CIplus
Band 9/2020

EventDetectR – An Open-Source Event Detection System

Sowmya Chandrasekaran, Margarita Rebolledo,
Thomas Bartz-Beielstein

Technology
Arts Sciences
TH Köln

EventDetectR- An Open-Source Event Detection System

Sowmya Chandrasekaran, Margarita Rebolledo, Thomas Bartz-Beielstein
Institute for Data Science, Engineering, and Analytics, TH Köln
Steinmüllerallee 1, 51643 Gummersbach, Germany

CONTENTS

I Introduction 3

II Background of the Research Project 4
- II-A Open Water Open Source(OWOS) Project . . . 4
- II-B Drinking Water Supply and Distribution Systems . . . 5
- II-C What is an Outlier, Event, and a Baseline Change? . . . 5
- II-D Challenges faced by an Event Detection System . . . 6

III Background on Event Detection Algorithms in Drinking Water Systems 6

IV EventDetectR- R Package 8
- IV-A The Event Detection Phases . . . 10
  - IV-A1 Data Pre-Processing . . . 10
  - IV-A2 Model Building: Event Detection Algorithms . . . 12
  - IV-A3 Post-Processing: Residual Classification . . . 13
- IV-B Event Simulation Techniques . . . 14
- IV-C EventDetectGUI- R Package . . . 15

V Performance Measures 15
- V-A Confusion Matrix . . . 15
- V-B Receiver Operating Curve (ROC) . . . 15
- V-C Sensitivity and Specificity . . . 16

VI Case Study 17

VII Summary and Outlook 21

References 21

List of Figures

1 Sensors installed at Monitoring Stations . . . 6
2 Representation of an Outlier, Event and a Baseline Change . . . 7
3 EventDetectR-Process Flow . . . 9
4 EventDetectR-Methodology Flowchart . . . 10
5 EventDetectR-Phases . . . 11
6 Event Simulation on Parameter Leit from geccoIC2018Train Data-set . . . 14
7 An Ideal ROC Curve . . . 16
8 Description of the GECCO2018 Challenge Parameters. . . . 18
9 Resulting Plot for the ForecastArima Model from EventDetectR on the Example Data-set . . . 19
10 ROC Curve for the ForecastArima Model from EventDetectR on the Example Data-set 20

List of Tables

I Available Imputation Techniques . . . 11
II Available Forecasting Models . . . 12
III Confusion Matrix that Summarizes the Event Detection Algorithms Performance . . 15
IV Confusion Matrix for the ForecastArima Model from EventDetectR on the Example Data-set . . . 19

Abstract

EventDetectR: An efficient Event Detection System (EDS) capable of detecting unexpected water quality conditions. This approach uses multiple algorithms to model the relationship between various multivariate water quality signals. Then the residuals of the models were utilized in constructing the event detection algorithm, which provides a continuous measure of the probability of an event at every time step. The proposed framework was tested for water contamination events with industrial data from automated water quality sensors. The results showed that the framework is reliable with better performance and is highly suitable for event detection.

I. Introduction

The quality of drinking water is a major concern as it directly affects public health and safety. Hence the protection of the drinking water supply and distribution systems against accidental or intentional contamination is crucial. The current technological advancements have made more and more affordable online sensors available. The recent rapid breakthroughs in statistical analysis and more low-cost online sensors have enabled the development of Event Detection System (EDS).

EDS is required to analyze the huge volume of water quality data from various monitoring stations, to detect unexpected water quality conditions, and to alarm the operator to handle these situations. Such an EDS can be combined with static set-point or threshold alarms. A clear difference between set-point alarms and EDS is that set-point alarms are simply triggered when a water quality signal breaches the specified control limits. In contrast, EDS is designed to identify the abnormal behavior of the water quality.

In [1], an extensive study about the recent advances in the online drinking water monitoring systems is carried out. Their report conveys that mostly commercially available tools provide reliable performance.

Some of the commercial tools include OptiEDS-OptiWater [2], GuardianBlue Event Detection System by HACH [3], ana::tool an event detection software [4]. Except for commercial software, there are only a few event detection systems like CANARY [5]. Therefore, we intend to develop an open-source Event Detection System that is efficient and is capable of detecting unexpected water quality conditions. The proposed framework can be easily extended as a real-time monitoring system.

The rest of this report is structured as follows: Section II presents an introduction and background to the research project. Then Section III clearly outlines the existing state of the art event detection methods. Thereafter, Section IV explains the proposed event detection approach. It also focuses on training the event detection framework and identifying the appropriate parameter settings. Then in Section V, the suitable performance metrics are discussed. In Section VI a case

study is presented. Finally, a summary and an outlook is presented in Section VII.

II. Background of the Research Project

With the current advancements in measurement technology, affordable online sensors are being installed in the drinking water production and distribution systems. They provide a huge amount of real data for analysis. However, a huge fraction of this data is not effectively used by the operators to gain a better understanding of the current status of their system. This brings in the necessity to devise an EDS that could automatically utilize the available data and infer vital information about the water quality. This can then aid the operators with easy proactive maintenance and can help in providing safe drinking water to the public.

A. Open Water Open Source(OWOS) Project

The OWOS project research focuses mainly on the drinking water quality in Germany, taking into account the effects of climate change, energy efficiency, and protection against environmental disasters and terrorist attacks. As an outcome of this project, the EventDetectR and EventDetectGUI Packages are developed as an open-source project. It focuses on the goal of monitoring the quality of drinking water in Germany.

The major goals of the project are:

- How can we ensure drinking water quality with increasing extreme weather conditions?
- How can drinking water suppliers control their processes in an energy and resource-efficient way?
- How can consumers be protected from the effects of environmental disasters?

In order to handle the above-discussed goals, the project research interests are:

- Process for the determination of trend analysis, long-term water supply forecasting
- Area-specific adaptation of the monitoring networks and monitoring programs, thus enabling reliable estimates of changes in water quality
- Methods for intelligent resource management
- Network monitoring to adapt the individual drinking water process

- Optimization of the sensor placement in order to better detect disturbances, drift, and failures
- Optimization of the drinking water network
- Simulation of different contamination scenarios
- Development of an event detection system
- Providing an event simulator
- Development of an alarm system

B. Drinking Water Supply and Distribution Systems

The drinking water supply and distribution systems generally include storage tanks, pipes, pumps, valves, reservoirs, meters, fittings, and other hydraulic appliances. Their network distribution ranges from source water well to a water treatment plant and traverses through various complex channels until it reaches the end customers' tap. Along with the system, the possibility of contamination injection into the distribution system either intentionally or unintentionally is significant. Fool-proof physical securing of the complete distribution network is not realizable. A reliable alternative is the installation of monitoring stations to measure the water quality with the help of sensors located at each monitoring station, as shown in Figure 1.

C. What is an Outlier, Event, and a Baseline Change?

An event can be defined as a significant deviation of one or more water quality parameters from their normal behavior for more than a specific period. As it can be seen in Figure 2, a significant deviation of a water quality parameter at a single time step is an outlier. This is mostly due to a short-term measurement problem (e.g., air bubbles in the line) and should not lead to an alarm message. When multiple outliers occur over a specific period, then it is defined as an event. An event is an actual deviation from normal behavior that is to be reported. The deviation must last for a certain period. The reported deviation from normal behavior does not necessarily have to represent an issue in the quality of the water. Changes in operation or maintenance should also be reported as a deviation. After a change in certain operating parameters, a baseline change is said to have occurred. A baseline change is a new normal behavior. In such a case, no more events should be reported. These time limits for events and baseline changes differ from utility to utility and are user-defined.

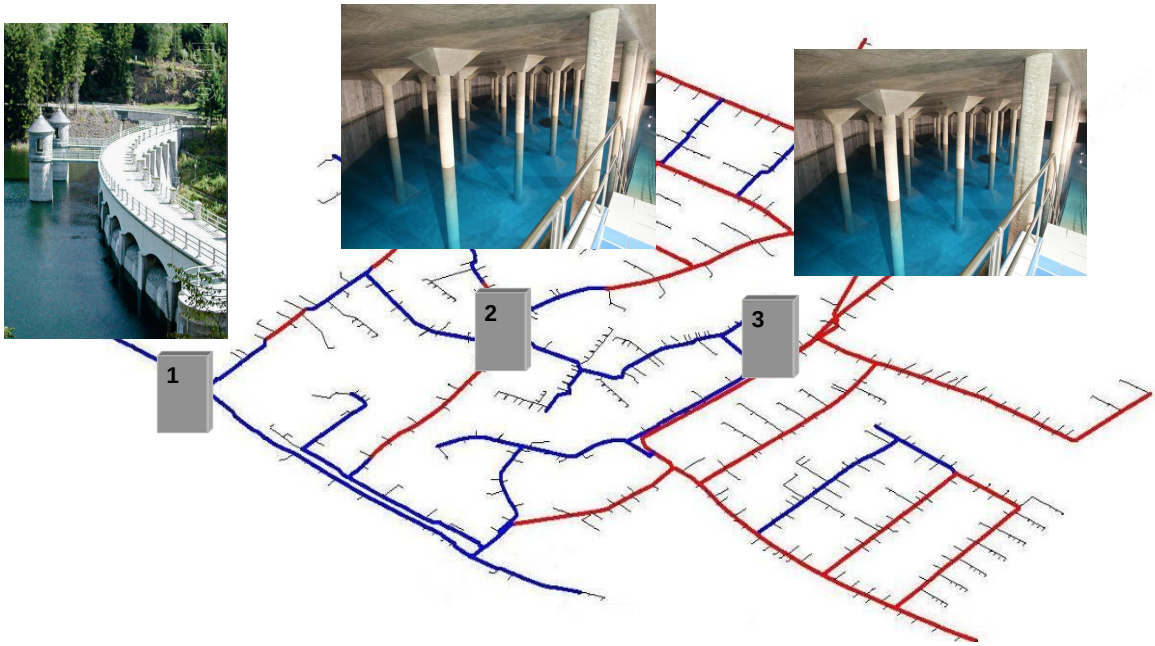


Fig. 1. Sensors installed at various Monitoring Station in the Water Distribution Network

D. Challenges faced by an Event Detection System

The main objective of an EDS is to capture the unexpected behavior as quick as possible and alert the operator. The major challenge is the rate of false alarms, which should be kept at a minimum. There is always a trade-off required between accuracy, false alarm, and quick signaling of unexpected behavior. The following challenges may lead to a false alarm:

- Differentiating a contamination event from a naturally deviating water quality characteristic
- Accurate testing of an EDS is very complicated
- Malfunctioning of a sensor due to various external or internal factors.
- Data transmission issues which may lead to incomplete data

III. Background on Event Detection Algorithms in Drinking Water Systems

The use of data mining algorithms for event detection in drinking water quality control can be traced back to [6] and [7]. A Linear Prediction Coefficient Filter (LPCF) algorithm was introduced for water quality event detection. It predicts the water quality at a future time step using recent observations. It uses an Auto Regressive (AR) model and then evaluates the residuals

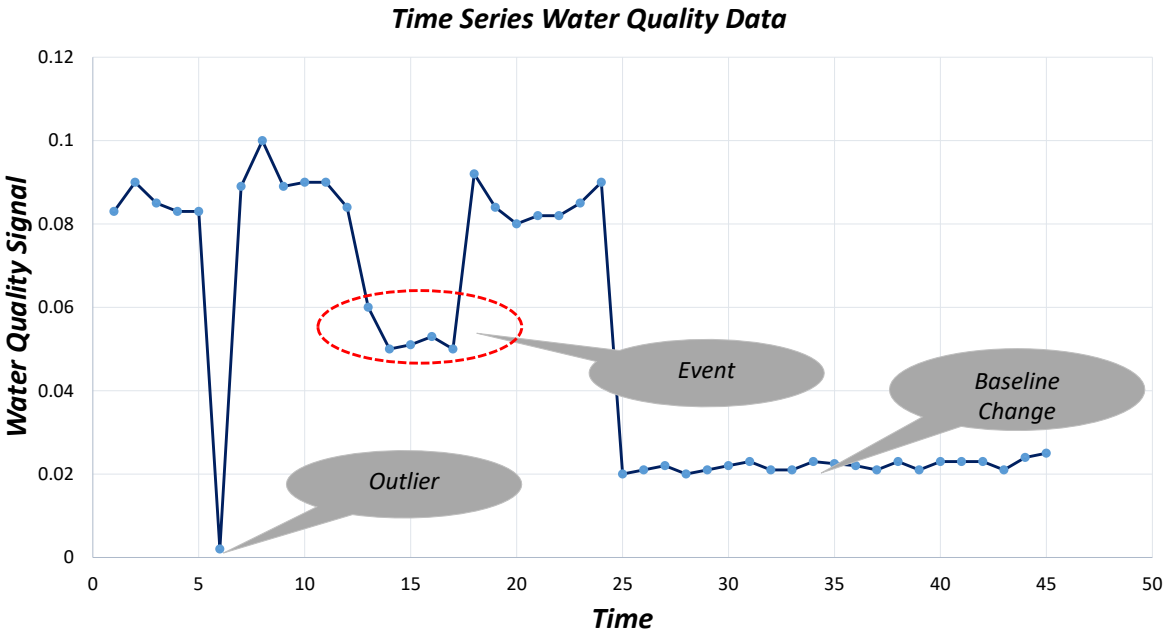


Fig. 2. Representation of an Outlier, Event and a Baseline Change

between predicted and measured water quality values. When the threshold value for the residuals is exceeded, events are triggered. Also, Multivariate Nearest Neighbour (MVNN) algorithm is used to classify the current observation as normal or anomalous by calculating multivariate Euclidean distance was developed.

In [8], the use of the Statistical Process Control (SPC) techniques that deal with turbidity data is demonstrated. The issue of auto-correlation and its effect on the performance of SPC was extensively discussed. The usage of ARIMA model residuals for the construction of SPC charts was recommended.

CANARY was introduced as an online streaming event detection tool based on MATLAB that can process data from multiple sensor locations [5]. Algorithms in CANARY include Set Point Proximity Exponential (SPPE) and Set Point Proximity Beta (SPPB) algorithms, used for the proximity of the event calculation. It also provides options to combine either of the set-point

algorithms with LPCF or MVNN using Consensus MAXimum (CMAX) or Consensus AVErage (CAVE) algorithms. The CMAX algorithm takes the maximum probability from both algorithms that are combined, while CAVE algorithm averages the results of both. One existing issue with CANARY is altering the threshold dramatically changes the performance of the event detection and the ability to detect anomalous water quality signals.

The use of ANN for classification of water quality signals into normal and anomalous classes is performed in [9]. Here, ANN is used to detect possible outliers. Then the probability of an event is updated using Bayes' rule. The probabilities are used for identifying anomalous behavior. An improvement for [9] was proposed in [10], which involved dynamic threshold limits. In [11], an event detection technique based on multi-sensor fusion using an extended Dempster Shafer (DS) method was proposed. Initially, an AR model is used to predict future time step values, then probabilities are assigned to the residuals obtained at each time step. Then DS fusion method searches for anomalous probabilities of the residuals.

A classification-based approach for event detection is carried out in [12], where support vector machines were employed to detect outliers, and sequential analysis is carried out to classify events.

In [13], clustering based analysis is implemented to classify the type of contaminant present in the water systems. The Mahalanobis distance of the water quality vectors classifies the type of contaminant in the water distribution systems based on the similarities of sensors responses to predefined classes.

A cloud based EDS is presented in [14]. In [15], various approaches for monitoring drinking water in real time are presented. Related studies were also recently carried out in [16], [17], [18].

IV. EventDetectR- R Package

The EventDetectR Framework detects events in a multi-dimensional time-series data. For each incoming new water quality parameter, the framework has to classify whether this multivariate measurement is a contamination event or not. This is done through various steps using model

residuals, thresholds, and the probability of an event.

The event detection process flow can be visualized in Figure 3 and Flowchart 4. The $windowSize$ is the sliding window, which represents the number of past observations used to build the model at each step. At each classification iteration, a window of $windowSize$ datapoints (shown in blue) enters into the data preparation phase and the older ones exit in a first in first out fashion. Then, a suitable model is fitted and used to predict the next $nIterationsRefit$ data into the future. The real data (shown in red) is then compared to the prediction and the difference between them is obtained as residuals. These residuals, together with the specified classification thresholds, are used to decide which data is considered an outlier and which is considered as normal behavior. BED is then employed to determine the probability of an event. After the classification, the window is moved by $nIterationsRefit$ rows in the data, and the procedure is repeated until the end of the time-series is reached, and thus all elements are classified.

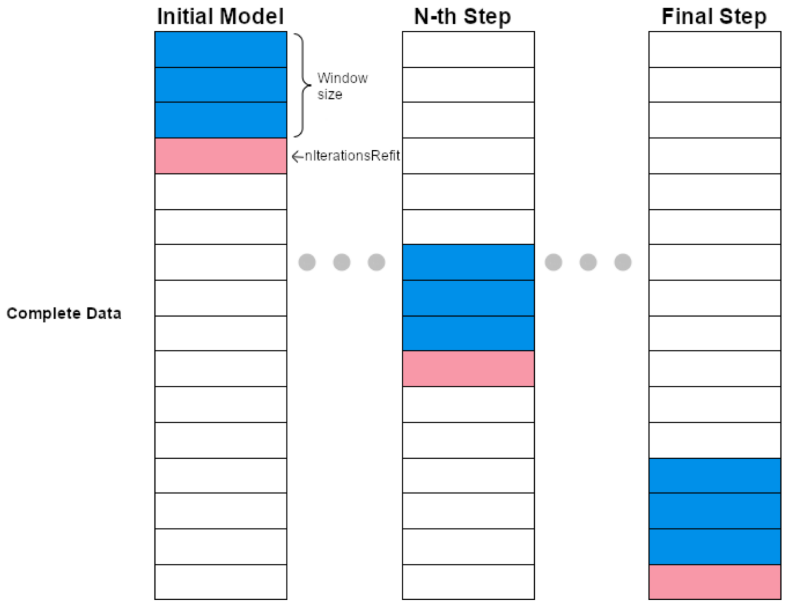


Fig. 3. EventDetectR-Process Flow

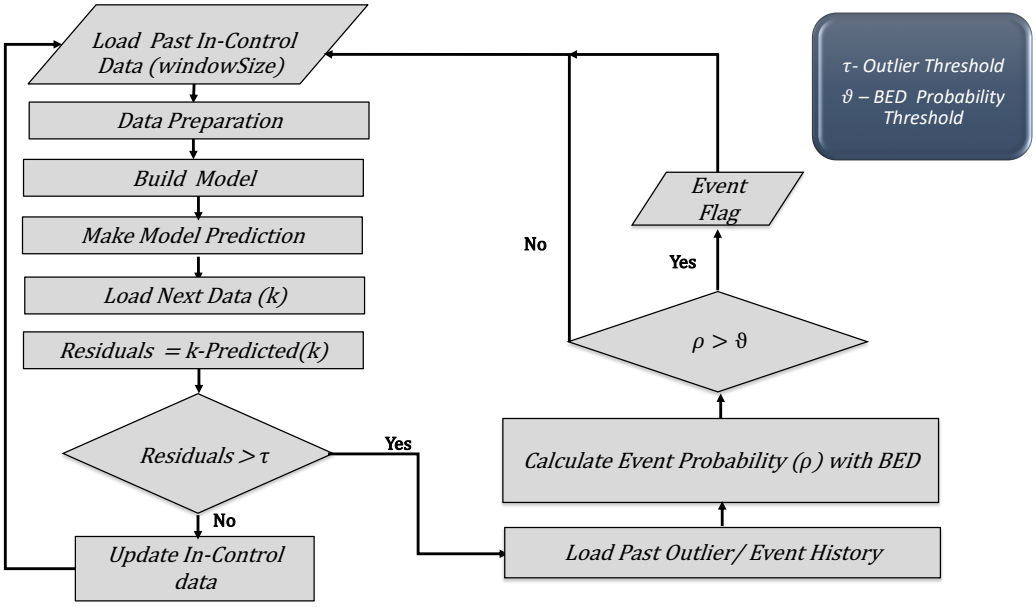


Fig. 4. EventDetectR- Methodology Flowchart

A. The Event Detection Phases

The event detection process is carried out as a three phase approach, namely, the data preparation phase, model building phase, and the post processing phase as shown in Figure 5.

1) Data Pre-Processing: The handling of missing values is particularly an important task for event detection. If the measured values are missing, machine learning models can sometimes not be trained, or the predictions are worse. Due to various challenges in sensor transmission and data acquisition, missing values are often found in the data.

In [19], the authors state that there exists no single univariate imputation technique that is suitable for all types of data patterns. Hence based on the observed data pattern, a suitable imputation technique has to be performed. To cater to this purpose, ImputeTS package [20] is utilized to perform imputation. The list of available imputation techniques is in Table I.

An additional data preparation step is data normalization. Normalization is a scaling method

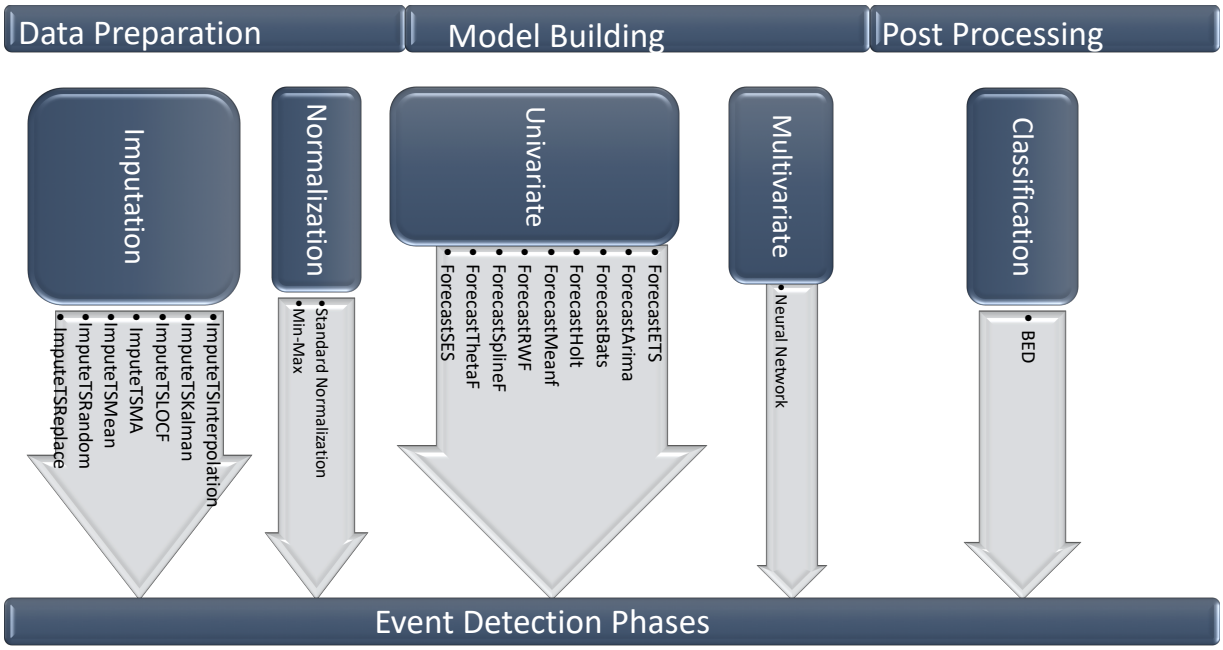


Fig. 5. EventDetectR-Phases

TABLE I. AVAILABLE IMPUTATION TECHNIQUES

Imputation Technique	Description
ImputeTSInterpolation	Imputation by Interpolation
ImputeTSKalman	Imputation by Kalman Smoothing and State Space Models
ImputeTSLOCF	Imputation by Last Observation Carried Forward
ImputeTSMA	Imputation by Weighted Moving Average
ImputeTSMean	Imputation by Mean Value
ImputeTSReplace	Imputation by a Defined Value

that is used to transform the values of various parameters into a single common scale i.e., within a specific range. The Z-score standardization is available, which transforms the raw data into a scale of zero mean and standard deviation one. If neural networks are going to be used in the modeling step, the data can be transformed with min-max normalization. Such kind of normalizing helps in preventing the network from ill-conditioning. In essence, this normalization is done to have the same range of values for each of the inputs to the model. This guarantees stable convergence of weight and biases.

TABLE II. AVAILABLE FORECASTING MODELS

Model Name	Description
ForecastETS	Forecasting using ETS models
ForecastArima	Forecasting using ARIMA or ARFIMA models
ForecastBats	Forecasting using BATS and TBATS models
ForecastHolt	Forecasting using Holt-Winters method
ForecastMeanf	Mean Forecasting method
ForecastRWF	Naive and Random Walk Forecasting method
ForecastSplineF	Cubic Spline Forecasting method
ForecastThetaf	Theta forecasting method
ForecastSES	Exponential smoothing forecasting method

2) Model Building: Event Detection Algorithms: EventDetectR supports univariate and multivariate modeling of water quality time series data. The field of univariate time series forecasting is well explored in [21]. In [22], various forecasting algorithms are implemented and made available as an open-source R package. Considering the efficiency of these forecasting models for time series data, we intend to utilize these models to predict the quality of drinking water. Nine different forecast models are made available, as listed in Table II. Based on the pattern of the time series data, appropriate forecasting models can be chosen.

Also included is a multivariate neural network algorithm that models the water quality parameters through nonlinear, weighted, parametric functions. The advantage of this model is that it requires less formal statistical training. A neural network model is formulated for each of the water quality parameters using its own lagged values and the rest of all parameters involved. The error is back-propagated to the network, and the weights are adjusted back in order to reduce the error with each iteration. The neuralnet package is used for this purpose [23]. For instance, considering 3 parameters $p1$, $p2$, $p3$, the value of $p1$ at time step i is calculated as $p1[i] = p2[i] + p3[i] + p1[i-1]$ and the value of $p2$ at time step i is calculated as $p2[i] = p1[i] + p3[i] + p2[i-1]$. Finally, the value of $p3$ at time step i is calculated as $p3[i] = p1[i] + p2[i] + p3[i-1]$.

The EventDetectR framework is developed in a generic fashion, allowing other similar models to be easily integrated.

3) Post-Processing: Residual Classification: The residuals from the model determines if each trial is a success or a failure. When residuals are less than the user-defined outlier threshold τ, it refers to success or an inlier, whereas if residuals are greater than the outlier threshold τ, then it refers to a failure or an outlier. After this classification at each time step, the Binomial Event Discriminator (BED) is employed to obtain the continuous probability of an event (ρ) from the count of outliers in a specific interval [5]. This specific interval is termed as a window, $BEDWindowSize$, the size of which is defined based on the user requirement. Most commonly, $BEDWindowSize$ is smaller than the sliding window $windowSize$. When ρ exceeds the user-defined BED probability threshold ϑ, then it signifies the occurrence of an event.

The binomial distribution is a discrete distribution that yields the probability of the number of success in a sequence of n independent trials.

The probability that the water quality represents expected normal behavior in n trials is represented as

$$B(r; n, p) = \frac{n!}{r!(n-r)!} p^r q^{(n-r)}, \tag{1}$$

where the n trials is given by our BED window size $BEDWindowSize$, q represents the probability that a trial succeeds and p represents the probability that a trial fails as an outlier. We keep the value of both p and q as 0.5 and hence equation 1 is simplified to

$$B(r; n, p) = \frac{n!}{r!(n-r)!} 0.5^n.$$

In order to increase the probability of failure p with an increase in the count of failures, the cumulative distribution function of the binomial distribution is employed as

$$P(r \leq r_t) = \sum_{i=0}^{r_t} B(r; n, p),$$

where r_t, is the probability threshold value. The advantage of this BED is that it helps in reducing the false alarm. However, as a result, there is some delay in the identification of a true event.

B. Event Simulation Techniques

Validating the event detection algorithms is a challenging task. In most cases, there are no data on actual events, or the occurrence of actual events is rare. Accordingly, real test data is rare. Hence, an easier solution would be computer generated events. The package supports four different types of events. Namely, Sinusoidal, Ramp, Slow-sinusoidal and Square signal events. The strength and duration of the event can be controlled by the user. These artificial events are superimposed on the input data and enable the validation of the algorithms as the locations of events are known.

The usage of the function can be illustrated using the data-set *geccoIC2018Train* included on the package. An inconsistency following a square pattern can be introduced as shown in Figure 6. It can be seen that from time index 2000 to 2500, the events are superimposed (showed in red).

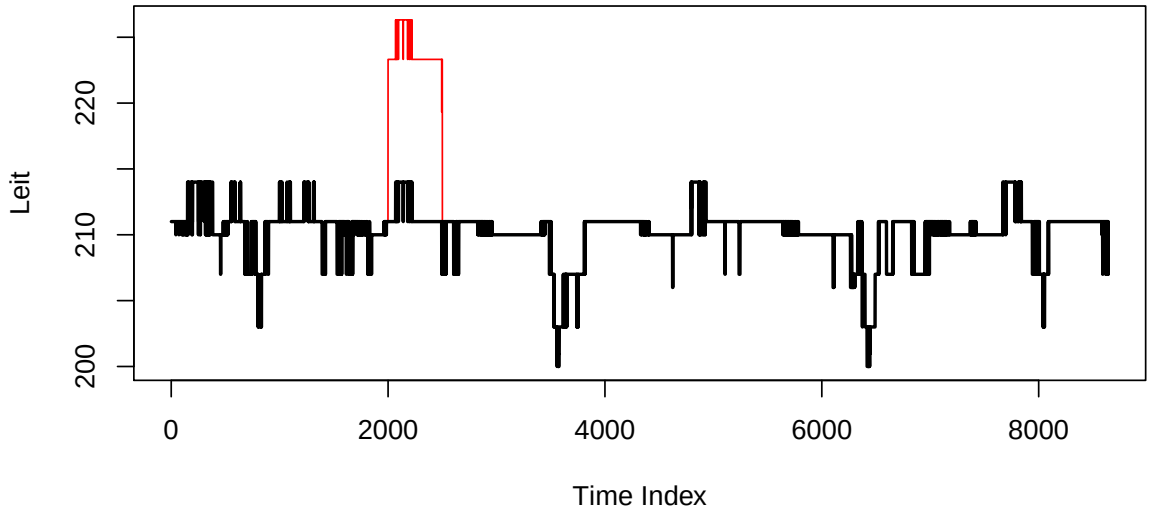


Fig. 6. Event Simulation on Parameter Leit from geccoIC2018Train Data-set

TABLE III. CONFUSION MATRIX THAT SUMMARIZES THE EVENT DETECTION ALGORITHMS PERFORMANCE

	Actual Event	No Actual Event
Predicted Event	TP	FP
Predicted No Event	FN	TN

C. *EventDetectGUI- R Package*

A graphical user interface (GUI) for the EventDetectR package is also available. It enables the use of the EventDetectR package for a wide range of users, mainly for practitioners without R programming skills. The user can load data into the system and visualize the data with interactive plots. After understanding the data, the user can interactively configure the desired algorithms from EventDetectR through the GUI. The event detection results are then available to the user in a tabular fashion. Interactive graphical visualization of the results is also enabled. Additionally, the results can be exported to other formats.

V. PERFORMANCE MEASURES

In order to evaluate the performance of the event detection algorithms, the Confusion Matrix and the Reciever Operating Curve are recommended as the most suitable performance metrics.

A. *Confusion Matrix*

The confusion matrix is formulated, as shown in Table III. Events that are correctly detected are the true positives (TP). True events that are not detected are the false negatives (FN). Non-events that are classified as events (false alarms) represents the false positives (FP). Non-events that are not declared as events are the true negatives (TN).

B. *Receiver Operating Curve (ROC)*

A considerable trade-off is required between the numbers of TP and FP. The ROC curve is used to visualize this trade-off between the True Positive Rate (TPR) and the False Positive Rate (FPR). For a perfect event detection algorithm, irrespective of the value of the threshold, the TPR should be equal to one, and FPR should be zero. This is shown in Figure 7. The area under

the curve, as shown in the plot is one, which is the maximum limit. The 45-degree line from the lower left to the upper right signifies the worst performing algorithm, which is the same as that of the random guess.

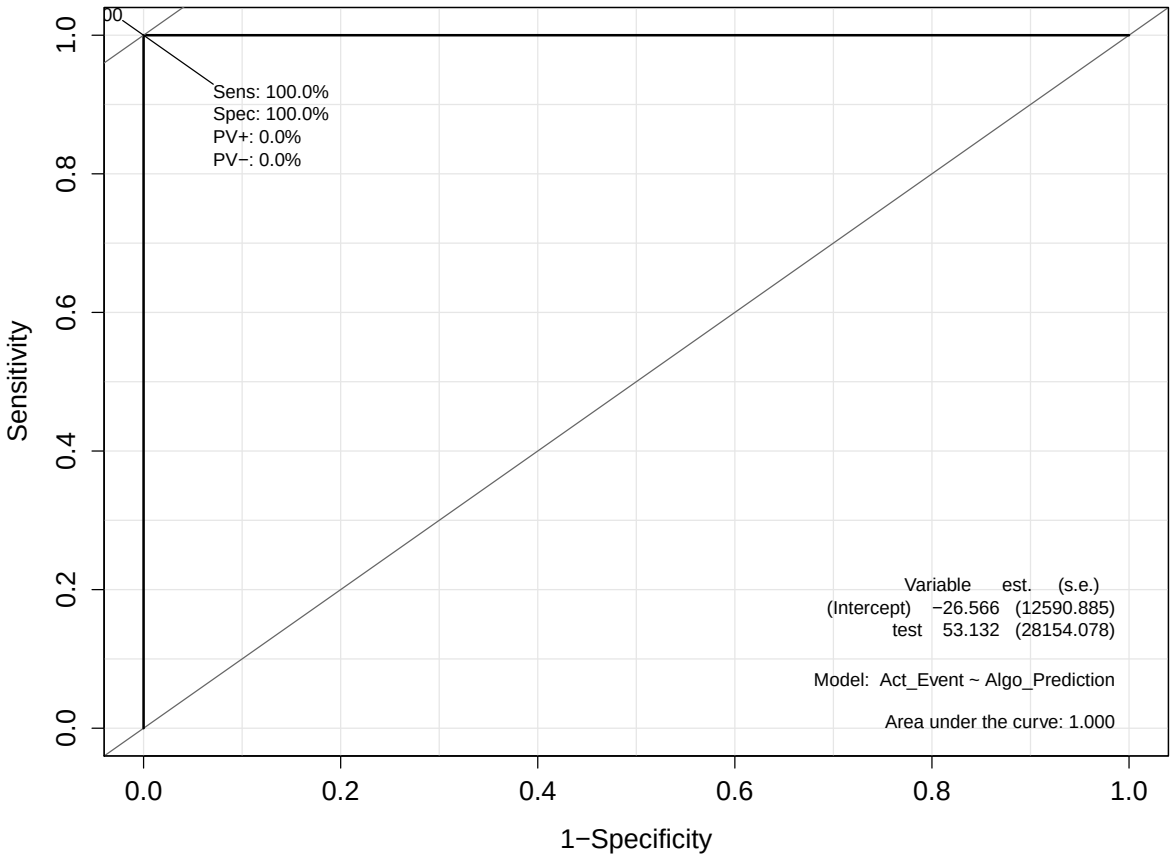


Fig. 7. An Ideal ROC Curve

C. Sensitivity and Specificity

Sensitivity (or TPR) measures the ability of the algorithm to correctly identify an event among all the actual events. Specificity measures how accurate the event detection algorithm is against false positives or false alarms. The Accuracy, TPR rate or sensitivity, specificity, and FPR are

obtained as below

$$Accuracy = \frac{(TP + TN)}{(P + N)}, \tag{2}$$

$$TPR = Sensitivity = \frac{TP}{P}, \tag{3}$$

$$FPR = \frac{FP}{N}, \tag{4}$$

$$Specificity = 1 - FPR, \tag{5}$$

$$\tag{6}$$

where P denotes the total number of events in the data-set and N denotes the total number of non-events.

VI. Case Study

The availability of a data-set for evaluating the performance of an event detection algorithm is very difficult. As a step to help the research community, the real data-set from the water industry is made available in the GECCO 2017 Industrial Challenge [24] and the GECCO 2018 Industrial Challenge [25]. Some of these data-sets are also included as a part of the EventDetectR package.

Let us consider the data from GECCO 2018 Industrial Challenge [25]. Figure 8 describes the water quality data-set, which contains minutely sensor data. It consists of the amount of chlorine dioxide in the water, its pH value, the redox potential, its electric conductivity and the turbidity of the water. These are water quality parameters and any change in these is considered as an event. The flow rate and the temperature of the water are considered as operational data. Changes in these values may indicate changes in the related quality values but are not considered as events.

For demonstration purposes, let us consider a small time frame of one week from the GECCO 2018 Industrial Challenge data-set [25]. The ForecastArima algorithm is chosen as the model. The configuration in R is done as shown below

```
d_int<- interval(start = as.POSIXct("2016-08-14"),
end = as.POSIXct("2016-08-20"))
```

Column name	Description
Time	Time of measurement, given in following format: yyyy-mm-dd HH:MM:SS
Tp	The temperature of the water, given in °C.
Cl	Amount of chlorine dioxide in the water, given in mg/L (MS1)
pH	PH value of the water
Redox	Redox potential, given in mV
Leit	Electric conductivity of the water, given in μS/cm
Trueb	Turbidity of the water, given in NTU
Cl_2	Amount of chlorine dioxide in the water, given in mg/L (MS2)
Fm	Flow rate at water line 1, given in m^3/h
Fm_2	Flow rate at water line 2, given in m^3/h
EVENT	Marker if this entry should be considered as a remarkable change resp. event, given in boolean.

Fig. 8. Description of the GECCO2018 Challenge Parameters.

```
sgecco<-geccoIC2018Train[geccoIC2018Train$Time %within% d_int , ]
ed_arima <- detectEvents(gecco_data[,c(3,4,5,6,7,8)],windowSize = 1000,
 nIterationsRefit = 5,verbosityLevel = 2,
buildModelAlgo = "ForecastArima",
postProcessorControl = list(nStandardDeviationseventThreshold = 2,
eventThreshold = .7, bedWindowSize =10))
qualityStatistics(ed_arima, sgecco$EVENT)
plot(ed_arima)
```

The results obtained are shown in the form of a confusion matrix in Table IV. Out of 144 actual events in the given time interval, 115 events were identified. There were 1748 false positives, which can be reduced with further tuning of the algorithm parameters and with the choice of other algorithms. The ROC curve is shown in Figure 10. The plot shows 79.9% sensitivity and 79.4% specificity. This indicates better performance. The resulting data plot with marked events is shown in Figure 9. The event positions are marked for each variable in red. This plot enables further analysis to identifying the root cause of the event.

TABLE IV. CONFUSION MATRIX FOR THE FORECASTARIMA MODEL FROM EVENTDETECTR ON THE EXAMPLE DATA-SET

	Actual Event	No Actual Event
Predicted Event	115	1748
Predicted No Event	29	6749

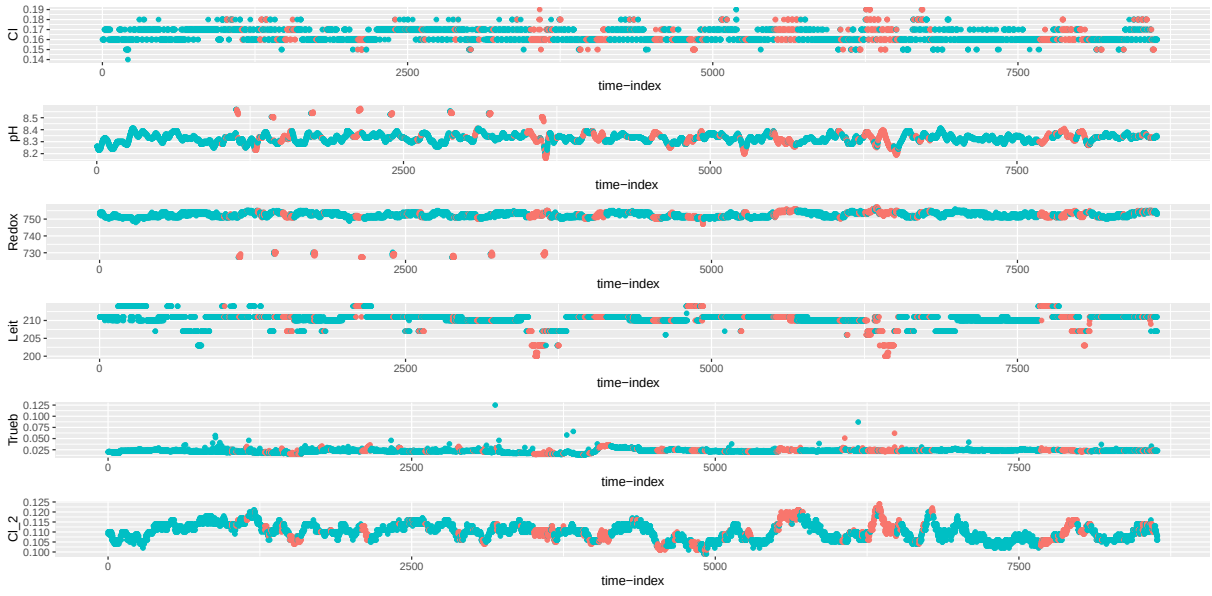


Fig. 9. Resulting Plot for the ForecastArima Model from EventDetectR on the Example Data-set

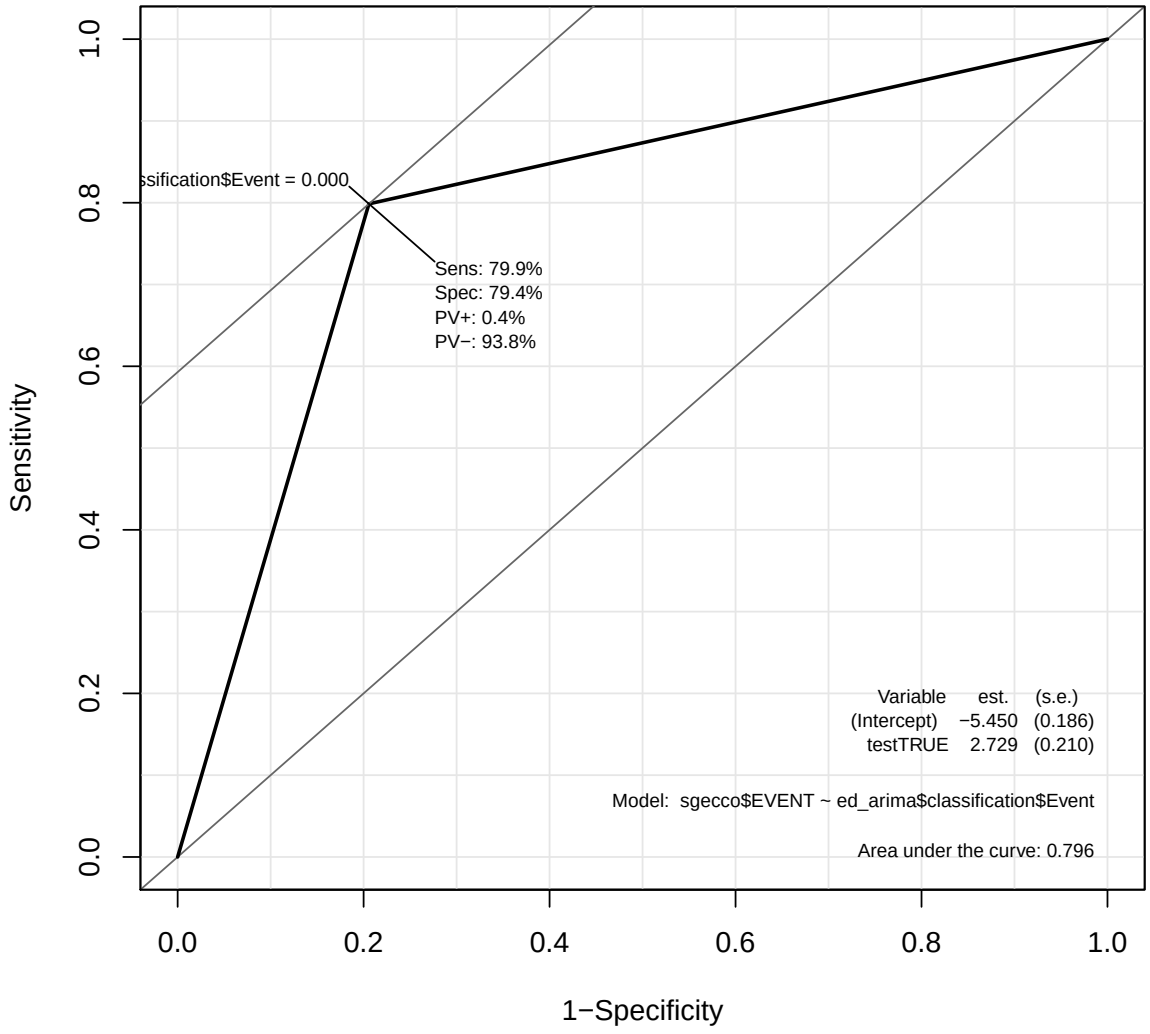


Fig. 10. ROC Curve for the ForecastArima Model from EventDetectR on the Example Data-set

VII. Summary and Outlook

EventDetectR package is capable of simulating, detecting and classifying events in time-series data. It delivers an easily configurable tool for event detection. Wide range of pre-processing techniques and event detection algorithms are available. Based on the nature of the time series data, suitable pre-processing techniques and event detection algorithms can be chosen by the user. Also, the threshold limits can be modified by the user, based on their requirements. The framework is reliable with better performance and is suitable for real time event detection. The EventDetectGUI package, a graphical user interface, enables the usage of the EventDetectR package easier, especially for non-programming users.

Acknowledgments

This research work is funded by project Open Water Open Source(OWOS) (reference number: 005-1703-0011) and is kindly supported by the FH Zeit für Forschung, Ministerium fur Innovation, Wissenschaft und Forschung des Landes NRW.

References

[1] M. V. Storey, B. van der Gaag, and B. P. Burns, "Advances in on-line drinking water quality monitoring and early warning systems," *Water Research*, vol. 45, no. 2, pp. 741–747, 2011.

[2] A. Ostfeld and E. Salomons, "Optimal layout of early warning detection stations for water distribution systems security," *Journal of Water Resources Planning and Management*, vol. 130, no. 5, pp. 377–385, 2004.

[3] "hach." [Online]. Available: https://www.hach.com/event-detection-and-security/guardianblue-early-warning-system/family?productCategoryId=35547627778

[4] "s::can." [Online]. Available: http://moni-tool.at/index.php/moni-tool/event-detection

[5] R. Murray, T. Haxton, S. McKenna, D. Hart, K. Klise, M. Koch, E. Vugrin, S. Martin, M. Wilson, V. Cruze *et al.*, "Water quality event detection systems for drinking water contamination warning systems—development, testing, and application of canary," *EPAI600IR-lOI036, US*, 2010.

[6] D. Hart, S. A. McKenna, K. Klise, V. Cruz, and M. Wilson, "Canary: a water quality event detection algorithm development tool," in *Proceedings of the World Environmental and Water Resources Congress, ASCE, Reston, VA*, 2007, pp. 1–9.

[7] K. A. Klise and S. A. McKenna, "Water quality change detection: multivariate algorithms," in *Defense and Security Symposium*. International Society for Optics and Photonics, 2006, pp. 62 030J–62 030J.

[8] E. Smeti, L. Kousouris, P. Tzoumerkas, and S. Golfinopoulos, "Statsitical process control techniques on autocorrelated turbidity data from finished water tank," in *Proceedings of An International Conference on Water Science and Technology Integrated Management on Water Resources, AQUA*, 2006.

[9] L. Perelman, J. Arad, M. Housh, and A. Ostfeld, "Event detection in water distribution systems from multivariate water quality time series," *Environmental science & technology*, vol. 46, no. 15, pp. 8212–8219, 2012.

[10] J. Arad, M. Housh, L. Perelman, and A. Ostfeld, "A dynamic thresholds scheme for contaminant event detection in water distribution systems," *Water research*, vol. 47, no. 5, pp. 1899–1908, 2013.

[11] D. Hou, H. He, P. Huang, G. Zhang, and H. Loaiciga, "Detection of water-quality contamination events based on multi-sensor fusion using an extented dempster–shafer method," *Measurement Science and Technology*, vol. 24, no. 5, p. 055801, 2013.

[12] N. Oliker and A. Ostfeld, "A coupled classification–evolutionary optimization model for contamination event detection in water distribution systems," *Water research*, vol. 51, pp. 234–245, 2014.

[13] S. Liu, H. Che, K. Smith, and T. Chang, "A real time method of contaminant classification using conventional water quality sensors," *Journal of environmental management*, vol. 154, pp. 13–21, 2015.

[14] C. Kühnert, M. Baruthio, T. Bernard, C. Steinmetz, and J.-M. Weber, "Cloud-based event detection platform for water distribution networks using machine-learning algorithms," *Procedia Engineering*, vol. 119, pp. 901–907, 2015.

[15] V. Puig, C. Ocampo-Martínez, R. Pérez, G. Cembrano, J. Quevedo, and T. Escobet, *Real-time Monitoring and Operational Control of Drinking-Water Systems*. Springer, 2017.

[16] A. Leow, J. Burkhardt, W. E. Platten III, B. Zimmerman, N. E. Brinkman, A. Turner, R. Murray, G. Sorial, and J. Garland, "Application of the canary event detection software for real-time performance monitoring of decentralized water reuse systems," *Environmental Science: Water Research & Technology*, vol. 3, no. 2, pp. 224–234, 2017.

[17] M. Bilal, A. Gani, M. Marjani, and N. Malik, "A study on detection and monitoring of water quality and flow," in *2018 12th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS)*. IEEE, 2018, pp. 1–6.

[18] F. Muharemi, D. Logofătu, C. Andersson, and F. Leon, "Approaches to building a detection model for water quality: a case study," in *Modern Approaches for Intelligent Information and Database Systems*. Springer, 2018, pp. 173–183.

[19] S. Moritz, A. Sardá, T. Bartz-Beielstein, M. Zaefferer, and J. Stork, "Comparison of different methods for univariate time series imputation in r," *arXiv preprint arXiv:1510.03924*, 2015.

[20] S. Moritz and T. Bartz-Beielstein, "imputets: time series missing value imputation in r." *R J.*, vol. 9, no. 1, p. 207, 2017.

[21] R. J. Hyndman, Y. Khandakar *et al.*, *Automatic time series for forecasting: the forecast package for R*. Monash University, Department of Econometrics and Business Statistics ..., 2007, no. 6/07.

[22] R. Hyndman, G. Athanasopoulos, C. Bergmeir, G. Caceres, L. Chhay, M. O'Hara-Wild, F. Petropoulos, S. Razbash, E. Wang, and F. Yasmeen, "forecast: Forecasting functions for time series and linear models," Apr. 2018.

[23] S. Fritsch, F. Guenther, and M. F. Guenther, "Package 'neuralnet'," *The Comprehensive R Archive Network*, 2016.

[24] S. Chandrasekaran, M. Rebolledo, M. Friese, S. Jörg, and T. Bartz-Beielstein, "Gecco 2017 industrial challenge: Monitoring of drinking-water quality," 2017.

[25] F. Rehbach, S. Moritz, S. Chandrasekaran, M. Rebolledo, M. Friese, and T. Bartz-Beielstein, "Gecco 2018 industrial challenge: Monitoring of drinking-water quality," 2018.

Kontakt/Impressum

Diese Veröffentlichungen erscheinen im Rahmen der Schriftenreihe "CIplus". Alle Veröffentlichungen dieser Reihe können unter
https://cos.bibl.th-koeln.de/home
abgerufen werden.

Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.
Datum der Veröffentlichung: 16.11.2020

Herausgeber / Editorship

Prof. Dr. Thomas Bartz-Beielstein,
Prof. Dr. Wolfgang Konen,
Prof. Dr. Boris Naujoks,
Institute for Data Science, Engineering, and Analytics,
Institute of Computer Science,
Faculty of Computer Science and Engineering Science,
TH Köln,
Steinmüllerallee 1,
51643 Gummersbach
url: www.ciplus-research.de

Schriftleitung und Ansprechpartner/ Contact editor's office

Prof. Dr. Thomas Bartz-Beielstein,
Institute for Data Science, Engineering, and Analytics,
Faculty of Computer Science and Engineering Science,
TH Köln,
Steinmüllerallee 1, 51643 Gummersbach
phone: +49 2261 8196 6391
url: http://www.spotseven.de
eMail: thomas.bartz-beielstein@th-koeln.de

ISSN (online) 2194-2870

Technology
Arts Sciences
TH Köln

Technology
Arts Sciences
TH Köln