Modified Self-organizing map (MSOM) method: A SOM algorithm was used in [1] to filter the IN corrupted pixels in images. In general, the goal of a SOM is to approximate the topological structure of inputs’ patterns [2]. This is achieved by using neurons arranged usually two-dimensional lattice patterns. Therefore, SOM’s are a class of ANN’s. The weights of the neurons (i.e., , where is the index of the neuron) are considered to be the coordinates of the neurons approximating the topological structure of the input. For example, Figure 1 (a) demonstrates a schema of a SOM where the input space may have any dimension while the output space has two dimensions, and the input is approximated using 9 neurons. As a result, the input vector is projected to a 2-D space and the ’s can be considered to be located in an output domain with x and y coordinates. Then, the SOM algorithm changes ’s to approximate the topological structure of the input. The final values of ’s are called feature maps. In Figure 1 (b), an example, where the 2-D feature map of a 2-D input is created using 9 neurons, is displayed.

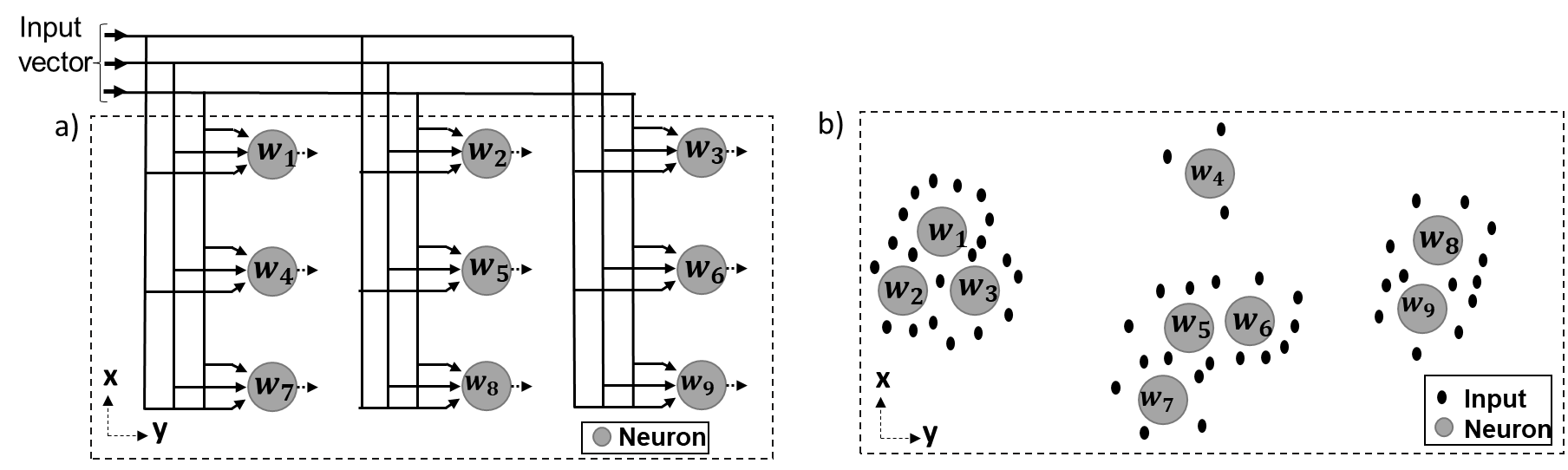


Figure 1. SOM: (a) a structure of an ANN in SOM having 9 neurons (b) the 2-D feature map of a 2-D input (example).

The steps of the SOM algorithm for input and output spaces having the same dimensions are as following:

**1. Initialization:** Determine the number of neurons (i.e., ) and choose random values for ’s.

**2. Sampling:** Draw one sample (i.e., ) from the 2-D input (i.e., *u*)) with a certain probability at each iteration (i.e., ).

**3. Similarity matching:** Find the best-matching neuron (i.e., , where is the index of the best-matching neuron) through the minimum distance criterion at the iteration .

(1)

**4.Updating:** Update ’s:

(2)

(3)

(4)

(5)

In Eq.2, is the iteration (time) dependent learning-rate parameter and is the iteration-dependent neighborhood function. The purpose of is to control the amount of the change in ’s and it is defined in Eq.3 where is a time constant. The purpose of is to restrict the updating procedure to the neurons that are located inside the topological neighborhood of the winning neuron (i.e., )). Consequently, not only the winning neuron is updated, but the neurons around the winning neuron are updated as well. This function is defined in Eq.4, where is the distance between and ’s and ) and is the iteration (time) dependent width of the neighborhood function. Eq. 5 shows the formulation of where is another time constant. Higher values of indicates larger coverage of . Initially, almost all ’s should be included in . Values of and decrease over time as it is expected that ’s approach to the topological structure of the input space.

5. Continuation: Continue with step 2 until no change in ’s is observed.

In the IN detection method used in [1], a two-dimensional input space was created where one dimension was the value of a pixel and the second dimension was the difference of the pixel value with the median of the surrounding pixels. Then, the SOM algorithm was used to classify the corrupted and uncorrupted pixels. In this study, this approach is modified to be applicable for one-dimensional dynamic LDV signals that have similar noise characteristics as (e.g., ). The modified method is called Modified SOM (i.e., MSOM) method. Its steps are:

First, a two-dimensional input space (i.e.,) of a moving LDV signal (e.g., ) is generated:

(6)

(7)

In Eq. 6, is the moving median of . In Eq. 7, is the 4th order difference of values of , is the moving median of . Two zeros are added at the beginning and the end of since the fourth-order difference of a signal decrease the length of the signal by 4 points. Selection of for and are based on the length of IN observed in and . This is because the moving median is robust against artificial peaks with a length of+1 [3]. To obtain for (i.e., ), of and were chosen as 8 and 62 since the maximum lengths of artificial peaks in and were 9 and 63, respectively. The reason of selecting a larger window size for is because appear as artificial peaks in due to 4th order differentiation. Figure 2 (a) shows a segment of and while Figure 2 (b) shows a segment of and .

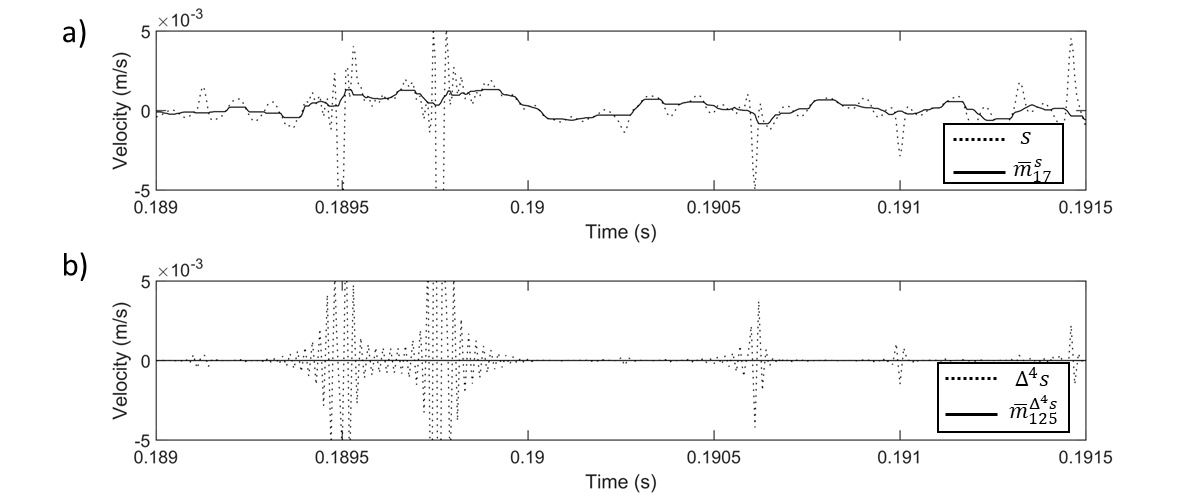


Figure 2. Signals required for : (a) a segment of and (b) a segment of and .

Next, create a SOM for . Then, the points in were assigned to the closest neuron through their minimum Euclidian distance to the neuron:

(22)

where represents a signal point assigned to a neuron (e.g., ). A 70 by 70 two-dimensional ANN was chosen to be used in the SOM algorithm to capture the topological structure of as shown in Figure 3 (c).

Afterward, find the main trends in **.** To achieve this, calculate the mean of the distances between a neuron and the closest six neurons (i.e., ). Next, sort ’s in ascending order. Then, use the L-shape method to find the knee point [4]. Discard the points beyond this knee point. This is because some extreme points corrupt the L-shape of the curve. Then, again apply the L-shape method to the remaining points and find the second knee point. Figure 3 (d) shows the ’s computed for and the second knee point.

Then, find the neurons whose values are lower than the second knee point and consider them as the neurons representing the points in that are not corrupted by IN. Figure 3 (e) shows the points in that are not corrupted by IN. The remaining points are considered to be corrupted by IN. Find their corresponding points of in and consider them as points that are corrupted by IN.

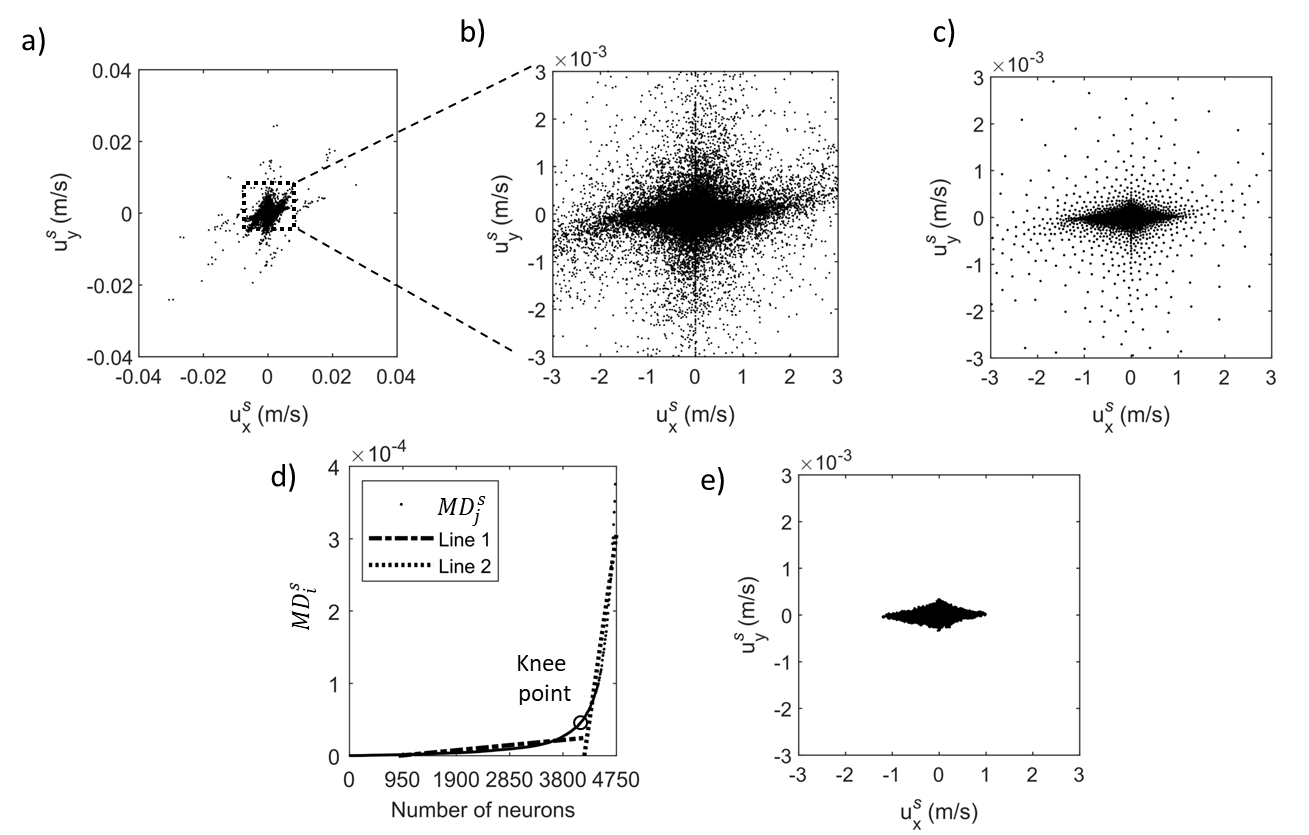


Figure 3. The SOM classification of *s*: (a) (b) zoomed view of (c) the neurons of the SOM applied to (d) average of the distances between a neuron and its neighbors (e) the points classified as uncorrupted in .

The IN corrupted points detected by MSOM algorithm is shown on a short section of the signal (see the figure below).

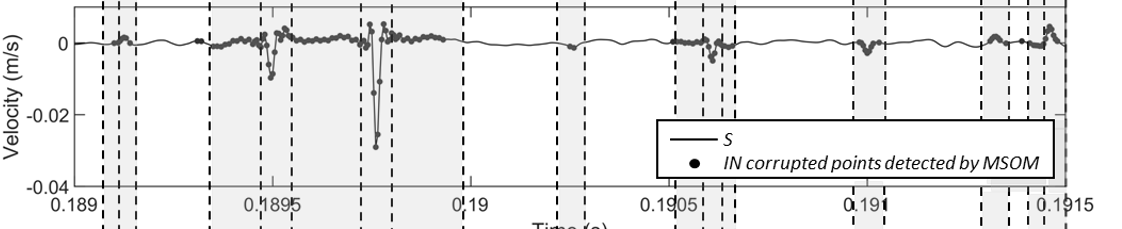


Figure 4. Detected IN corrupted points (by MSOM).

[1] H. Kong and L. Guan, “A neural network adaptive filter for the removal of impulse noise in digital images,” *Neural Networks*, vol. 9, no. 3, pp. 373–378, 1996, doi: 10.1016/0893-6080(95)00128-X.

[2] S. Haykin, *Neural Networks and Learning Machines*. New Jersey: Pearson, 2009.

[3] R. K. Pearson, Y. Neuvo, J. Astola, and M. Gabbouj, “Generalized Hampel Filters,” *EURASIP J. Adv. Signal Process.*, vol. 2016, no. 1, 2016, doi: 10.1186/s13634-016-0383-6.

[4] S. Salvador and P. Chan, “Determining the Number of Clusters / Segments in Hierarchical Clustering / Segmentation Algorithms,” in *16th IEEE International Conference on Tools with Artificial Intelligence*, 2004, pp. 576–584.