**The republıc of turkey**

**BAHCESEHIR UNIVERSITY**

**A HYBRID METHOD FOR tıme serıes anomaly detection**

**Master Thesis**

**samet yazak**

**ISTANBUL, 2018**

**The rebuplıc of turkey**

**BAHCESEHIR unıversıty**

**graduate school of natural and applıed scıences**

**computer engıneerıng**

**A HYBRID METHOD FOR TIME SERIES anomaly detectıon**

**Master Thesis**

**samet yazak**

**Supervisor: AsSt. prof. cemal okan şakar**

**İSTANBUL, 2018**

**The republıc of turkey**

**BAHCESEHIR UNIVERSITY**

**GRADUATE SCHOOL of natural and applıed scıences**

**computer engıneerıng**

Name of the thesis: A Hybrid Method for Time Series Anomaly Detection

Name/Last Name of the Student: Samet Yazak

Date of the Defense of Thesis:

The thesis has been approved by the Graduate School of Natural and Applied Sciences.

Asst. Prof. Yücel Batu SALMAN

Graduate School Director

Signature

I certify that this thesis meets all the requirements as a thesis for the degree of Master of Arts.

Asst. Prof. Tarkan AYDIN

Program Coordinator

Signature

This is to certify that we have read this thesis and we find it fully adequate in scope, quality and content, as a thesis for the degree of Master of Arts.

Examining Committee Members Signature\_\_\_\_

Thesis Supervisor -----------------------------------

Asst. Prof. Cemal Okan ŞAKAR

Member -----------------------------------

Title Name and Surname

Member -----------------------------------

Title Name and Surname

**ABSTRACT**

A HYBRID METHOD FOR TIME Serıes anomaly detectıon

Samet Yazak

Computer Engineering

Thesis Supervisor: Asst. Prof. Cemal Okan Şakar

June 2018, 33 pages

This study deals with anomaly detection for time series data using unsupervised methods. There are many robust techniques that only work well on specific domains. It is challenging to find a complete method for detecting various anomaly types in different datasets. For this reason, this study aims to find such a method (or ensemble of methods).

**Keywords**: Anomaly Detection, Time Series Data

**ÖZET**

ZAMAN SERİSİ ANOMALİ TESPİTİ İÇİN HİBRİt BİT YÖNTEM

Samet Yazak

Bilgisayar Mühendisliği

Tez Danışmanı: Yrd. Doç. Cemal Okan Şakar

Haziran 2018, 33 sayfa

Bu çalışma, denetimsiz yöntemler kullanarak, zaman serisi verisinde anomali tespiti üzerinedir. Bu alanda, çeşitli problem kümelerine özgü etkili yöntemler bulunmaktadır. Ancak, farklı çalışma alanlarında, farklı anomali tiplerini tespit etmekte yeterli değiller. Bu sebeple, bu çalışma kendi alanlarında etkili yöntemleri birleştirerek, tüm anomali tiplerinde tespit etmede başarılı bir yöntem bulmayı amaçlamaktadır.

**Anahtar Kelimeler**: Anomali Tespiti, Zaman Serisi Verisi

**CONTENTS**

[**TABLES vii**](#_Toc514967793)

[**FIGURES viii**](#_Toc514967794)

[**ABBREVIATIONS ix**](#_Toc514967795)

[**1. INTRODUCTION 1**](#_Toc514967796)

[**1.1 TIME SERIES PROPERTIES 2**](#_Toc514967797)

[**1.1.1 Univariate vs Multivariate 2**](#_Toc514967798)

[**1.1.2 Time Series Characteristics 2**](#_Toc514967799)

[**1.2 ANOMALY TYPES 3**](#_Toc514967800)

[**1.2.2 Point Anomaly 3**](#_Toc514967801)

[**1.2.2 Contextual Anomaly 4**](#_Toc514967805)

[**1.2.3 Collective Anomaly 5**](#_Toc514967806)

[**1.3 ANOMALY DETECTION APPLICATIONS 6**](#_Toc514967807)

[**1.3.1 Intrusion Detection 6**](#_Toc514967808)

[**1.3.2 Fraud Detection 6**](#_Toc514967809)

[**1.3.3 Medical Diagnosis 6**](#_Toc514967810)

[**1.3.4 Flight Safety 6**](#_Toc514967811)

[**2. RELATED WORKS 7**](#_Toc514967812)

[**3. MATERIALS AND METHODS 9**](#_Toc514967813)

[**3.1 DATASETS 9**](#_Toc514967814)

[**3.1.1 Data Preparation 9**](#_Toc514967815)

[**3.2 METHODS 9**](#_Toc514967816)

[**3.2.1 Moving Average 10**](#_Toc514967817)

[**3.2.2 Auto-Regressive Integrated Moving Average (ARIMA) 10**](#_Toc514967818)

[**3.2.3 Kalman Filter 14**](#_Toc514967819)

[**3.2.4 Time Series Decomposition 17**](#_Toc514967820)

[**4. RESULTS 20**](#_Toc514967821)

[**5. DISCUSSION 25**](#_Toc514967822)

[**5.1 POINT ANOMALY 25**](#_Toc514967823)

[**5.2 CONTEXTUAL ANOMALY 26**](#_Toc514967824)

[**5.3 COLLECTIVE ANOMALY 26**](#_Toc514967825)

[**5.4 COMPARISON WITH LIBRARIES 27**](#_Toc514967826)

[**6. CONCLUSION 32**](#_Toc514967827)

[**REFERENCES 34**](#_Toc514967828)

# **TABLES**

Table 4.1: Moving Average Results 20

Table 4.2: ARIMA Results 21

Table 4.3: Kalman Filter Results 22

Table 4.4: Time Series Decomposition Results 23

Table 4.5: Algorithm Comparison 23

Table 4.6: Results for anomaly types 24

Table 5.1: Twitter library results 27

# **FIGURES**

Figure 1.1: Seasonal time series 2

Figure 1.2: Trend in time series 3

Figure 1.3: Point Anomaly 4

Figure 1.4: Contextual Anomaly 5

Figure 1.5: Collective Anomaly 6

Figure 3.1: Original Data ADF Test 12

Figure 3.2: Differenced Data ADF Test 12

Figure 3.3: ACF Plot 13

Figure 3.4: PACF Plot 13

Figure 3.5: Reference values for uncertainty parameters 15

Figure 3.6: Lower value for covariance uncertainty 15

Figure 3.7: Higher value for environment uncertainty 16

Figure 3.8: Lower value for environment uncertainty 16

Figure 3.9: STL Output 18

Figure 3.10: Decompose Output 18

Figure 3.11: Moving Average vs Moving Median 19

Figure 5.1: True anomaly set in sample series 28

Figure 5.2: Moving Average sample result 28

Figure 5.3: ARIMA sample result 29

Figure 5.4: Kalman Filter sample result 30

Figure 5.5: Time Series Decomposition sample result 30

Figure 5.6: Twitter library sample result 31

# **ABBREVIATIONS**

ARIMA : Auto-Regressive Integrated Moving Average

CI : Confidence Interval

DFT : Discrete Fourier Transform

DTFT : Discrete-Time Fourier Transform

ESD : Extreme Studentized Deviate

LSTM : Long short term memory

PCA : Principal Component Analysis

RPCA : Robust Principal Component Analysis

Sd : Standard Deviation

# **INTRODUCTION**

Anomaly is part of a data or event that does not conform to the excepted behavior. Ex; a web site that is normally receives 0 to 10 requests per hour, suddenly gets 1000 requests. This kind of behavior can also be called outlier, discord, exception. For the rest of this experiment, it will be called anomaly for consistency.

Anomaly detection is finding unexpected behavior of data or events (anomalies). It is studied in many years in diverse domains such as security, finance, health care, fraud etc. Comprehensive surveys of anomaly detection techniques are provided by Hodge and Austin (2004), Agmeyang et al (2006) and Chandola et al (2008). Also, many problem specific studies exist such as Aleskorov (1997) on credit card fraud, Axelsson (2000) on intrusion detection, Spence et al (2001) on MRI image anomaly detection, Markov and Singh (2003a, 2003b) on novelty detection, Fujumaki et al (2005) on air craft signal processing for anomaly detection.

Time series anomaly detection is different from traditional techniques because of nature and behavior of sequence data. Techniques for time series data focus on previously seen observations while traditional techniques focus on general behavior. These techniques can be classified into three categories as Kernel Based, Window Based and Markovian (Chandola et al 2009). Chandola et al (2009) provides a survey on time series anomaly detection with variety of techniques in each group.

Motivation of this experiment is to observe the strengths and drawbacks of present time series anomaly detection techniques on 3 different anomaly types and assemble different methods in one technique that will work well on all types on real time. Supervised techniques that based on sliding window algorithm which will be detailed in Section 3.2 will be investigated. Results of used techniques will be provided with recall, f1 score and performance measures.

Rest of this thesis is organized as follows. Current literature will be provided briefly in Section 2. Datasets, pre-processing tasks and techniques will be described in Section 3. Section 4 consists results of experiments with F-1 score, recall. Section 5 discusses strengths and drawbacks of techniques, which technique is best for what type of anomaly and what can be done beyond this experiment to get better results. Also, it contains comparison of used techniques with open source time series anomaly detection library of Twitter. Section 6 includes closing statements.

## **TIME SERIES PROPERTIES**

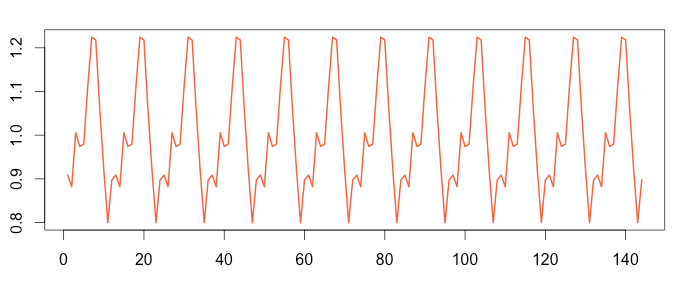
### **Univariate vs Multivariate**

Univariate data consists of single observation that is collected over equal time intervals while multivariate means data consists of many attributes. In this study, univariate time series were used.

### **Time Series Characteristics**

**Seasonality:** Fixed length periodic fluctuations in data. See Figure 1.1.

**Figure 1.1: Seasonal time series**



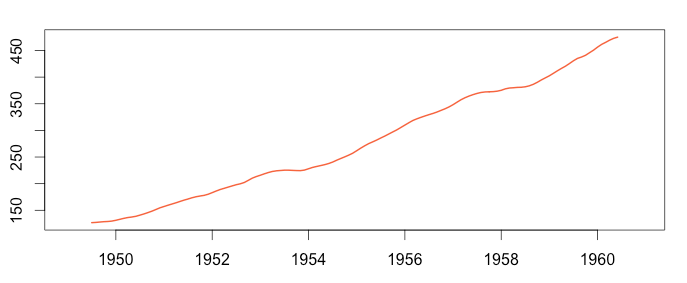
**Cyclic:** Periodic fluctuations with variable length.

**Trend:** Long-term increase/decrease in data.

**Stationary:** Mean and variance does not change over time.

**Residual:** Remainder of time series when season and trend components are removed.

**Figure 1.2: Trend in time series**



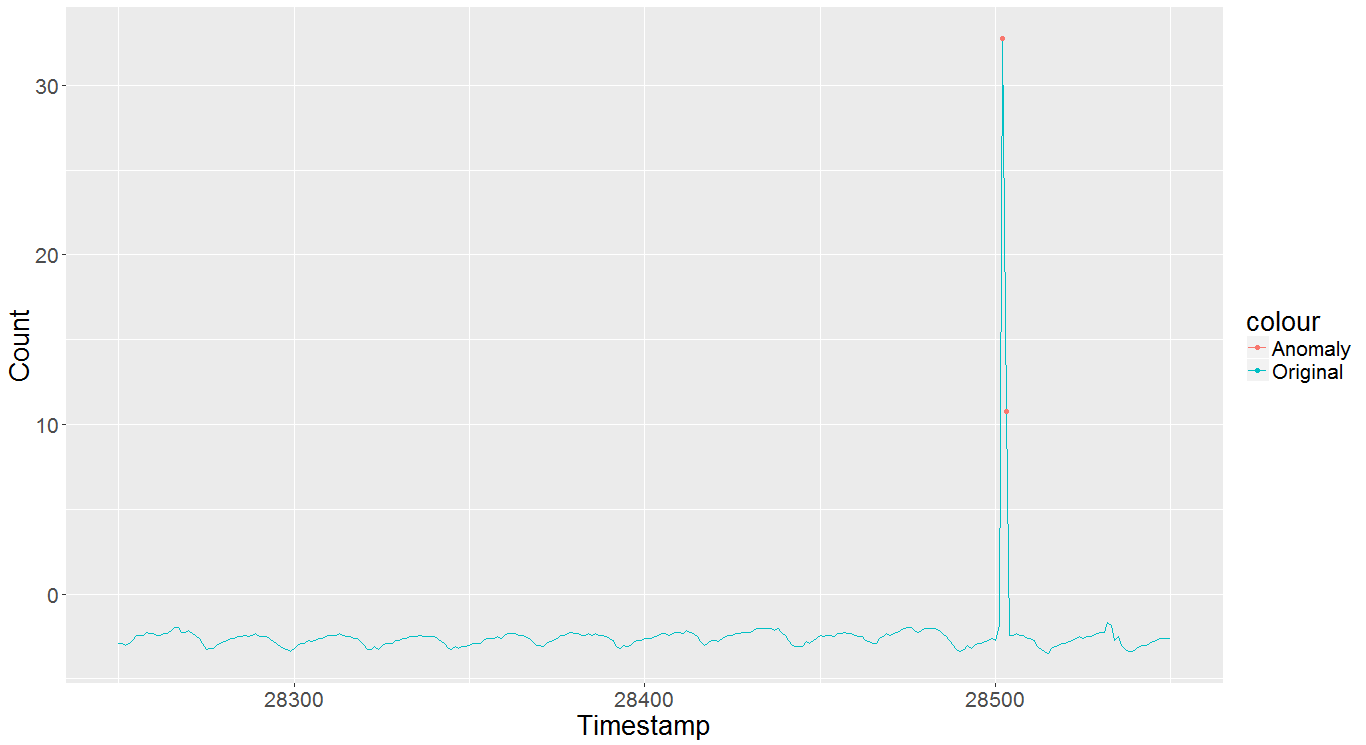
## **ANOMALY TYPES**

It is important to know what kind of anomaly/anomalies that problem includes before determining or proposing a method to sort it out. Because, each type requires different interests and algorithms to detect anomalies. In the literature, anomalies are categorized into three groups (Chandola et al 2009).

### **Point Anomaly**

Individual data instance that can be identified as anomalous is named point anomaly. Figure 1.1 shows an example of this type. If data is considered as web site traffic per hour, red points does not conform with the nature of previous data instances, so they can be identified as anomalous.

**Figure 1.3: Point Anomaly**

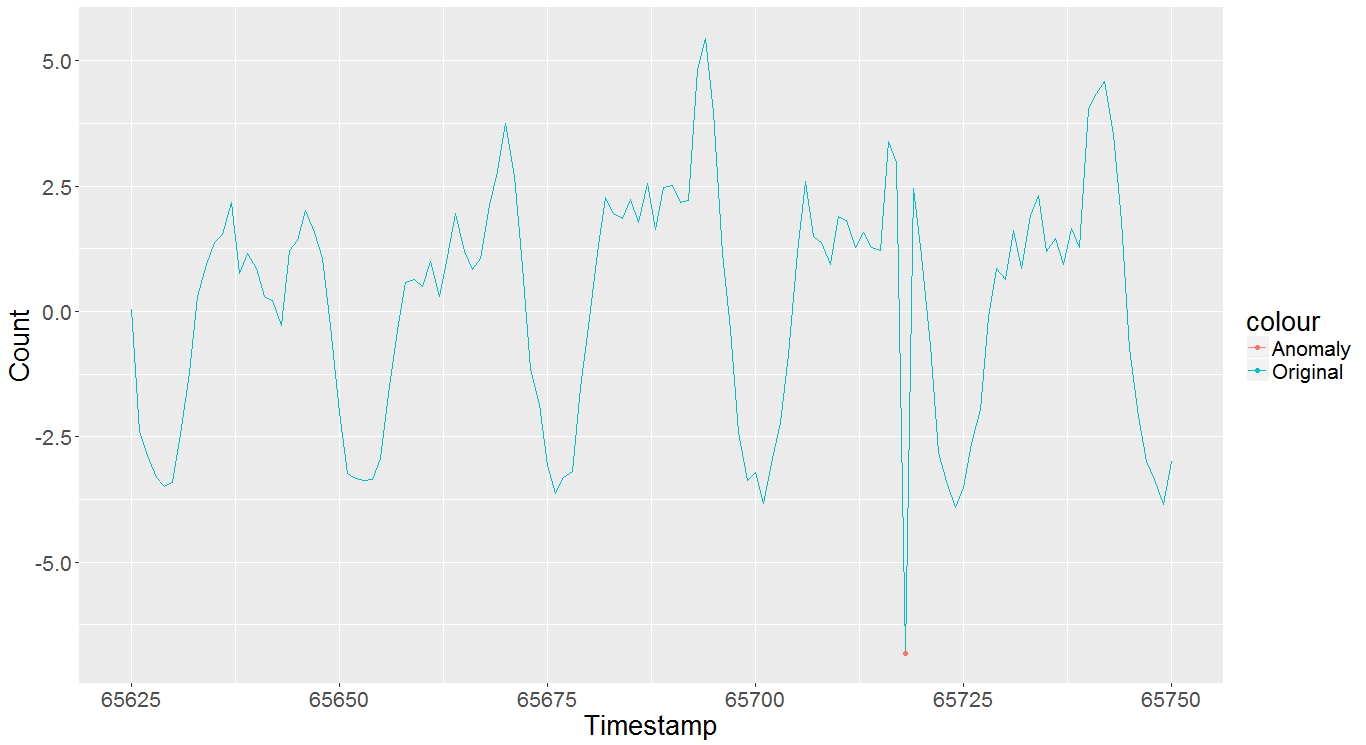




### **Contextual Anomaly**

If data instance is anomalous only in a certain context but not otherwise, then it is a contextual anomaly. Anomaly in Figure 1.2 is a contextual anomaly for breaking periodicity in current context. As in the previous example, if we consider data as web traffic per hour in a web site or purchase of certain product online, this type of anomaly detection can point to a short time of a system failure.

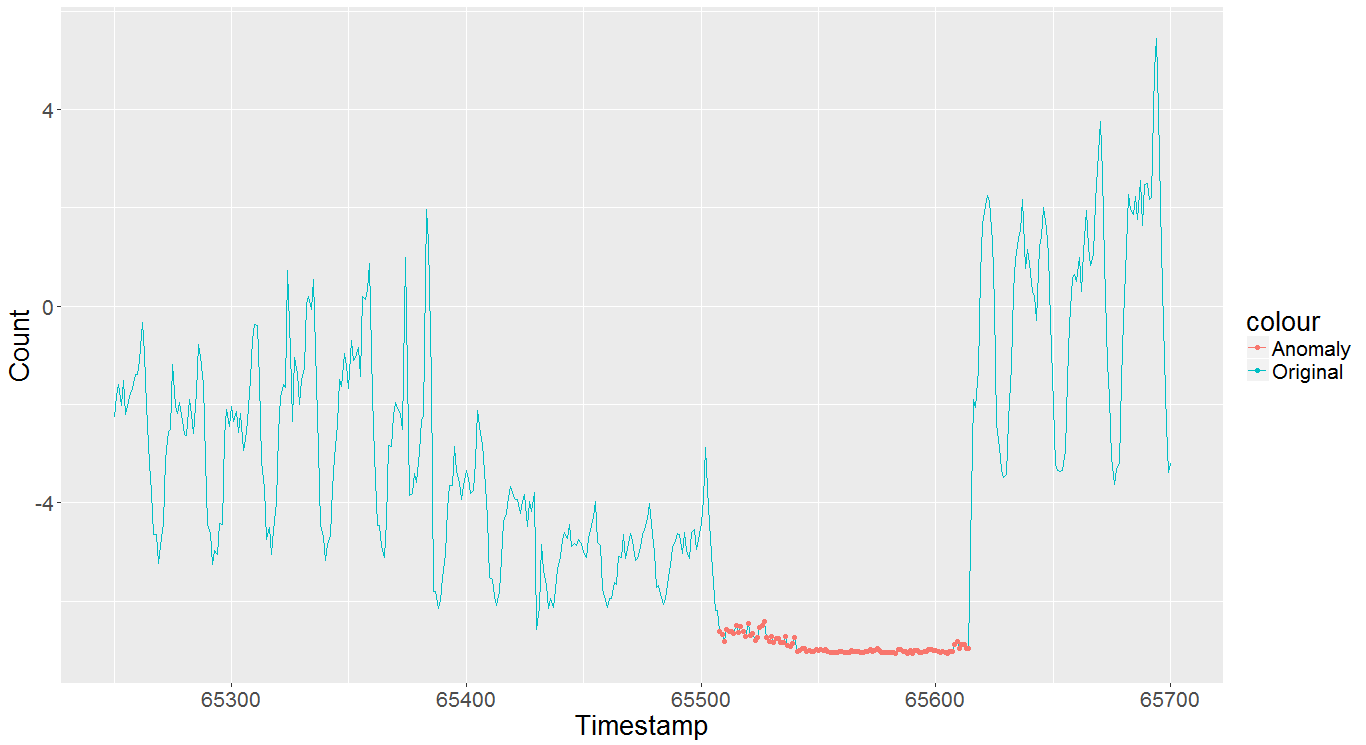
**Figure 1.4: Contextual Anomaly**



### **Collective Anomaly**

If a group of data instances does not conform with respect to the nature of complete data, but they cannot be declared as anomalous individually, then this group of instances are named as collective anomaly. Figure 1.3 shows an example of a collective anomaly. For a considerable amount of time, web site traffic is measured at lowest level (although few data instances can have low values) which can point a serious system failure. Techniques used for detecting collective anomalies are different from point and contextual anomalies and requires detailed examination.

**Figure 1.5: Collective Anomaly**



## **ANOMALY DETECTION APPLICATIONS**

### **Intrusion Detection**

Intrusion detection is a software tool to detect malicious activity or unauthorized access to a computer or network. Since anomaly means different pattern from normal behavior of data, intrusion detection software is good for anomaly detection techniques and they are widely studied (Denning 1987, Axelsson 2000).

### **Fraud Detection**

Fraud detection is a billion-dollar industry that aims to protect customers from unauthorized transactions. There are many applicable domains for fraud like credit cards, banks, stock markets. It generally requires detecting malicious activity in real time to prevent financial loss.

### **Medical Diagnosis**

Medical diagnosis is a very crucial domain and this type of anomaly model needs high accuracy. It typically uses patient history or recent measurements to find irregularities. Electrocardiogram (ECG) records are example of a medical time series data.

### **Flight Safety**

To monitor flight status, sequences of data are collected from aircrafts with various sensors. These sequential data are used to detect faults or unexpected behavior of signals.

# **RELATED WORKS**

Anomaly detection on time series data is a widely studied area on various domains. Especially in recent years, increasing amount of sequential data and evolving computer and internet technologies create interest on the field to monitor abnormal or malicious behavior.

Fox (1972) is one of the early researchers to detect anomalies using maximum likelihood ratio tests. Aleskerov et al (1997) on credit card fraud detection, Axelsson (2000) on network intrusion detection systems, Spence et al (2001) on MRI image analysis, Markov and Singh (2003a, 2003b) on novelty detection, Fujumaki et al (2005) on air craft signal processing, Keogh et al (2006) on unusual patterns in time series database and many others have key contributions on time series anomaly detection.

Hodge and Austin (2004) groups number of existing algorithms in three categories as statistical, machine learning and hybrid methods and provides comparative results of each algorithm. Agmeyang et al (2006), similarly, studies existing algorithms in general as numerical, clustering and symbolic techniques. Chandola et al (2009) provides a more comprehensive survey that studies algorithms in three groups; window, markovian and kernel. They also compare algorithms with various datasets and discusses advantages and disadvantages of each technique with specific dataset. Ahmed et al (2016) provides a survey of network anomaly detection with classification, clustering and information theory algorithms.

Statistical methods are easy to implement and used widely for detecting anomaly in sequential data. One of the popular statistical detection technique Extreme Studentized Deviate (ESD) (Grubbs 1969) was used by Walfish (2006) and Hodge and Austin (2004). Also, Paul and Fung (1991) and Vallis et al (2014) showed that generalized ESD can be used for find anomalous data. Decomposing time series data is not directly a detection algorithm but it helps detection process by removing seasonal and trend part of data which is essential in anomaly detection. Vallis et at (2014) uses decomposition to form more appropriate data for ESD.

Predictive method ARIMA (Bianco et al 2001) was used by Moayedi and Shirazi (2008) and Yaacob and Chien (2010) for network anomaly detection. It also has an R package created by Hyndman and Khandakar (2008). Kalman Filter (Kalman 1960) was used in anomaly detection with combining statistical methods by Soule et al (2005) and it was used by Knorn and Leith (2008) for detecting fault or failures in software appliances.

Long short term memory (LSTM) networks are being used extensively in recent years for time series anomaly detection: Malhotra et al (2015), Marchi et al (2015). Bontemps et al (2016) combines LSTM and recurrent neural networks for collective anomalies.

Big internet companies create and publish their own anomaly detection libraries. Yahoo provides both a benchmark dataset and a java library (EGADS) to detect anomalies in large scale in real time and contains a number of anomaly detection techniques. Twitter has an open source R package that uses Seasonal ESD. It employs piecewise median algorithm to calculate trend in long time series. Netflix created an algorithm that uses Robust Principal Component Analysis (RPCA) that is proposed by Candes et al (2009). Numenta provides an anomaly detection scoring mechanism that evaluates algorithms for anomaly detection in streaming and real-time applications.

On Yahoo! Webscope dataset, Nayyar et al (2015) forms an anomaly detection process that results 0.65 f1 score using sliding window algorithm. Till et al (2017) presents comparative study implementing Simple Online Regression Anomaly Detector (SORAD), Offline-RAD and feature generation using sliding windows and comparing these algorithms with existing libraries like Hierarchical Temporal Memory (HTM) of Numenta and Twitter's ADVec algorithm. They get best results using SORAD with 0.67 f1 score. Hyndman et al (2015) use Yahoo! dataset to compare entire series with each other using Principal Component Analysis (PCA) to detect anomalies. According to their result, they have 80 per cent success rate. Suh et al (2016) presents a new technique that combines variational autoencoder and echo-state network and compares it with generalized ESD.

# **MATERIALS AND METHODS**

## **3.1 DATASETS**

In this experiment, a benchmark dataset that is created for Time Series Anomaly Detection by Yahoo is used. Dataset is combination of four parts A1, A2, A3, A4 with each has different characteristics. Two of those sets are used in experiments; A1 and A2. A1 consists real world web request statistics of some Yahoo properties. A2 is a synthetic data which anomalies, trend and seasonality are placed randomly. Both has three attributes:

1. Timestamp: hourly representation of time when data is measured
2. Value: measurement of data
3. Is\_anomaly: indicates whether current instance is anomalous or not, 0 for non-anomalous, 1 for anomalous.

### **3.1.1 Data Preparation**

A1 and A2 datasets are separated into multiple files (67 files for A1, 100 files for A2) and each file has different magnitude of measurement value for instances. So, to merge them into single file with compatible value attributes, a pre-processing method has to applied to standardize data (Nayyar et al 2016). These steps can be listed as:

1. Scale each file around mean
2. Calculate local minimum and maximum values for each file
3. Calculate scale factor that will be applied all files with following formula:

avgMean = mean of local minimums

avgMax = mean of local maximums

scaleFactor = (avgMin + avgMax) / 2

1. Multiply each value in all files with scale factor calculated in previous step.
2. Merge all files into one.

## **3.2 METHODS**

In this chapter, methods that are applied to datasets during experiment will be explained in detail. Essential part of methods is based on same algorithm which is to calculate a confidence interval that next expected instance will most likely be in, using previous observed instances. These previous observed instances are formed using Sliding Window technique. Sliding Window handles fixed number of instances (window) at a time, the earliest instance is dropped out of window when a new instance is observed. Window size is hyper parameter of this method. Algorithm 3.1 explains calculating confidence interval.

**Algorithm 3.1:**

1. Set an appropriate window size for data, n.
2. Iterate fixed size window though whole date set.

window-1 = {1, 2, 3, 4, …, n - 1, n}

window-2 = {2, 3, 4, …, n, n + 1}

window-3 = {3, 4, …, n + 1, n + 2}

1. Calculate confidence interval (CI) for next instance. (Calculate lower and upper bounds)
2. If next observed instance is inside of CI, it is not anomalous, otherwise, mark it as anomaly.
3. If record is mark anomaly, then replace it with non-anomalous record, else do nothing.

Remainder of this chapter will provide methods to calculate CI for step 3.

### **3.2.1 Moving Average**

Moving Average is a method that only uses mean and standard deviation of window to calculate CI. It is based on the assumption that next instance will be most likely be similar with the window records. In this domain, it is mean and standard deviation that makes a record coherent with entire window. One can generate many useful formulas that can be effective for calculating CI, here is a sample to express idea better:

|  |  |
| --- | --- |
|  | (3.1) |

where mean is mean, sd is standard deviation of window records, n is window size and m is a hyper parameter to adjust confidence interval.

### **3.2.2 Auto-Regressive Integrated Moving Average (ARIMA)**

ARIMA is a popular and widely used model for time series forecasting and data analysis. Forecasting the next instance can result calculating better confidence intervals, which leads sliding window technique to apply better anomaly test.

ARIMA is a generalization of Auto-Regressive Moving Average (ARMA) model. Fitting an ARIMA model is sometimes called Box-Jenkins Model. It has three key parts, hence it is notated by 3 parameters as ARIMA (p, d, q). There are briefly listed here:

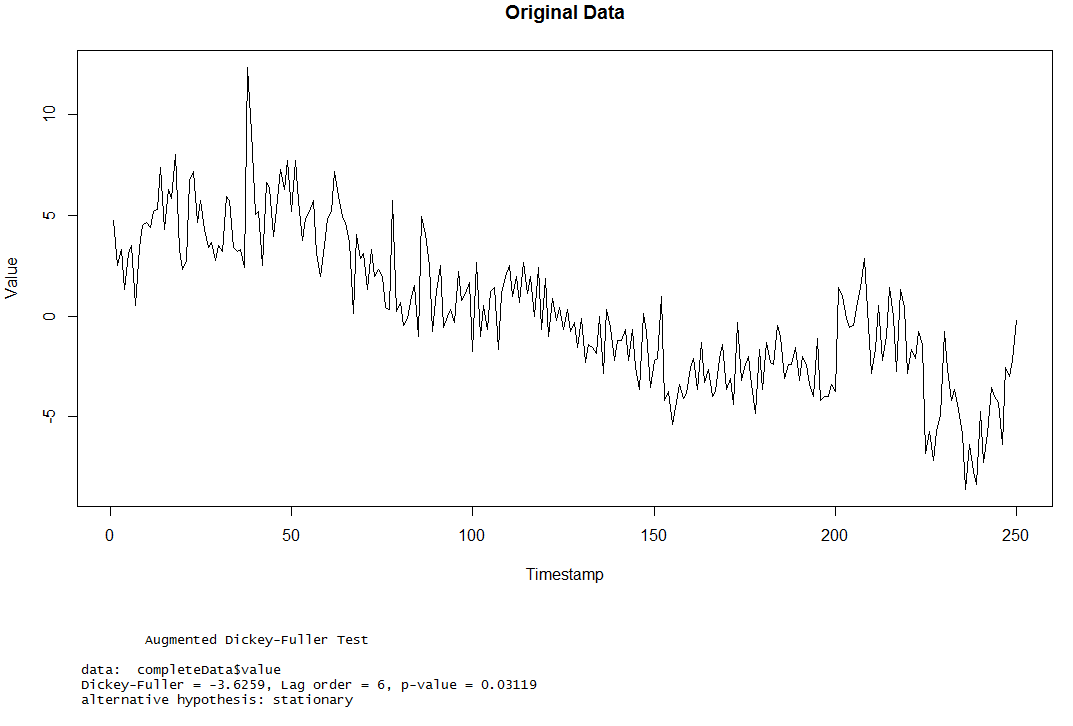
* AR(p): Auto-regression. Number of lags used in the model.
* I(d): Integrated. Degree of differencing between past and current values d times. It is used to make series stationary.
* MA(q): Moving average. MA refers to residual error of model and q is the number of instances in moving window.

In order to make ARIMA model forecast better results, following steps must be applied.

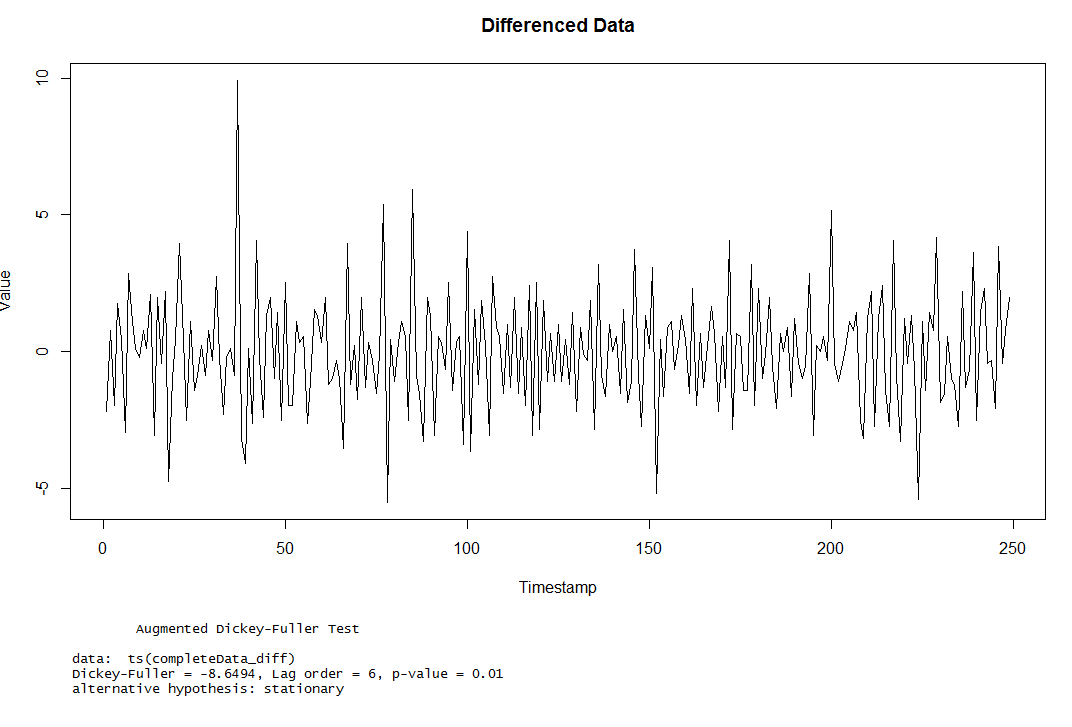
1. Visualize data: To model a successful model, trend, seasonality or random behavior must be determined. Because, ARIMA works best if data is stable and stationary.
2. Make data stationary: Augmented Dickey-Fuller (ADF) test is a popular stationary test for ARIMA models. As a result of the test, if data is non-stationary, differencing which is Integration part of model can be applied to data. Decomposition is also an effective method. It will be detailed in Section 3.2.4.

The lower p-value output is generated from ADF test; the better d parameter is gained for model. To determine d parameter, first, original data is passed to ADF test (Figure 3.1). Then, a differenced version of data with degree 1 is tested. Figure 3.2 shows an ADF test with differenced data and after this test d can be determined as 1.

**Figure 3.1: Original Data ADF Test**

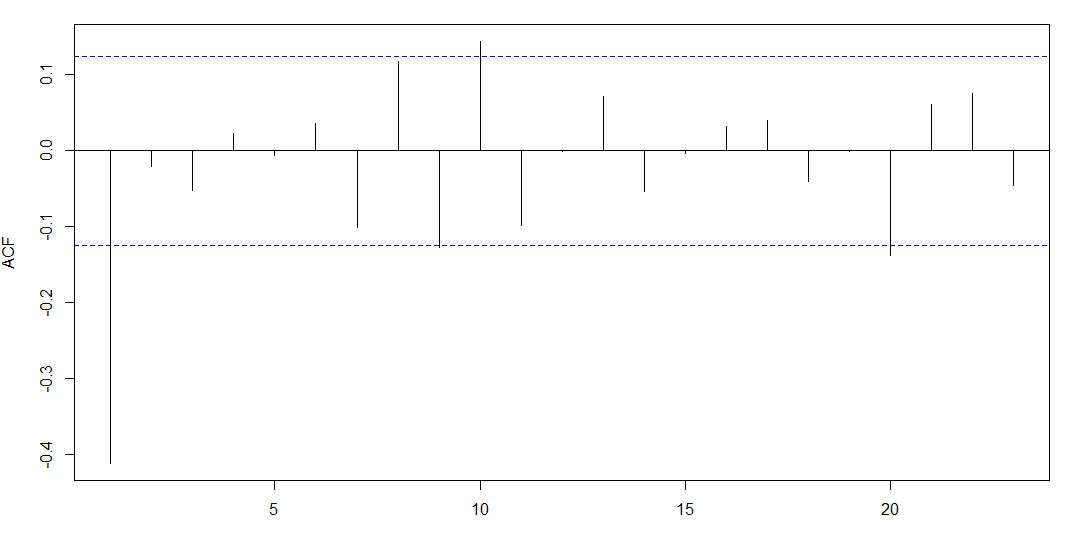


**Figure 3.2: Differenced Data ADF Test**

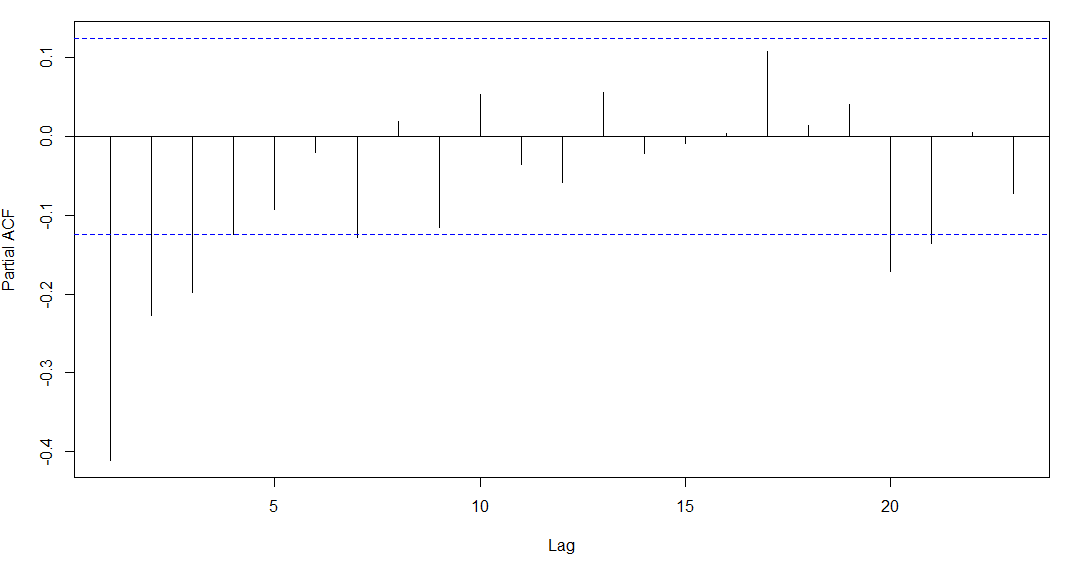


1. Find p and q: Other two essential parameters of model, p and q can be found using ACF (Figure 3.3) and PACF (Figure 3.4) plots. Figures show that both have a peak at Lag 1. This means p and q can be 1.

**Figure 3.3: ACF Plot**



**Figure 3.4: PACF Plot**



1. Build ARIMA model with p, d and q: Detail of building is beyond this experiment, hence a popular online R library “forecast” is used to build a model.

Main goal of building an ARIMA is to calculate a better CI. “Forecast” library also provides parametric CI results. Thus, for this model, probability of CI that next instance will most likely be in is a hyper-parameter. As it is known anomaly is a rare case, probability can be between 90 per cent and 99 per cent.

### **3.2.3 Kalman Filter**

Kalman Filter (Kalman 1960) is an algorithm that uses Gaussian Distribution and previously measured instances to produce estimation of unknown variables. Despite its almost 60-year history, it is still a powerful solution to tracking and data prediction tasks. Typical usage areas of Kalman Filter are global positioning system, guidance of aircrafts, data smoothing, robotic motion control, time series signal processing etc.

In this experiment, Kalman Filter was used to predict next time series instance value and then calculate CI based on it. ARIMA can fail producing good results with volatile time series. It is essential to make data stationary before building ARIMA model. Kalman Filter is an alternative solution to smooth data and make predictions about future. Algorithm 3.2 explains how Kalman Filter smooths time series data:

**Algorithm 3.2**

1. Let be previous observed value, be current value, be covariance uncertainty, be uncertainty of environment like noise, be error.
2. Initialize starting error with 1.
3. Calculate covariance;

|  |  |
| --- | --- |
|  | (3.2) |

1. Calculate gain;

|  |  |
| --- | --- |
|  | (3.3) |

1. Predict next value using previous value and gain;

|  |  |
| --- | --- |
|  | (3.4) |

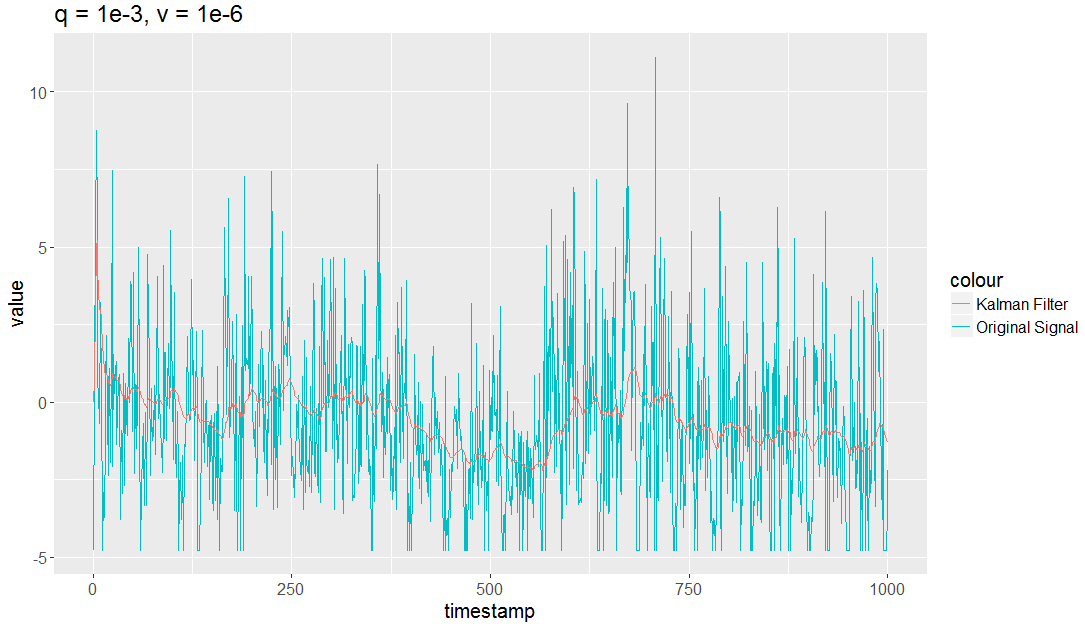
1. Calculate error which will be used in next iteration;

|  |  |
| --- | --- |
|  | (3.5) |

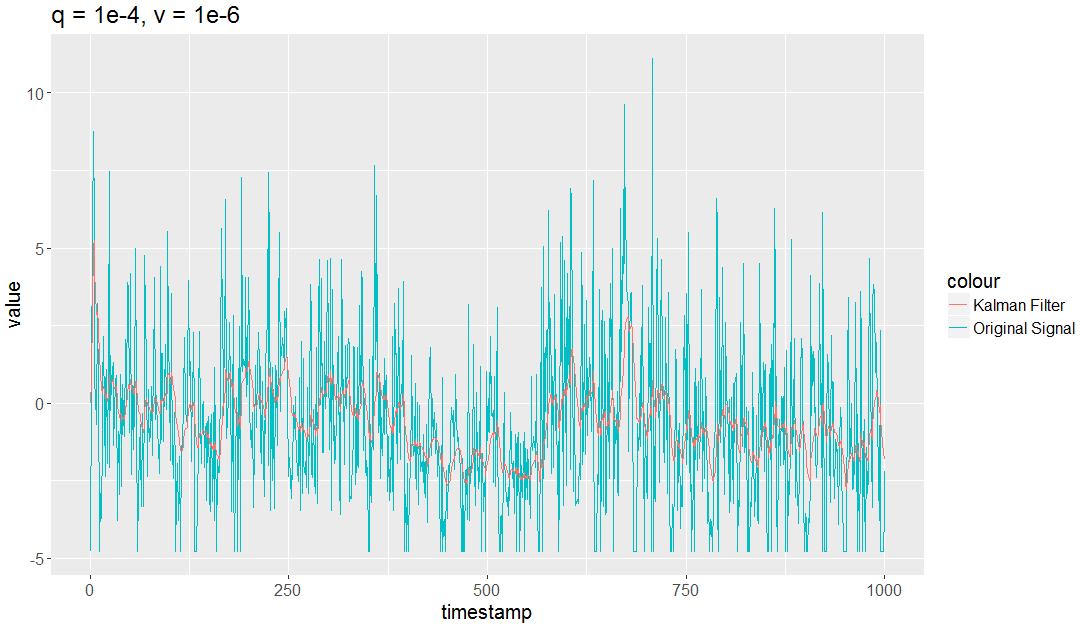
1. Iterate step 3 to 6 until all records are processed.

In algorithm 3.2, and are hyper-parameters. Following images demonstrate effects of different parameters on signal smoothing.

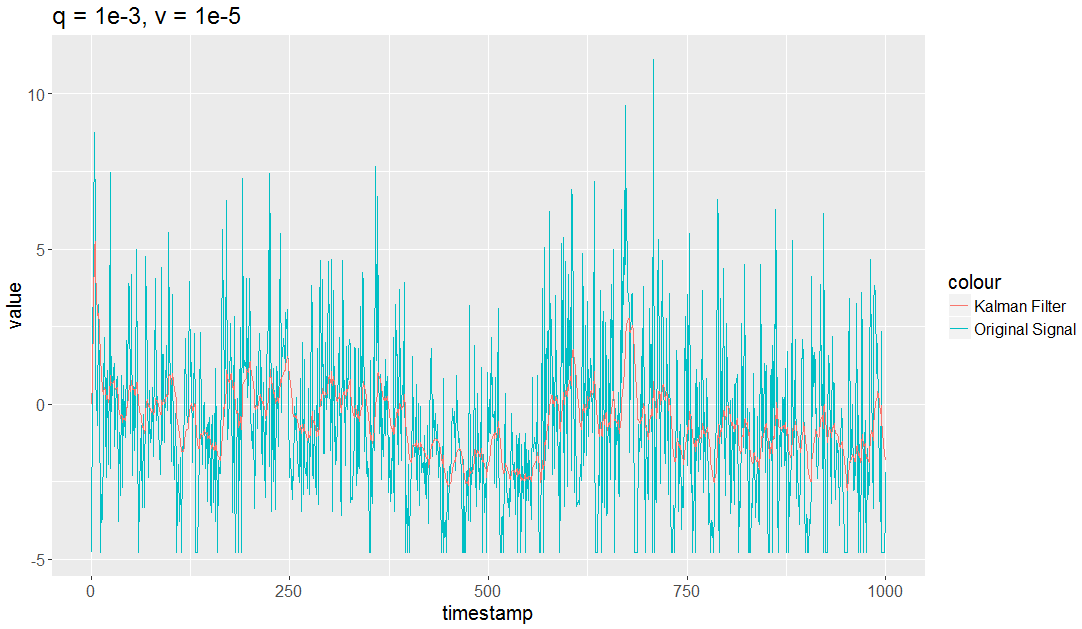
**Figure 3.5: Reference values for uncertainty parameters**



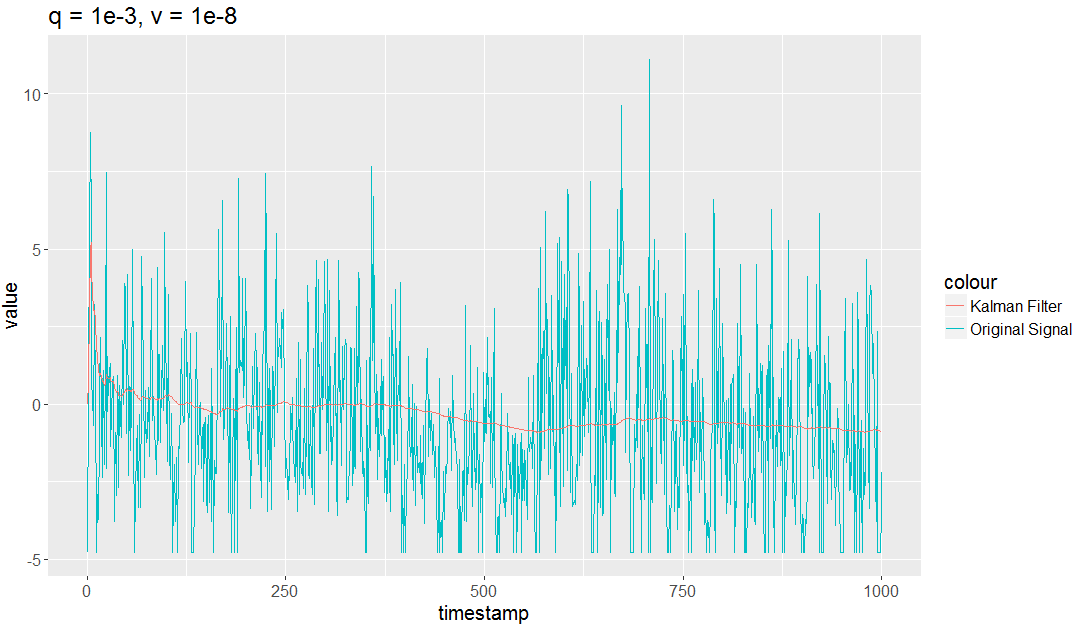
**Figure 3.6: Lower value for covariance uncertainty**



**Figure 3.7: Higher value for environment uncertainty**



**Figure 3.8: Lower value for environment uncertainty**



Difference of Figure 3.5 and Figure 3.6 shows that for a lower covariance uncertainty, more volatile signal is obtained. For environment uncertainty, Figure 3.7 and Figure 3.8 demonstrate that high value gets a volatile signal while low value gets a smother signal.

Once signal is smoothed, CI can be calculated using equation 3.1 and signal properties like mean and standard deviation.

### **3.2.4 Time Series Decomposition**

Time series decomposition is splitting time series data into three parts; seasonality, trend and residual. Residual part is used to detect anomalies.

Residual is remainder of data when seasonality and trend are removed. To remove them, first, one must find seasonal behavior of data and calculate trend. In this experiment 2 different seasonality determination method are used; Discrete Fourier Transform and correlation of time series windows.

**Discrete Fourier Transform:** DFT is one of the most powerful digital signal processing tools. It is basically converting finite sequence of data into same-length of Discrete-Time Fourier Transform (DTFT) sequences. It can provide multiple period lengths for seasonality of data. In this experiment, period which is the smallest amongst numbers that are greater than 24 is used for sliding windows. The reason of number 24 is that data is collected by hourly observation and 24 hours mean a day. A day of periodicity looks reasonable for data.

Drawback of DFT is it always gives a periodicity length result whether data has seasonality component or not. Threating data like it has seasonality results incorrect residual calculation, thus it results false anomaly detections.

**Correlation:** If time series data has seasonality, then successive periods of data should be highly correlated. Based on this definition, if a data window has seasonality, period length can be found using Algorithm 3.3.

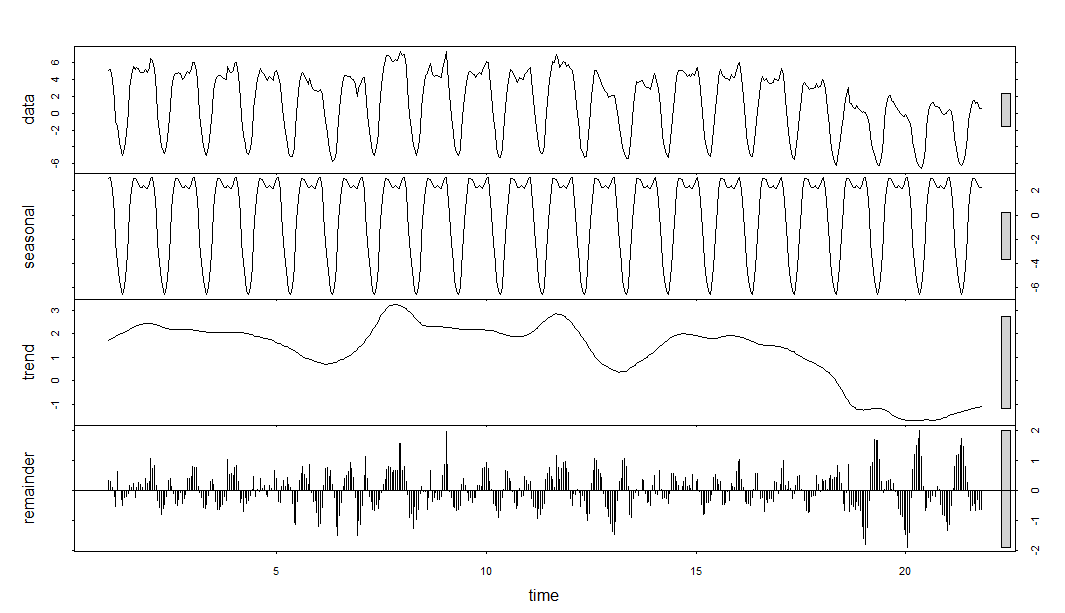
**Algorithm 3.3**

1. Set a possible number of period length (p) values.
2. **S**elect a new p value and split data into p parts.
3. Calculate correlation of all periods with each other.
4. If correlation value is greater than threshold, then p is seasonal length.
5. If correlation value is lower than threshold, repeat Step 2.

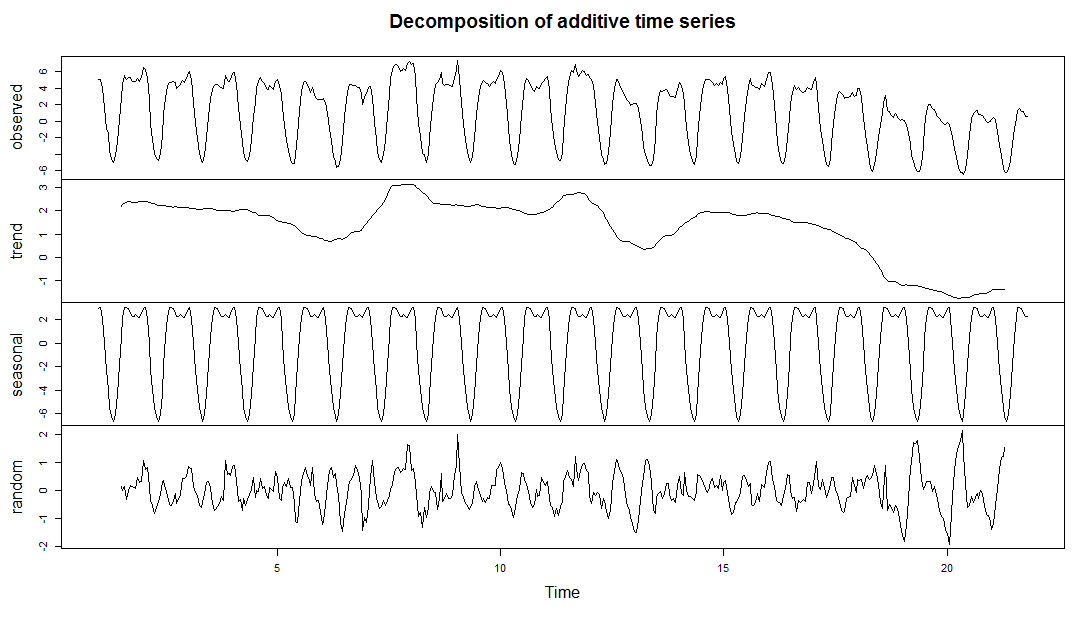
Drawback of Algorithm 3.3 is complexity of calculation which is O(n2) in worst case.

As an alternative of these two methods, built-in R methods stl() and decompose() can be used. But both methods require period length as input. Figure 3.9 and Figure 3.10 show sample outputs of stl() and decompose(). As It can be seen on images, outputs also consist trend and residual part which is important for anomaly detection procedures.

**Figure 3.9: STL Output**

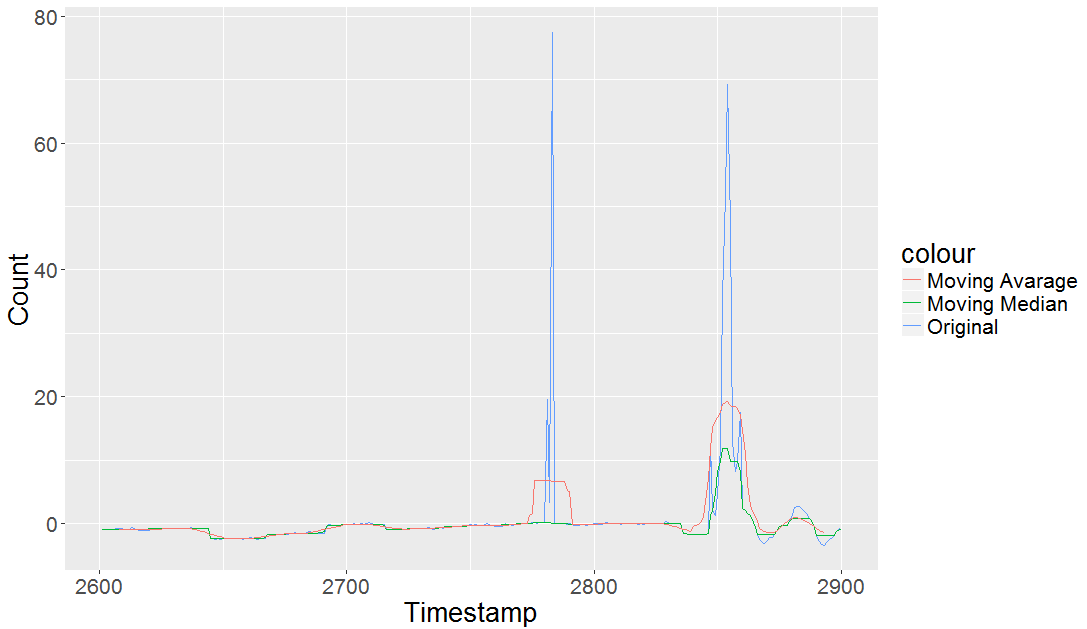


**Figure 3.10: Decompose Output**



Before starting anomaly detection, trend in data must also be calculated. Moving average and moving median both can be used to calculate trend. However, moving median is more robust to anomalies, thus, it fits better for anomaly detection procedures. Figure 3.11 shows the difference between two methods in case of anomaly exists in data. Moving average has higher spikes than moving median in case of anomalies.

**Figure 3.11: Moving Average vs Moving Median**



As it is mentioned before, residual is what is left when seasonality and trend have been removed. Confidence interval (CI) can be calculated based on mean and standard deviation of residual according to Equation 3.1.

# **RESULTS**

In this section results of methods that are described in 3.2 are provided in terms of precision, recall, f1 score and relationship of hyper-parameters with successful and failed experiments. In addition, strengths and drawbacks of implemented methods while detecting different anomaly types will be pointed out. At the end of section, each method will be compared with each other using these metrics and run time performances.

The idea of this study is to detect anomalies with accuracy and propose a method that works efficiently with all kind of anomaly types. According to this statement, Moving Average method produces good results detecting anomalies with high accuracy as window size is increased. Table 4.1 shows that it produces best F1 score with window size 25 and interval multiplier 30. Increasing window size helps Moving Average technique to analyze sliding window records better and produce more accurate mean and standard deviation values those are key parameters of Equation 3.1 to calculate confidence interval. But when window size gets bigger than 25, f1 score tends to decrease, because window mean and standard deviation are affected by volatile signals.

**Table 4.1: Moving Average Results**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Window Size** | **Multiplier** | **TP** | **FP** | **TN** | **FN** | **Recall** | **Precision** | **F1 score** |
| 5 | 20 | 1443 | 1090 | 92102 | 226 | 0.86459 | 0.56968 | 0.686816 |
| 8 | 20 | 1435 | 691 | 92498 | 234 | 0.859796 | 0.674976 | 0.756258 |
| 11 | 20 | 1422 | 664 | 92522 | 247 | 0.852007 | 0.681687 | 0.75739 |
| 14 | 20 | 1427 | 700 | 92483 | 242 | 0.855003 | 0.670898 | 0.751844 |
| 10 | 25 | 1382 | 423 | 92764 | 287 | 0.828041 | 0.765651 | 0.795625 |
| 15 | 25 | 1350 | 371 | 92811 | 319 | 0.808868 | 0.784428 | 0.79646 |
| 20 | 25 | 1347 | 304 | 92873 | 322 | 0.80707 | 0.815869 | 0.811446 |
| 25 | 25 | 1342 | 309 | 92863 | 327 | 0.804074 | 0.812841 | 0.808434 |
| 30 | 25 | 1344 | 386 | 92781 | 325 | 0.805273 | 0.776879 | 0.790821 |
| 10 | 30 | 1345 | 316 | 92871 | 324 | 0.805872 | 0.809753 | 0.807808 |
| 15 | 30 | 1296 | 269 | 92913 | 373 | 0.776513 | 0.828115 | 0.801484 |
| 20 | 30 | 1281 | 213 | 92964 | 388 | 0.767525 | 0.85743 | 0.809991 |
| 25 | 30 | 1280 | 199 | 92973 | 389 | 0.766926 | 0.86545 | 0.813215 |
| 30 | 30 | 1293 | 238 | 92929 | 376 | 0.774715 | 0.844546 | 0.808125 |

Another result that can be extracted from Table 4.1 is that when interval multiplier is increased, better f1 score can be obtained. Bigger interval multiplier means wider confidence interval, thus, there is a tradeoff between true results and false results. When confidence interval gets wider, number of true/false positive results decrease while true/false negative results increase. As ratio of true results are bigger than false results, it can be tolerated. Optimal value for this study according to Table 4.1 is 25.

Moving average does not consider characteristics of time series data while it calculates confidence interval. So, it fails to detect contextual or collective anomalies that are only considered anomaly under certain circumstances (Section 1.2.2 and 1.2.3). However, it provides good results in case of sudden changes in streaming data which are called point anomalies (Section 1.2.1). But in case of several close point anomalies, it fails to detect recent ones as it adopts to anomalous mean and standard deviation.

ARIMA needs bigger sliding window size when it is compared to moving average algorithm, since it involves a learning process inside. Thus, bigger windows produce higher accuracy f1 score (Table 4.2). But in general ARIMA fails to generate high accuracy results because of too many false positive results. The reason of these mislabeled records and possible actions to make it work better will be discussed in next section.

ARIMA considers seasonality and trend of data while its model is being determined. Thus, it can produce good results with both contextual and point anomalies.

**Table 4.2: ARIMA Results**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Window Size** | **TP** | **FP** | **TN** | **FN** | **Recall** | **Precision** | **F1 score** |
| 150 | 970 | 2950 | 89747 | 699 | 0.581186 | 0.247449 | 0.34711 |
| 200 | 952 | 2372 | 90425 | 717 | 0.570401 | 0.286402 | 0.381334 |
| 250 | 964 | 2389 | 90358 | 705 | 0.577591 | 0.287504 | 0.383911 |
| 300 | 948 | 2204 | 90643 | 721 | 0.568005 | 0.300761 | 0.393279 |
| 350 | 954 | 2034 | 90813 | 715 | 0.5716 | 0.319277 | 0.409706 |

Kalman filter has also a learning process, thus, it produces better results with bigger window size according to Table 4.3. It has best results between 150 and 250. Over 250, despite increasing number of true positive records, because of overfitting, it generates too many false positive records. Thus, it generates low f1 score. Two hyper-parameters of this algorithm is also decisive to build a model; environment and covariance uncertainty. According to result table, as environment uncertainty decreases which means a smoother signal (Figure 3.8), f1 score increases. This fact states that smoother signal produces better anomaly detection results. Covariance uncertainty has its optimum value at 0,001. However, its variety does not affect result as much as environment uncertainty.

**Table 4.3: Kalman Filter Results**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **WS** | **Env. Uncert.** | **Cov. Uncert.** | **TP** | **FP** | **TN** | **FN** | **Recall** | **F1 score** |
| 50 | 1,00E-04 | 5,00E-04 | 440 | 202 | 92945 | 1229 | 0.263631 | 0.380788 |
| 150 | 1,00E-04 | 5,00E-04 | 667 | 431 | 92616 | 1002 | 0.399641 | 0.482111 |
| 250 | 1,00E-04 | 5,00E-04 | 822 | 818 | 92129 | 847 | 0.49251 | 0.496827 |
| 350 | 1,00E-04 | 5,00E-04 | 940 | 1723 | 91124 | 729 | 0.563212 | 0.43398 |
| 450 | 1,00E-04 | 5,00E-04 | 964 | 2389 | 90358 | 705 | 0.577591 | 0.383911 |
| 150 | 1,00E-04 | 5,00E-04 | 572 | 252 | 92795 | 1097 | 0.34272 | 0.458885 |
| 250 | 1,00E-04 | 5,00E-04 | 758 | 690 | 92257 | 911 | 0.454164 | 0.486365 |
| 350 | 1,00E-04 | 5,00E-04 | 867 | 1190 | 91657 | 802 | 0.519473 | 0.465378 |
| 150 | 1,00E-04 | 0,01 | 662 | 386 | 92661 | 1007 | 0.396645 | 0.487302 |
| 250 | 1,00E-04 | 0,01 | 834 | 1015 | 91932 | 835 | 0.4997 | 0.474133 |
| 350 | 1,00E-04 | 0,01 | 639 | 801 | 46015 | 436 | 0.594419 | 0.508151 |
| 150 | 1,00E-06 | 0,001 | 731 | 478 | 92569 | 938 | 0.437987 | 0.507992 |
| 250 | 1,00E-06 | 0,001 | 875 | 1164 | 91783 | 794 | 0.524266 | 0.471953 |
| 350 | 1,00E-06 | 0,001 | 954 | 2034 | 90813 | 715 | 0.5716 | 0.409706 |
| 150 | 1,00E-07 | 0,001 | 768 | 540 | 92507 | 901 | 0.460156 | 0.515956 |
| 250 | 1,00E-07 | 0,001 | 884 | 1112 | 91835 | 785 | 0.529658 | 0.482401 |
| 350 | 1,00E-07 | 0,001 | 948 | 2204 | 90643 | 721 | 0.568005 | 0.393279 |
| 150 | 1,00E-06 | 0,1 | 729 | 505 | 90719 | 896 | 0.448615 | 0.509969 |
| 150 | 1,00E-07 | 0,001 | 828 | 964 | 92083 | 841 | 0.496105 | 0.478474 |

Kalman Filter does not consider seasonality while it is making predictions about next records but trend is included to its calculations. Its basic feature to predict next records approximately based on previously observed sequences and uncertainty. So, it fails to detect contextual and collective anomalies but it produces good results for point anomalies.

Time series decomposition technique extracts seasonality and trend from original signal, thus this provides good results with point and contextual anomalies, it can also detect collective anomalies if sliding window records are formed correctly. Choosing windows correctly will be discussed in detail in Section 5.

As it can be seen in Table 4.4, f1 score does not change significantly when window size is increased. The reason of this situation is that time series decomposition does not learn from previous records, it splits signal into seasonal, trend and residual components which it uses to detect anomalies.

**Table 4.4: Time Series Decomposition Results**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Window Size | TP | FP | TN | FN | Recall | Precision | F1 score |
| 96 | 483 | 1838 | 91359 | 1186 | 0.289395 | 0.2081 | 0.242105 |
| 120 | 516 | 1896 | 91272 | 1182 | 0.303887 | 0.21393 | 0.251095 |
| 144 | 491 | 1878 | 91300 | 1197 | 0.290877 | 0.20726 | 0.242051 |
| 168 | 495 | 1925 | 91343 | 1103 | 0.309762 | 0.204545 | 0.246391 |
| 192 | 507 | 1942 | 91325 | 1092 | 0.317073 | 0.207023 | 0.250494 |
| 216 | 505 | 2053 | 91311 | 997 | 0.336218 | 0.19742 | 0.248768 |

Despite its good accuracy for point and contextual anomalies, in this experiment, time series decomposition generates poor f1 score according to other algorithms. It is due to the fact that applied algorithm can be improved to get better results. This part will also be discussed in Section 5.

Combining all applied algorithms together (Table 4.5), moving average has the best f1 score while time series decomposition has the worst. ARIMA and Kalman Filter has close results but Kalman leads it by 0.1 margin.

**Table 4.5: Algorithm Comparison**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Recall** | **F1 score** | **Run time (minutes)** |
| **Moving Average** | 0.8136609 | 0.8143939 | 5 |
| **ARIMA** | 0.5811863 | 0.4097058 | 200 |
| **Kalman Filter** | 0.5944186 | 0.5199131 | 12 |
| **Time Series Dec.** | 0.3362184 | 0.2504941 | 10 |

In terms of completion time of algorithms, ARIMA has the slowest execution of all. The other three has close results but moving average with its simple calculation tasks is the fastest among all. ARIMA’s slow execution time can be explained with its complex steps like such as ACF, PACF examinations, ADF Test, stationary process, learning in model building phase. Other three algorithms have simpler tasks to build a working model.

For different anomaly types, success rates of algorithms are listed in Table 4.6. For all three anomaly types, time series composition has best results as it is the only one that can detect collective anomalies. Point anomalies are detected by all algorithms. Contextual anomalies are detected by ARIMA and time series decomposition that use seasonal and trend components in signal.

**Table 4.6: Results for anomaly types**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Point Anomaly** | **Contextual Anomaly** | **Collective Anomaly** |
| **Moving Average** | Success | Fail | Fail |
| **ARIMA** | Success | Success | Fail |
| **Kalman Filter** | Success | Fail | Fail |
| **Time Series Dec.** | Success | Success | Success/Fail |

# **DISCUSSION**

In this section, results of experiments with detail, strengths and drawbacks of algorithms, obstacles that cause experiments to produce low f1 scores will be discussed. Algorithms will be compared with each other and an open source time series anomaly detection library that is created by Twitter. Twitter also uses time series decomposition to detect anomalies. It differs from algorithm explained in Section 3.2.4 with its determination of seasonal and trend components.

## **POINT ANOMALY**

Point anomalies can be detected by all algorithms explained in Section 3 successfully. But in case of several anomalous point in same window can result false negatives for Moving Average, ARIMA and Kalman Filter. Because these methods include mean and standard deviation for confidence interval calculation. Mean and standard deviation is not robust to anomalies. To resolve this issue, median and median absolute deviation can be used instead of mean and standard deviation.

Kalman Filter and ARIMA learns from previously observed records to predict the next one. If there are many anomalous points in previous records, these algorithms learn and predict signal from some data that does not reflect the normal behavior. Thus, next predicted records and confidence interval can be calculated incorrectly. Predicting incorrectly can produce many false positive/negative results which leads to an unreliable model. In this experiment, two different methods were used to solve this problem; removing points that are predicted as anomalous by model from window before sending it to prediction algorithm and replacing them with a convenient value. Former one can be used in case of few anomalies in window but if majority of window records are anomalous, prediction is done with only few observations. Predictions with insufficient number of instances cannot produce good results. Latter method should also be handled with care since replacing instances with wrong values is a big problem for prediction task. Different calculations are used for each algorithm to overcome this problem. For Moving Average and ARIMA, anomalous instances were replaced with average of calculated confidence interval, for Kalman Filter, they were replaced with output of algorithm (prediction) and for time series decomposition, a replacement wasn’t needed. The reason of choosing these calculations for replacement is that they are cross-validated with different calculations and these are ones with best results.

## **CONTEXTUAL ANOMALY**

Contextual anomalies are more difficult to detect than point anomalies according to experiments that are done in this study. Because, these type of instances are not anomalous individually but can be considered as one in certain contexts. So, algorithms that does not take current context of instance into account while determining whether it is anomalous or not does not produce good results with this kind of anomaly.

To understand the current context of instance, characteristics of time series data (Section 1.1.2) can be used. If data has seasonal behavior, one can expect the next instance to be close to previous periodic windows. Moving Average and Kalman Filter does not calculate any seasonality or trend, so they fail to detect contextual anomalies unless they are also point anomalies.

ARIMA and Time Series Decomposition use seasonality and trend to detect anomalies. Thus, they can be used for contextual anomaly detection. The reason of low f1 score for Time Series Decomposition despite its capabilities in detecting contextual anomaly is sliding window algorithm. Sliding window algorithm always keeps fixed number of instances in a window and removes first record from it when a new instance is observed. Hence, both seasonal and normal behavior can be represented in a window. Question here is which behavior should be considered while calculating confidence interval? According to experiments in this study, the answer is none. Different behaviors should be considered differently for accurate confidence interval values. To avoid multiple behaviors in a window, seasonality starting and ending point should be determined and windows should be formed with changing points as their boundaries. Time series change point detection algorithms can be useful to find such points in data. Cook and Aminikhanghahi (2017) provides a survey with wide range of change point detection algorithms.

## **COLLECTIVE ANOMALY**

Collective anomalies are the most difficult type amongst others to detect. In this study, any of the algorithms used does not produce robust result for this type of anomaly. Time series decomposition has the closest results but anomalous instances are considered as collective when a group of related instances do not conform with the rest of data. Hence, with sliding window algorithm, first observations of these groups are detected as contextual anomaly since, at first, they are individual instances out of current context. So, from this study, it is concluded that collective anomaly needs different kind of approach than algorithms in Section 3. This approach will be left as a future work.

## **COMPARISON WITH LIBRARIES**

Twitter library execution results with same dataset that is used by algorithms described in Section 3 are demonstrated in Table 5.1. It produces close recall, precision and f1 score values with results provided in Section 4. But it also produces low value of false positive records. The reason of high false negative results is that it also can’t detect collective anomalies and in addition, it has low accuracy on multiple point or contextual anomalies close to each other. Because it only detects certain percentage of records in dataset using generalized Extreme Studentized Deviate. Thus, high number of anomalous point are being regarded in a close group.

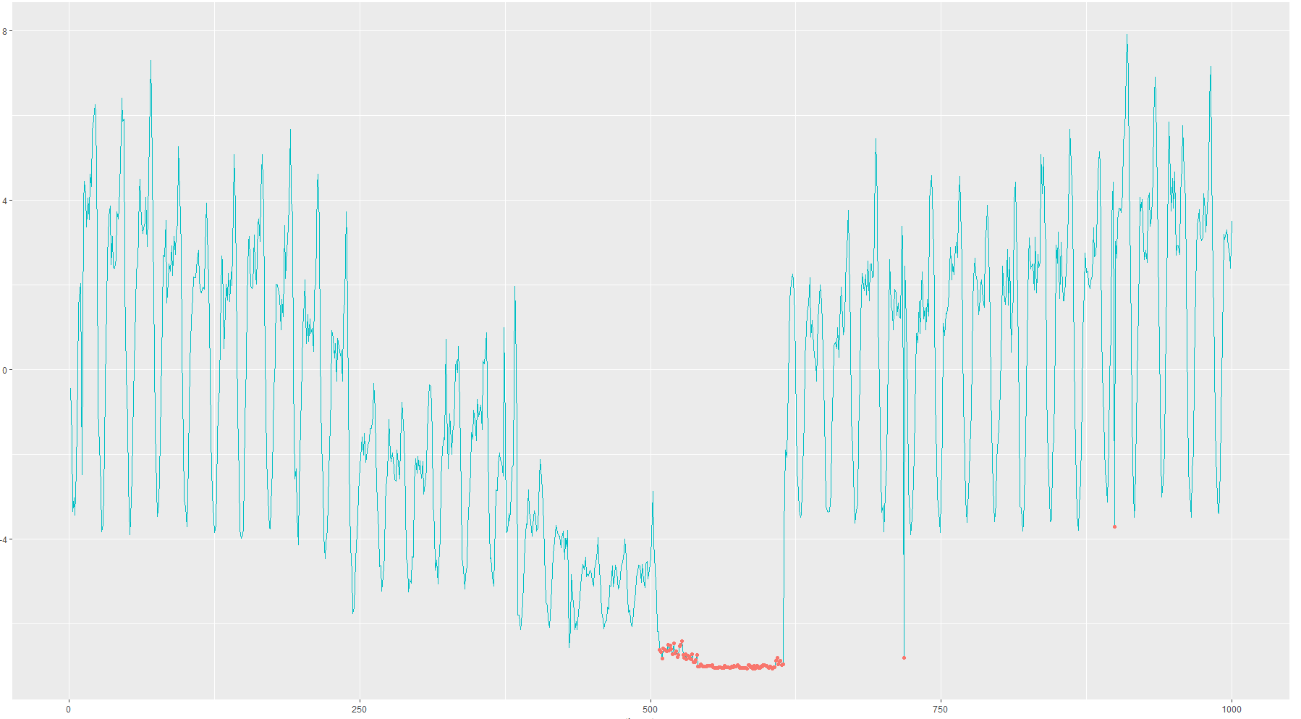
**Table 5.1: Twitter library results**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP** | **FP** | **TN** | **FN** | **Recall** | **Precision** | **F1 score** | **Time** |
| **Count** | 573 | 384 | 92813 | 1096 | 0.3433 | 0.5987 | 0.4364 | 20 min. |

Twitter library uses time series decomposition for anomaly detection. Hence, it has similarities between algorithm that is used in this study. However, it has good results in change points and handles these groups better. But it takes more time for Twitter library to finish all data, approximately 20 minutes.

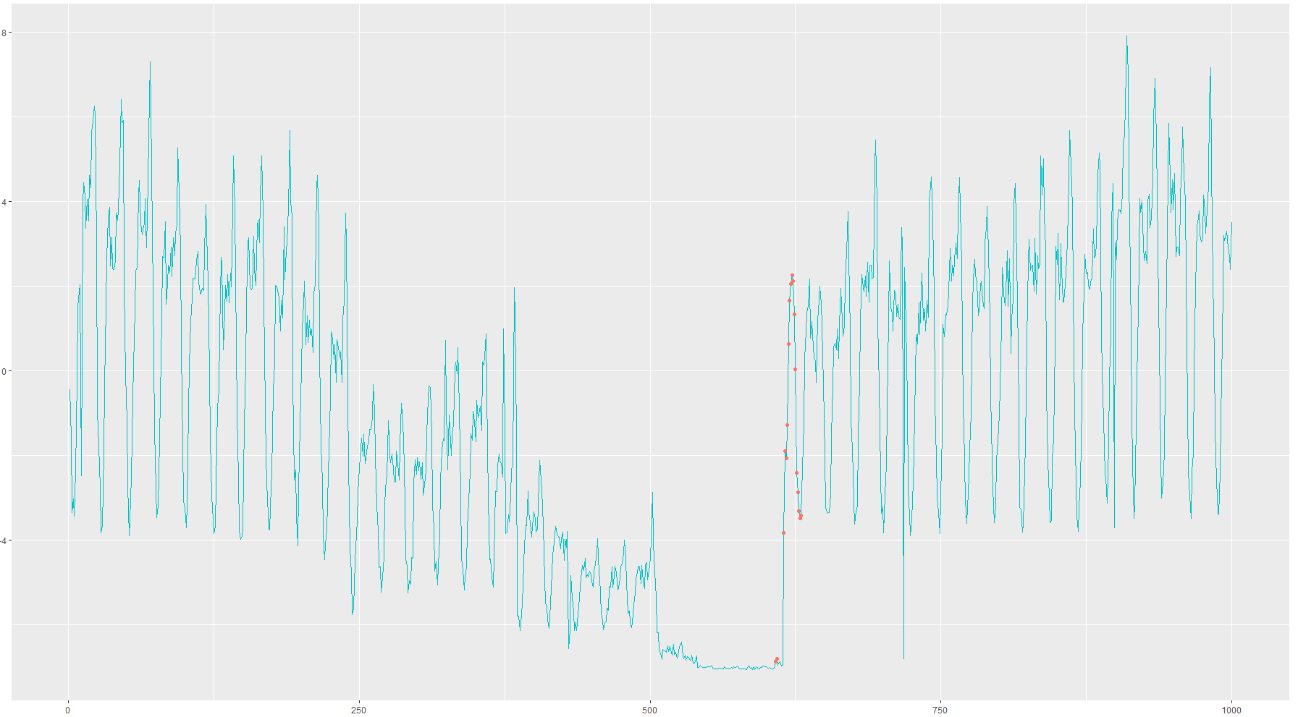
Comparison of Twitter library with implemented algorithms in detection accuracy can be visualized in Figures 5.1 to Figure 5.6. Figure 5.1 is the source data with labelled anomalies. This window has been chosen specifically to illustrate differences because it contains both collective and contextual anomalies. Sequence between 500 and 600 forms a collective anomaly because of breaking seasonality and generating steady signal instead. Other two anomalies are contextual anomalies. They break seasonality individually. Point anomalies are not considered here since all algorithms covered here produces robust results with them.

**Figure 5.1: True anomaly set in sample series**



Moving Average detects records that are last part of collective anomaly and beginning of seasonal behavior. For this algorithm, big margin change on new data is considered as anomaly, thus beginning part of new seasonal window is labeled as anomalous (Figure 5.2). It continues to detect them as anomalous until standard deviation and mean gets close enough to seasonal part.

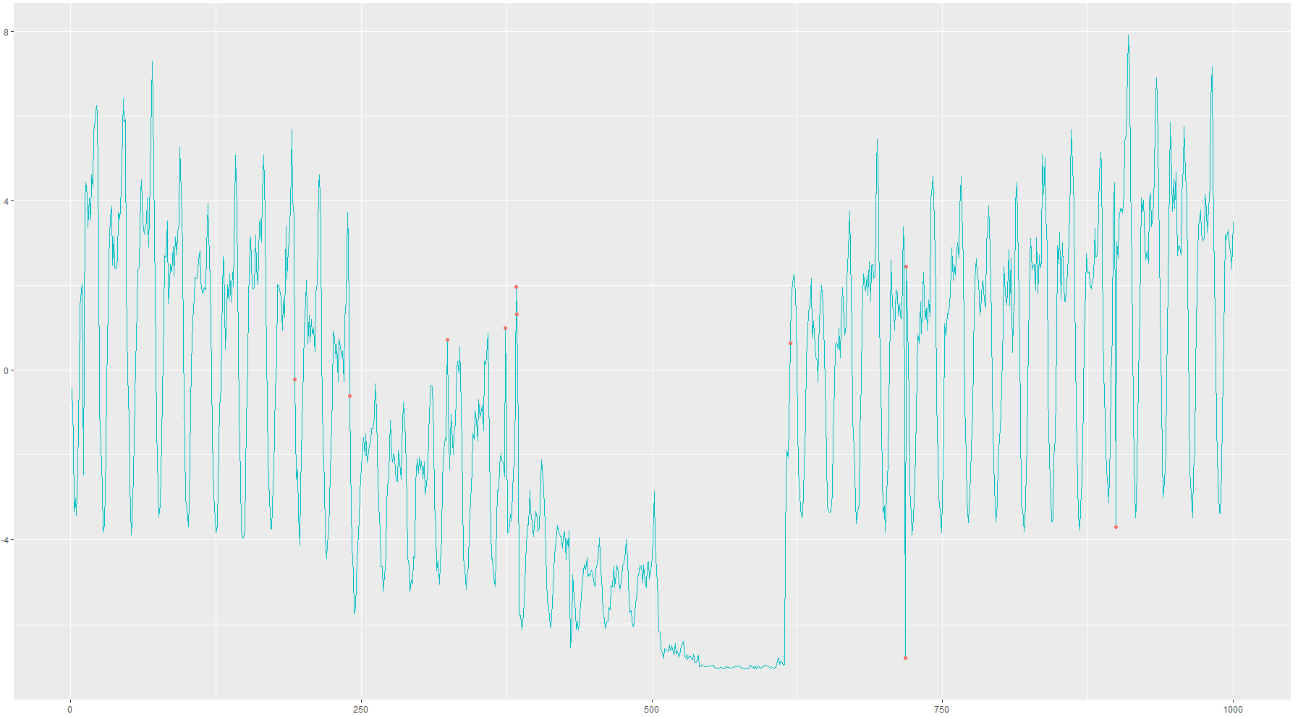
**Figure 5.2: Moving Average sample result**



Also, it cannot detect contextual anomalies since bigger standard deviation cause larger confidence intervals. Although they are anomalous according to current context, they are in calculated intervals.

ARIMA correctly classifies contextual anomalies (Figure 5.3) and generates some false positives. But, they are acceptable because there some differences between seasonal windows, thus, they can be outside of confidence interval for that reason. Like Moving Average, ARIMA cannot detect collective anomalies.

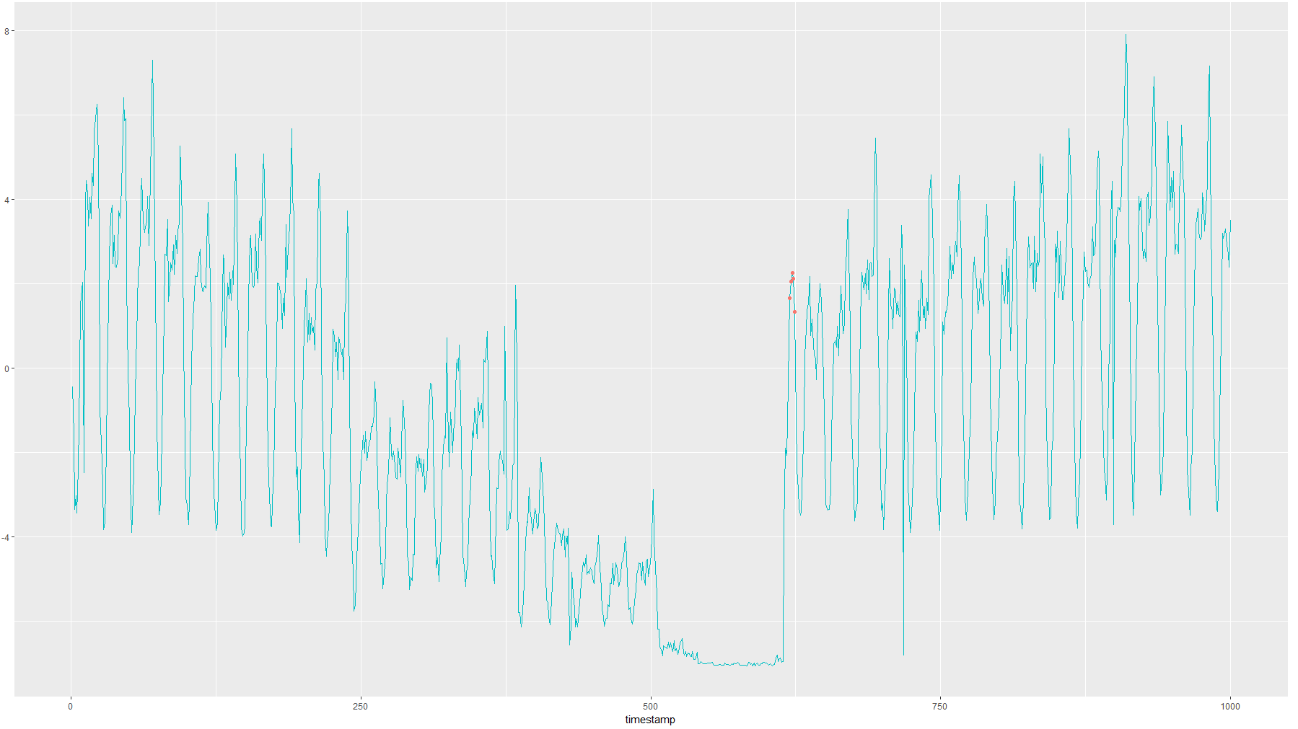
**Figure 5.3: ARIMA sample result**



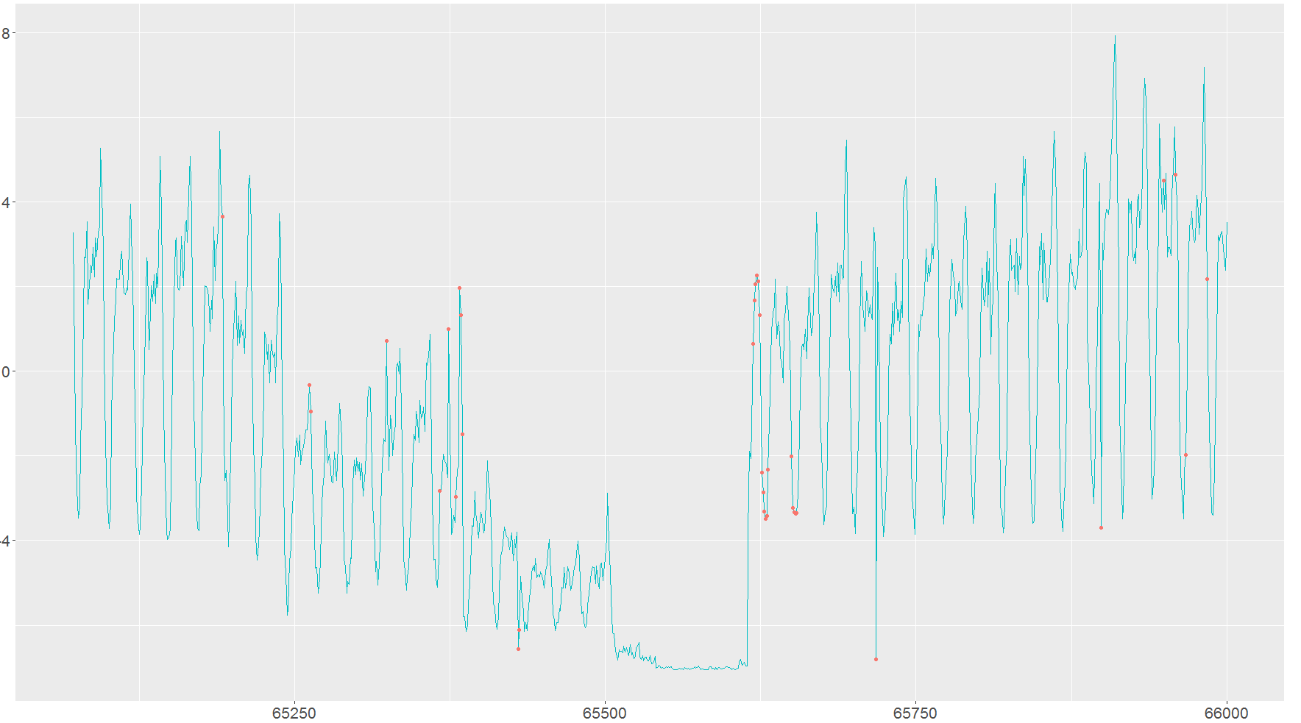
As it is stated before, Kalman Filter produces its best results for point anomalies. Figure 5.4 confirms that it detects some records as anomaly when sudden change in data value occurs. But unlike Moving Average, it adopts more quickly to change.

Time Series Decomposition produces similar results (Figure 5.5) with ARIMA. It detects contextual anomalies and produces some false positives. But if sliding windows are selected better it can be more accurate. As with other three algorithms, it fails to detect collective anomalies in sample set.

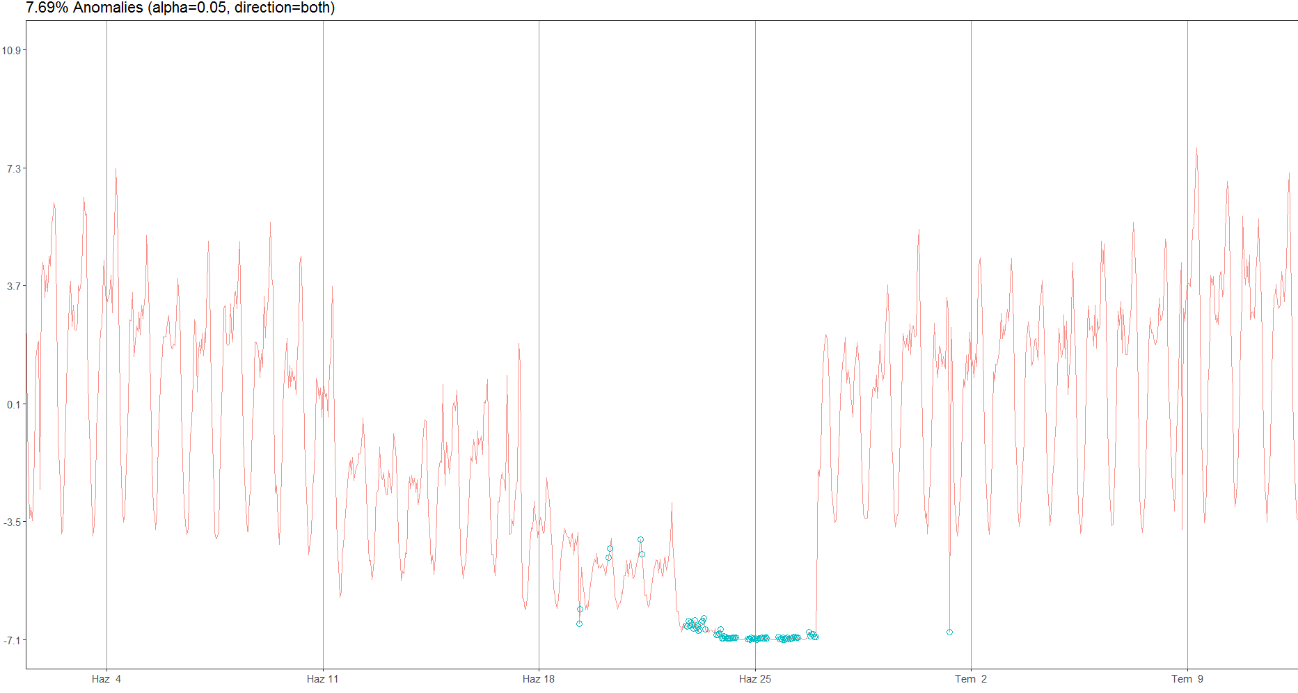
**Figure 5.4: Kalman Filter sample result**



**Figure 5.5: Time Series Decomposition sample result**



**Figure 5.6: Twitter library sample result**



Twitter library produces different results with implemented algorithms in this study. Figure 5.6 shows result of library with sample set. It detects some of collective anomaly records as anomalous but not all of them. It is because, it considers that part as steady signal and small fluctuations cause them to be labeled anomalous. More steady signal in collective anomaly part may cause them to be not detected. This experiment is left as a future work.

Twitter library also detect only one contextual anomaly although there are two and produces some false positives which is similar with ARIMA and Time Series Decomposition.

# **CONCLUSION**

Motivation of this study was detecting anomalous instances in time series data with high positive accuracy and proposing a method that can generate robust results for all anomaly types in Section 1.2. To achieve this task, Moving Average, ARIMA, Kalman Filter and Time Series Decomposition techniques were implemented based on sliding window algorithm. Key of this algorithm is to find a confidence interval that next observation most likely will be in.

The goal is partially succeeded, since, only point and contextual anomaly types are detected with high accuracy. Also, ensemble of implemented algorithms could not be tested to get an even better results.

For the future of not achieved task, collective anomalies can be determined with a new domain and algorithm since current one was not able to detect with good results. Also, implemented algorithms can be assembled into one algorithm. For example, Kalman Filter is the best point anomaly detector in this study. It can be merged with ARIMA that can handle contextual anomalies. One algorithm that can run Kalman Filter for point anomalies and ARIMA for contextual type can produce high accuracy results. Since each algorithm has different strengths, a voting schema can be implemented to get more robust results.

Time series decomposition algorithm still has the challenge to determine change points between seasonal and normal behavior. Algorithm is tested with manually separated windows and it is observed that it can produce good results. Thus, it will be important to automatize it.

Detecting collective anomalies is not a successful part of this study. It differs from other anomaly types for number of instances required to determine them anomalous. While a single instance can be detected as both point and contextual anomaly, collective anomaly needs a group of instances. These instances can be considered as group of contextual anomalies. Algorithms implemented in this study are not capable of detecting such anomalies, because, they adapt existence of anomaly after some point and instances are detected as non-anomalous. Amongst all of algorithms, time series decomposition is the closest one to be able to detect collective anomalies. But, it can consider them as change points and detect as non-anomalous. For these reason, collective anomaly types should be studied exclusively.

In this study, a benchmark dataset of Yahoo! Webscope program is used. For diversity of data and generalization of algorithms, datasets from another source(s) should be included and results should be compared with Yahoo! Dataset.

# **REFERENCES**

***Books***

Kumar, V., N., Steinbach, M, Tan, P. 2014. *Introduction to data mining.*

Dubes. R. C., Jain, A. K. 1988. *Algorithms for Clustering Data*. Prentice-Hall, Inc.

***Periodicals***

Banerjee, A., Chandola, V., Kumar, V. 2009. Anomaly detection: A survey. *ACM Comput. Survey.* 41, 3

Chandola, V., Mithal, V., Kumar, V. 2008. A comparative evaluation of anomaly detection techniques for sequence data. *In Proceedings of International Conference* *on Data Mining*.

Denning, D.E., 1987. An Intrusion Detection Model. *Software Engineering, IEEE Transactions,* 222-32

Axelsson, S., 2000. Intrusion Detection Systems: A Survey and Taxonomy

Chakravarty, S., Dabiri, S., Nayyar, K., Vishwasrao, S. 2016. Anomaly Detection for Univariate Time-Series Data. *ICML 2015*.

Kalman, R., E. 1960. A new approach to linear filtering and prediction problems. *J. Basic Eng.* 82, 1.

Ahmed, M., Mahmood, A. 2014. Network traffic analysis based on collective anomaly detection. *IEEE 9th conference on industrial electronic sand applications (ICIEA)*. 1141–46.

Agyemang, M., Alhajj, R., Barker, K. 2006. A comprehensive survey of numeric and symbolic outlier mining techniques. *Intell. Data Anal.* 521–538.

Keogh, W., Wei, L., Xi, X. 2006. Saxually explicit images: Finding unusual shapes. *ICDM ’06. Sixth International Conference*. 711–720.

Fox, A. J. 1972. Outliers in time series. *Journal of the Royal Statistical Society. Series B(Methodological).* 34(3):350–363

Ma, J., Perkins, S. 2003. Time-series novelty detection using one-class support vector machines. 1741–1745.

Austin, J., Hodge, V. 2004. A survey of outlier detection methodologies. *Artif. Intel. Rev*. 22, 2, 85–126.

Aleskerov, E., Freisleben, B., Rao, B. 1997. Cardwatch: A neural network based database mining system for credit card fraud detection. *In Proceedings of the IEEE Conference on Computational Intelligence for Financial Engineering*. 220–226.

Herle, H. V., Keogh, E., Lee, S.H., Lin, J. 2006. Finding the most unusual time series subsequence: Algorithms and applications. *Knowl. Inform. Syst*. 11, 1, 1–27.

Chiu, B. Y., Keogh, E., Lonardi, S. 2002. Finding surprising patterns in a time series database in linear time and space. *In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press.* 550–556.

Janakiram, D., Kumar, A., Reddy, V. 2006. Outlier detection in wireless sensor networks using Bayesian belief networks. *In Proceedings of the 1st International Conference on Communication System Software and Middleware*. 1–6.

Kumar, V. 2005. Parallel and distributed computing for cybersecurity. *IEEE Distrib. Syst. Online.* 6, 10.

Parra, L., Sajda, P., Spence, C. 2001. Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model. *In Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis. IEEE Computer Society*.

Fujimaki, R., Machida, K., Yairi, T. 2005. An approach to spacecraft anomaly detection problem using kernel feature space. *In Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. ACM Press, 401–410.

Chen, K., Lu, S., Teng, H. 1990. Adaptive real-time anomaly detection using inductively generated sequential patterns. *In Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy*. IEEE Computer Society Press, 278–284.

Leroy, A. M., Rousseeuw, P. J. 1987. Robust Regression and Outlier Detection.

Markou, M. Singh, S. 2003a. Novelty detection: A review-part 1: Statistical approaches. *Sig. Proc*. 83, 12, 2481–2497.

Markou, M. Singh, S. 2003b. Novelty detection: A review-part 2: Neural network based approaches. *Sig. Proc*. 83, 12, 2499–2521.

Edgeworth, F. Y. 1887. On discordant observations. *Philosoph. Mag*. 23, 5, 364–375.

Hochenbaum, J., Kejariwal, A., Vallis, O. *Twitter Inc.*

Aminikhanghahi, S., Cook, J. D. 2007. A Survey of Methods for Time Series Change Point Detection. *Knowl Inf Syst.* 339–367.

Malhotra, P., Shroff, G., Vig, L. 2015. Long short term memory networks for anomaly detection in time series. *23rd European Symposium on Artificial Neural Networks, Computer Intelligence and Machine Learning*.

Eyben, F., Marchi, E., Schuller, B, Squartini, S., Vesperini, F. 2015. A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional lstm neural networks. *IEEE International Conference on Acoustics, Speech and Signal Processing.* 1996-2000.

Clifton, D.A., Clifton, Pimentel, M.A.F., L., Tarassenko, L. 2014. A review of novelty detection. *Signal Processing.* 215-249.

Ahmed, M., Mahmood, A.N., Hu, J. 2016. A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* 19-31.

Bontemps, L., Cao, V.L., Le-Khac, NA., McDermott, J. 2016 Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Networks. *Future Data and Security Engineering. FDSE 2016. Lecture Notes in Computer Science, vol 10018*. 141-152.

Akouemo, H.N., Povinelli, R.J. 2015. Probabilistic anomaly detection in natural gas time series data. *Int. J. Forecast.* 948-956.

Bianco, A.M., García B. M., Martínez, E.J., Yohai, V.J. 2001. Outlier detection in regression models with ARIMA errors using robust estimates. *J. Forecast.* 565-579.

Hyndman, R.J., Khandakar, Y. 2008. Automatic time series forecasting: the forecast package for R Automatic time series forecasting: the forecast package for R. *J. Stat. Softw.* 1-22.

Rinehart, A.W., Simon, D.L. 2014. A model-based anomaly detection approach for analyzing streaming aircraft engine measurement data. *Proceedings of Turbo Expo 2014: Turbine Technical Conference and Exposition, ASME.* 665-672.

Alcock, C., Brodley, C.E., Protopapas, P., Rebbapragada, U. 2009. Finding anomalous periodic time series: An application to catalogs of periodic variable stars. *Mach. Learn.* 281-313.

Bäck, T., Konen, W., Thill, M. 2017. Online anomaly detection on the Webscope S5 dataset: A comparative study. *Evolving and Adaptive Intelligent Systems.*

Lee, H., Roberts, S.J. 2008. On-line novelty detection using the Kalman filter and extreme value theory. *Proceedings of the 19th International Conference on Pattern Recognition.* 1-4.

Hyndman, R.J., Wang, E., Laptev, N. 2015. Large-Scale Unusual Time Series Detection. *15th International Conference on Data Mining Workshops.* 1616–1619.

Chae, D. H., Choi, S, Kang, H.G, Suh, S. 2016. Echo-state conditional variational autoencoder for anomaly detection. *Neural Networks (IJCNN), 2016 Int. Joint Conf. on. IEEE*. 1015–1022.

Candes, E.J., Li, X., Wright, J., Yi Ma, Y. 2009. Robust Principal Component Analysis.

Walfish, S. 2006. A review of statistical outlier methods. *Pharmaceutical technology*.

Fung, K.Y., Paul S.R. 199. A generalized extreme studentized residual multiple-outlier-detection procedure in linear regression. *Technometrics*.

Leith, D. J., Knorn, F. 2008. Adaptive Kalman filtering for anomaly detection in software appliances. *IEEE Conference on Computer Communications Workshops*. 1–6.

***Other Publication***

Yahoo! Webscope dataset. <http://labs.yahoo.com/Academic_Relations> [accessed 6 November 2017]

Forecast R Package. <https://cran.r-project.org/web/packages/forecast/index.html> [accessed 12 December 2017]

Time Series Components. <https://www.otexts.org/fpp/6/1> [accessed 1 April 2018]

Twitter Engineering: Introducing Practical and Robust Anomaly Detection in a Time Series. <https://blog.twitter.com/engineering/en_us/a/2015/introducing-practical-and-robust-anomaly-detection-in-a-time-series.html>. [accessed 23 March 2018]