Anomaly Detection in Multi-Seasonal Time Series Data

Ashton Taylor Williams
Wright State University

Follow this and additional works at: https://corescholar.libraries.wright.edu/etd_all

Part of the Computer Engineering Commons, and the Computer Sciences Commons

Repository Citation
Williams, Ashton Taylor, "Anomaly Detection in Multi-Seasonal Time Series Data" (2023). Browse all Theses and Dissertations. 2811.
https://corescholar.libraries.wright.edu/etd_all/2811

This Thesis is brought to you for free and open access by the Theses and Dissertations at CORE Scholar. It has been accepted for inclusion in Browse all Theses and Dissertations by an authorized administrator of CORE Scholar. For more information, please contact library-corescholar@wright.edu.
ANOMALY DETECTION IN MULTI-SEASONAL TIME SERIES DATA

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science

By

ASHTON TAYLOR WILLIAMS
B.S.C.S., Wright State University, 2022

2023
Wright State University
WRIGHT STATE UNIVERSITY

COLLEGE OF GRADUATE PROGRAMS AND HONORS STUDIES

April 20, 2023

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Ashton Taylor Williams ENTITLED Anomaly Detection in Multi-Seasonal Time Series Data BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science.

_________________________________
Soon M. Chung, Ph.D.
Thesis Director

_________________________________
Thomas Wischgoll, Ph.D.
Chair, Department of Computer Science and Engineering

Committee on Final Examination:

_________________________________
Soon M. Chung, Ph.D.

_________________________________
Nikolaos Bourbakis, Ph.D.

_________________________________
Vincent A. Schmidt, Ph.D.

_________________________________
Shu Schiller, Ph.D.
Interim Dean, College of Graduate Programs & Honors Studies
ABSTRACT


Most of today’s time series data contain anomalies and multiple seasonalities [7], and accurate anomaly detection in these data is critical to almost any type of business. However, most mainstream forecasting models used for anomaly detection can only incorporate one or no seasonal component into their forecasts and cannot capture every known seasonal pattern in time series data. In this thesis, we propose a new multi-seasonal forecasting model for anomaly detection in time series data that extends the popular Seasonal Autoregressive Integrated Moving Average (SARIMA) model. Our model, named multi-SARIMA, utilizes a time series dataset’s multiple pre-determined seasonal trends to increase anomaly detection accuracy even more than the original SARIMA model. Our experimental results demonstrate the higher accuracy of multi-SARIMA when multiple seasonalities are present than most models with one or no seasonal component, although with more processing time.
# TABLE OF CONTENTS

CHAPTER 1 ............................................................................................................................................... 1

INTRODUCTION ........................................................................................................................................ 1

CHAPTER 2 ............................................................................................................................................... 4

BACKGROUND ......................................................................................................................................... 4

2.1 Anomaly Definition ................................................................................................................................. 4

2.2 Prior Work ............................................................................................................................................ 4

2.3 Desired Qualities .................................................................................................................................. 5

CHAPTER 3 ............................................................................................................................................... 7

EXISTING DETECTION METHODS ............................................................................................................ 7

3.1 Anomaly Labeling .................................................................................................................................. 7

3.2 Moving Average (MA) ............................................................................................................................ 8

3.3 Seasonal Integrated Moving Average (SIMA) ....................................................................................... 8

3.4 Seasonal Autoregressive Integrated Moving Average (SARIMA) ...................................................... 9

3.5 Hierarchical Temporal Memory (HTM) .............................................................................................. 11

3.6 TBATS .................................................................................................................................................. 13

CHAPTER 4 ............................................................................................................................................ 16

PROPOSED DETECTION METHODS ......................................................................................................... 16

4.1 Multi-SARIMA ...................................................................................................................................... 16

4.2 Two-Step Approach .............................................................................................................................. 18

CHAPTER 5 ............................................................................................................................................. 20

DATASETS .............................................................................................................................................. 20

5.1 Dataset Overview ................................................................................................................................. 20

5.2 MSTL Seasonal Decomposition ......................................................................................................... 24

5.3 Differencing ....................................................................................................................................... 28
CHAPTER 6

EXPERIMENTAL RESULTS

6.1 Implementation

6.2 Single-Step Experimental Results

6.3 Two-Step Experimental Results

CHAPTER 7

CONCLUSIONS AND FUTURE TOPICS

REFERENCES
LIST OF FIGURES

FIGURE 1: NYC TAXI DATASET ........................................................................................................... 21
FIGURE 2: SYNTHETIC DATASET 3 ...................................................................................................... 22
FIGURE 3: HOTGYM DATASET ............................................................................................................ 23
FIGURE 4: MSTL DECOMPOSITION OF THE NYC TAXI DATASET ................................................... 25
FIGURE 5: MSTL DECOMPOSITION OF THE SYNTHETIC DATASET ................................................ 26
FIGURE 6: MSTL DECOMPOSITION OF THE HOTGYM DATASET ..................................................... 27
FIGURE 7: NYC TAXI DATASET AFTER FIRST-ORDER SEASONAL DIFFERENCING ...................... 28
FIGURE 8: SYNTHETIC DATASET 3 AFTER FIRST-ORDER SEASONAL DIFFERENCING ............... 29
FIGURE 9: THE HOTGYM DATASET AFTER FIRST-ORDER SEASONAL DIFFERENCING ............... 29
LIST OF TABLES

TABLE 1: OVERVIEW OF THE DATASETS ........................................................................ 20
TABLE 2: SINGLE-STEP EXPERIMENTAL RESULTS...................................................... 31
TABLE 3: TWO-STEP EXPERIMENTAL RESULTS.......................................................... 32
ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Soon Chung, for his guidance throughout my graduate education. I would also like to thank my committee members, Drs. Nikolaos Bourbakis and Vincent Schmidt, for their time. Lastly, I would like to thank my older brother, Tristan, for always being a role model to me and sparking my interest in computer science. I would not be where I am today without them.
CHAPTER 1
INTRODUCTION

Nowadays there are many data sources, such as sensors, producing time series data, which is a sequence of data points indexed in time order. These data points typically consist of successive measurements made from the same source over a fixed time interval and are used to track change over time [16]. Anomalies (i.e., outliers) are data points that significantly deviate from their expected value [4], and early detection of anomalies is important to mitigate harmful effects, particularly in critical systems where failure can be catastrophic [3]. For example, a hospital can detect abnormal body signals of its patients and notify professionals before it's too late [2].

For anomaly detection in time series data, forecasting models are used to compare forecasted values to actual values to determine if a point is anomalous. While some deviation is expected when comparing a forecasted value to its real counterpart, if the predicted value deviates significantly from the actual value, then the data point is most likely an anomaly [2].

Some time series data contain a seasonality, which is a pattern that repeats at specific time intervals [1]. For example, CPU usage rate of a server may have a daily seasonal trend. The popular Seasonal Autoregressive Integrated Moving Average (SARIMA) forecasting model can represent a seasonal trend in its forecasting of time series data. However, SARIMA can implement only one seasonal trend in its forecasting. Allowing only one seasonal trend is unfortunate because some time series data contain more than one seasonality [7]. For example, New York City (NYC) taxi traffic data has both daily and weekly seasonal trends. Thus, utilizing all known seasonal effects in time series data can play an important role in data forecasting and anomaly detection [1].
In this thesis, we propose a new multi-seasonal model, named multi-SARIMA, for anomaly detection in time series data that extends the SARIMA model by allowing multiple seasonal components. The multi-SARIMA utilizes a dataset's multiple predetermined seasonal trends to increase anomaly detection accuracy. To compare with our multi-SARIMA, we also implemented other anomaly detection models, including Moving Average (MA), Seasonal Integrated Moving Average (SIMA), the original SARIMA, Numenta’s Hierarchical Temporal Memory (HTM) [9, 10], and another multi-seasonal model TBATS which stands for Trigonometric seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal components [8]. Additionally, we implemented the two-step approach proposed in [2] with our multi-SARIMA as the second step. The multi-SARIMA produced better anomaly detection results than the original SARIMA for every dataset we tested and, in most cases, outperformed HTM and TBATS.

This thesis is organized as follows: Chapter 2 contains our definition of anomalies, prior work with anomaly detection in time series data, and what traits detectors should have. Chapter 3 defines our anomaly labeling method and explains the existing anomaly detection models used in this thesis. Chapter 4 explains our proposed multi-SARIMA model and the two-step approach from [2] with our multi-SARIMA model as the second step. Chapter 5 describes the datasets used for testing and their properties, the Multiple Seasonal-Trend decomposition using Locally Estimated Scatterplot Smoothing (Loess) (MSTL) decomposition [7] we used to verify the seasonal trends in each dataset, and the differencing used on our datasets. Chapter 6 describes the implementation of different models, single-step and two-step test results, and comparisons based on the detection
accuracy and runtime. Lastly, chapter 7 contains our final thoughts with a conclusion and possible future research topics.
CHAPTER 2
BACKGROUND

2.1 Anomaly Definition

Anomalies are data points that significantly deviate from their expected value and are usually uncommon in real-world data [4]. Anomalies can contain useful information about the abnormal characteristics in a dataset [4]. The occurrence of an anomaly could be positive or negative [4]. For example, increased traffic on a website could indicate to a company that they had more clicks than usual, while the fluctuation in the turbine rotation frequency of a jet engine could indicate possible failure [10]. There are also different types of anomalies. A spatial anomaly is a data point that deviates from the expected value concerning the rest of the data and is usually easier to detect [10]. A temporal anomaly is a data instance that is only anomalous in a specific temporal context and is subtle and harder to detect [10].

2.2 Prior Work

Most prior work for anomaly detection involves data batching techniques that require all the data to be readily available [10]. These methods won’t work with real-time datasets where data is still being gathered or gathering never ends [10]. Previous work for real-time anomaly detection includes data forecasting algorithms where the forecasting models learn from past data, then predict values based on what they learned. From there, the predicted values are compared to the actual values and are given an anomaly score based on their difference to determine if they are anomalous. These algorithms use
statistical techniques like sliding thresholds, outlier tests, changepoint detection, statistical hypotheses testing, and exponential smoothing [10]. Most of today’s research for real-time anomaly detection is improving the accuracy of data forecasts.

Prior work involving the use of seasonality in time series forecasting is significant. The implementation of seasonality to the ARIMA model and exponential smoothing techniques are the most common approaches to adding seasonality to time series forecasting [1]. However, these basic models are only suited for single seasonality, and cannot account for multiple seasonality [1]. When it comes to multiple seasonality time series forecasting, many studies have been conducted to extend the original forecasting models to accommodate multiple seasonal patterns including double seasonal ARIMA and the exponential smoothing technique adapted from the simple Holt-Winters method [1]. We discovered the double seasonal ARIMA model after the development of our multi-SARIMA. The double seasonal ARIMA contains a similar approach to our multi-SARIMA model, however, our model differs from how it conducts its forecasting. New standalone models have also emerged for multi-seasonal time series forecasting including TBATS, MS, and Facebook’s Prophet [1, 8]. Currently, there is no research focused on the specific use of these multi-seasonal time series forecasting models for anomaly detection in time series data.

2.3 Desired Qualities

An ideal anomaly detector should be able to identify anomalies quickly and accurately. Accurately enough that false alarms are uncommon and real anomalies are detected almost every time. Quickly enough that anomalies are detected as early as possible
so that critical systems can negate possible catastrophic failure. However, anomaly
detectors need to find a perfect balance between early detection and accuracy. Too early
can lead to worse accuracy by nature and not early enough can improve accuracy but not
detect anomalies before it’s too late [10]. A balanced detector would be able to detect
anomalies early enough while maintaining a low number of false positives.
CHAPTER 3
EXISTING DETECTION METHODS

3.1 Anomaly Labeling

Time series anomaly detection models use a calculated numeric metric called an anomaly score to determine if a data point is an anomaly or not [4]. In our case, we determine the anomaly score using the error between the predicted value and the actual value. If the anomaly score exceeds the threshold, the data point is labeled as an anomaly [10].

In some cases, the threshold is fixed, however, a fixed threshold is not suitable in our case because the variance can change over time. The dynamic threshold must be calculated using sample metrics since we cannot assume the detector has access to the entire dataset as values are collected in real-time, such that only past values are available. For our dynamic threshold, we used the mean absolute deviation (MAD) calculated as:

$$MAD = \text{median}(|X_i - \text{median}(X)|)$$

where $X$ is the portion of data values in a rolling sample window to limit the impact of older data values [2]. Unlike the mean and standard deviation, MAD is robust when anomalies are present in the data sample, and there is no distortion unless at least a half of the sample is composed of anomalies [2, 5]. We then multiply the threshold with a constant to adjust the sensitivity of our anomaly detection, resulting in our anomaly detection metric defined as:

$$AnomalyDetected_t = |\epsilon_t| > |c \times MAD|$$

where $\epsilon_t$ is the anomaly score error metric and $c$ is the sensitivity constant [2].
3.2 Moving Average (MA)

Time series data is usually produced by a monitoring device and can range from spatial data in medical imaging to sequential data in network security [2, 12]. Let \( X = \{X_1, X_2, ..., X_t\} \) be a one-dimensional time series with evenly spaced discrete time where \( X_t \) is a value \( X \) at time \( t \) [2, 12]. Values that are before \( t \) are considered its lags such that \( X_{t-i} \) is the value \( i \) steps back in the time series [2, 12]. The backshift operator \( B \) yields the lags in a time series and is defined as:

\[
B^iX_t = X_{t-i}
\]

for all \( t > i \) [2, 12].

The moving average (MA) model is a simple and common approach to forecasting time series data. The moving average of order \( q \), denoted as MA\((q)\), predicts the value \( X \) at time \( t \) as:

\[
X_t = \mu + \varepsilon_t + \sum_{i=1}^{q} b_i B^i \varepsilon_t
\]

for all \( t > q \), where \( X_t \) is stationary, \( \varepsilon_t \) is the white noise error term, \( \mu \) is the mean of the series, and \( b = \{b_1, b_2, ..., b_q\} \) is the \( q \) parameters for the model [2, 12].

3.3 Seasonal Integrated Moving Average (SIMA)

The seasonal integrated moving average (SIMA) model is an extension of the MA model where one seasonal component is considered for the data forecast. SIMA denoted SIMA\((d, q)_m\), forecasts using MA\((q)\) with a seasonally differenced time series. Differencing is used to eliminate trends that are apparent in a dataset to make the data
stationary [2]. Differencing is done by replacing every value with the difference between itself and its first lag [2]. Let $\nabla X = \{X_1, X_2, ..., X_t\}$ be a first-order differenced time series such that:

$$\nabla X_t = X_t - X_{t-1} = (1 - B)X_t$$

for all $t > 1$ [2]. The order of differencing can be represented by a symbol $d$, such that $\nabla^d X_t$ denotes the $d$th-order differenced time series. For example, when $d = 2$ it is the second-order differenced time series [2]. Therefore, a differenced time series can be written more generally as:

$$\nabla^d X_t = (1 - B)^d X_t$$

for all $t > d$ [2]. However, no amount of differencing will remove a seasonal trend from data. Seasonal trends can be eliminated by seasonal differencing, which differences against the previous season instead of the first lag [2]. Let $\nabla^d m X$ be a seasonally differenced time series where $m$ is the period of the seasonal trend, then it is defined as:

$$\nabla^d m X_t = (1 - B^m)^d X_t$$

for all $t > d * m$ [2].

### 3.4 Seasonal Autoregressive Integrated Moving Average (SARIMA)

The seasonal autoregressive integrated moving average (SARIMA) denoted as $\text{SARIMA}(p, d, q)_m$ is an extension of the autoregressive integrated moving average (ARIMA) model denoted as $\text{ARIMA}(p, d, q)$ by incorporating a seasonal component into its forecasting model. Both models are based on a combination of the autoregressive (AR) model and moving-average (MA) model. The autoregressive model predicts $X$ using its
most recent lags [2]. Let AR(\(p\)) be an autoregressive model of order \(p\) that predicts the value of \(X\) at time \(t\) as:

\[
X_t = c + \varepsilon_t + \sum_{i=1}^{p} a_i B^i X_t
\]

for all \(t > p\) where \(X_t\) is stationary, \(a = \{a_1, a_2, ..., a_p\}\) is the \(p\) parameters for the model, \(c\) is a constant term, and \(\varepsilon_t\) is the unpredictable white noise [2, 12]. Let ARMA(\(p, q\)) be an autoregressive moving-average (ARMA) model, where \(p\) represents the order of AR and \(q\) represents the order of MA, defined as:

\[
X_t = c + \varepsilon_t + \sum_{i=1}^{p} a_i B^i X_t + \sum_{j=1}^{q} b_j B^j \varepsilon_t
\]

for all \(t > \max\{p, q\}\) where \(X_t\) is stationary [2, 12]. The ARIMA(\(p, d, q\)) model predicts \(X\) by modeling the differenced series \(\nabla^d X\) with an ARMA(\(p, q\)) model [2]. The SARIMA(\(p, d, q\)) model predicts \(X\) by modeling the seasonally differenced series \(\nabla_m^d X_t\) as an ARMA(\(p, q\)) model:

\[
X_t = c + \varepsilon_t + \left( \sum_{i=1}^{p} a_i B^i \nabla_m^d X_t \right) + \left( \sum_{j=1}^{q} b_j B^j \varepsilon_t \right)
\]

where \(X_t = \nabla_m^d X_t + \sum_{i=0}^{d-1} \nabla_m^i B^m X_t\) [2]. Although the SARIMA model is one of the best and most common time series forecasting models, it is unable to incorporate more than one seasonal trend into its forecasting.
3.5 Hierarchical Temporal Memory (HTM)

Hierarchical Temporal Memory (HTM) is a neural network-based machine learning algorithm derived from neuroscience that models spatial and temporal patterns in streaming data [9]. HTM works by simulating how the neocortex works in the human brain [13]. It is versatile and tolerable to noisy data and can detect even the most subtle anomalies, resulting in a low false positive rate with most real anomalies detected [3].

The learning of HTM can be broken down into three main parts: The first part is the encoder and the Sparse Distributed Representations (SDRs) [3]. SDRs help explain how brains can make semantic generalizations [13]. SDRs are represented by vectors that contain thousands of bits, and the encoder gives the bits meaning by encoding them to represent the properties of a representation [13]. The encoded properties of two SDRs are compared, and if they have 1-bit in the same location, then they share some similarities [13]. The more 1-bits the two SDRs share, the more semantically similar the two representations are [13].

The second part is the spatial pooler. The spatial pooler is responsible for learning spatial patterns present in the data. It starts by taking in a fixed number of encoded SDR bits then assigns a layer containing columns [13]. Each column has a set of potential synapses, a connection to the previous layer representing a subset of the input bits [13]. Connections between the layers are then determined based on the comparison between performance values and a performance threshold [13]. The active synapses of each column are then determined based on how many connected columns exist [13]. As more data are collected, the spatial pooler determines how many connected synapses of each column overlap with the input SDR bits, and activates columns with the most overlap [3]. Only
active columns update their connections, then the network boosts or hinders columns accordingly to prevent columns from being too dominant [2].

The third and final part of the HTM learning model is the temporal memory. Temporal memory does two things: learns the sequences of SDRs produced by the spatial pooler and makes predictions [13]. The temporal memory establishes connections between cells in the spatial pooler’s columns, then learns the connections between cells that reside in the same layer [13]. An active cell forms connections to other cells that were just active. This way, the cells can predict when they will likely become active by referring to their current connections [13].

HTM also calculates its own anomaly score by measuring the deviation between its predicted input and the actual input [9]. The anomaly score at time $t$ denoted as $s_t$, is given as:

$$s_t = 1 - \frac{\pi(X_{t-1}) \cdot a(X_t)}{|a(X_t)|}$$

where $a(X_t)$ is the sparse encoded value of the input at time $t$, $|a(X_t)|$ is the total number of 1 bits in $a(X_t)$, and $\pi(X_{t-1})$ is the internal prediction of $a(X_t)$ [10]. The anomaly score will be 0 if the current input is perfectly predicted or 1 otherwise [9]. To increase the anomaly detection accuracy, a short-term average of the prediction errors is computed, then a threshold is applied to the Gaussian tail probability to determine if a data point is truly an anomaly [10]. This second step in determining the anomaly score is the compliment of the tail probability and is defined as the anomaly likelihood:

$$L_t = 1 - Q\left(\frac{\mu'_{t} - \mu_{t}}{\sigma_t}\right)$$
where $\mu'_t = \frac{\sum_{i=W'-1}^{W'} s_{t-i}}{W'}$, $\mu_t$ is the mean of the sample of past anomaly scores, $\sigma_t$ is the standard deviation of the sample of past anomaly scores, $\mu'_t$ is the short-term average, $Q$ is the Gaussian tail probability function, and $W'$ is a window for a short term moving average [9, 10].

### 3.6 TBATS

The Trigonometric seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal components model, denoted as TBATS, is a forecasting model for complex time series that can include multiple seasonal periods, high-frequency seasonality, non-integer seasonality, and dual calendar effects [8]. TBATS is currently one of the best multi-seasonal time series forecasting models and is the most common [1]. It utilizes a framework that incorporates Box-Cox transformations, Fourier representations with time-varying coefficients, and ARMA error correction [8]. The TBATS model requires pre-specified seasonal periods that are then modeled by a trigonometric representation based on the Fourier series [1, 11].

TBATS is an extension of the Box-Cox transformation, ARMA errors, Trend, and Seasonal components (BATS) model, where the addition of trigonometric seasonality creates a more flexible parsimonious approach [8]. BATS, however, is an extension of exponential smoothing methods that combine its other components like Box-Cox transformations and ARMA errors to produce a better forecasting model [15]. The exponential smoothing in BATS utilizes the Holt-Winters method that handles time series with a trend and a single seasonality [8, 15]. The exponential smoothing works by having
future values be weighted averages of past values [15]. The Box-Cox transformation in the model stabilizes the variance and mean over time, making the time series stationary. ARMA errors in the model are applied to the residuals to capture any leftover information [15]. The trend captures long-term changes in the mean. Lastly, the seasonal component captures a time series’ periodical variation [15].

The BATS model was improved to forecast time series with multiple seasonal components with the addition of trigonometric seasonality as well as updated versions of some methods used in BATS to create the TBATS model [8, 15]. The trigonometric seasonality in TBATS represents each seasonal component in a time series as a trigonometric representation based on the Fourier series [8, 15]. This addition allows the model to fit multiple, larger, and non-integer seasonal components with less run-time than the original BATS model [15].

The BATS model can be represented as:

\[ y_t^{(\lambda)} = l_{t-1} + \phi b_{t-1} + \sum_{i=1}^{T} s_{t-m}^{(i)} + d_t \]

\[ l_t = l_{t-1} + \phi b_{t-1} + \alpha d_t \]

\[ b_t = \phi b_{t-1} + \beta d_t \]

\[ d_t = \sum_{i=1}^{p} \varphi_i d_{t-1} + \sum_{i=1}^{q} \theta_i e_{t-i} + e_t \]

where \( y_t^{(\lambda)} \) is the Box-Cox transformed time series at time \( t \), \( s_t^{(i)} \) is the \( i \)th seasonal component, \( l_t \) is the local level at time \( t \), \( b_t \) is the trend with damping at time \( t \), \( d_t \) is the ARIMA\((p, q)\) process, \( e_t \) is white noise, \( \varphi \) and \( \theta \) are the ARIMA\((p, q)\) coefficients, \( \phi \) is the trend damping, \( \alpha \) and \( \beta \) are the smoothing, \( T \) is the amount of seasonalities, \( \lambda \) is the Box-
Cox transformation, and $m_i$ is the length of the $i$th seasonal period [8, 11]. The BATS model is then extended by adding the trigonometric seasonal model and is represented as seasonal components based on the Fourier series:

$$s_{t}^{(i)} = \sum_{j=1}^{k_i} s_{j,t}^{(i)}$$

$$s_{j,t}^{(i)} = s_{j,t-1}^{(i)} \cos \omega_j^{(i)} + s_{j,t-1}^{*^{(i)}} \sin \omega_j^{(i)} + \gamma_1^{(i)} d_t$$

$$s_{j,t}^{*^{(i)}} = -s_{j,t-1}^{(i)} + \sin \omega_j^{(i)} + s_{j,t}^{*^{(i)}} \cos \omega_j^{(i)} + \gamma_2^{(i)} d_t$$

where $\gamma_1^{(i)}$ and $\gamma_2^{(i)}$ are the smoothing parameters, $k_i$ is the number of harmonics for the $i$th seasonal period, and $\omega_j^{(i)} = \omega_j^{(i)} 2\pi j / m_i$ [8, 11].
CHAPTER 4
PROPOSED DETECTION METHODS

4.1 Multi-SARIMA

Most forecasting models today include at most one seasonal component, and this
unnecessary restriction only hinders their potential. Allowing a model to take full advantage
of every known seasonal pattern in a dataset gives more options and possibilities for it to
perform better. Our proposed model, named multi-SARIMA, extends the original
SARIMA\((p, d, q)\) model and is denoted as SARIMA\((p_1, d_1, q_1)_{m_1} \times (p_2, d_2, q_2)_{m_2}\). It
predicts \(X\) by modeling the seasonal differenced series \(\nabla_{m_2}^{d_2} X\) with two SARIMA\((p, d, q)\) models:

\[
\nabla_{m_2}^{d_2} X_t = (\sum_{i=1}^{p_1} a_{1,i} B_{m_1}^{i}) \nabla_{m_2}^{d_2} X_t + (\sum_{i=1}^{p_2} a_{2,i} B_{m_2}^{i}) \nabla_{m_2}^{d_2} X_t - (\sum_{j=1}^{p_2} \sum_{i=1}^{p_1} a_{1,i} a_{2,j} B_{m_1}^{i+m_2} j) \nabla_{m_2}^{d_2} X_t + \epsilon_t + (\sum_{i=1}^{q_1} b_{1,i} B_{m_1}^{i}) \epsilon_t + (\sum_{i=1}^{q_2} b_{2,i} B_{m_2}^{i}) \epsilon_t + (\sum_{j=1}^{q_2} \sum_{i=1}^{q_1} b_{1,i} b_{2,j} B_{m_1}^{i+m_2} j) \epsilon_t
\]

where \(X_t = \nabla_{m_2}^{d_2} X_t + \sum_{i=0}^{d_2-1} B_{m_2}^{i} \nabla_{m_2}^{i} X_t\), \(m_1\) is the shorter seasonal period, \(m_2\) is the longer
seasonal period, \(d_2\) is the order of differencing, \(a_1\) is the \(p\) parameters for the shorter period,
\(b_1\) is the \(q\) parameters for the shorter period, \(q_1\) is the seasonal MA order of the shorter
period, \(p_1\) is the seasonal AR order for the shorter period, \(a_2\) is the \(p\) parameters for the
longer period, \(b_2\) is the \(q\) parameters for the longer period, \(q_2\) is the seasonal MA order of
the longer period, and \(p_2\) is the seasonal AR order for the longer period. The multi-SARIMA
equation was derived by extending the original SARIMA equation. The multi-SARIMA
equation contains seasonal AR and MA terms for individual season lengths \(m_1\) and \(m_2\),
followed by terms that account for the combination of the two seasonal trends and the
backshift operator being scaled by the season length. From there, we distribute the factors
and solve for the lone $X_t$ to get the final equation depicted above. We concluded that we
only need to difference using $d_2$ and $m_2$ since differencing over the longer seasonal trend
captures both seasonalities and makes the data stationary. We set $d_1 = 0$ since we difference
the data and obtain a stationary version using $d_2$, eliminating $d_1$ from appearing in the
multi-SARIMA equation.

In our approach, the first model is trained on three iterations of the shorter seasonal
trend, while the second model is trained on three iterations of the longer seasonal trend.
From the first model, we obtain the seasonal and non-seasonal autoregressive and moving
average parameters, the residuals, and the constant. From the second model, we obtain just
the seasonal autoregressive and moving average parameters. During the prediction, we
apply the values from both models to the multi-SARIMA equation to get the prediction $X$
at time $t$.

We expect the multi-SARIMA model to perform well, compared to other models,
when it is used with datasets that contain two meaningful seasonal trends. If a dataset
doesn’t have meaningful seasonal patterns, the multi-SARIMA model is not expected to
perform better. If a dataset contains two seasonal trends, but they are insignificant, then the
multi-SARIMA model is not guaranteed to perform better.

Better performance entails that the multi-SARIMA model has higher anomaly
detection accuracy, meaning a higher true positive rate with a lower false positive rate.
With this higher precision, however, we also expect the runtime of the multi-SARIMA to
be somewhat longer than those of other models. This is because the multi-SARIMA model
requires more fitting and learning than other models as it uses two different models and learns over two seasonal periods.

4.2 Two-Step Approach

The two-step approach for anomaly detection was initially proposed by Sperl and Chung in [2]. The algorithm consists of a simpler model that can label data fast with less accuracy and a more complex model that can label data accurately but requires more time [2]. In the two-step approach, the first step does the initial labeling with the faster but less accurate model, then the second step verifies the first step's labels with the slower but more accurate model [2]. The first model must pick up as many true positives as possible, then the second step denies most of its false positives and verifies its true positives [2]. So, this combined approach is limited to the true positive rate of the first model but reduces the false positive rate [2]. In the worst case, the first model finds every data point anomalous, causing the second step to verify every data point in the dataset [2]. The runtime of the two-step is at best slightly slower than the first model and at worst slightly faster than the second model.

Although the two-step approach is not new by itself, using our multi-SARIMA as the second step in the two-step approach is. Since we expect our multi-SARIMA to perform better than other models, when it is used with datasets that contain two meaningful seasonal trends, we also expect our multi-SARIMA to perform well when it is used as the second step in the two-step approach. For the two-step approach, better results entail maintaining the true positive rate of first step model while significantly reducing its false positive rate.
For our experiments, we used the MA and SIMA models as our first step to create the initial labeling, then verified the labels with SARIMA, our multi-SARIMA, and TBATS. We denote a combination of two models used in the two-step approach as ‘first step + second step’. For example, a two-step approach that uses MA as the first step and SARIMA as the second step is denoted as MA + SARIMA.
CHAPTER 5
DATASETS

5.1 Dataset Overview

We evaluated all models on three different datasets. Two datasets are from the Numenta Anomaly Benchmark (NAB), a collection of labeled, univariate, real-world time series data [6]. The third dataset is a synthetic time series dataset we created using our data generation tool. Since we are focused on multi-seasonal anomaly detection in time series data, the three datasets contain two meaningful seasonal trends, numerous hand-labeled anomalies, and enough data points to train and test models on. A general summary of each dataset is given in Table 1. We used a smaller version of the HotGym dataset as one anomaly occurred within the first three weeks of the data, causing training issues with some models.

\[\text{Table 1: Overview of the Datasets}\]

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Interval</th>
<th>Seasonality #1</th>
<th>Seasonality #2</th>
<th>Anomalies</th>
<th>Total Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>NYC Taxi</td>
<td>30 min</td>
<td>Daily</td>
<td>Weekly</td>
<td>5</td>
<td>10,320</td>
</tr>
<tr>
<td>Synthetic</td>
<td>1 hour</td>
<td>Daily</td>
<td>Weekly</td>
<td>5</td>
<td>8,664</td>
</tr>
<tr>
<td>Dataset 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HotGym</td>
<td>1 hour</td>
<td>Daily</td>
<td>Weekly</td>
<td>5</td>
<td>3,887</td>
</tr>
</tbody>
</table>
Figure 1: NYC Taxi dataset. Red lines depict anomalies.

The first NAB dataset, the NYC Taxi dataset, contains the number of NYC taxi passengers from 2014 to 2015. Shown in Table 1 the NYC Taxi dataset is our largest dataset with over 10,000 data points, has daily and weekly seasonality, contains a data point every 30 minutes, and has five hand-labeled anomalies that occur during the NYC marathon, Thanksgiving, Christmas, New Year’s Day, and a snowstorm. Figure 1 shows the NYC Taxi dataset with its anomalies labeled by the red lines.
As shown in Table 1, our Synthetic Dataset was generated to include a year of synthetic data with a data point every hour with a total of over 8,000 data points. It has a strong weekly and daily seasonal trend. The Synthetic Dataset has five manually placed anomalies with each one being harder to detect than the previous one. Figure 2 depicts the Synthetic Dataset with its anomalies labeled by the red lines.

Our data generation tool used to create the dataset depicted in Figure 2 is a python program we made to generate a seasonally heavy and controlled dataset to test the full potential of models that can utilize every seasonal trend in the dataset. The program creates a year of data with a data point every hour for 2020. Every data point has a starting value that is determined by the time of day. For example, morning has a starting value of 20 that consistently increases to a max of 40 around midday. Then, the starting values decrease to 20 again as they approach midnight. Also, weekends are handled differently by subtracting one from each starting value to give them smaller values than the weekdays. The tool
generates data like this to simulate a typical workday by having larger values during busier times and smaller values during less busy times. It also follows a typical workweek with weekends having slightly smaller values than weekdays. From there, we incorporate random noise by adding a random value from 1.0 to 1.5 to every data point. Once all of that is complete, anomalies are inserted manually to get a variety of detection difficulties. For example, harder to detect anomalies deviate slightly from the norm, like the first three anomalies in Figure 2. While easier to detect anomalies significantly deviate from the norm like the last two anomalies in Figure 2.

![HotGym.csv](image)

**Figure 3: HotGym dataset. Red lines depict anomalies.**

The third and final dataset is the HotGym dataset. This is another NAB dataset that includes data on the kilowatt energy consumption of a gym center in Australia [3]. As shown in Table 1, our shortened HotGym dataset contains about half a year of data from 2010 with almost 4,000 data points. It also has daily and weekly seasonality, has data points
every hour, and has five hand-labeled anomalies. Figure 3 shows the HotGym dataset with its anomalies labeled by red lines.

5.2 MSTL Seasonal Decomposition

Since our multi-SARIMA model utilizes two seasonal components, we should confirm that our test datasets contain two meaningful seasonal trends. For that purpose, we used Multiple Seasonal-Trend decomposition using Locally Estimated Scatterplot Smoothing (Loess) (MSTL) [7]. There are a few multi-seasonal time series decomposition methods available, including Facebook’s Prophet, TBATS, and Seasonal-Trend Decomposition using Regression (STR); however, we chose MSTL because it produces the lowest root mean squared error, is robust to outliers, has the smallest execution time, and is easy to use as it requires minimal parameters [7, 14].

MSTL decomposes an additive time series into a trend component, given seasonal components, and a residual component [7, 14]. MSTL is an extension of the Seasonal-Trend decomposition using Loess (STL) model as STL is only able to decompose time series with one seasonal component [7, 14]. Loess is a scatterplot smoothing technique that fits a curve to a scatterplot to determine the degree of the polynomial [14]. STL applies Loess to various transformations of the given time series and then extracts the trend and one seasonal component [14]. MSTL extracts each known seasonal component in a time series using STL one by one [7, 14]. MSTL first orders the given seasonal periods from shortest to longest to avoid shorter seasonal periods from being interlaced with the longer seasonal periods [7]. MSTL then applies STL iteratively on each identified seasonal period [7]. The MSTL additive decomposition of a time series can be defined as:
\[ X_t = S^1_t + S^2_t + \cdots + S^n_t + T_t + R_t \]

where \( S^1_t, S^2_t, \ldots, S^n_t \) denotes the seasonal components, \( T_t \) denotes the trend, and \( R_t \) denotes the remainder [7]. We used Python’s statsmodels MSTL package on a Linux virtual machine to perform the MSTL decomposition on our datasets as depicted in Figures 4-6.

Figure 4: MSTL decomposition of the NYC Taxi dataset for one week of data. The top graph depicts the dataset’s daily seasonal trend while the bottom graph depicts its weekly seasonal trend with the weekend highlighted in red.

Figure 4 depicts one week of MSTL decomposition on the NYC taxi dataset, where the vertical axis represents the smoothing for the seasonal component given. The seasonality in the data follows a typical workweek and makes sense, considering that the original data represents taxi passengers in New York City. The daily trend is very low early in the morning, then has spikes before midday since everyone is trying to get to work, a dip around noon since no one is out and about, the highest spikes in the afternoon when
everyone is heading home or traveling around the city, then ends with very a low dip late at night since everyone is at home. The weekly trend follows a typical workweek with the weekdays maintaining the same taxi usage pattern until the weekend showing a different pattern and higher spikes during later hours.

Figure 5: MSTL decomposition of the Synthetic Dataset for one week of data. The top graph depicts the dataset’s daily seasonal trend while the bottom graph depicts its weekly seasonal trend with the weekend highlighted in red.

Figure 5 depicts one week of MSTL decomposition on our Synthetic dataset. We generated our synthetic data to simulate a typical work schedule. The daily trend shows that the data values tend to be very low early in the morning, then has a spike before midday, a small dip around noon, another spike in the afternoon, and ends with very a low dip late
at night. The weekly trend shows that the trend is consistent throughout the weekdays, then shifting to having lower values during the weekend.

![MSTL decomposition of the HotGym dataset for one week of data.](image)

*Figure 6: MSTL decomposition of the HotGym dataset for one week of data. The top graph depicts the dataset’s daily seasonal trend while the bottom graph depicts its weekly seasonal trend with the weekend highlighted in red.*

Figure 6 depicts one week of MSTL decomposition on the HotGym dataset. The seasonality in the data follows a typical workweek and makes sense, considering that the original data represents a gym’s energy consumption in Australia. The daily trend shows that the data values tend to have a small spike at midnight and then are very low early in the morning until midday where there is the highest spike during the hottest and busiest time of the day, then decreases back down for the rest of the day as the sun goes down and people go home. The weekly trend shows that throughout the weekdays, the trend seems to be somewhat consistent as people tend to visit the gym regularly during the week, until
the weekend that has no high spikes but very low dips as not many people are going to the gym or it is closed during different hours.

5.3 Differencing

For the differencing of our test datasets in order to make them stationary, we decided to use seasonal differencing since they contain apparent seasonal trends, specifically daily and weekly seasonalities. So, we differenced using first-order seasonal differencing with a period of one week. This captures both the daily and weekly seasonal trends and produces stationary data. Figures 7-9 depicts our datasets after the first-order seasonal differencing was applied. The beginning of each graph has a flat line because the first week has no prior data to difference against [2].

![NYC Taxi](image)

**Figure 7: The NYC Taxi dataset after first-order seasonal differencing with a period of 1 week.**
Figure 8: Synthetic Dataset 3 after first-order seasonal differencing with a period of 1 week.

Figure 9: The HotGym dataset after first-order seasonal differencing with a period of 1 week.
6.1 Implementation

To properly compare our multi-SARIMA model, we used existing forecasting models MA, SIMA, SARIMA, TBATS, and HTM. MA, SIMA, SARIMA, TBATS, and our proposed multi-SARIMA were implemented in Python 3.8.5 on a Windows 10 computer with an Intel i7 8-core processor operating at 3.80 GHz, 16 GB of memory, and a 1 TB SSD. Numenta’s HTM algorithm was implemented on the same machine, using Python 2.7. The optimal parameters for each model were determined by a grid search. We used open-source python libraries provided by their authors for our implementations of HTM and TBATS. For MA, SIMA, and SARIMA we used Python’s statsmodels package. For the two-step approach, we used MA and SIMA as the first step, and SARIMA, TBATS, and our proposed multi-SARIMA as the second step.

All models were trained on the first three weeks of the data, then evaluated on the remaining data. We made sure there were no anomalies present in the training portion of each dataset. Since there are a very small number of anomalies within a large number of data points in each dataset, comparing performance based on accuracy percentage is ineffective as a model that never labels any data point as anomalous would achieve more than 90% accuracy [2]. Instead, we focus on which models produce the most true positives with the lowest number of false positives. This means, the best models would be able to label all anomalies correctly while not labeling other non-anomalous data points as anomalies. Table 2 shows the final single-step results of all models, and Table 3 shows the
final two-step results, where TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives.

6.2 Single-Step Experimental Results

Table 2: Single-Step Experimental Results.

<table>
<thead>
<tr>
<th>Detector</th>
<th>NYC_Taxi Dataset</th>
<th>Synthetic Dataset 3</th>
<th>HotGym Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TP</td>
<td>FP</td>
<td>FN</td>
</tr>
<tr>
<td>MA</td>
<td>2</td>
<td>654</td>
<td>3</td>
</tr>
<tr>
<td>SIMA</td>
<td>3</td>
<td>1587</td>
<td>2</td>
</tr>
<tr>
<td>SARIMA</td>
<td>2</td>
<td>1464</td>
<td>3</td>
</tr>
<tr>
<td>Multi-SARIMA</td>
<td>4</td>
<td>1425</td>
<td>1</td>
</tr>
<tr>
<td>TBATS</td>
<td>3</td>
<td>1391</td>
<td>2</td>
</tr>
<tr>
<td>HTM</td>
<td>4</td>
<td>178</td>
<td>1</td>
</tr>
</tbody>
</table>

For our single-step experimental results shown in Table 2, our multi-SARIMA model had the highest number of true positives for every dataset while maintaining fewer false positives than the SARIMA model for every dataset, although with longer runtime. Our multi-SARIMA had either the best or second-best results for every dataset.

The multi-SARIMA had the highest runtime compared to other models because the multi-SARIMA is the only model that combines the results from two models which train over the two seasonal periods of one day and one week, respectively. Since every other seasonal model but TBATS is limited to one seasonal trend, they are trained over the period of one day as that is their stronger seasonality. Training two models and having one training over a week required the extra time but produced better results. Specifically, the runtime of the multi-SARIMA on the NYC Taxi dataset was unexpectedly long. This is because the NYC Taxi dataset is the only dataset with a data point every 30 minutes instead of every hour, causing the 3-week training data to contain a large amount of data for the models to
train on. The other multi-seasonal model, TBATS, was also slow and had the second longest runtime for every dataset. TBATS may be a more refined model, but it still requires more time since that is the nature of learning multiple seasonal patterns.

Our multi-SARIMA was the only model that achieved the same number of true positives as HTM for the NYC Taxi dataset and outperformed every model for the HotGym dataset. The multi-SARIMA doubled the true positive rate of HTM and TBATS for the HotGym dataset while still maintaining the second lowest false positive rate among all models.

Expectedly, the two multi-seasonal models performed the best for the Synthetic dataset. Most models detected all five anomalies, but TBATS and multi-SARIMA did so with under ten false positives. HTM performed very poorly with this dataset, and we think that is because the dataset was created using randomness, throwing off the learning of HTM.

Notably, TBATS had either the same or higher true positive rate than the original SARIMA for every dataset, while maintaining a lower false positive rate.

### 6.3 Two-Step Experimental Results

*Table 3: Two-Step Experimental Results.*

<table>
<thead>
<tr>
<th>Detector</th>
<th>NYC_Taxi Dataset</th>
<th></th>
<th></th>
<th></th>
<th>Synthetic Dataset 3</th>
<th></th>
<th></th>
<th></th>
<th>HotGym Dataset</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TP</td>
<td>FP</td>
<td>FN</td>
<td>Runtime (sec)</td>
<td>TP</td>
<td>FP</td>
<td>FN</td>
<td>Runtime (sec)</td>
<td>TP</td>
<td>FP</td>
<td>FN</td>
</tr>
<tr>
<td>MA + SARIMA</td>
<td>2</td>
<td>131</td>
<td>3</td>
<td>2.753</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3.23</td>
<td>3</td>
<td>120</td>
<td>2</td>
</tr>
<tr>
<td>SIMA + SARIMA</td>
<td>3</td>
<td>1072</td>
<td>2</td>
<td>3.627</td>
<td>5</td>
<td>91</td>
<td>0</td>
<td>3.207</td>
<td>2</td>
<td>547</td>
<td>3</td>
</tr>
<tr>
<td>MA + Multi-SARIMA</td>
<td>2</td>
<td>93</td>
<td>3</td>
<td>787.783</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>69.95</td>
<td>3</td>
<td>53</td>
<td>2</td>
</tr>
<tr>
<td>SIMA + Multi-SARIMA</td>
<td>3</td>
<td>475</td>
<td>2</td>
<td>697.393</td>
<td>5</td>
<td>50</td>
<td>0</td>
<td>70.612</td>
<td>2</td>
<td>220</td>
<td>3</td>
</tr>
<tr>
<td>MA + TBATS</td>
<td>2</td>
<td>122</td>
<td>3</td>
<td>76.867</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>45.359</td>
<td>3</td>
<td>68</td>
<td>2</td>
</tr>
<tr>
<td>SIMA + TBATS</td>
<td>3</td>
<td>1156</td>
<td>2</td>
<td>79.023</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>47.08</td>
<td>2</td>
<td>306</td>
<td>3</td>
</tr>
</tbody>
</table>
For our two-step experimental results shown in Table 3, all two-step algorithms, each of which uses a combination of two models, have less false positives than their standalone first step results shown in Table 2, except for MA + SARIMA for the Synthetic dataset which produced the same results as MA. This is because MA’s false positives were already very low for that dataset. Also, most two-step algorithms have significantly less false positives than their standalone second step results shown in Table 2, but have less true positives because they are limited to the true positive rate of the first step.

Although with more processing time, the multi-SARIMA as the second step produced significantly less false positives than the original SARIMA as the second step for every dataset. The only case that produced less false positives than the multi-SARIMA is TBATS for the Synthetic dataset, which was expected as TBATS did better on that dataset. Also, the two-step approach using multi-SARIMA as the second step improved the runtime, compared to the standalone multi-SARIMA, as it worked on less data points. Notably, TBATS did better as the second step than the original SARIMA for every dataset, but worse than multi-SARIMA for two of the three datasets.

All two-step algorithms could not detect the four true positives that MA originally detected for the HotGym dataset. We believe this is because other models could not detect the fourth anomaly detected by MA, causing them to label it as non-anomalous when they were used in the second step.
CHAPTER 7
CONCLUSIONS AND FUTURE TOPICS

When data contains repeated patterns such as seasonality, they can be learned and applied to a forecasting model to improve the accuracy of the model. Today, time series data containing multiple seasonalities are common in real-world applications [7]. However, most existing models for anomaly detection in time series data can include just one or no seasonal component, so they cannot capture every seasonal trend that appears in datasets.

Our multi-SARIMA model takes the original SARIMA model one step forward by including multiple seasonal components instead of just one. The multi-SARIMA produced better anomaly detection results than the original SARIMA for every dataset we tested and, in most cases, outperformed well-known HTM and TBATS. Also, we proved that our multi-SARIMA produces better results than SARIMA when used as the second step in the two-step approach we proposed in [2].

In addition to our multi-SARIMA model, we showed the anomaly detecting capability of an existing multi-seasonal forecasting model TBATS, which also outperformed SARIMA and HTM.

Different time series datasets have different characteristics, such that no one model could be the best for every case. However, our multi-SARIMA model showed very accurate detection performance on various datasets we used for evaluation and better overall results than other models.

In the future, some improvements could be incorporated to the multi-SARIMA model, including the runtime reduction, the ability to capture more than two seasonal trends, and a better way to choose optimal parameters.
REFERENCES


