Iran J Sci Technol Trans Civ Eng (2019) 43:197–214
https://doi.org/10.1007/s40996-018-0131-2

RESEARCH PAPER

# Detection of Outlier in 3D Flow Velocity Collection in an Open-Channel Bend Using Various Data Mining Techniques

Mohammad Vaghefi[1] · Kumars Mahmoodi[2] · Maryam Akbari[3]

Received: 6 July 2017 / Accepted: 4 July 2018 / Published online: 10 July 2018

**Abstract**
Data collection related to the flow pattern has always been associated with outliers due to various reasons. Outlier detection in flow pattern experiments is of high importance and results in a better and more accurate understanding of the flow pattern. In this study, six data mining methods have been used to identify the outliers in flow pattern experiments. The discussed methods include box plot, histograms, linear regression, *k*-nearest neighbors, local outlier factor, *k*-medoids clustering, multilayer perceptron, and self-organizing map. The main aim of this study is to detect the outliers in data collection in order to conduct flow pattern experiments using the data mining methods. These methods have been analyzed and compared with each other in a case study and their performance evaluated. The experimental outliers under investigation were emanated from flow pattern experiments around a spur dike located in a 90° bend using Vectrino velocimeter (ADV). The range of velocity measurement of this device is between ± 0.01 and ± 4 m/s, and measurement accuracy is 1 mm/s. Also, the frequency is set at 50 Hz. The comparisons of different outlier detection methods results demonstrated that the box plot and the local outlier factor methods have the best performance.

**Keywords** Flow pattern · Outlier detection · Data mining · ADV

## 1 Introduction

Understanding the flow pattern has a fundamental role in determining the behavior of its related phenomena. Using experimental and field studies, the anticipation of the flow pattern behavior around the structures located at river bends is possible. Spur dikes are hydraulic engineering structures for preserving the desired water depth, deflecting the main current in the harbor channels and rivers, and protecting river banks. Spur dikes have always been used as an economical way to protect the river banks in their outer banks (Vaghefi et al. 2015b). In order to study the flow pattern around these structures, 3D flow velocities are collected via different velocimeters (Sulaiman et al. 2013; Xiekang and Xingnian 2016), and various parameters including shear stress (Vaghefi et al. 2015a), kinetic energy, and turbulence intensity (Kang 2013) are calculated. Yet by considering human factors and using different devices for data collection or change in measuring conditions, some of the data are collected as outliers (Alih and Ong 2015; Dhhan et al. 2015). Identifying these outliers and reducing their effects in measurements could be effective in presenting the authentic flow pattern. As a result, outlier detection during the data collection for specifying the flow pattern is an undeniable necessity. In previous studies, most researchers examined errors in calculations or in relations obtained from experimental data. Furthermore, they mentioned that the measuring tolerance of the device is the system errors, and the errors incurred in data collection are less likely discussed. Many researchers such as Nikora and Goring (2000), Goring and Nikora

✉ Mohammad Vaghefi
vaghefi@pgu.ac.ir

Kumars Mahmoodi
kumarsmahmoodi@aut.ac.ir

Maryam Akbari
m.akbari@pgu.ac.ir

[1] Department of Civil Engineering, Persian Gulf University, P.O. Box 7516913817, Bushehr, Iran

[2] Department of Marine Engineering, Amirkabir University of Technology, Tehran, Iran

[3] Department of Civil Engineering, Persian Gulf University, Bushehr, Iran

(2002), Cea et al. (2007), Khorsandi et al. (2012), Islam and Zhu (2013), Durgesh et al. (2014), Yafei (2015), and Hejazi et al. (2016) used filtration methods in relation to data cleaning and the separation of normal and raw data. They used such methods on data collection pertinent to flow velocity using Vectrino, which has the same performance as ADV. Results demonstrated that errors in data collection do not have a strong influence on the mean velocity due to the large number of data, whereas calculating the Reynolds shear stresses and other turbulence parameters may cause unrealistic values (Vaghefi et al. 2010; Mahmoodi et al. 2013a, b). Hence, it is necessary to identify errors and correct or remove them from measurements.

This study aims to identify the outliers in data collection in order to conduct flow pattern experiments using conventional data mining methods. Data mining is a branch of computer science that discovers hidden knowledge, patterns, and relationships of valid data, which have been so far unknown using data mining tools (Han and Kamber 2006; Mahmoodi et al. 2013a, b). These methods could be statistical models, mathematical algorithms, and learning methods. Discussed methods include box plot, histogram, linear regression (Shamim et al. 2015), *k*-nearest neighbors (*k*NN) (Yang et al. 2015), local outlier factor (LOF), *k*-medoids clustering (Alarcon-Aquino et al. 2011), multilayer perceptron (Heidari et al. 2016), and self-organizing map (Olawoyin et al. 2013).

In order to evaluate the performance of these methods in detecting outliers, their performance is reviewed in a case study that aims to determine the flow pattern around a T-shaped spur dike located in a 90° bend.

In this study, an outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism (Hawkins 1980). For example, Fig. 1 represents a data set with five outliers that are marked by O1, O2, O3, O4, and O5 labels. As evident, these points were inconsistent with the rest of the samples and fall away from the overall data pattern.

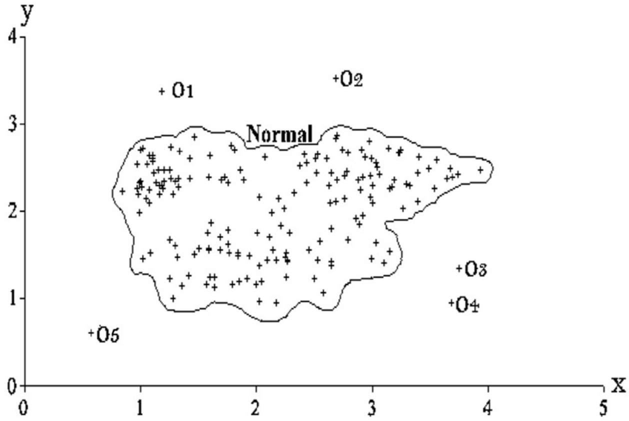

**Fig. 1** A data set with five outliers (O1, O2, O3, O4, and O5)

## 2 Materials and Methods

In this section, case study, data collection devices, discussed methods, and the measurement criteria for precision of methods are introduced.

### 2.1 Case Study

The experimental outliers under investigation were emanated from experiments determining the flow pattern around a single spur dike located in a 90° bend in the Hydraulic Laboratory of Tarbiat Modares University in Iran (Ghodsian and Vaghefi 2009). Figure 2 shows a view of the laboratory and the desired channel. The channel is composed of a 7.1-m-long upstream and 5.2-m-long downstream straight reach, both of which are connected via a 90° bend with an external and internal radius of curvature of 2.7 and 2.1 m, respectively (Vaghefi et al. 2012). The ratio of the curvature radius to channel width is 4, its height is 70 cm and width 60 cm. The channel is made of glass and its stability is maintained by steel frames. The channel bed is rigid and covered with uniform sediment with an average diameter of 1.28 mm and standard deviation of 1.3 mm. The flow discharge is adjusted by a calibrated orifice, is constant, and is equal to 25 l/s in this experiment (Vaghefi et al. 2009). A butterfly gate, installed at the end of the channel, is used to control the flow depth. The Froude and Reynolds numbers are, respectively, 0.34 and 30,120. The rectangular plate spur dike with T-shaped plan is made of Plexiglas. The spur dike used in this experiment is T-shaped. The length of wing (*L*) and that of web (*l*) is equal to 9 cm and is 65 cm in height. This spur dike is vertical and unsubmerged in a 45° position (Vaghefi et al. 2010; Mahmoodi et al. 2013a, b).

### 2.2 Data Collection System

In order to determine the flow pattern, Vectrino velocity meter is used to collect 3D velocities. Vectrino is the new generation of ADV and is an advanced device of its kind used in laboratory researches on account of its high accuracy of velocity measurement and most importantly, its ability to measure the flow velocity in three-dimensional coordinates. This device is formed of two main parts: sensor and cylindrical case (Nortek 2004). Measuring the flow velocity 5 cm away from the sensor tip is one of the characteristics of this device. For this reason, the side-looking sensor measures the velocity near the water surface, while the down-looking sensor is used at other layers.

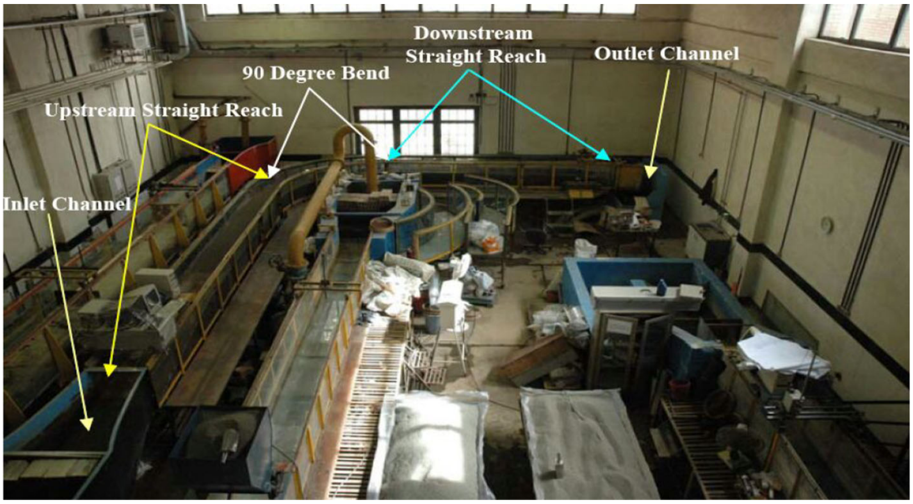

**Fig. 2** A view of the laboratory and the channels

Both the placement of this device on the channel and its two sensors are illustrated in Fig. 3. The range of velocity measurement of this device is between ± 0.01 and ± 4 m/s, and measurement accuracy is 1 mm/s. The frequency is between 50 and 200 Hz (the frequency for this experiment is set at 50 Hz), and the time of sample measuring in this velocity meter is 1–5 min. Based on its users' preferences, Vectrino can take 60,000 flow samples every 5 min in each direction and save the information in the format of binary files on the hard drive of the computer to which it is connected. The saved data are analyzed and averaged using software programs Vectrino$^+$ and Explorer V (Nortek 2004), and the average of $U$, $V$, $W$ velocities and other relevant parameters such as shear stress and kinetic turbulent energy are measured (Vaghefi et al. 2010; Mahmoodi et al. 2013a, b).

## 2.3 Data Mining Algorithms

### 2.3.1 Box Plot Method

Box plot (Solberg and Lahti 2005) is a graphical technique that calculates data distribution using five main characteristics: (1) smallest normal observation (min), (2) lower quartile ($Q1$), (3) median, (4) upper quartile ($Q3$), and (5) largest normal observation (max). The value of $Q3 - Q1$ specifies the interquartile range ($IQR$). The normal and abnormal data can be identified by this parameter. Samples

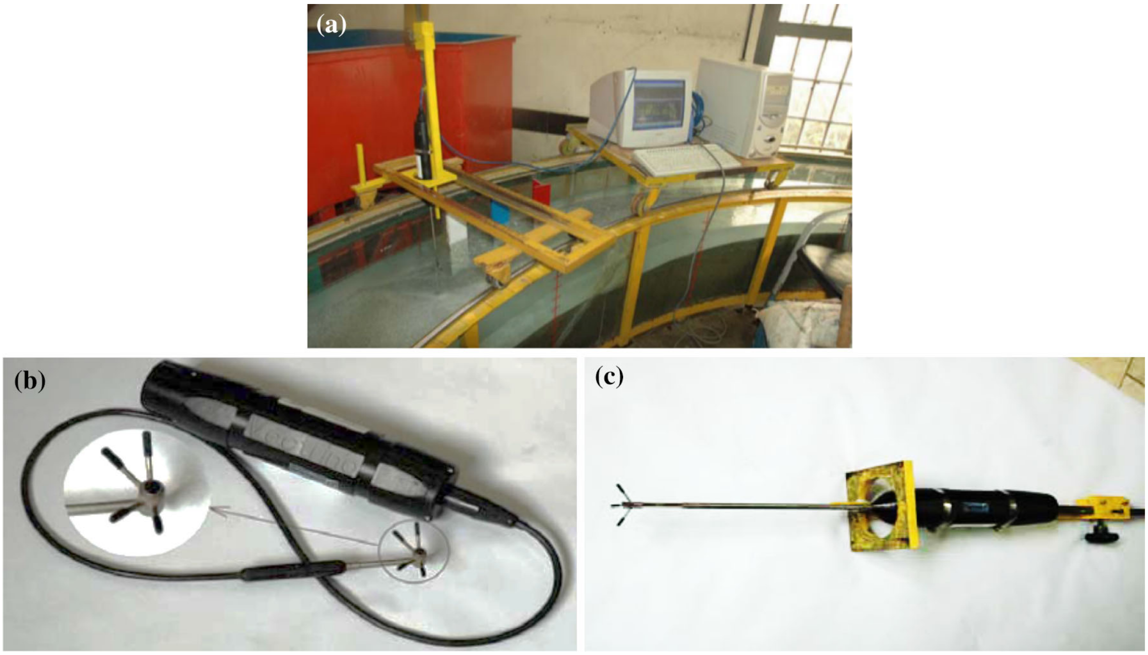

**Fig. 3** **a** Placement of the Vectrino velocity meter system, **b** side-looking sensor, and **c** down-looking sensor

$1.5 \times IQR$ times smaller than $Q1$, or $1.5 \times IQR$ times greater than $Q3$ could be considered as outliers. The mentioned concepts are shown in Fig. 4.

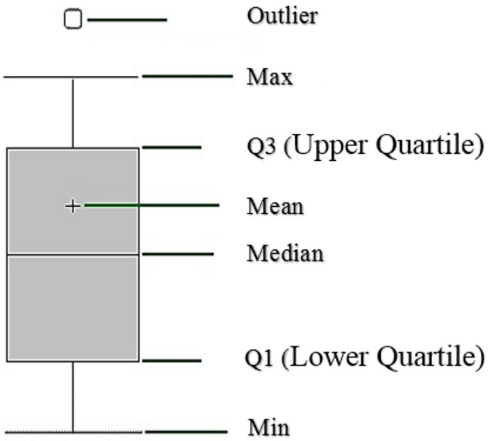

**Fig. 4** Box plot and its concepts

### 2.3.2 Histogram Method

Histogram techniques are dependent on the frequency or number of samples. The histogram can be graphically represented. Mathematically, the histogram of a variable contains the number of discrete bins in which the height of each bin represents the frequency (number) of samples that are located within a bin. If the samples in a bin are less than a user-defined threshold, it can be said that all samples located in the bin are candidates for outliers (Eskin 2000). For example, in Fig. 5, the histogram of a data set is shown. This graph contains eight bins. The samples shown with a vector mark in a bin could be indicative of an outlier. As clearly demonstrated, the frequency of this bin is considerably less than other bins.

### 2.3.3 Linear Regression Method

Regression analysis is used to determine the relationship between the dependent variable $y$ and one (or more) independent variable $x$. The simplest form of regression is linear, in which there are one dependent variable and one independent variable. The linear regression uses the formula of straight line, $\hat{y}_i = \hat{\alpha} + \hat{b}x_i + e_i$. In this formula, the values of $\hat{\alpha}$ and $\hat{b}$ variables are used to predict approximate values of $\hat{y}$ based on the values of $x$.

Values of the variables $\hat{\alpha}$ and $\hat{b}$ can be calculated from Eqs. (1)–(6):

$$\hat{\alpha} = \bar{y} - \hat{b}\bar{x} \tag{1}$$

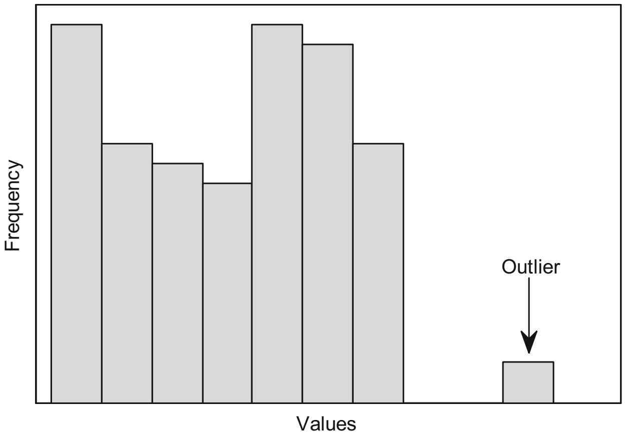

**Fig. 5** Histogram of a data set

$$\hat{b} = \frac{S_{xy}}{S_{xx}} \tag{2}$$

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{3}$$

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{4}$$

$$S_{xy} = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \tag{5}$$

$$S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2, \tag{6}$$

where $e_i$ specifies the residual values or errors. The regression line must be estimated in such a way that the sum-squared errors (SSE) is minimized. This method is called least-squares error. Thus, for each observation, $e_i = y_i - \hat{y}_i$ is the error of regression prediction that represents the difference between the $i$th of the $y_i$ observation and its result through regression line of $\hat{y}_i$. If the error of $i$th observation ($e_i$) is remarkably larger than the error of the other members of the sample, it could be stated that this observation is a candidate for being an outlier (Srimani and Koti 2012).

### 2.3.4 *k*-Nearest Neighbors Method

The $k$-nearest neighbors ($k$NN) algorithm is used for finding $k$-nearest neighbors of $p{\cdot}q$ from data set $D$ which is in the neighborhood of $p$ if its distance from $p$ is less than or equal to specified distance $d$:

$$k\_\text{Nearest Neighbors} = \{q \in D | \text{Dist}(q, p) \le d\} \tag{7}$$

In this case, $q$ is in $d$ neighborhood of $p$. In the above definition, $Dist$ represents the measuring distance between

$p$ and $q$. Euclidean distance measurement is used in this study to measure the distance between points.

To identify outliers using this method, the data located in the $d$ neighborhood of each data point is first calculated. If the number of points is less than a certain threshold $k$, then that data point could be a candidate for an outlier, otherwise it is a normal member of the data set. The values of $k$ and $d$ are determined based on the physical nature of matter and by trial and error (Ramaswamy et al. 2002; Amiri et al. 2016).

#### 2.3.5 Local Outlier Factor Method

The local outlier factor method (LOF) (Srimani and Koti 2012) is one of the most powerful methods in machine learning that can be used to identify anomalies in data. This method detects the outlier by calculating the local neighborhood density of each sample and assigning a factor to each of them which calculates the amount of inconsistency with other members of the data set. This factor is called the local outlier factor (LOF). The values of this factor depend on the isolation of a sample when it is compared to its local neighbors. Intuitively, large amounts of the LOF can be a representation of an outlier, while lower values indicate normality. For calculating the LOF, the following steps should be done:

**2.3.5.1 Step One: Calculating *k*-Distance of *p*** For any object $p$, $k$-distance ($p$) is the $k$th nearest neighbor of $p$. To calculate this parameter, the $k$th nearest neighbor of $p$ is initially determined, and then the distance from this neighbor to $p$ is selected as $k$-distance ($p$). This parameter gives an estimation of the local neighborhood density of $p$.

**2.3.5.2 Step Two: Finding *k*-Distance Neighborhood of *p*** Each $q$ whose distance from $p$ is less than or equal to $k$-distance ($p$) is located in $k$th distance neighborhood of $p$:

$$N_{k-\text{distance}(p)}(p) = \{q \in D\backslash\{p\} | d(p,q) \leq k - \text{distance}(p)\}. \tag{8}$$

**2.3.5.3 Step Three: Calculating the Reachability Distance of *p* with Respect to Object *o*** For any object $o$ which is located within k-distance neighborhood of $p$, reachability distance of $p$ with respect to object $o$ is defined as Eq. (9):

$$\text{Reachdist}_k(p,o) = \max\{k - \text{distance}(o), d(p,o)\}. \tag{9}$$

Figure 6 shows an example of the reachability distance for $k = 4$. If $p$ is located out of $k$-distance ($o$) ($p2$ in the figure), reachability density would be the distance between $d(o, p2)$. If the distance is less than the $k$-distance ($o$), then reachability distance is equal to $k$-distance ($o$).

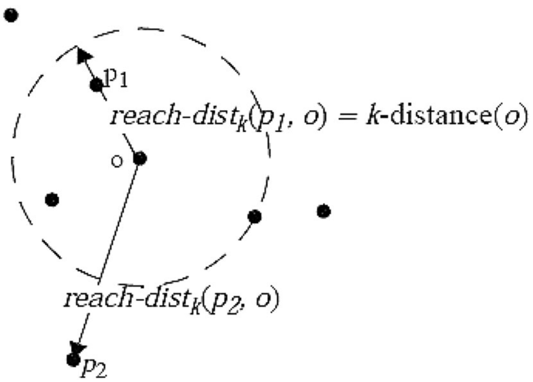

**Fig. 6** Concepts of reachability distance for $k = 4$ (Breunig et al. 2000)

**2.3.5.4 Step Four: Calculating the Local Reachability Density of *p*** Local reachability density of $p$ is the reversed average of reachability density $k$ to its close neighbors:

$$lrd_k(p) = \left[\frac{\sum_{o\epsilon N_{K(p)}} \text{Reach} - \text{dist}_K(p,o)}{|N_k(p)|}\right]^{-1} \tag{10}$$

The LOF is calculated using the value of parameter $lrd_k$.

**2.3.5.5 Step Five: Calculating the LOF** The LOF is used in order to detect outliers or normality of the data. The LOF($p$) is the average ratio of the local reachability density of $p$ and its $k$ neighbors:

$$\text{LOF}_k(p) = \frac{\sum_{o\epsilon N_k(p)} \frac{lrd_k(o)}{lrd_k(p)}}{|N_k(p)|} \tag{11}$$

#### 2.3.6 *k*-Medoids Clustering Method

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than those in other groups (clusters). Many data mining algorithms in the literature find outliers as a by-product of clustering algorithms themselves and define outliers as points that do not lie in or are located far apart from any clusters (Agrawal et al. 1998, 1999; Liu et al. 2015; Rashedi et al. 2015; Zhang et al. 2014; Rehman et al. 2014; Zhang 2008). Thus, the clustering techniques implicitly define outliers as the background noise of clusters. Clustering algorithms can be categorized based on their cluster model. Partitioning clustering is one of the clustering categories that perform clustering by partitioning the data set into a specific number of clusters. The number of clusters to be obtained, denoted by $k$, is specified by human users. Partitioning clustering methods typically start with an initial partition of the data set and then iteratively optimize the objective function until it reaches the optimal for the data set (Zhang 2008).

*k*-medoids clustering (Kaufman and Rousseeuw 1987) is a classical partitioning technique of clustering that clusters the data set of *n* objects into *k* clusters known a priori. It is more robust to noise and outliers as compared to *k*-means clustering (MacQueen 1967), since it minimizes a sum of pairwise dissimilarities instead of a sum of squared Euclidean distances. A medoid can be defined as the object of a cluster, whose average dissimilarity to all the objects in the cluster is minimal, i.e., it is the most centrally located point in the cluster. A typical *k*-medoids algorithm for partitioning based on medoid or central objects is as follows (Theodoridis and Koutroumbas 2006):

*Input:* *k*: The number of clusters; *D*: A data set containing *n* objects.

*Output:* A set of *k* clusters that minimizes the sum of the dissimilarities of all the objects to their nearest medoid.

*Method:* Initialize: randomly select (without replacement) *k* of the *n* data points as the medoids.

Associate each data point to the closest medoid. (“closest” here is defined using any valid distance metric, most commonly Euclidean distance (Deza and Deza 2009), Manhattan distance (Krause 1986), or Minkowski distance (Burago et al. 2001; Papadopoulos 2014).

For each medoid *m*:

For each non-medoid data point *o*.

Swap *m* and *o* and compute the total cost of the configuration.

Select the configuration with the lowest cost.

Repeat steps 2–4 until there is no change in the medoid.

### 2.3.7 Multilayer Perceptron (MLP)

Multilayer perceptron (MLP) is one of the most practical architectures of artificial neural networks, which is capable of performing regression and classification problems. A typical MLP consists of an input layer, a number of hidden layers, and an output layer each having a number of processing neurons (nodes) with varying weights representing the relative influence of the different neuron inputs to the other neurons (Azari et al. 2015; Heidari et al. 2016). The number of neurons in the input and output layer is equal to input and output variables, respectively. The number of hidden layers, neurons in the hidden layer, and linking weights is usually determined in the training process with trial-and-error procedure. It has been proven that a single hidden layer MLP network, given enough hidden neurons and suitable activation functions, can approximate any nonlinear relation (Hornik 1991). In the MLP network, the output of the *j*th neuron ($y_j$) can be found as follows:

$$y_j = f\left(\sum_{i=1}^{M} w_{ij}x_{ij} + b_j\right), \tag{12}$$

where $w_{ij}$ and $x_{ij}$ represent the link weights between the *i*th neuron in the previous layer and the *j*th neuron in the current layer that were selected randomly in the network training process, and also the input from the *i*th neuron to the *j*th neuron, respectively. *M* denotes the total number of neurons in the previous layer, and $b_j$ represents the bias associated with the *j*th neuron. *f* is the nonlinear activation transfer function which for the current work is hyperbolic tangent sigmoid function. In Eq. (12), weights and biases are unknown. In this study, back-propagation learning algorithm is employed to find unknowns.

The aim of this study is to examine the applicability of MLP network in outlier detection in flow pattern experiments. To do this, at first, the best MLP model will be created for each data set. Then, for each observation, the residual value ($e_i = y_i - \hat{y}_i$), which is the difference between real and output model, is calculated. The best network architecture is selected based on two statistical criteria, including coefficient of determination ($R^2$) and root-mean-squared error (RMSE), as follows:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{13}$$

$$R = \frac{\sum_{i=1}^{n}(y_i - \bar{y})\left(\hat{y}_i - \bar{\hat{y}}\right)}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2\sum_{i=1}^{n}\left(\hat{y}_i - \bar{\hat{y}}\right)^2}}, \tag{14}$$

where *n* represents the total number of observations, while $y_i$ and $\hat{y}_i$ are representative of real and predicted values using models, respectively. Moreover, $\bar{y}$ and $\bar{\hat{y}}$ are the average of mentioned data.

### 2.3.8 Self-Organizing Map (SOM)

Neural networks have been extensively used for outlier detection (Hoz et al. 2015; Wang et al. 2015; Fustes et al. 2013; Yan 2011). So far, different types of neural networks have been used for outlier detection. In this paper, SOM (Olawoyin et al. 2013; Corona et al. 2010) is selected, because this method has not yet been widely applied to the field of outlier detection in flow pattern experiments. SOM is an unsupervised neural network which clusters the input data into a fixed number of units. It consists of two layers of one-dimensional array of input units and a two-dimensional array of output units. These units are called neurons. The units in one layer are fully connected with the units in another layer. If input data set consists of *n* observations belonging to *d*-dimensional space, then the input layer must have *d* units and the output layer has *R* × *C* units, where *R* and *C* represent the number of rows and the number of columns of the SOM output array, respectively (Yan 2011). In this configuration, each map unit has a unique (*i*, *j*) coordinate. This makes it easy to reference a unit in the network and to calculate the distances between units. Each unit is associated with a weight vector of the same dimension as the input data vectors, and a position in the map space. SOM projects the input data set in a nonlinear way onto a rectangular grid laid out on a hexagonal lattice. It has a feed-forward structure with a single computational layer, which applies competitive learning as opposed to error correction learning and uses a neighborhood function to preserve the topological properties of the input space. The general structure of SOM networks is shown in Fig. 7.

The self-organization process involves five major components (Giraudel and Lek 2001): (1) All the connection weights are initialized with small random values; (2) A vector is chosen in a random way from the input data set and presented to the network; (3) Every unit in the network is examined to calculate which ones' weights are most like the input vector using a discriminant function (such as Euclidean distance) which provides the basis for competition. The particular neuron with the smallest value of the discriminant function is declared the winner. The winning neuron is commonly known as the best-matching unit (BMU); (4) The radius of the neighborhood of the BMU is calculated. The units in the neighborhood of the BMU are updated by pulling them closer to the input vector; (5) Repeat stage (2) for N iterations.

If the input space is *d*-dimensional, we can write the input patterns as $D = \{p_i : i = 1, 2, \ldots, d\}$, and the connection weights between the input units *i* and the neurons *j* in the output layer can be written $W_j = \{w_{ji} : j = R \times C; i = 1, \ldots, d\}$, where $R \times C$ is the total number of neurons in the output layer. At each training step *t*, a sample data vector $p(t) = [p_1, p_2, \ldots, p_d]$ is randomly chosen from the input data set and Euclidian distances between $p(t)$ and all the weight vectors are computed. The winning neuron $u_c$ (the neuron whose weight vector comes closest to the input vector) is determined by Eq. (15):

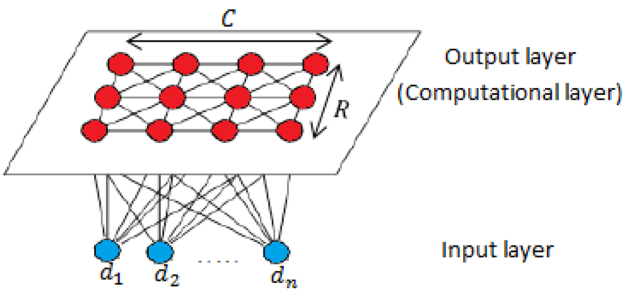

**Fig. 7** A two-dimensional SOM, each circle denotes one neuron at the input and output layer

$$\|p(t) - w_c(t)\| = \min_j \{\|p(t) - w_j(t)\|\}. \tag{15}$$

The equations for updating weights are:

$$W_j(t+1) = \begin{cases} W_j(t) + \alpha(t, c, j) \cdot (p(t) - W_j(t)), & \text{if} \quad j \in L_c(t) \\ W_j(t), & \text{if} \quad j \notin L_c(t) \end{cases}, \tag{16}$$

where $L_c(t)$ is a set of neighboring neuron of the winning neuron, and $\alpha(t, c, j)$ is the neighborhood kernel function (Wu and Chow 2004) around the winning neuron *c* at time *t*.

In this research, to detect outliers using the SOM method, based on the two-dimensional plane and the topology, a quasi-3 $\delta$ edit rule (Yan 2011) is applied. Suppose that the obtained weight vectors of SOM is $W_j = \{w_{ji} : j = R \times C; i = 1, \ldots, d\}$. The procedure of quasi-3 $\delta$ edit rule is as follows:

Determine the median of the weight vector, $W_{\text{median}}$, as:

$$W_{i,\,\text{median}} = \text{median}(W_{i,1}, W_{i,2}, \ldots, W_{i,j}, W_{i,R \times C}), \quad i = 1, 2, \ldots, d, \tag{17}$$

where $W_{i,\text{median}}$ is the *i*th element of $W_{\text{median}}$ and $W_{i,j}$ is the *i*th element of $W_j$.

Calculate the Euclidean distance $d_j$ between $W_{\text{median}}$ and $W_j$ as:

$$d_j = \left[\sum_{k=1}^{d} (W_{k,j} - W_{k,\,\text{median}})^2\right]^{\frac{1}{2}}, \quad j = 1, 2, \ldots, R \times C. \tag{18}$$

Determine the median of the Euclidean distance(s), $d_{\text{median}}$, as:

$$d_{\text{median}} = \text{median}(d_1, d_2, \ldots, d_j, \ldots, d_{R \times C}). \tag{19}$$

Calculate the median absolute deviation from $d_{\text{median}}$, $d_{\text{MAD}}$, as:

$$d_{\text{MAD}} = 1.4826 \times \text{median}(|d_1 - d_{\text{median}}|, |d_2 - d_{\text{median}}|, \ldots, |d_{R \times C} - d_{\text{median}}|). \tag{20}$$

Detect the outlier neurons based on the following rule:

$$\begin{cases} \text{Outlier neuron}, & \text{if} \quad R_j = \left|\dfrac{d_j - d_{\text{median}}}{d_{\text{MAD}}}\right| > 3 \\ \text{Normal neuron}, & \text{if} \quad R_j = \left|\dfrac{d_j - d_{\text{median}}}{d_{\text{MAD}}}\right| \le 3 \end{cases}, \quad (j = 1, 2, \ldots, R \times C). \tag{21}$$

The data objects projected on the outlier neuron are the outlier candidates.

**Table 1** Details of tested data sets

| Number | Data set | Total number of points | Total number of outliers |
|---|---|---|---|
| 1 | *U* (velocity in *x* direction) | 2751 | 13 |
| 2 | *V* (velocity in *y* direction) | 2751 | 10 |
| 3 | *W* (velocity in *z* direction) | 2751 | 17 |
| 4 | *U*–*V* (velocity in *x* direction–velocity in *y* direction) | 2751 | 41 |
| 5 | *U*–*W* (velocity in *x* direction–velocity in *z* direction) | 2751 | 31 |
| 6 | *V*–*W* (velocity in *y* direction–velocity in direction *z*) | 2751 | 22 |

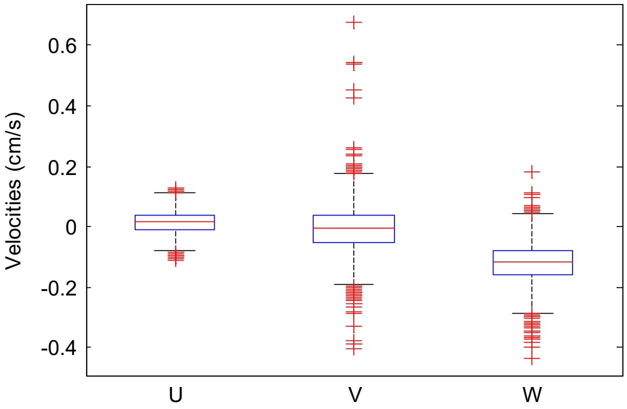

**Fig. 8** Box plot of *U*, *V*, and *W* velocities

**Table 2** Results of the box plot method on data sets

| Detection rate (%) | False alarm rate (%) | Data set |
|---|---|---|
| 100 | 0.62 | *U* |
| 100 | 2.11 | *V* |
| 100 | 1.28 | *W* |

## 2.4 Measurement Criteria for Precision of Methods

To select the best performance of the method (methods) for identifying outliers, it is essential to define criteria used to evaluate the performance of the algorithms. Algorithms for identifying anomalies in the data are typically evaluated by criteria like “Detection rate” and “False Alarm Rate” (Provost and Fawcett 2001):

$$\text{Detection Rate} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{22}$$

$$\text{False Alarm Rate} = \frac{\text{FP}}{\text{FP} + \text{TN}}, \tag{23}$$

where TP is the actual number of anomalous samples that are correctly diagnosed as anomalous samples, FN is the actual number of anomalous samples that are incorrectly diagnosed as normal samples, FP is the actual number of normal samples that are incorrectly diagnosed as anomalous samples, and TN is the actual number of normal samples that are correctly diagnosed as normal samples.

Detection rate criterion provides information regarding the relative number of correctly detected anomalous samples. False alarm rate represents the relative number of anomalous samples that might have been mistakenly interpreted as normal. If the detection rate is high and the false alarm rate is low, the method is more accurate. Its reverse is also true.

## 3 Potential Errors in Data Collection

In this study, based on conducted experiments, errors are divided into three categories: inherent errors, observation errors, and statistical errors (Vaghefi et al. 2010; Mahmoodi et al. 2013a, b).

### 3.1 Inherent Errors

These errors occur because of the circumstances of data collection and represent the error inherent in the collected data. In Vectrino velocity meter, inherent error can occur with changing flow pattern. During data collection, the required time to produce quasi-steady and quasi-permanent conditions at the start of the experiment and the restart time of the pump was taken into account. Moreover, a 10-s period of time was considered to remove local fluctuations occurring during position sensor change for 1-min collection of a point. As such, this error was reduced to the minimum possible value. Due to minor power fluctuations and its effect on the production discharge by the pump, there is a possibility of slight variations in the velocity

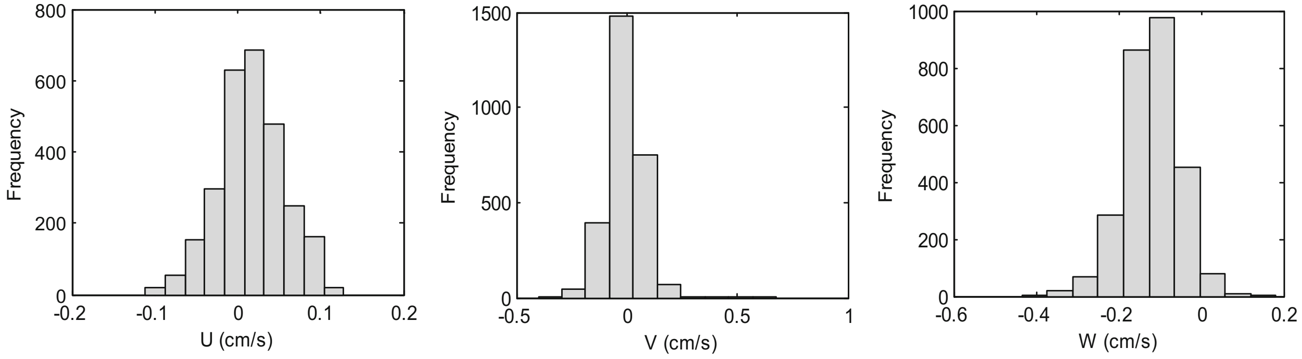

**Fig. 9** Histograms of *U*, *V*, and *W* velocities

**Table 3** Number of data located within each data set bin

| Data set | Bin 1 | Bin 2 | Bin 3 | Bin 4 | Bin 5 | Bin 6 | Bin 7 | Bin 8 | Bin 9 | Bin 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| *U* | 18 | 54 | 154 | 295 | 631 | 687 | 479 | 250 | 162 | 21 |
| *V* | 5 | 45 | 395 | 1482 | 750 | 66 | 3 | 2 | 2 | 1 |
| *W* | 4 | 19 | 65 | 283 | 864 | 977 | 451 | 81 | 6 | 1 |

**Table 4** Results of the histogram on data sets

| Data set | False alarm rate (%) | Detection rate (%) |
|---|---|---|
| *U* | 1.18 | 100 |
| *V* | 0.11 | 100 |
| *W* | 0.00 | 29.41 |

collection. Compared to the actual value, the current error is not significant since 3000 data (in 1 min) had been collected in any direction and at any point.

## 3.2 Observation Error

The major error observed in this part is in the adjustment of the coordinates of collected points of the velocity meter and balancing the spur dike in the considered positions along the bend. Consider the fact that the longitudinal, transverse, and vertical rulers with an accuracy of 0.1 mm are used to adjust the coordinates of points. The adjustment of the longitudinal cart, transverse movement rail, and vertically movable shaft is done by the user employing the mentioned rulers. The error rate is 0.1 mm if there is an error in the coordinates of points.

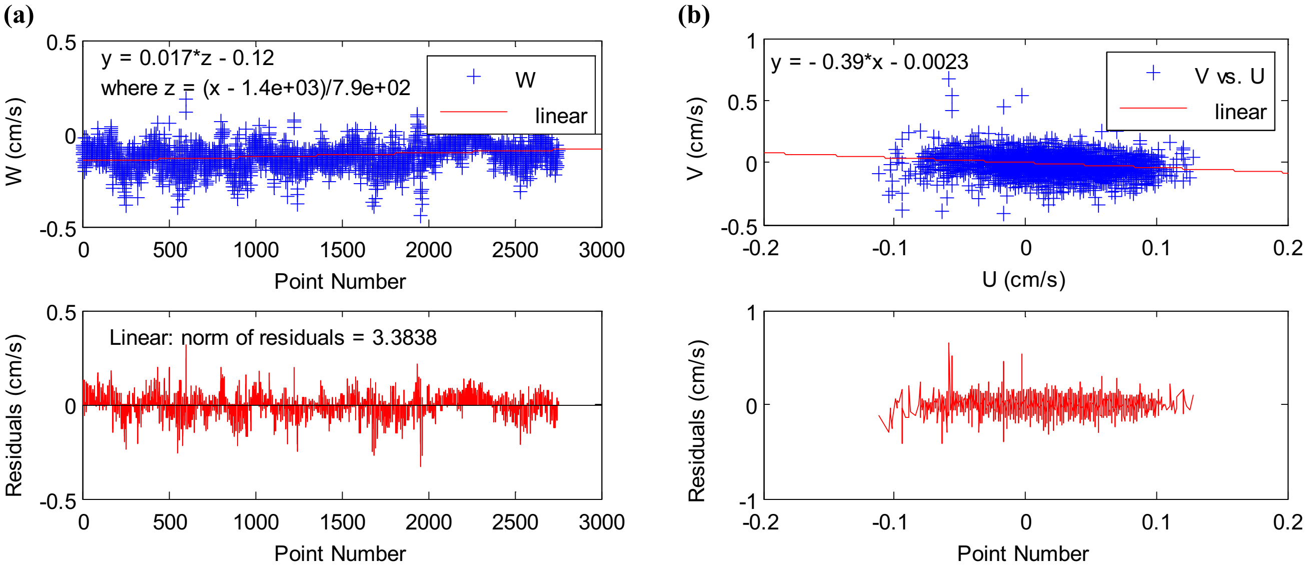

**Fig. 10** Applying the regression on data set: **a** *U* and **b** *U*–*W*

**Table 5** Results of linear regression on data sets

| Data set | False alarm rate (%) | Detection rate (%) |
|---|---|---|
| *U* | 1.61 | 100 |
| *V* | 1.93 | 100 |
| *W* | 1.72 | 100 |
| *U*–*V* | 1.40 | 51.22 |
| *U*–*W* | 2.13 | 29.03 |
| *V*–*W* | 1.91 | 54.54 |

**Table 6** Results of the *k*NN method on data sets

| Data set | Neighborhood radius (cm) | False alarm rate (%) | Detection rate (%) |
|---|---|---|---|
| *U* | 0.02 | 0.62 | 100 |
| *V* | 0.06 | 0.62 | 100 |
| *W* | 0.07 | 0.04 | 100 |
| *U*–*V* | 0.04 | 5.06 | 100 |
| *U*–*W* | 0.04 | 5.11 | 100 |
| *V*–*W* | 0.10 | 1.21 | 100 |

**Table 7** Results of the LOF method on data sets

| Data set | False alarm rate (%) | Detection rate (%) |
|---|---|---|
| *U* | 1.83 | 100 |
| *V* | 1.71 | 100 |
| *W* | 0.29 | 100 |
| *U*–*V* | 1.40 | 100 |
| *U*–*W* | 1.43 | 100 |
| *V*–*W* | 1.25 | 100 |

**Table 8** Results of the *k*-medoids on data sets

| Data set | False alarm rate (%) | Detection rate (%) |
|---|---|---|
| *U* | 0.18 | 100 |
| *V* | 0.25 | 50 |
| *W* | 0.07 | 17.65 |
| *U*–*V* | 2.24 | 5.23 |
| *U*–*W* | 3.01 | 4.80 |
| *V*–*W* | 1.87 | 5.03 |

### 3.3 Statistical Error

Statistical errors include errors incurred after data collection. In the collected velocity data using Vectrino, which has the same performance of the ADV, it is seen that in some of the recorded data some values are outside the range of other data. These errors are known as Spike.

**Table 9** Characteristics of selected MLP networks

| | |
|---|---|
| Training subset | 70% of data set |
| Validation subset | 15% of data set |
| Test subset | 15% of data set |
| Number of input layer neurons | 4 |
| Number of output layer neurons | 1 |
| Number of hidden layer neurons | 10 |
| Hidden layer activation function | Hyperbolic tangent |
| Output layer activation function | Linear |
| Training algorithm | Levenberg–Marquardt |
| Maximum number of training epochs | 1000 |

## 4 Results and Discussion

In this section, the researchers identify outliers in the data collected from the case study using the above-mentioned methods. Among the collected points in this study, the ability of the methods to detect outliers for the coordinates of a point (*U*, *V*, and *W* are the velocity values in the direction of *x*, *y*, and *z*, respectively) has been analyzed. Details of studied data sets are presented in Table 1. To assess the performance of methods in detecting outliers, using pretests and conducted studies, the outliers in each data set were detected. If a method (methods) detects all or most of the outliers without the slightest error, it will have the best performance and can be used in future studies to identify outliers. Using the algorithm of each method, a computer code was written in MATLAB software to identify the outliers. This code receives the raw data in Excel format as the input data, which then automatically saves the filtered files and outlier files in Excel format before providing the user with them.

The box plot is a univariate method. In other words, it is only applicable for univariate data sets. Therefore, this method can only be used with *U*, *V*, and *W* data sets. In Fig. 8, the box plots of these data sets are shown. In each graph, the central line of the box represents the median, the edges of the box represent the 25th and 75th percentiles, and the trailing edges represent the normal samples. Samples that fall outside of this range represent the outliers. These samples are marked with "+" sign in the figure. The summary of results of outliers detected by this method for data sets is presented in Table 2.

The histogram is a univariate analysis, too. In Fig. 9, the histograms of the test data sets are shown. Each of these graphs has 10 bins. In this study, the number of the bin is

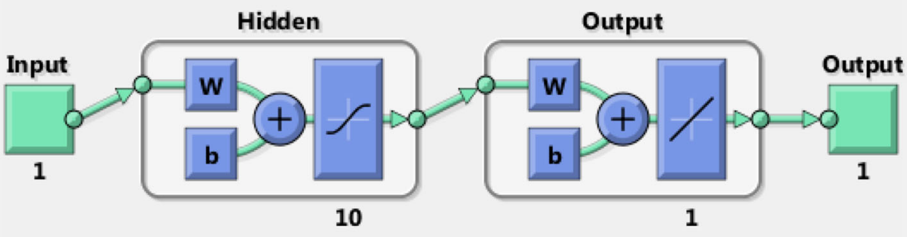

**Fig. 11** Schematic of defined MLP network

**Table 10** Results of the MLP method on data sets

| Data set | False alarm rate (%) | Detection rate (%) |
|---|---|---|
| $U$ | 1.82 | 100 |
| $V$ | 1.97 | 100 |
| $W$ | 2.23 | 100 |
| $U$–$V$ | 1.47 | 46.34 |
| $U$–$W$ | 1.87 | 38.70 |
| $V$–$W$ | 1.79 | 63.63 |

considered on the left side of the histogram. Table 3 shows the frequency of samples located in each bin. If the frequency of a bin compared to other bins is considerably less frequent, it can be said that all the samples located in those bins are a candidate for outliers. According to this definition, the data histogram $U$ data set located in bin 1, the data histogram $V$ data set located in bins 1, 7, 8, 9, and 10, and the data histogram $W$ data set in bins 1, 9, and 10 are the candidates for outliers.

The summary of the results of outliers detected by this method for data sets is presented in Table 4.

Simple linear regression can be applied to two-dimensional data sets. Hence, this method can be used in all data sets. The residual values should be calculated to identify outliers using regression models (the difference between the actual values and the values estimated by the regression line). Samples with values that differ greater than a threshold in comparison with other samples could be a candidate for outliers. After calculating the residual values for each data point, the following equation could be used to detect outliers:

$$G = \frac{|r_i - \bar{r}|}{\mathrm{SD}}. \tag{24}$$

In the above equation, $r_i$ is an element of the residual values, $\bar{r}$ the average, and SD standard deviation of the residual values. In this method, the value of $G$ for any residual value is calculated. If the value is greater than a threshold value of $t$, then the sample can be a candidate for outliers. In this study, $t$ is considered 2.5. For example, in Fig. 10, the results of applying linear regression on $W$ and $U$–$V$ data sets are shown. In this figure, the trend line on the data sets and the graph of the residuals are depicted. Due to the limited page numbers of this paper, outlining the results of applying regression for all data sets is avoided. The summary of the results of the identified outliers for each data set using this method is presented in Table 5.

For applying the $k$NN algorithm on data sets in order to identify potential outliers, the determination of the parameters of the number of neighbors ($k$) and radius of the neighborhood ($d$) is required. As previously mentioned, the correct values of these parameters depend on the physical nature of the matter and are usually obtained through trial and error. The value of $k$ is considered 50, which suggests that for each data item, the 50 nearest neighbors are defined as the area of the neighborhood. The reason is that around 50 samples are collected each second while collecting data. Therefore, samples collected in 1 s are considered as the neighboring data. The parameter $d$ is also specified based on the nature of each sample and the pre-performed tests. The Euclidean function is used to measure distance between the points. The $k$NN test results on data sets are shown in Table 6.

The number of neighbors ($k$) and threshold parameter ($t$) for applying LOF algorithm on data sets needs to be determined. Here, the value of $k$ is considered 50. After calculating LOF for each sample, the falsity or the normality is acknowledged based on this value. According to the LOF formula, if all samples are sorted and put side by side exactly with the same distance on the plane, then the LOF of all samples (except for the boundary samples) will be 1. Also, as the neighborhood density increases, the factor is closer to zero, otherwise this factor is greater than 1 and may even become a larger number. Therefore, a number greater than 1 should be selected as the threshold in each problem to identify outliers (Mahmoodi et al. 2013a, b). In this study, the number 1.3 is selected as the threshold parameter value due to the nature of the data. This means that samples with a neighborhood density less than 30% of their uniform density are considered as candidate for outliers. Table 7 presents the LOF test results for each data set.

Applying $k$-medoids algorithm to data set requires the determination of the number of clusters $k$ and the distance measurement function. The most important parameter is

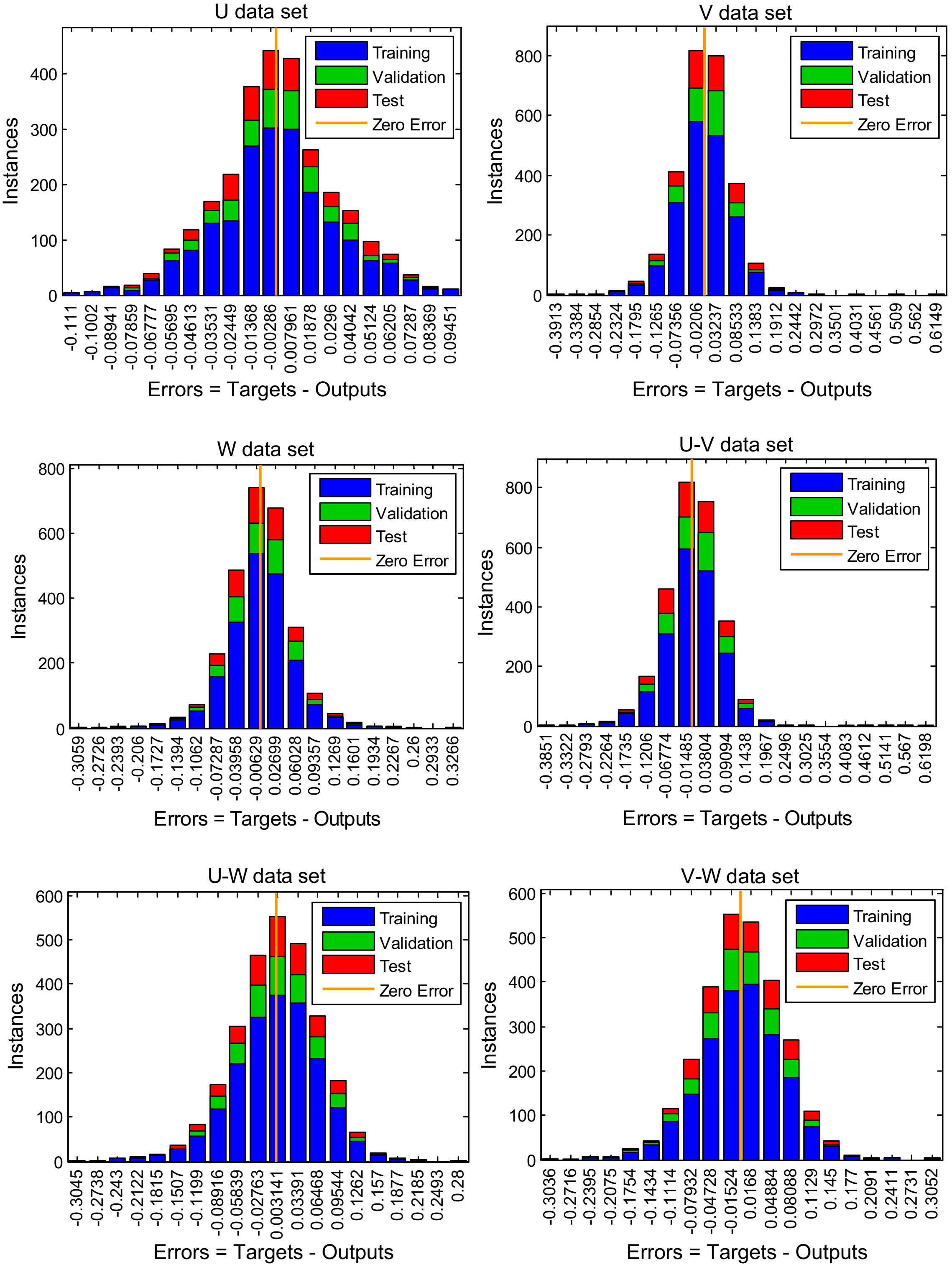

**Fig. 12** Error histogram of the best obtained models

finding the true value for *k*. There are no explicit rules as to how to find the true value of such a parameter, and it depends on the input nature. Typically, the algorithm is applied on different amounts of *k* so as to select the most appropriate one. Here, *k* is considered 40 for all data sets. In order to measure the distance, Euclidean distance has been used in this research. If the number of data located in a cluster is smaller than a threshold parameter *t*, then all the

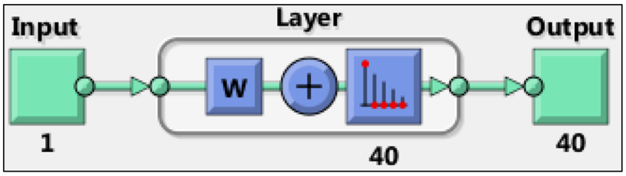

**Fig. 13** Schematic of defined SOM network

**Table 11** Selected SOM network parameters

| Parameter | Value |
|---|---|
| Map dimensions | 5 × 8 |
| Number of neurons | 40 |
| Layer topology function | Hexagonal |
| Neuron distance function | Link distance function |
| Training algorithm | Batch unsupervised weight/bias training |
| Performance function | Mean squared normalized error |
| Initial neighborhood size | 3 |
| Number of training steps for initial covering of the input space | 100 |
| Number of epochs | 300 |

**Table 12** Results of the SOM method on data sets

| Data set | False alarm rate (%) | Detection rate (%) |
|---|---|---|
| *U* | 0.77 | 100 |
| *V* | 0.84 | 100 |
| *W* | 0.91 | 100 |
| *U*–*V* | 1.99 | 53.65 |
| *U*–*W* | 1.14 | 50.33 |
| *V*–*W* | 1.06 | 73.72 |

data in that cluster will be taken as candidates for outliers. The threshold parameter value has been selected to be 19 in all data sets. The results of the application of this method to data sets are presented in Table 8.

Development of reliable ANN models for prediction problems requires determination of the ANN architecture, i.e., the number of hidden layers, the number of neurons in the hidden layers, learning algorithm, and the activation transfer functions. The suitable selection of these values is based on trial-and-error procedure. The MLP network usually has one or more hidden layers, since according to Bishop's study (Bishop 1995), more than one hidden layer is often not necessary; so our architectures have only one hidden layer. To determine the best MLP network architecture, several models were created with varying network parameters. The parameters of optimum network structure and its schematic are shown in Table 9 and Fig. 11, respectively.

The results of MLP method on the data sets are presented in Table 10. Also, the error histogram for the best obtained models is presented in Fig. 12.

To cluster input data sets using self-organizing map, a 5-by-8 two-dimensional map of 40 neurons is used. The map size was determined empirically by trial and error. Figure 13 represents the schematic of defined SOM network. The batch SOM algorithm is used for training because it is more stable than the online version and in addition, it is faster and can be parallelized to reduce computational time (Fustes et al. 2013). The selected networks parameters are shown in Table 11.

Table 12 provides the results of SOM method on the data sets. Figure 14 indicates distances between neighboring of all studied stations. This figure uses the following color coding: (1) The blue hexagons represent the neurons; (2) The red lines connect neighboring neurons; (3) The colors in the regions containing the red lines indicate the distances between neurons; (4) The darker colors represent larger distances; (5) The lighter colors represent smaller distances. Figure 15 shows how many data points are associated with each neuron of all studied stations. Neurons with lower sample hits are outlier candidates.

The conclusion of the results of all tests is shown in Table 13. The average of false alarm rate and detection rate derived from the execution of all methods on tested data sets has been provided in this table. A method works best with the lowest average of false alarm rate and the highest average of detection rate. According to the results in Table 13, we can suggest that the local outlier factor (LOF) and the box plot methods had the best performance. The performance of the *k*-nearest neighbors was acceptable, and its rate of false alarm rate is slightly higher than the LOF and the box plot methods. On the other hand, the lowest performance is related to the *k*-medoids method. This is because such method was unable to cluster the data properly with the selected values for input parameters of the algorithm. The small values of detection rate in this method are due to the fact that a large number of outliers have been placed in normal clusters by mistake. Thus, it can be said that the method had not been able to differentiate the data properly in most of the data sets.

As outlined in Table 13, most methods have given satisfactory results. It should be noted, though, that the nature of the collected data from various experiments is different. Hence, there is not a superior method compared to other methods, and a method may be highly efficient for a particular data set while not having acceptable performance for other data sets. As such, it is recommended to use the process employed in this study when working with different data.

**Fig. 14** Neural network training SOM neighbor weight distances of all data sets

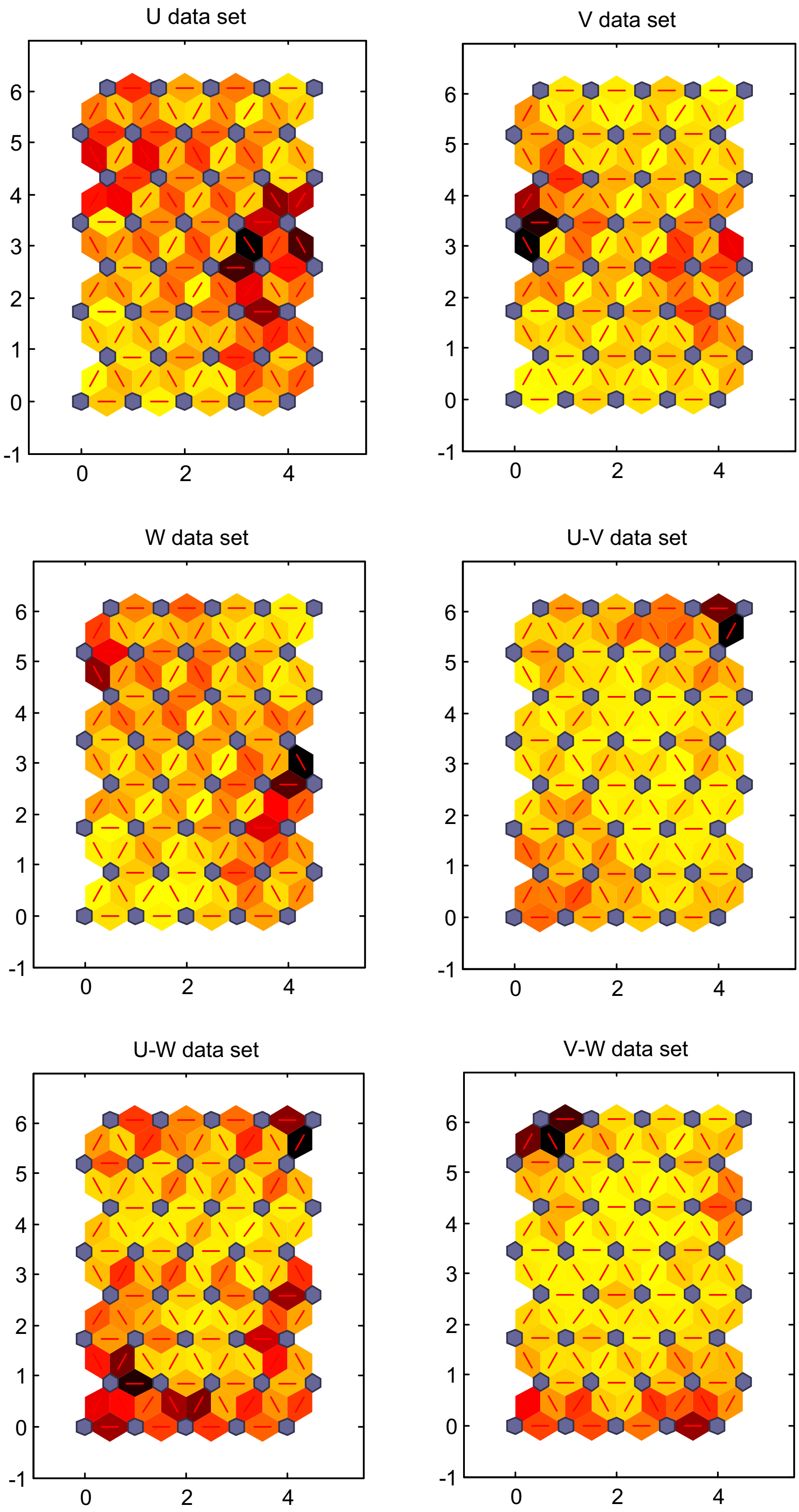

**Fig. 15** Neural network training SOM sample hits of all data sets

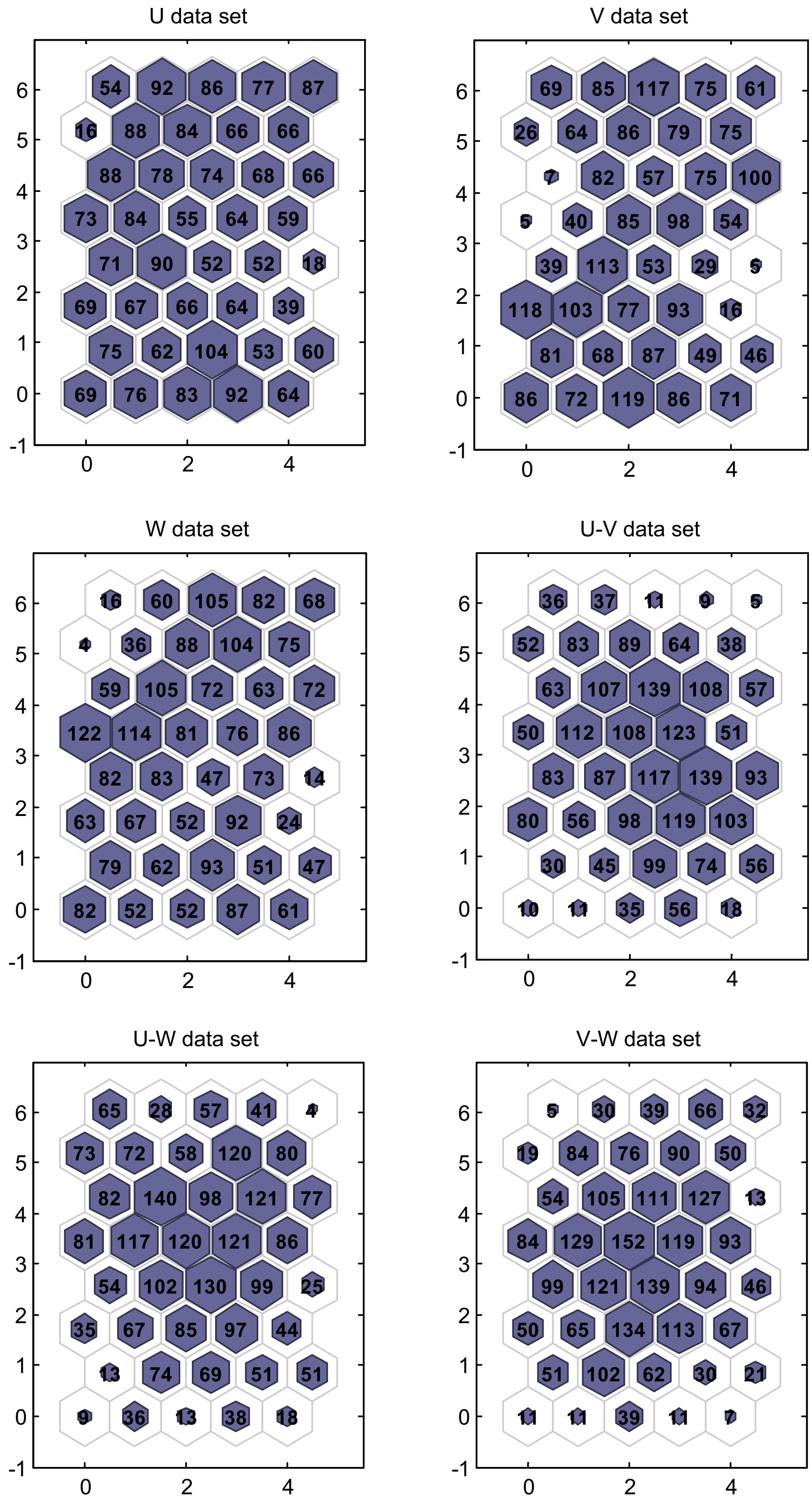

**Table 13** Conclusions of the results of all tests on data sets

| Method | False alarm rate average (%) | Detection rate average (%) |
|---|---|---|
| Box plot | 1.34 | 100 |
| Histograms | 0.09 | 74.47 |
| Linear regression | 1.78 | 72.47 |
| *k*-nearest neighbors | 2.11 | 100 |
| Local outlier factor | 1.33 | 100 |
| *k*-medoids clustering | 1.27 | 30.45 |
| Multilayer perceptron | 2 | 75 |
| Self-organizing map | 1 | 80 |

Interestingly, the samples which these methods have selected as the outlier candidates may not truly reflect the errors in the study system, as they may have been created due to changes in natural conditions (e.g., changes in flow pattern). Therefore, we should measure different aspects of outliers' falsity after identifying them in order to either eliminate or correct them. It is also worth mentioning that in a particular experiment, the method of selecting the correct input parameters for each algorithm on its performance in the detection of outlier is effective. For example, if the threshold value of the LOF algorithm is chosen higher than 1.3, some outliers may go outside their domain and be considered as the normal sample. Also, if the threshold value is selected less than 1.3, some normal samples may go outside of their domain and be considered as the outlier. In general, there is no rule specifying the correct choice of algorithm parameters, and their correct selection is dependent on the physical nature of matter, the nature of the data and related professional person's experience.

## 5 Conclusions

Experimental data collection has always been associated with numerous outliers. These outliers cause problems in data analysis and lead to incorrect conclusions. Hence, outlier detection is required before the processing of data. In this study, the box plot, histograms, linear regression, *k*-nearest neighbors, local outlier factor, *k*-medoids clustering, multilayer perceptron, and self-organizing map methods and the way they are employed to identify outliers in a case study were discussed. The performance of these methods has been analyzed in identifying the outliers in a case study, the purpose of which is to determine the flow pattern around a T-shaped spur dike located in a 90° bend. The outliers present in data collection for the case study are caused by Vectrino 3D velocimeter, change in measuring conditions, and the problems occurred during the data collection.

The results indicated that most methods have given satisfactory results, but the box plot and the local outlier factor methods held the best performance among all methods (because of the lowest average of false alarm rate and the highest average of detection rate). Moreover, the poorest performance is observed in the *k*-medoids method. This is because such a method was unable to cluster the data properly with the selected values for input parameters of the algorithm. However, it should be noted that the nature of the collected data from various experiments is different. Hence, there is not one method superior to other methods, and a method may be highly efficient for a particular data set while it may not have an acceptable performance for other data sets. Therefore, the authors of this paper suggest using these methods to identify the outliers before analyzing the data collected from the flow pattern experiments.

## References

Aggarwal CC, Procopiuc CM, Wolf JL, Yu PS (1999) Fast algorithms for projected clustering. In: Proceeding of international conference on management of data, Philadelphia

Agrawal R, Gehrke J, Gunopulos D, Raghavan P (1998) Automatic subspace clustering of high dimensional data for data mining applications. In: Proceeding of international conference on management of data, Seattle

Alarcon-Aquino V, Garcia-Baleon HA, Ramirez-Cortes JM, Gomez-Gil P, Starostenko O (2011) Biometric cryptosystem based on keystroke dynamics and *k*-Medoids. IETE J Res 57:385–394. https://doi.org/10.4103/0377-2063.86341

Alih E, Ong HC (2015) Cluster-based multivariate outlier identification and re-weighted regression in linear models. J Appl Stat 42:938–955. https://doi.org/10.1080/02664763.2014.993366

Amiri M, Amnieh HB, Hasanipanah M, Khanli LM (2016) A new combination of artificial neural network and *k*-nearest neighbors models to predict blast-induced ground vibration and air-overpressure. Eng Comput. https://doi.org/10.1007/s00366-016-0442-5

Azari T, Samani N, Mansoori E (2015) An artificial neural network model for the determination of leaky confined aquifer parameters: an accurate alternative to type curve matching methods. Iran J Sci Technol 39:463–472

Bishop C (1995) Neural networks for pattern recognition. Oxford University, New York

Breunig MM, Kriegel HP, Ng RT, Sander J (2000) LOF: identifying density-based local outliers. In: Proceedings of the 2000 ACM Sigmod international conference on management of data, vol 29. ACM, New York, NY, USA, pp 93–104

Burago D, Burago YD, Ivanov S (2001) A course in metric geometry. American Mathematical Society, Rhode Island

Cea L, Puertas J, Pena L (2007) Velocity measurements on highly turbulent free surface flow using ADV. Exp Fluids 42:333–348. https://doi.org/10.1007/s00348-006-0237-3

Corona F, Mulas M, Baratti R, Romagnoli JA (2010) On the topological modeling and analysis of industrial process data using the SOM. Comput Chem Eng 34:2022–2032. https://doi.org/10.1016/j.compchemeng.2010.07.002

De la Hoz E, De La Hoz E, Ortiz A, Ortega J, Prieto B (2015) PCA filtering and probabilistic SOM for network intrusion detection. Neurocomp 164:71–81. https://doi.org/10.1016/j.neucom.2014.09.083

Deza E, Deza MM (2009) Encyclopedia of distances. Springer, New York

Dhhan W, Rana S, Midi H (2015) Non-sparse ∊-insensitive support vector regression for outlier detection. J Appl Stat 42:1723–1739. https://doi.org/10.1080/02664763.2015.1005064

Durgesh V, Thomson J, Richmond MC, Polagye BL (2014) Noise correction of turbulent spectra obtained from acoustic doppler velocimeters. Flow Meas Instrum 37:29–41. https://doi.org/10.1016/j.flowmeasinst.2014.03.001

Eskin E (2000) Anomaly detection over noisy data using learned probability distributions. In: Proceeding of 7th international conference on machine learning, Stanford

Fustes D, Dafonte C, Arcay B, Manteiga M, Smith K, Vallenari A, Luri X (2013) SOM ensemble for unsupervised outlier analysis. Application to outlier identification in the Gaia astronomical survey. Expert Syst Appl 40:1530–1541. https://doi.org/10.1016/j.eswa.2012.08.069

Ghodsian M, Vaghefi M (2009) Experimental study on scour and flow field in a scour hole around a T-shaped spur dike in a 90 degree bend. J Sediment Res 24:145–158. https://doi.org/10.1016/S1001-6279(09)60022-6

Giraudel JL, Lek S (2001) A comparison of self-organizing map algorithm and some conventional statistical methods for ecological community ordination. Ecol Model 146:329–339. https://doi.org/10.1016/S0304-3800(01)00324-6

Goring DG, Nikora VI (2002) Despiking acoustic doppler velocimeter data. J Hydraul Eng 128:117–126. https://doi.org/10.1061/(ASCE)0733-9429(2002)128:1(117)

Han J, Kamber M (2006) Data mining: concepts and techniques. Morgan Kaufmann Publishers, San Francisco

Hawkins D (1980) Identification of outliers. Chapman and Hall, London

Heidari E, Sobati MA, Movahedirad S (2016) Accurate prediction of nanofluid viscosity using a multilayer perceptron artificial neural network (MLP-ANN). Chemom Intell Lab 155:73–85. https://doi.org/10.1016/j.chemolab.2016.03.031

Hejazi K, Falconer RA, Seifi E (2016) Denoising and despiking ADV velocity and salinity concentration data in turbulent stratified flows. Flow Meas Instrum 52:83–91. https://doi.org/10.1016/j.flowmeasinst.2016.09.010

Hornik K (1991) Approximation capabilities of multilayer feedforward networks. Neural Netw 4:251–257. https://doi.org/10.1016/0893-6080(91)90009-T

Islam MR, Zhu DZ (2013) Kernel density–based algorithm for despiking ADV data. J Hydraul Eng 139:785–793. https://doi.org/10.1061/(ASCE)HY.1943-7900.0000734

Kang H (2013) Flow characteristics and morphological changes in open-channel flows with alternate vegetation zones. KSCE J Civ Eng 17:1157–1165. https://doi.org/10.1007/s12205-013-0346-5

Kaufman L, Rousseeuw PJ (1987) Clustering by means of medoids, in statistical data analysis based on the L1-norm and related methods. North-Holland, New York

Khorsandi B, Mydlarski L, Gaskin S (2012) Noise in turbulence measurements using acoustic Doppler velocimetry. J Hydraul Eng 138:829–838. https://doi.org/10.1061/(ASCE)HY.1943-7900.0000589

Krause EF (1986) Taxicab geometry: an adventure in non-Euclidean geometry. Courier Dover, New York

Liu X, Wanga X, Pedryczc W (2015) Fuzzy clustering with semantic interpretation. J Appl Soft Com 26:21–30. https://doi.org/10.1016/j.asoc.2014.09.037

MacQueen J (1967) Some methods for classification and analysis of multivariate observations. In: Proceeding of 5th Berkeley symposium on mathematical statistics and probability, Berkeley

Mahmoodi K, Rostami H, Saybany M, Moradi A (2013) An overview of the science of data mining and its applications in the offshore industry. In: Proceeding of 5th national offshore industries conference, Tehran

Mahmoodi K, Vaghefi M, Moradi A, Sayehbany M (2013) Identifying the errors in the data collection related to the flow and score pattern using the local outlier factor. In: Proceeding of 5th national offshore industries conference, Tehran

Nikora VI, Goring DG (2000) Flow turbulence over fixed weakly mobile gravel beds. J Hydraul Eng 126:679–690. https://doi.org/10.1061/(ASCE)0733-9429(2000)126:9(679)

Nortek AS (2004) Nortek Vectrino velocimeter user guide. Nortek, Norway

Olawoyin R, Nieto A, Larry Grayson R, Hardisty F, Oyewole S (2013) Application of artificial neural network (ANN)—self-organizing map (SOM) for the categorization of water, soil and sediment quality in petrochemical regions. Expert Syst Appl 40:3634–3648. https://doi.org/10.1016/j.eswa.2012.12.069

Papadopoulos A (2014) Metric spaces, convexity and nonpositive curvature. European Mathematical Society, Strasbourg

Provost F, Fawcett T (2001) Robust classification for imprecise environments. Mach Learn 42:203–231. https://doi.org/10.1023/A:1007601015854

Ramaswamy S, Rastogi R, Kyuseok S (2002) Efficient algorithms for mining outliers from large data sets. In: Proceeding international conference on management of data, Madison

Rashedi E, Mirzaei A, Rahmati M (2015) An information theoretic approach to hierarchical clustering combination. J Neurocomput 148:487–497. https://doi.org/10.1016/j.neucom.2014.07.014

Rehman MZ, Li T, Yang Y, Wang H (2014) Hyper-ellipsoidal clustering technique for evolving data stream. J Knowl Based Syst 70:3–14. https://doi.org/10.1016/j.knosys.2013.11.022

Shamim MA, Hassan M, Ahmad S, Zeeshan M (2015) A comparison of artificial neural networks (ANN) and local linear regression (LLR) techniques for predicting monthly reservoir levels. KSCE J Civ Eng. https://doi.org/10.1007/s12205-015-0298-z

Solberg HE, Lahti A (2005) Detection of outliers in reference distributions: performance of Horn's algorithm. Clin Chem 51:2326–2332

Srimani PK, Koti MS (2012) Outliers mining in medical databases by using statistical methods. Int J Eng Sci Technol 4:239–246

Sulaiman MS, Sinnakaudan SK, Shukor MR (2013) Near bed turbulence measurement with acoustic doppler velocimeter (ADV). KSCE J Civ Eng 17:1515–1528. https://doi.org/10.1007/s12205-013-0084-8

Theodoridis S, Koutroumbas K (2006) Pattern recognition. Academic Press, Inc., Orlando

Vaghefi M, Ghodsian M, Salehi Neyshabori SAA (2009) Experimental study on the effect of a T-shaped spur dike length on scour in a 90 degree channel bend. Arab J Sci Eng 34:337–348

Vaghefi M, Ghodsian M, Adib A (2010) Review of errors in data recovery laboratory. In: Proceeding of 9th Iranian hydraulic conference, Tehran

Vaghefi M, Ghodsian M, Salehi Neyshabori SAA (2012) Experimental study on scour around a T-shaped spur dike in a channel bend. J Hydraul Eng 138:471–474. https://doi.org/10.1061/(ASCE)HY.1943-7900.0000536

Vaghefi M, Akbari M, Fiouz AR (2015a) An experimental study of mean and turbulent flow in a 180 degree sharp open channel bend: secondary flow and bed shear stress. KSCE J Civ Eng. https://doi.org/10.1007/s12205-015-1560-0

Vaghefi M, Safarpoor Y, Akbari M (2015b) Numerical investigation of flow pattern and components of three-dimensionalv around a submerged T-shaped spur dike in a 90 degree bend. J Cent South Univ 0: 1–15

Wang Y, Zhang M, Wilson PA, Liu X (2015) Adaptive neural network-based backstepping fault tolerant control for underwater vehicles with thruster fault. Ocean Eng 110:15–24. https://doi.org/10.1016/j.oceaneng.2015.09.035

Wu ST, Chow TWS (2004) Clustering of the self-organizing map using a clustering validity index based on inter-cluster and intra-cluster density. Pattern Recogn 37:175–188. https://doi.org/10.1016/S0031-3203(03)00237-1

Xiekang W, Xingnian L (2016) Experimental investigation of flow structures and bed deformation with small width-to-depth ratio in a bend flume. KSCE J Civ Eng 20:497–508. https://doi.org/10.1007/s12205-015-0654-z

Yafei H (2015) Discussion on the development of algorithm for despiking ADV data. Int J Sci Res 4:1018–1020

Yan X (2011) Multivariate outlier detection based on self-organizing map and adaptive nonlinear map and its application. Chemom Intell Lab 107:251–257. https://doi.org/10.1016/j.chemolab.2011.04.007

Yang B, Zhang Q, Zhou Z (2015) Solving truss topological optimization via swarm intelligence. KSCE J Civ Eng. https://doi.org/10.1007/s12205-015-0501-2

Zhang J (2008) Towards outlier detection for high-dimensional data streams using projected outlier analysis strategy. Dissertation, Dalhousie University

Zhang T, Chen L, Ma F (2014) A modified rough c-means clustering algorithm based on hybrid imbalanced measure of distance and density. Intl J Approx Reason 55:1805–1818. https://doi.org/10.1016/j.ijar.2014.05.004