encapsulated in frameworks and R-packages that were made widely available for the research community, including new sets of metrics that address the demand for soft evaluation of event detection methods. Moreover, this thesis confirmed the contribution of a lagged prediction of the inertial component of a time series to the problem of online change point event detection, ultimately producing a novel event detection method with the potential to lower detection lags, especially for applications characterized by high volatility levels.

References

Afzalan, M., Jazizadeh, F., and Wang, J. Self-configuring event detection in electricity monitoring for human-building interaction. *Energy and Buildings*, 187:95–109, 2019.

Aggarwal, C. C. *Outlier Analysis*. Springer Science & Business Media, 2013.

Ahmad, S., Lavin, A., Purdy, S., and Agha, Z. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017.

Ahmed, M., Mahmood, A., and Islam, M. A survey of anomaly detection techniques in financial domain. *Future Generation Computer Systems*, 55:278–288, 2016.

Alevizos, E., Skarlatidis, A., Artikis, A., and Paliouras, G. Probabilistic complex event recognition: A survey. *ACM Computing Surveys*, 50(5), 2017.

Alexopoulos, C. Statistical analysis of simulation output: state of the art. In *2007 Winter Simulation Conference*, pages 150–161. IEEE, 2007.

Ali, A., Khelil, A., Shaikh, F., and Suri, N. MPM: Map based Predictive Monitoring for Wireless Sensor Networks. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, 23 LNICST:79–95, 2010.

Ali, A., Khelil, A., Shaikh, F., and Suri, N. Efficient predictive monitoring of wireless sensor networks. *International Journal of Autonomous and Adaptive Communications Systems*, 5(3):233–254, 2012.

Alkhamees, N. and Fasli, M. Event detection from time-series streams using directional change and dynamic thresholds. In *Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017*, volume 2018-January, pages 1882–1891, 2017.

Aminikhanghahi, S. and Cook, D. A survey of methods for time series change point detection. *Knowledge and Information Systems*, 51(2):339–367, 2017.

Amodeo, G., Blanco, R., and Brefeld, U. Hybrid models for future event prediction. In *International Conference on Information and Knowledge Management, Proceedings*, pages 1981–1984, 2011.

An, X., Jiang, D., Liu, C., and Zhao, M. Wind farm power prediction based on wavelet decomposition and chaotic time series. *Expert Systems with Applications*, 38(9):11280–11285, 2011.

Andrade, A., Salles, R., Carvalho, F., da Silva, E. B., Soares, J., Souza, C., Gonzalez, P. H., and Ogasawara, E. Uso de ciência de dados para predição do consumo de fertilizantes no brasil. In *Anais do XIV Brazilian e-Science Workshop*, pages 9–16. SBC, 2020.

Andrews, D. W. K. Tests for parameter instability and structural change with unknown change point. *Econometrica*, 61(4):821–856, 1993.

Andrienko, G., Andrienko, N., Mladenov, M., Mock, M., and Poelitz, C. Extracting events from spatial time series. In *Proceedings of the International Conference on Information Visualisation*, pages 48–53, 2010.

Ariyaluran Habeeb, R., Nasaruddin, F., Gani, A., Targio Hashem, I., Ahmed, E., and Imran, M. Real-time big data processing for anomaly detection: A Survey. *International Journal of Information Management*, 45:289–307, 2019.

Ba, A. and McKenna, S. Water quality monitoring with online change-point detection methods. *Journal of Hydroinformatics*, 17(1):7–19, 2015.

Bai, X., Xiong, Y., Zhu, Y., Liu, Q., and Chen, Z. Co-anomaly event detection in multiple temperature series. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8041 LNAI:1–14, 2013.

Batal, I., Cooper, G., Fradkin, D., Harrison, J., J., Moerchen, F., and Hauskrecht, M. An efficient pattern mining approach for event detection in multivariate temporal data. *Knowledge and Information Systems*, 46(1):115–150, 2016.

Batal, I., Fradkin, D., Harrison, J., Moerchen, F., and Hauskrecht, M. Mining recent temporal patterns for event detection in multivariate time series data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 280–288, 2012.

Ben-Shachar, M. S., Lüdecke, D., and Makowski, D. effectsize: Estimation of effect size indices and standardized parameters. *Journal of Open Source Software*, 5(56):2815, 2020.

Blázquez-García, A., Conde, A., Mori, U., and Lozano, J. A Review on Outlier/Anomaly Detection in Time Series Data. *ACM Computing Surveys*, 54(3), 2021.

Borges, H., Dutra, M., Bazaz, A., Coutinho, R., Perosi, F., Porto, F., Masseglia, F., Pacitti, E., and Ogasawara, E. Spatial-time motifs discovery. *Intelligent Data Analysis*, 24(5):1121–1140, 2020.

Braei, M. and Wagner, S. Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art, 2020.

Breiman, L. Random forests. *Machine Learning*, 45(1):5–32, 2001.

Buza, K. Time series classification and its applications. In *ACM International Conference Proceeding Series*, 2018.

Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), 2009.

Chandola, V., Mithal, V., and Kumar, V. Comparative evaluation of anomaly detection techniques for sequence data. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 743–748, 2008.

Chauhan, V., Dahiya, K., and Sharma, A. Problem formulations and solvers in linear SVM: a review. *Artificial Intelligence Review*, 52(2):803–855, 2019.

Chen, H. and Zhang, N. Graph-based change-point detection. *Annals of Statistics*, 43(1):139–176, 2015.

Chen, X., Mueen, A., Narayanan, V., Karampatziakis, N., Bansal, G., and Kumar, V. Online discovery of group level events in time series. In *SIAM International Conference on Data Mining 2014, SDM 2014*, volume 2, pages 632–640, 2014.

Cheng, C., Sa-Ngasoongsong, A., Beyca, O., Le, T., Yang, H., Kong, Z., and Bukkapatnam, S. Time series forecasting for nonlinear and non-stationary processes: A review and comparative study. *IIE Transactions (Institute of Industrial Engineers)*, 47(10):1053–1071, 2015.

Chiroma, H., Abdul-Kareem, S., Shukri Mohd Noor, A., Abubakar, A., Sohrabi Safa, N., Shuib, L., Fatihu Hamza, M., Ya'u Gital, A., and Herawan, T. A Review on Artificial

Intelligence Methodologies for the Forecasting of Crude Oil Price. *Intelligent Automation and Soft Computing*, 22(3):449–462, 2016.

Chiu, B., Keogh, E., and Lonardi, S. Probabilistic discovery of time series motifs. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 493–498, 2003.

Choudhary, D., Kejariwal, A., and Orsini, F. On the Runtime-Efficacy Trade-off of Anomaly Detection Techniques for Real-Time Streaming Data, 2017.

Chow, G. C. Tests of equality between sets of coefficients in two linear regressions. *Econometrica: Journal of the Econometric Society*, pages 591–605, 1960.

Chu, C.-S. J., Stinchcombe, M., and White, H. Monitoring structural change. *Econometrica*, 64(5):1045–1065, 1996.

Clements, M. P. and Hendry, D. F. *A Companion to Economic Forecasting*. John Wiley & Sons, 2008.

Conejo, A., Plazas, M., Espínola, R., and Molina, A. Day-ahead electricity price forecasting using the wavelet transform and ARIMA models. *IEEE Transactions on Power Systems*, 20(2):1035–1042, 2005.

Cook, A., Misirli, G., and Fan, Z. Anomaly Detection for IoT Time-Series Data: A Survey. *IEEE Internet of Things Journal*, 7(7):6481–6494, 2020.

Cook, D. J. and Krishnan, N. C. *Activity Learning: Discovering, Recognizing, and Predicting Human Behavior from Sensor Data*. John Wiley & Sons, 2015.

da Silva, D., Simões, A., Cardoso, C., de Oliveira, D., Rittmeyer, J., Wehmuth, K., Lustosa, H., Pereira, R., Souto, Y., Vignoli, L., Salles, R., de Heleno, S.C., J., Ziviani, A., Ogasawara, E., Delicato, F., de Pires, P., da Pinto, H., Maia, L., and Porto, F. A conceptual vision toward the management of machine learning models. In *CEUR Workshop Proceedings*, volume 2469, pages 15–27, 2019.

Damle, C. and Yalcin, A. Flood prediction using Time Series Data Mining. *Journal of Hydrology*, 333(2-4):305–316, 2007.

Davydenko, A. and Fildes, R. Measuring Forecasting Accuracy: The Case Of Judgmental Adjustments To Sku-Level Demand Forecasts. *International Journal of Forecasting*, 29(3):510–522, 2013.

De Paepe, D., Vanden Hautte, S., Steenwinckel, B., De Turck, F., Ongenae, F., Janssens, O., and Van Hoecke, S. A generalized matrix profile framework with support for contextual series analysis. *Engineering Applications of Artificial Intelligence*, 90, 2020.

DiCecco, J., Salisbury, J., and Sun, Y. A sequential algorithm for biological event detection using statistical nonstationarity. In *Proceedings of the IEEE Annual Northeast Bioengineering Conference, NEBEC*, volume 2006, pages 147–148, 2006.

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. Querying and mining of time series data: Experimental comparison of representations and distance measures. In *Proceedings of the VLDB Endowment*, volume 1, pages 1542–1552, 2008.

Ding, X., Li, Y., Belatreche, A., and Maguire, L. An experimental evaluation of novelty detection methods. *Neurocomputing*, 135:313–327, 2014.

Ditzler, G., Roveri, M., Alippi, C., and Polikar, R. Learning in Nonstationary Environments: A Survey. *IEEE Computational Intelligence Magazine*, 10(4):12–25, 2015.

Dudek, G. Neural networks for pattern-based short-term load forecasting: A comparative study. *Neurocomputing*, 205:64–74, 2016.

Efros, P., Buchmann, E., Englhardt, A., and Böhm, K. How to Quantify the Impact of Lossy Transformations on Event Detection. *Big Data Research*, 9:84–97, 2017.

Eichmann, P., Tatbul, N., Solleza, F., and Zdonik, S. Visual exploration of time series anomalies with metro-viz. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1901–1904, 2019.

Escobar, L., Salles, R., Lima, J., Gea, C., Baroni, L., Ziviani, A., Pires, P., Delicato, F., Coutinho, R., Assis, L., and Ogasawara, E. Evaluating Temporal Bias in Time Series Event Detection Methods. *Journal of Information and Data Management*, 12(3), 2021.

Esling, P. and Agon, C. Time-series data mining. *ACM Computing Surveys*, 45(1), 2012.

Fanaee-T, H. and Gama, J. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 2(2-3):113–127, 2014.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. Efficient and robust automated machine learning. In *Advances in neural information processing systems*, pages 2962–2970, 2015.

Flach, M., Gans, F., Brenning, A., Denzler, J., Reichstein, M., Rodner, E., Bathiany, S., Bodesheim, P., Guanche, Y., Sippel, S., and Mahecha, M. Multivariate anomaly detection for Earth observations: A comparison of algorithms and feature extraction techniques. *Earth System Dynamics*, 8(3):677–696, 2017.

Fontugne, R., Borgnat, P., Abry, P., and Fukuda, K. MAWILab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In *Proceedings of the 6th International Conference on Emerging Networking Experiments and Technologies, Co-NEXT'10*, 2010.

Fox, J. *Applied Regression Analysis and Generalized Linear Models*. SAGE Publications, 2015.

Funder, D. C. and Ozer, D. J. Evaluating effect size in psychological research: Sense and nonsense. *Advances in Methods and Practices in Psychological Science*, 2(2):156–168, 2019.

Gabarda, S. and Cristóbal, G. Detection of events in seismic time series by time-frequency methods. *IET Signal Processing*, 4(4):413–420, 2010.

Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4), 2014.

Gensler, A. and Sick, B. Performing event detection in time series with SwiftEvent: an algorithm with supervised learning of detection criteria. *Pattern Analysis and Applications*, 21(2):543–562, 2018.

Gholami, E. and Borujerdi, M. Fuzzy knowledge discovery from time series data for events prediction. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5351 LNAI:646–657, 2008.

Girish, G. and Tiwari, A. A comparison of different univariate forecasting models for spot electricity price in India. *Economics Bulletin*, 36(2):1039–1057, 2016.

Gmati, F., Chakhar, S., Chaari, W., and Xu, M. A taxonomy of event prediction methods. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11606 LNAI:12–26, 2019.

Greff, K., Srivastava, R., Koutnik, J., Steunebrink, B., and Schmidhuber, J. LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, 2017.

Guanche García, Y., Shadaydeh, M., Mahecha, M., and Denzler, J. Extreme anomaly event detection in biosphere using linear regression and a spatiotemporal MRF model. *Natural Hazards*, 98(3):849–867, 2019.

Gujarati, D. N. *Essentials of Econometrics*. SAGE, 2021.

Guo, T., Dong, J., Li, H., and Gao, Y. Simple convolutional neural network on image classification. In *2017 IEEE 2nd International Conference on Big Data Analysis, ICBDA 2017*, pages 721–724, 2017.

Gupta, M., Gao, J., Aggarwal, C., and Han, J. Outlier Detection for Temporal Data: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 26(9):2250–2267, 2014.

Guralnik, V. and Srivastava, J. Event Detection from Time Series Data. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '99, pages 33–42, New York, NY, USA. ACM, 1999.

Han, J., Kamber, M., and Pei, J. *Data Mining: Concepts and Techniques*, 2012.

Han, J., Pei, J., and Tong, H. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, Cambridge, MA, 4th edition edition, 2022.

Hanssens, D. M., Parsons, L. J., and Schultz, R. L. *Market Response Models: Econometric and Time Series Analysis*. Springer Science & Business Media, 2006.

Haykin, S. O. *Neural Networks and Learning Machines*. Pearson Education, 2011.

Henriques, J., Carvalho, P., Rocha, T., Paredes, S., and Morais, J. Multi-parametric trends analysis and events prediction in the context of a cardiac rehabilitation system. In *IFMBE Proceedings*, volume 45, pages 678–681, 2015.

Hodge, V. and Austin, J. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

Hollander, M., Wolfe, D. A., and Chicken, E. *Nonparametric statistical methods*. John Wiley & Sons, 2013.

Huang, G.-B., Zhu, Q.-Y., and Siew, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

Hüllermeier, E. Fuzzy sets in machine learning and data mining. *Applied Soft Computing*, 11(2):1493–1505, 2011.

Hunter, J. and McIntosh, N. Knowledge-based event detection in complex time series data. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1620:271–280, 1999.

Hyndman, R. J. and Athanasopoulos, G. *Forecasting: principles and practice*. OTexts, 2018.

Ihler, A., Hutchins, J., and Smyth, P. Adaptive event detection with time - Varying poisson processes. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 2006, pages 207–216, 2006.

Ilhan, N. and Oguducu, S. Predicting community evolution based on time series modeling. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2015*, pages 1509–1516, 2015.

Ismaeel, S., Miri, A., and Chourishi, D. Using the Extreme Learning Machine (ELM) technique for heart disease diagnosis. In *2015 IEEE Canada International Humanitarian Technology Conference, IHTC 2015*, 2015.

Iwashita, A. and Papa, J. An Overview on Concept Drift Learning. *IEEE Access*, 7:1532–1547, 2019.

Izaú, L., Fortes, M., Ribeiro, V., Marques, C., Oliveira, C., Bezerra, E., Porto, F., Salles, R., and Ogasawara, E. Towards robust cluster-based hyperparameter optimization. In *Anais do XXXVII Simpósio Brasileiro de Bancos de Dados*, pages 439–444. SBC, 2022.

Jin, H., Song, Q., and Hu, X. Efficient neural architecture search with network morphism. *arXiv preprint arXiv:1806.10282*, 2018.

Joo, T. and Kim, S. Time series forecasting based on wavelet filtering. *Expert Systems with Applications*, 42(8):3868–3874, 2015.

Kassambara, A. rstatix: Pipe-Friendly Framework for Basic Statistical Tests, 2023.

Kattan, A., Fatima, S., and Arif, M. Time-series event-based prediction: An unsupervised learning framework based on genetic programming. *Information Sciences*, 301:99–123, 2015.

Keogh, E. and Lin, J. Clustering of time-series subsequences is meaningless: Implications for previous and future research. *Knowledge and Information Systems*, 8(2):154–177, 2005.

Kerby, D. S. The simple difference formula: An approach to teaching nonparametric correlation. *Comprehensive Psychology*, 3:11–IT, 2014.

Kiskin, I., Zilli, D., Li, Y., Sinka, M., Willis, K., and Roberts, S. Bioacoustic detection with wavelet-conditioned convolutional neural networks. *Neural Computing and Applications*, 32(4):915–927, 2020.

Kotthoff, L., Thornton, C., Hoos, H. H., Hutter, F., and Leyton-Brown, K. Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka. *The Journal of Machine Learning Research*, 18(1):826–830, 2017.

Lahmiri, S. Interest rate next-day variation prediction based on hybrid feedforward neural network, particle swarm optimization, and multiresolution techniques. *Physica A: Statistical Mechanics and its Applications*, 444:388–396, 2016.

Lajevardi, S. and Minaei-Bidgoli, B. Combination of time series, Decision Tree and clustering: A case study in Aerology event prediction. In *Proceedings of the 2008 International Conference on Computer and Electrical Engineering, ICCEE 2008*, pages 111–115, 2008a.

Lajevardi, S. and Minaei-Bidgoli, B. Comparison between ANN and decision tree in aerology event prediction. In *Proceedings - 2008 International Conference on Advanced Computer Theory and Engineering, ICACTE 2008*, pages 533–537, 2008b.

Lampert, C., Blaschko, M., and Hofmann, T. Beyond sliding windows: Object localization by efficient subwindow search. In *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2008.

Laptev, N., Yosinski, J., Li, L. E., and Smyl, S. Time-series extreme event forecasting with neural networks at uber. In *International conference on machine learning*, volume 34, pages 1–5, 2017.

Lavin, A. and Ahmad, S. Evaluating Real-Time Anomaly Detection Algorithms – The Numenta Anomaly Benchmark. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 38–44, 2015.

Lavin, A. and Ahmad, S. Evaluating real-time anomaly detection algorithms - The numenta anomaly benchmark. In *Proceedings - 2015 IEEE 14th International Conference on Machine Learning and Applications, ICMLA 2015*, pages 38–44, 2016.

Law, A. M. Statistical analysis of simulation output data: the practical state of the art. In *2020 winter simulation conference (WSC)*, pages 1117–1127. IEEE, 2020.

Lawhern, V., Hairston, W., and Robbins, K. DETECT: A MATLAB Toolbox for Event Detection and Identification in Time Series, with Applications to Artifact Detection in EEG Signals. *PLoS ONE*, 8(4), 2013.

Le Page, M., Jarlan, L., El Hajj, M., Zribi, M., Baghdadi, N., and Boone, A. Potential for the detection of irrigation events on maize plots using Sentinel-1 soil moisture products. *Remote Sensing*, 12(10), 2020.

Lesti, G. and Spiegel, S. A sliding window filter for time series streams. In *CEUR Workshop Proceedings*, volume 1958, 2017.

Lim, B. and Zohren, S. Time-series forecasting with deep learning: A survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194), 2021.

Lima, J., Salles, R., Porto, F., Coutinho, R., Alpis, P., Escobar, L., Pacitti, E., and Ogasawara, E. Forward and Backward Inertial Anomaly Detector: A Novel Time Series Event Detection Method. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2022-July, 2022.

Liu, S., Smith, K., and Che, H. A multivariate based event detection method and performance comparison with two baseline methods. *Water Research*, 80:109–118, 2015.

Lomio, F., Baselga, D. M., Moreschini, S., Huttunen, H., and Taibi, D. Rare: a labeled dataset for cloud-native memory anomalies. In *Proceedings of the 4th ACM SIGSOFT International Workshop on Machine-Learning Techniques for Software-Quality Evaluation*, pages 19–24, 2020.

Los, C. A. *Financial Market Risk: Measurement and Analysis*. Routledge, 2003.

Loureiro, D., Amado, C., Martins, A., Vitorino, D., Mamade, A., and Coelho, S. Water distribution systems flow monitoring and anomalous event detection: A practical approach. *Urban Water Journal*, 13(3):242–252, 2016.

Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., and Zhang, G. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering*, 31(12):2346–2363, 2018.

Machado, E., Serqueira, M., Ogasawara, E., Ogando, R., Maia, M., Da Costa, L., Campisano, R., Paiva Guedes, G., and Bezerra, E. Exploring machine learning methods for the Star/Galaxy Separation Problem. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2016-October, pages 123–130, 2016.

Madani, P. and Vlajic, N. Effective segmentation of rssi timeseries produced by stationary iot nodes: Comparative study. In *International Workshop on Attacks and Defenses for Internet-of-Things*, pages 84–101. Springer, 2022.

Marik, R. and Bohac, L. Non-stationary Events Detection Based on Extrema Value Theory. In *2018 5th International Conference on Systems and Informatics, ICSAI 2018*, pages 889–894, 2019.

Marrocu, E. An investigation of the effects of data transformation on nonlinearity. *Empirical Economics*, 31(4):801–820, 2006.

McKenna, S., Hart, D., Klise, K., Cruz, V., and Wilson, M. Event detection from water quality time series. In *Restoring Our Natural Habitat - Proceedings of the 2007 World Environmental and Water Resources Congress*, 2007.

Meenakshi, S. and Raghavan, S. Forecasting and event detection in internet resource dynamics using time series models. *Engineering Letters*, 23(4):245–257, 2015.

Mehrmolaei, S. EPTs-TL: A two-level approach for efficient event prediction in healthcare. *Artificial Intelligence in Medicine*, 111, 2021.

Mehrmolaei, S. and Keyvanpourr, M. A brief survey on event prediction methods in time series. *Advances in Intelligent Systems and Computing*, 347:235–246, 2015.

Memarzadeh, M., Matthews, B., and Avrekh, I. Unsupervised anomaly detection in flight data using convolutional variational auto-encoder. *Aerospace*, 7(8), 2020.

Molaei, S. and Keyvanpour, M. An analytical review for event prediction system on time series. In *2015 2nd International Conference on Pattern Recognition and Image Analysis, IPRIA 2015*, 2015.

Mondal, T., Akbarinia, R., and Masseglia, F. knn matrix profile for knowledge discovery from time series. *Data Mining and Knowledge Discovery*, 37(3):1055–1089, 2023.

Morana, C. Multivariate modelling of long memory processes with common components. *Computational Statistics and Data Analysis*, 52(2):919–934, 2007.

Mueen, A. Time series motif discovery: Dimensions and applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(2):152–159, 2014.

Munir, M., Chattha, M., Dengel, A., and Ahmed, S. A comparative analysis of traditional and deep learning-based anomaly detection methods for streaming data. In *Proceedings - 18th IEEE International Conference on Machine Learning and Applications, ICMLA 2019*, pages 561–566, 2019.

Muniyandi, A., Rajeswari, R., and Rajaram, R. Network anomaly detection by cascading k-Means clustering and C4.5 decision tree algorithm. In *Procedia Engineering*, volume 30, pages 174–182, 2012.

Nachane, D. and Clavel, J. Forecasting interest rates: A comparative assessment of some second-generation nonlinear models. *Journal of Applied Statistics*, 35(5):493–514, 2008.

Noy, N. F. and Hafner, C. D. The state of the art in ontology design: A survey and comparative review. *AI magazine*, 18(3):53–53, 1997.

Ogasawara, E., Martinez, L., De Oliveira, D., Zimbrão, G., Pappa, G., and Mattoso, M. Adaptive Normalization: A novel data normalization approach for non-stationary time series. In *Proceedings of the International Joint Conference on Neural Networks*, 2010.

Ogasawara, E., Salles, R., Escobar, L., Baroni, L., Lima, J., and Porto, F. Online event detection for sensor data. In *Ibero-Latin American Congress on Computational Methods in Engineering*, Rio de Janeiro, RJ, 2021.

Ogasawara, E., Salles, R., Porto, F., and Pacitti, E. *Event Detection in Time Series (to appear)*. Springer Nature, 2024.

Paixão, B., Baroni, L., Pedroso, M., Salles, R., Escobar, L., de Sousa, C., de Freitas Saldanha, R., Soares, J., Coutinho, R., Porto, F., et al. Estimation of covid-19 underreporting in the brazilian states through sari. *New Generation Computing*, pages 1–23, 2021.

Pang, G., Shen, C., Cao, L., and Hengel, A. Deep Learning for Anomaly Detection: A Review. *ACM Computing Surveys*, 54(2), 2021.

Park, P., Di Marco, P., Shin, H., and Bang, J. Fault detection and diagnosis using combined autoencoder and long short-term memory network. *Sensors (Switzerland)*, 19(21), 2019.

Pelech-Pilichowski, T. and Duda, J. General structure of T-lymphocyte applied to event detection in financial time series. *Systems Science*, 32(4):65–71, 2006.

Perelman, L., Arad, J., Housh, M., and Ostfeld, A. Event detection in water distribution systems from multivariate water quality time series. *Environmental Science and Technology*, 46(15):8212–8219, 2012.

Pimentel, M., Clifton, D., Clifton, L., and Tarassenko, L. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

Poggio, T., Laboratory, M. I. o. T. A. I., and Girosi, F. *A Theory of Networks for Approximation and Learning*. Massachusetts Institute of Technology, Artificial Intelligence Laboratory and Center for Biological Information Processing, Whitaker College, 1989.

Polz, J., Chwala, C., Graf, M., and Kunstmann, H. Rain event detection in commercial microwave link attenuation data using convolutional neural networks. *Atmospheric Measurement Techniques*, 13(7):3835–3853, 2020.

Porto, F., Ferro, M., Ogasawara, E., Moeda, T., de Barros, C. D. T., Silva, A. C., Zorrilla, R., Pereira, R. S., Castro, R. N., Silva, J. V., et al. Machine learning approaches to extreme weather events forecast in urban areas: Challenges and initial results. *Supercomputing Frontiers and Innovations*, 9(1):49–73, 2022.

Pratama, M., Lu, J., Lughofer, E., Zhang, G., and Er, M. An Incremental Learning of Concept Drifts Using Evolving Type-2 Recurrent Fuzzy Neural Networks. *IEEE Transactions on Fuzzy Systems*, 25(5):1175–1192, 2017.

Provotar, O., Linder, Y., and Veres, M. Unsupervised Anomaly Detection in Time Series Using LSTM-Based Autoencoders. In *2019 IEEE International Conference on Advanced Trends in Information Theory, ATIT 2019 - Proceedings*, pages 513–517, 2019.

Rahman, A., McCulloch, J., and Mamun, Q. Prediction with uncertainty: A novel framework for analyzing sensor data streams. *IEEE Sensors Journal*, 15(1):382–386, 2015.

Rahul and Choudhary, B. An Advanced Genetic Algorithm with Improved Support Vector Machine for Multi-Class Classification of Real Power Quality Events. *Electric Power Systems Research*, 191, 2021.

Raza, H., Prasad, G., and Li, Y. EWMA model based shift-detection methods for detecting covariate shifts in non-stationary environments. *Pattern Recognition*, 48(3):659–669, 2015.

Rehbach, F., Moritz, S., Chandrasekaran, S., Rebolledo, M., Friese, M., and Bartz-Beielstein, T. GECCO Industrial Challenge 2018 Dataset. Technical report, https://zenodo.org/record/3884398#.Y9qShK3MJD8, 2018.

Riese, F. and Keller, S. Supervised, semi-supervised, and unsupervised learning for hyperspectral regression. *Advances in Computer Vision and Pattern Recognition*, pages 187–232, 2020.

Rocha, T., Paredes, S., Carvalho, P., and Henriques, J. A wavelet-based approach for time series pattern detection and events prediction applied to telemonitoring data. In *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pages 6037–6040, 2011.

Ronald, A., Salles, R., Belloze, K., Pastore, D., and Ogasawara, E. Modelo autorregressivo de integração adaptativa. In *Anais do XXXIV Simpósio Brasileiro de Banco de Dados*, pages 175–180. SBC, 2019.

Roshan, W., Gopura, R., and Jayasekara, A. Financial forecasting based on artificial neural networks: Promising directions for modeling. In *2011 6th International Conference on Industrial and Information Systems, ICIIS 2011 - Conference Proceedings*, pages 322–327, 2011.

Salfner, F., Lenk, M., and Malek, M. A survey of online failure prediction methods. *ACM Computing Surveys*, 42(3), 2010.

Salles, R., Assis, L., Guedes, G., Bezerra, E., Porto, F., and Ogasawara, E. A framework for benchmarking machine learning methods using linear models for univariate time series prediction. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2017-May, pages 2338–2345, 2017.

Salles, R., Belloze, K., Porto, F., Gonzalez, P., and Ogasawara, E. Nonstationary time series transformation methods: An experimental review. *Knowledge-Based Systems*, 164:274–291, 2019.

Salles, R., Bezerra, E., Soares, J., and Ogasawara, E. Evaluating Linear Models as a Baseline for Time Series Imputation. In *XXX Simpósio Brasileiro de Banco de Dados*, Petrópolis, RJ, 2015.

Salles, R., Escobar, L., Baroni, L., Zorrilla, R., Ziviani, A., Kreischer, V., Delicato, F., Pires, P. F., Maia, L., Coutinho, R., Assis, L., and Ogasawara, E. Harbinger: Um framework para integração e análise de métodos de detecção de eventos em séries temporais. In *Anais do Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 73–84. SBC, 2020.

Salles, R., Lima, J., Baroni, L., Castro, A., Carvalho, L., Borges, H., Carvalho, D., Coutinho, R., Bezerra, E., Pacitti, E., Porto, F., Ogasawara, E., and Janeiro (CEFET/RJ), F. C. f. T. E. o. R. d. harbinger: An Unified Time Series Event Detection Framework, 2023a.

Salles, R., Lima, J., Coutinho, R., Pacitti, E., Masseglia, F., Akbarinia, R., Chen, C., Garibaldi, J., Porto, F., and Ogasawara, E. Softed: Metrics for soft evaluation of time series event detection, 2023b.

Salles, R. and Ogasawara, E. TSPred: Functions for Benchmarking Time Series Prediction. Technical report, https://cran.r-project.org/web/packages/TSPred/index.html, 2021.

Salles, R., Pacitti, E., Bezerra, E., Marques, C., Pacheco, C., Oliveira, C., Porto, F., and Ogasawara, E. Tspredit: Integrated tuning of data preprocessing and time series prediction models. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems LIV: Special Issue on Data Management-Principles, Technologies, and Applications*, pages 41–55. Springer, 2023c.

Salles, R., Pacitti, E., Bezerra, E., Porto, F., and Ogasawara, E. TSPred: A framework for nonstationary time series prediction. *Neurocomputing*, 467:197–202, 2022.

Sayed-Mouchaweh, M. and Lughofer, E. *Learning in Non-Stationary Environments: Methods and Applications*. Springer Science & Business Media, 2012.

Schmidl, S., Wenig, P., and Papenbrock, T. Anomaly detection in time series: A comprehensive evaluation. *Proceedings of the VLDB Endowment*, 15(9):1779–1797, 2022.

Sebestyen, G., Hangan, A., Czako, Z., and Kovacs, G. A taxonomy and platform for anomaly detection. In *2018 IEEE International Conference on Automation, Quality and Testing, Robotics, AQTR 2018 - THETA 21st Edition, Proceedings*, pages 1–6, 2018.

Seth, T. and Chaudhary, V. A Predictive Analytics Framework for Insider Trading Events. In *Proceedings - 2020 IEEE International Conference on Big Data, Big Data 2020*, pages 218–225, 2020.

Shu, M.-H., Hsu, B.-M., Nguyen, T.-L., Lu, C., and Huang, J.-C. Forecasting cargo throughput with modified seasonal ARIMA models. *WSEAS Transactions on Mathematics*, 13:171–181, 2014.

Shumway, R. H. and Stoffer, D. S. *Time Series Analysis and Its Applications: With R Examples*. Springer, 2017.

Singh, N. and Olinsky, C. Demystifying Numenta anomaly benchmark. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2017-May, pages 1570–1577, 2017.

Song, W., Wang, L., Xiang, Y., and Zomaya, A. Geographic spatiotemporal big data correlation analysis via the Hilbert–Huang transformation. *Journal of Computer and System Sciences*, 89:130–141, 2017.

Souto, G. and Liebig, T. On event detection from spatial time series for urban traffic applications. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9580:221–233, 2016.

Strassburger, K. and Bretz, F. Compatible simultaneous lower confidence bounds for the holm procedure and other bonferroni-based closed tests. *Statistics in medicine*, 27(24):4914–4927, 2008.

Sun, G., Chen, T., Wei, Z., Sun, Y., Zang, H., and Chen, S. A carbon price forecasting model based on variational mode decomposition and spiking neural networks. *Energies*, 9(1), 2016.

Taha, A. and Hadi, A. Anomaly detection methods for categorical data: A review. *ACM Computing Surveys*, 52(2), 2019.

Takeuchi, J.-I. and Yamanishi, K. A unifying framework for detecting outliers and change points from time series. *IEEE Transactions on Knowledge and Data Engineering*, 18(4):482–492, 2006.

Talagala, P., Hyndman, R., Smith-Miles, K., Kandanaarachchi, S., and Muñoz, M. Anomaly Detection in Streaming Nonstationary Temporal Data. *Journal of Computational and Graphical Statistics*, 29(1):13–27, 2020.

Tan, T. and Darken, C. Learning and prediction of relational time series. *Computational and Mathematical Organization Theory*, 21(2):210–241, 2015.

Tang, J., Deng, C., and Huang, G.-B. Extreme Learning Machine for Multilayer Perceptron. *IEEE Transactions on Neural Networks and Learning Systems*, 27(4):809–821, 2016.

Tatbul, N., Lee, T., Zdonik, S., Alam, M., and Gottschlich, J. Precision and recall for time series. In *Advances in Neural Information Processing Systems*, volume 2018-December, pages 1920–1930, 2018.

Tettamanzi, A. and Tomassini, M. *Soft Computing: Integrating Evolutionary, Neural, and Fuzzy Systems*. Springer Science & Business Media, 2013.

Thudumu, S., Branch, P., Jin, J., and Singh, J. A comprehensive survey of anomaly detection techniques for high dimensional big data. *Journal of Big Data*, 7(1), 2020.

Truong, C., Oudre, L., and Vayatis, N. Selective review of offline change point detection methods. *Signal Processing*, 167, 2020.

Tsay, R. S. *Analysis of Financial Time Series*. Wiley, 2010.

Uschold, M. and Gruninger, M. Ontologies: Principles, methods and applications. *The knowledge engineering review*, 11(2):93–136, 1996.

Van Hoan, M. and Exbrayat, M. Time series symbolization and search for frequent patterns. In *ACM International Conference Proceeding Series*, pages 108–117, 2013.

Verma, T. and Pearl, J. *Equivalence and Synthese of Causal Models*. University of California (Los Angeles). Computer Science Department, 1991.

Vishnusai, Y., Kulakarni, T. R., and Sowmya Nag, K. Ablation of artificial neural networks. *Lecture Notes on Data Engineering and Communications Technologies*, 46:453 – 460, 2020.

Wang, H., Bah, M., and Hammad, M. Progress in Outlier Detection Techniques: A Survey. *IEEE Access*, 7:107964–108000, 2019.

Wang, H., Liu, L., Dong, S., Qian, Z., and Wei, H. A novel work zone short-term vehicle-type specific traffic speed prediction model through the hybrid EMD–ARIMA framework. *Transportmetrica B*, 4(3):159–186, 2016.

Wang, W., Lv, W., and Lang, X. Impact of traffic event prediction based on time series prediction. In *ACM International Conference Proceeding Series*, volume Part F137704, pages 78–82, 2018.

Wang, Y., Vuran, M., and Goddard, S. Analysis of event detection delay in wireless sensor networks. In *Proceedings - IEEE INFOCOM*, pages 1296–1304, 2011.

Ward, J., Lukowicz, P., and Gellersen, H. Performance metrics for activity recognition. *ACM Transactions on Intelligent Systems and Technology*, 2(1), 2011.

Webscope, Y. Labeled anomaly detection dataset. Technical report, https://webscope.sandbox.yahoo.com/catalog.php?datatype=s, 2015.

Wickham, H. *Advanced R*. CRC Press, 2019.

Wu, R. and Keogh, E. Current Time Series Anomaly Detection Benchmarks are Flawed and are Creating the Illusion of Progress. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

Xi, L., Kong, L., Xiao, S., and Chen, G. Prediction model of non-stationary time series parameters for a complex blending process. In *26th Chinese Control and Decision Conference, CCDC 2014*, pages 1027–1030, 2014.

Yadav, R., Pradhan, A., and Kamwa, I. Real-Time Multiple Event Detection and Classification in Power System Using Signal Energy Transformations. *IEEE Transactions on Industrial Informatics*, 15(3):1521–1531, 2019.

Yan, X.-B., Tao, L., Li, Y.-J., and Cui, G.-B. Research on event prediction in time-series data. In *Proceedings of 2004 International Conference on Machine Learning and Cybernetics*, volume 5, pages 2874–2878, 2004.

Yang, W. and Zurbenko, I. Nonstationarity. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):107–115, 2010.

Yeh, C.-C., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, H., Zimmerman, Z., Silva, D., Mueen, A., and Keogh, E. Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile. *Data Mining and Knowledge Discovery*, 32(1):83–123, 2018.

Yeh, C.-C. M., Van Herle, H., and Keogh, E. Matrix Profile III: The Matrix Profile Allows Visualization of Salient Subsequences in Massive Time Series. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 579–588, 2016.

Yu, S., Gu, L., and Dai, W. Fast event detection on big time series. In *2014 IEEE/CIC International Conference on Communications in China, ICCC 2014*, pages 329–333, 2015.

Zadeh, L. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

Zeileis, A. A unified approach to structural change tests based on ml scores, f statistics, and ols residuals. *Econometric Reviews*, 24(4):445–466, 2005.

Zeileis, A., Leisch, F., Hornik, K., and Kleiber, C. strucchange: An r package for testing for structural change in linear regression models. *Journal of statistical software*, 7:1–38, 2002.

Zeileis, A., Shah, A., and Patnaik, I. Testing, monitoring, and dating structural changes in exchange rate regimes. *Computational Statistics & Data Analysis*, 54(6):1696–1706, 2010.

Zhang, C., Liu, Y., Wu, F., Fan, W., Tang, J., and Liu, H. Multi-Dimensional Joint Prediction Model for IoT Sensor Data Search. *IEEE Access*, 7:90863–90873, 2019.

Zhang, J., Wu, C., and Wang, Y. Human fall detection based on body posture spatio-temporal evolution. *Sensors (Switzerland)*, 20(3), 2020.

Zhang, Y., Meratnia, N., and Havinga, P. Outlier detection techniques for wireless sensor networks: A survey. *IEEE Communications Surveys and Tutorials*, 12(2):159–170, 2010.

Zhao, L. Event Prediction in the Big Data Era: A Systematic Survey. *ACM Computing Surveys*, 54(5), 2021.

Zhao, Q., Zhang, Y., Shi, Y., and Li, J. Analyzing and Visualizing Anomalies and Events in Time Series of Network Traffic. *Advances in Intelligent Systems and Computing*, 936:15–25, 2020.

Zhou, Y., Arghandeh, R., Zou, H., and Spanos, C. Nonparametric Event Detection in Multiple Time Series for Power Distribution Networks. *IEEE Transactions on Industrial Electronics*, 66(2):1619–1628, 2019.

Zhu, Y., Yeh, C.-C., Zimmerman, Z., Kamgar, K., and Keogh, E. Matrix Profile XI: SCRIMP++: Time Series Motif Discovery at Interactive Speeds. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, volume 2018-November, pages 837–846, 2018.

A- SoftED metrics

This Appendix is published as a part of the preprint paper by Salles et al. [2023b]. The paper formalizes and evaluates the SoftED metrics. The metrics are implemented and included in the latest version of the Harbinger R-package [Salles et al., 2023a]. This Appendix presents a self-contained idea of the SoftED metrics and their formalization. Related works and experimental evaluation can be found in the preprint paper by Salles et al. [2023b].

A.1- Metric formalization

This research adopts a distance-based approach to develop novel metrics designed to evaluate the performance of methods for detecting events in time series. The inspiration for the proposed solution is found in soft (or approximate) computing. Soft computing is a collection of methodologies that exploit tolerance for inaccuracy, uncertainty, and partial truth to achieve tractability, robustness, and low solution cost [Tettamanzi and Tomassini, 2013]. In this context, the main proposed idea is to soften the hard metrics (standard classification metrics) to incorporate temporal tolerance or inaccuracy in event detection. Such metrics seek to support the decision-making of the most appropriate method for a given application with a basis not only on the usual analysis of the detection accuracy but also on the analysis of the ability of a method to produce detections that are close enough to actual time series events. Henceforth, the proposed approach is named Soft classification metrics for Event Detection, or SoftED. This section formalizes the SoftED metrics.

A.1.1- Defining an event membership function

In order to soften the standard hard metrics, we incorporate a distance-based temporal tolerance for events. It is done by defining the relevance of a particular detection to an event. This section formalizes the proposed approach. Table 8 defines the main variables used in the formalization of SoftED. Given a time series X of length $|X|$ containing a set of m events, $E = \{e_1, e_2, \ldots, e_m\}$, where e_j, $j = 1, \ldots, m$, is the j–th event in E occurring at time point t_{e_j}. A particular detection method applied to X produces a set of n detections, $D = \{d_1, d_2, \ldots, d_n\}$, where d_i, $i = 1, \ldots, n$, is the i–th detection in D indicating the time point t_{d_i} as a detection occurrence.

Table 8 – Definition of main variables for the formalization of SoftED

Var.	Value	Description
E	$\{e_1, e_2, \ldots, e_m\}$	set of time series events
m	$\|E\|$	number of events
j	$1, \ldots, m$	event index
e_j	$-$	the j–th event in E
t_{e_j}	time point	time point where the e_j occurs
D	$\{d_1, d_2, \ldots, d_n\}$	set of detections
n	$\|D\|$	number of detections
i	$1, \ldots, n$	detection index
d_i	$-$	the i–th detection in D
t_{d_i}	time point	time point where d_i occurs
k	time duration	constant of tolerance for event detections

The degree to which a detection d_i is relevant to a particular event e_j is given by an event membership function $\mu_{e_j}(t)$ as defined in Equation 13 and illustrated in Figure 27a. The $\mu_{e_j}(t)$ represents a euclidean distance function, which is of simple computation and interpretation. Moreover, this solution was inspired by Fuzzy sets, where we innovate by fuzzifying the time dimension rather than the time series observations [Zadeh, 1965]. The definition of $\mu_{e_j}(t)$ considers the acceptable tolerance for inaccuracy in event detection for a particular domain application. The acceptable time range in which an event detection is relevant for allowing an adequate response reaction to a domain event is given by the constant k. Take for example a meteorologist that needs to provide alerts of extreme

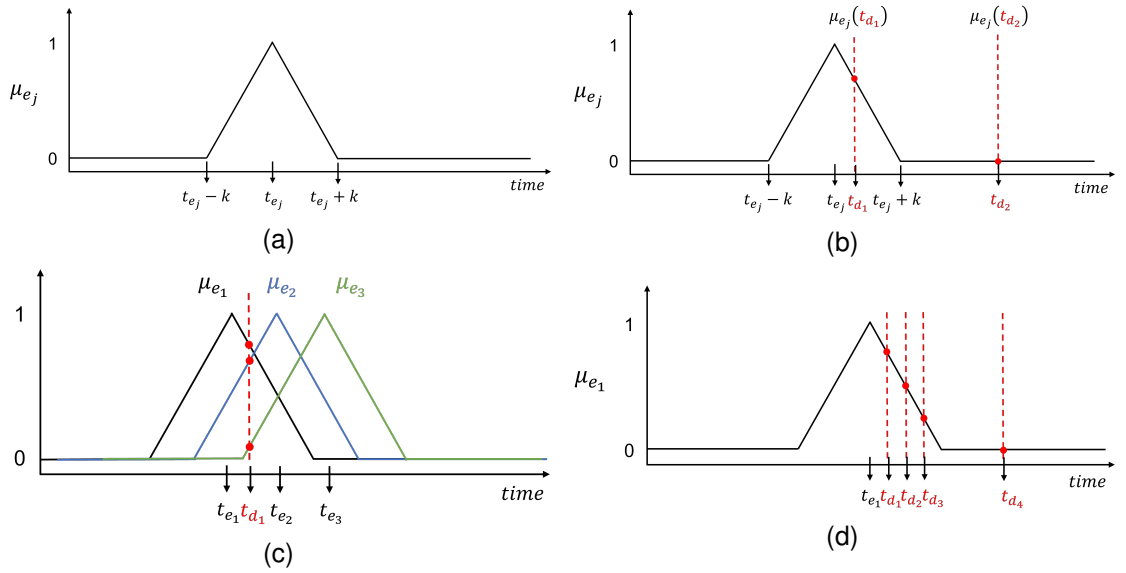


Figure 27 – Auxiliary plots for comprehension of SoftED. (a) represents an event membership function $\mu_{e_j}(t)$. (b) represents $\mu_{e_j}(t)$ for detections d_1 and d_2. (c) depicts the example scenario containing one detection to many events, motivating the first constraint of SoftED. (d) depicts the example scenario containing many detections to a single event, motivating the second constraint of SoftED.

weather events at least 6 hours in advance, or an engineer that needs to intervene in the operation of an overheated piece of machinery within 5 minutes before critical system failure. In this case, they might set k as ranging to 6 hours, or 5 minutes, respectively. However, this approach defines a domain-agnostic default value of k, which is set to 15, defining a tolerance window of 30 observations, enough to hold the central limit theorem.

$$\mu_{e_j}(t_{d_i}) = max\left(min\left(\frac{t_{d_i}-(t_{e_j}-k)}{k}, \frac{(t_{e_j}+k)-t_{d_i}}{k}\right), 0\right) \tag{13}$$

Figure 27b represents the evaluation of $\mu_{e_j}(t)$ for two detections, d_1 and d_2, produced by a particular detection method. In this context, $\mu_{e_j}(t_{d_i})$ gives the extent to which a detection d_i represents event e_j, or, in other words, its temporal closeness to a hard true positive (TP) regarding e_j. In that case, detection d_1 is closer to a TP, and d_2 lies outside the tolerance range given by k and could be considered a false positive.

A.1.2- Maintaining integrity with hard metrics

We are interested that the SoftED metrics still preserve concepts applicable to traditional (hard) metrics. In particular, the SoftED metrics are designed to express the same properties as their hard correspondents. Moreover, they are designed to maintain the reference to the perfect detection performance (score of 1 as in hard metrics) and indicate how close a detection method came to it. In order to achieve this goal, this approach defines constraints necessary for maintaining integrity concerning the standard hard metrics:

1. A given detection d_i must have only one associated score.
2. The total score associated with a given event e_j must not surpass 1.

The first constraint comes from the idea that the detection d_i should not be rewarded more than once. It avoids the possibility of the total score for d_i surpassing the perfect reference score of 1. Take, for example, the first scenario, presented in Figure 27c, in which we have one detection and many close events. The detection d_1 is evaluated for events e_1, e_2, and e_3 resulting in three different membership evaluation of $\mu_{e_1}(t_{d_1})$, $\mu_{e_2}(t_{d_1})$, and $\mu_{e_3}(t_{d_1})$, respectively. Nevertheless, to maintain integrity with hard metrics, a given detection d_1 must not have more than one score. Otherwise, d_1 would be rewarded three times, and its total score could surpass the score of a perfect match, which would be 1.

To address this issue, we devise a strategy for attributing each detection d_i to a particular event e_j. The adopted attribution strategy is based on the temporal distance between d_i and e_j. It facilitates interpretation and avoids the need for solving an optimization problem for each detection. In that case, we attribute d_i to the event e_j that maximizes the membership evaluation $\mu_{e_j}(t_{d_i})$. This attribution is given by E_{d_i} defined in Equation 14. According to Figure 27c, d_1 is attributed to event e_1 given the maximum membership evaluation of $\mu_{e_1}(t_{d_1})$. In case there is a tie for the maximum membership evaluation of two or more events, E_{d_i} represents the set of events to which d_i is attributed. As a consequence, we can also derive the set of detections attributed to each event e_j, D_{e_j}, defined by Equation 15. The addition of a detection d_i to the set D_{e_j} is further conditioned by the tolerance range, that is, a membership evaluation greater than 0 ($\mu_{e_j}(t_{d_i}) > 0$).

$$E_{d_i} = \arg\max_{e_j} \left(\mu_{e_j}(t_{d_i})\right) \tag{14}$$

$$D_{e_j} = \{d_i \mid E_{d_i} \supset e_j \wedge \mu_{e_j}(t_{d_i}) > 0\} \tag{15}$$

The second constraint defined by this approach comes from the idea that a particular detection method should not be rewarded more than once for detecting the same event e_j. It assures that the total score of detections for event e_j does not surpass the perfect reference score of 1. Take, for example, the second scenario, presented in Figure 27d, in which we have many detections attributed to the same event. The event e_1 is present in the sets E_{d_1}, E_{d_2}, E_{d_3} and E_{d_4}. Moreover, D_{e_1} contains d_1, d_2, and d_3. But in order to maintain integrity with hard metrics, the total score for D_{e_1} ($\mu_{e_1}(t_{d_1}) + \mu_{e_1}(t_{d_2}) + \mu_{e_1}(t_{d_3})$) must not surpass the score of a perfect match.

To address this issue we devise an analogous distance-based strategy for attributing a representative detection d_i to each event e_j. In that case, we attribute to event e_j the detection d_i, contained in D_{e_j}, that maximizes the membership evaluation $\mu_{e_j}(t_{d_i})$. This attribution is given by $\hat{d}_{e_j}$ defined in Equation 16. According to Figure 27d, e_1 is best represented by detection d_1 given the maximum membership evaluation of $\mu_{e_1}(t_{d_1})$. As a consequence, we can compute the associated score for each event e_j as $es(e_j)$, defined by Equation 17.

$$\hat{d}_{e_j} = \arg\max_{d_i} \left(\{\mu_{e_j}(t_{d_i}) \mid d_i \in D_{e_j}\}\right) \tag{16}$$

$$es(e_j) = \mu_{e_j}(t_{\hat{d}_{e_j}}) \tag{17}$$

Finally, each detection d_i produced by a particular detection method is scored by $ds(d_i)$ defined in Equation 18. Representative detections ($d_i = \hat{d}_{e_j}$) are scored based on $es(e_j)$. All other detections are scored 0. This definition ensures the total score for detections of a particular method does not surpass the number of real events m contained in the time series X. It holds true the Equation 19 and maintains the reference to the score of a perfect detection Recall according to usual hard metrics. Furthermore, it penalizes false positives and multiple detections for the same event e_j.

$$ds(d_i) = \begin{cases} es(e_j), & \text{if } \exists e_j \in E \mid d_i = \hat{d}_{e_j} \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

$$\sum_{i=1}^{n} ds(d_i) \leq m \tag{19}$$

The pseudocode for calculating $ds(d_i)$ for each detection d_i is given by Algorithm 2, which defines the function $soft_scores$. This function takes three arguments: $detections$, $events$, and k. The $detections$ argument is a logical vector of the same length as a time series X ($|X|$), indicating the detections by a given method, and $events$ is a logical vector of the same length indicating the true events in X. The k argument is an optional parameter (default is set to 15) that sets the temporal tolerance for a detection to be considered a match with a true event.

The function $soft_scores$ in Algorithm 2 first takes the vectors E and D containing the indices of all events and detections in the input vectors $events$ and $detections$, respectively. Then, for each i-th detection, it finds the indices of all matching events within a distance of k (line 7) and stores them in E_{match}. If no matching events exist, $ds(d_i)$ is 0. Otherwise, it calculates the membership scores between each matching event in E_{match} and the i-th detection (line 11). Finally, the soft score for each detection ($ds(d_i)$) is the maximum membership score among the matching events (line 12), indicating the best match between the current detection and the events. Lastly, a vector of soft scores for each detection (ds) is returned.

Algorithm 2 – Computing soft scores for each detection

Input: $detections =$ vector of detections by a given method;
$events =$ vector of true events in X;
$k =$ constant of temporal tolerance for a detection;
Output: $ds =$ vector of soft scores for each detection;

```
1  begin
2      E ← get_indices(events);
3      D ← get_indices(detections);
4      n ← length(D);
5      ds ← empty_vector(n);
6      foreach i ∈ 1 to n do
7          E_match ← matching_events(D[i], E, k);
8          if E_match is empty then
9              ds[i] ← 0;
10         else
11             scores ← membership_scores(D[i], E_match, k);
12             ds[i] ← max(scores);
13     return ds;
```

A.1.3- Computing the SoftED metrics

The scores computed for each detection d_i, $ds(d_i)$, are used to create soft versions of the hard metrics TP, FP, TN, and FN, as formalized in Table 9. In particular, while the value of TP gives the number of detections that perfectly matched an event (score of 1), the sum of $ds(d_i)$ scores indicate the degree to which the detections of a method approximate the m events contained in time series X given the temporal tolerance of k observations. Hence, the soft version of the TP metric, TP_s is given by $\sum_{i=1}^{n} ds(d_i)$. Conversely, the soft version of FN, FN_s, indicates the degree to which a detection method could not approximate the m events in an acceptable time range. The FN_s can then be defined as the difference between TP_s and the perfect Recall score ($m - \text{TP}_s$).

On the other hand, while the value of FP gives the number of detections that did not match an event (score of 0), its soft version, FP_s, indicates how far the detections of a method came to the events contained in time series X given the temporal tolerance of k observations. In that sense, FP_s is the complement of TP_s and can be defined by $\sum_{i=1}^{n} (1 - ds(d_i))$. Finally, the soft version of TN, TN_s, indicates the degree to which a detection method could avoid nonevent observations of X ($|X| - m$). The TN_s is given by the difference between FP_s and the perfect specificity score ($(|X| - m) - \text{FP}_s$).

Table 9 – Formalization of SoftED Metrics

$\text{TP}_s =$	$\sum_{i=1}^{n} ds(d_i)$	$\text{FN}_s =$	$m - \text{TP}_s$
$\text{FP}_s =$	$\sum_{i=1}^{n} (1 - ds(d_i))$	$\text{TN}_s =$	$(\|X\| - m) - \text{FP}_s$

Due to the imposed constraints described in Section A.1.2, the defined SoftED metrics TP_s, FP_s, TN_s, and FN_s, hold the same properties and scale as traditional hard metrics. Consequently, using their same characteristic formulas, they can derive soft versions of traditional scoring methods, such as Sensitivity, Specificity, Precision, Recall, and F1. Moreover, SoftED scoring methods still provide the same interpretation while including temporal tolerance for inaccuracy, which is pervasive in time series event detection applications. An implementation of SoftED metrics is publicly available at Github[1].

[1] SoftED implementation, datasets and experiment codes: `https://github.com/cefet-rj-dal/softed`