Maritime Transport Research 3 (2022) 100063

Contents lists available at ScienceDirect

Maritime Transport Research

journal homepage: www.sciencedirect.com/journal/maritime-transport-research

Filtering harbor craft vessels' fuel data using statistical, decomposition, and predictive methodologies

Januwar Hadi [a], Dimitrios Konovessis [b], Zhi Yung Tay [a,*]

[a] Engineering Cluster, Singapore Institute of Technology, 10 Dover Drive, 534038 Singapore
[b] Department of Naval Architecture, Ocean and Marine Engineering, University of Strathclyde, 100 Montrose St, Glasgow G4 0LZ, United Kingdom

ARTICLE INFO

Keywords:
Fuel oil consumption
Fuel efficiency
Climate change
Data filtering
Statistical analysis
Decomposition
Neural network

ABSTRACT

Filtering is the process of defining, recognizing, and correcting flaws in given data so that the influence of inaccuracies in input data on subsequent studies is minimized. This paper aims to discuss the characteristics of some filtering methods from various topics. Wavelet transform and frequency (Fourier) transform are considered for the decomposition methodologies whereas descriptive statistics is used for the statistical methodology. The Kalman filter and autoencoder neural network are also explored for the predictive methodologies. All the aforementioned methodologies are discussed empirically using two metrics of R-squared and mean absolute error. This paper aims to study the effectiveness of these filtering techniques in filtering noisy data collected from mass flowmeter reading in an unconventional situation i.e., on a tugboat while in operation to measure fuel consumption. Finally, the performance of various filtering methods is assessed, and their effectiveness in filtering noisy data is compared and discussed. It is found that the Haar wavelet transforms, Kalman filter and the descriptive statistics have a better performance as compared to their counterparts in filtering out spikes found in the mass flow data.

1. Introduction

The evidence of climate change has created issues throughout the world. The main factor of climate change is the increasing level of greenhouse gasses in the atmosphere creating a greenhouse effect. One type of greenhouse gasses is carbon dioxide from combustion engine emissions (Miller and Spoolman, 2009; Shaftel et al., 2021).

The negative effect of greenhouse gas in the maritime sector contributes to multiple research initiatives mainly targeting shipping vessels. The initiative analyzes various environmental and operating conditions in shipping vessels, to ensure that it operates at optimum fuel efficiency when travelling in the international ocean. However, it neglects the micro point of view that harbor craft vessels such as tugboats have also contributed significantly to greenhouse gas emissions when burning fuel in assisting the large ocean-going shipping vessels in docking operations. for the Singapore sea straits, tugboats alone led to an overall figure of 993 (tons/day) of CO^2 emissions, equivalent to the CO^2 emission of 78,793 cars in a single day (of Transportation 2018; Leong et al., 2014).

Although recording fuel consumption does not directly reduce carbon emission, it is still very much essential for any empirical study to begin with. The measurements from the record are useful information to determine if a particular operation is consuming a considerable amount of fuel for optimization opportunities (Bialystocki and Konovessis, 2016). One of the major parameters is the fuel flow commonly recorded by using the volumetric flow measurement. However, there are weaknesses in volumetric measurement, as

* Corresponding author.
E-mail address: zhiyung.tay@singaporetech.edu.sg (Z.Y. Tay).

https://doi.org/10.1016/j.martra.2022.100063
Received 15 February 2022; Received in revised form 18 May 2022; Accepted 20 May 2022
Available online 27 May 2022

Fig. 1. Case study vessel, POSH Grace tugboat.

Table 1
Vessel specifications.

Main particulars	Value
Length overall (LOA)	29m
Displacement	665tonnes
Maximum speed	12 knots
Main engines	NIIGATA 6L26HLX
Number of engines	2
Total BHP	4000BHP
Type of propulsion	Azimuth pod
Number of propulsors	2

marine fuel of certain mass will vary in volume at different temperatures. The effect of volume expansion may be exaggerated in tugboat and bunker vessels as these vessels have to carry a considerable amount of fuel, especially around the tropic region where the temperature varies widely between night and day. In addition to the volumetric expansion, the accuracy of the fuel measurement is also affected by the varying thermal expansion of the fuel. Given the volume and thermal expansion premises, measuring the fuel flow by its mass is thus simpler i.e., by having to deal with only one parameter of mass flow rather than two parameters of volume flow and temperature. However, noises sometimes occur in mass flowmeter records for vessels as most of the mass flowmeters are designed for use in facilities in which vibration is either at a minimum or non-existent at all. Therefore, using a mass flowmeter in a non-stationary setting such as tugboats may compromise its performance. One source of compromises is the engine room vibration that disrupts the Coriolis effect which the meter uses to make measurements (Raszillier and Durst, 1991; Chiang et al., 2015).

Another source of compromise is electromagnetic interference due to the presence of electric generators and solenoids in the engine room where the mass flowmeters are installed (Kaur et al., 2011). The energized coils from the generators and solenoids may create interference to the electromagnetic field used by the Coriolis mass flowmeter in measuring the vibration of the Coriolis effect from the fluid flow. Furthermore, in any case, a suspected anomaly is not guaranteed to be an error and if so, the source of the error and means in recreating the compromised information have to be sought.

In view of the noises that are present in the mass flowmeter data, data preprocessing has to be performed prior to the data being used for further analysis. The filtering approaches discussed in this paper apply and compare five denoising methods from various topics, i.e., wavelet transform, optimal estimation, frequency decomposition, descriptive statistics, and artificial neural network. Wavelet transform is an effective method to denoise raw data whose noise is in the form of spikes (Ehrentreich and Sümmchen, 2001). Kalman filter (KF) is from the optimal estimation camp that utilises a series of iterations for filtering the subsequent data point using information from the previous iteration or data point (Kim and Bang, 2019). The Fourier transform (FT) algorithm transforms the signal from the time domain to the frequency domain which is represented as power spectral density (PSD) from which the noise frequency is targeted and attenuated (Nussbaumer, 1981). Descriptive statistics supplies the techniques that help to condense large datasets by using tables, graphs, and summary measures. Descriptive statistics could be of immense importance because it provides efficient and effective methods for summarizing and analyzing information (Mann and Lacke, 2020). Finally, from the machine learning camp, this paper also considers the artificial neural network (ANN) which is a form of autoencoder network, as means of filtering methodology.

The result from the application of each method will be evaluated individually using two metrics, i.e. the mean absolute error (*MAE*) and R-square error (*R*2), and be compared among them to discuss and emphasize the characteristics of each method.

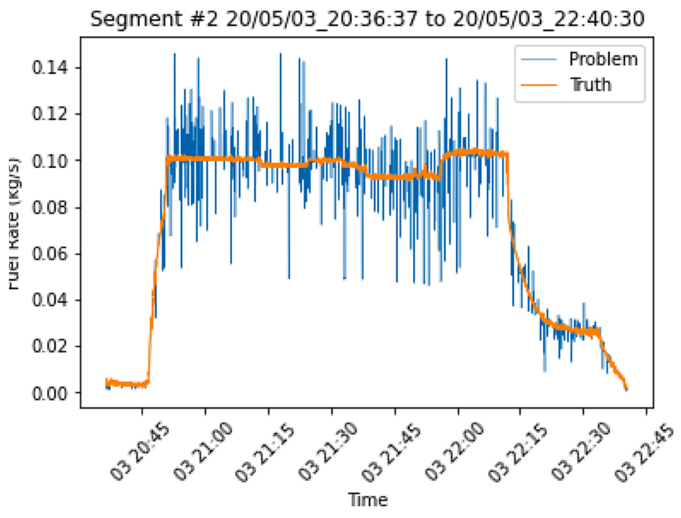


Fig. 2. Plot of Raw Data & Ground Truth.

2. Harbor craft vessel operational data: fuel consumption

2.1. Operational data

The operational data were collected from the data acquisition device installed onboard the traditional diesel-powered tugboat as shown in Fig. 1. The vessel specifications are given in Table 1. The main operational activity of the tugboat consists of anchoring, assisting large vessels in docking, and piloting around the southern sea of Singapore. The purpose of collecting the operational data was to conduct a research study on predicting the fuel rate to achieve fuel efficiency through data analytics and neural network modeling (Z.Y. Tay et al., 2021; Z.Y. Tay et al., 2021).

To facilitate the collection of the operational data, two Coriolis mass flowmeters were installed to record the fuel consumption from both the port and starboard main engines. The fuel consumption of the auxiliary engines (the electric power generators) is not considered in this case study, as it is significantly small compared to the main engines. Auxiliary engines are also highly predictable by the number of operational hours, due to their constant rate of fuel consumption. The operational data acquired over a six-month duration from April 2020 to September 2020 are used for this case study. Overall, the operational data composed of around 1.2 million data points were collected at one-second intervals.

The data collected from the mass flowmeter are prone to inaccuracy or error in the Coriolis effect measurement due to external factors. The Coriolis effect causes the fluid to deflect at different angles when passing the curved channel. In addition to the deflection angles, the curved channel also oscillates at varying frequencies in the process of determining the mass density of the flowing fluid. As a result, both the deflection and oscillation are highly susceptible to vibrations and electromagnetic interferences, thus causing the presence of noises in the raw data. It is to note that investigating the source of the noise is not part of the scope of this paper; instead, this paper aims to filter out the noise from the recorded mass flow data. The benefit of the filtering approach is that it is a software correction that requires only virtual resources. It would require much fewer resources if it were to be compared with hardware correction. Hardware correction requires resources for alteration or modification to the physical setup specific to a particular tugboat (i.e., rerouting fuel line, vibration reduction/dampening, or electromagnetic shielding). Another benefit of the filtering approach is that once established, it applies to other tugboats with similar problems regardless of make and design. It is noted here that the collected mass flow data is stored in segments. Each segment length is based on a duration of a job. Hence, each segment may have a unique length.

2.2. Ground truth

A sample of one segment of the raw data is plotted in Fig. 2. The raw data in blue of Fig. 2 (Problem) shows the noisy fuel rate versus time (day hour: minute). The noise is in form of spikes in either up or down direction throughout the segment. Fig. 2 also shows the clean data (Truth) and is plotted as the orange curve. The ground truth will be used as a benchmark with which the result of each filtering method is evaluated. The evaluation uses two types of key performance index (KPI) and is discussed in the next section.

During data collection (logging), irregularities were found at regular intervals. Logging at regular intervals is known as pooling mode. The irregularities are similar to the plot shown as the *Problem* in Fig. 2. Simultaneously, while using other logging modes which are event-based, it is discovered that there are invalid measurements during the second-pooling interval. The invalid measurement appears as a saturated number that is visually recognizable. Hence, the presence of invalid measurements invalidates other valid measurements in an interval. To generate the *Truth* in Fig. 2, the logging must be performed in event-based mode, and the invalid measurements removed and the valid measurements resampled to the pooling interval.

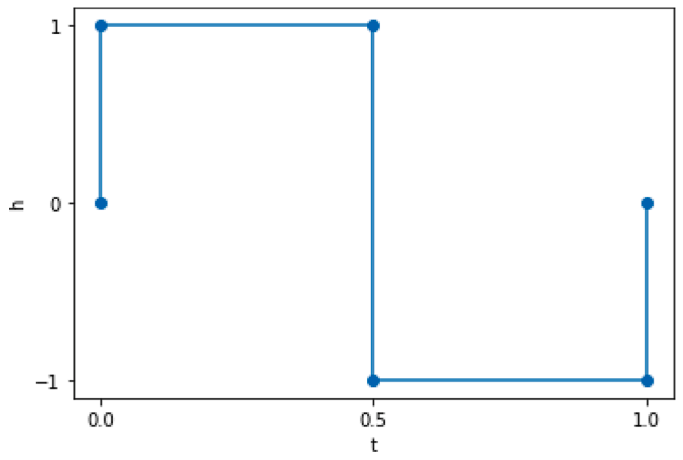


Fig. 3. Haar Wavelet.

During deployment, data logging in pooling mode is preferred over event-based mode. Within an interval, multiple events may present that trigger the creation of multiple entries per interval. This may create an overly large amount of data that burdens computing devices on the tugboat as large data requires more resources to record as well as to process.

2.3. Key performance index (KPI)

This sub-section discusses the two KPIs (also referred to as metrics), i.e., the *MAE* and *R2*, that will be used to assess the performance of each denoising method.

2.3.1. Mean absolute error (MAE)

The *MAE* score is calculated as the average of the absolute error values, as the name suggests.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}| \tag{1}$$

where n is the number of data points, y_i is the i-th value of ground truth, and $\widehat{y_i}$ is the i-th denoised value. As *MAE* is the measurement of error, the lower the *MAE* metric the higher the prediction accuracy is. The ideal *MAE* value is zero (Chai and Draxler, 2014).

2.3.2. R-Squared (R2 score)

In computing the *R*2 score as indicated by (4), the maximum score of 1.0 is subtracted with *RSS* (2) over *TSS* (3) where *RSS* is the residual sum of squared error whereas *TSS* is the total sum of squared difference of the Truth data point (y_i) from its mean ($\overline{y}$). Here, an error is defined as the difference in Truth data point y_i and denoised data point $\widehat{y_i}$.

*R*2 score uses the fluctuation of the valid data points to dynamically scale the effect of errors. In the case of extreme fluctuation throughout the entire data points (n), *TSS* will become very high. A very high *TSS* reduces the importance of *RSS* as far as *R*2 is concerned. This is usually true when the valid data points themselves are noise-like.

The ideal *R*2 score is when every $\widehat{y_i}$ is equal to y_i, regardless of how fluctuative or flat the data points are. In such a case, *RSS* is zero and the *R*2 score is at the maximum of 1.0. A Higher *R*2 score indicates better metrics. Therefore, the *R*2 is very useful to calculate the similarity between two sets of values (Mittlböck and Heinzl, 2001) where the *R*2 emphasizes more on accuracy than precision.

$$RSS = \sum_{i=1}^{n}(y_i - \widehat{y_i})^2 \tag{2}$$

$$TSS = \sum_{i=1}^{n}(y_i - \overline{y})^2 \tag{3}$$

$$R^2 = 1 - \frac{RSS}{TSS} \tag{4}$$

3. Methodology

This section explains five filtering methodologies from various topics, i.e., the Haar wavelet transform (HWT), Kalman filter (KF), Fourier transform (FT), descriptive statistics denoising (DSD) and artificial neural network (ANN).

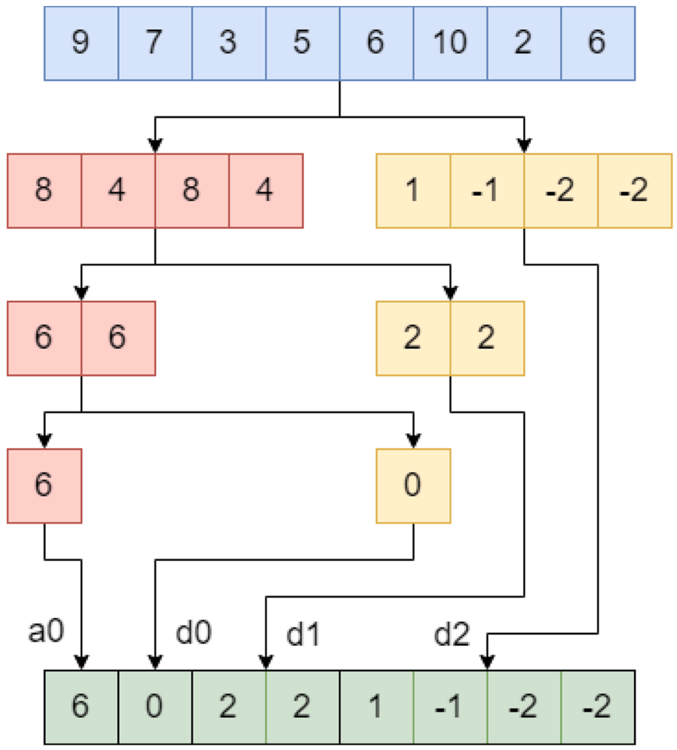


Fig. 4. Haar Transform/Decomposition.

3.1. Haar wavelet transform

The Haar wavelet transform is one of the simplest wavelet transforms (Stanković and Falkowski, 2003). Due to its simplicity, it is computationally less demanding. Despite its simplicity, it has an orthogonal property to decompose a signal sequence into transform elements for denoising (Luisier et al., 2009). The mathematical representation of the Haar Wavelet is expressed in (5), and the Haar wavelet is shown in Fig. 3.

$$h(t)=\begin{cases} 1 & 0 \le t < \frac{1}{2}, \\ -1 & \frac{1}{2} \le t < 1, \\ 0 & otherwise. \end{cases} \tag{5}$$

In the HWT, the length of the data points to be transformed is determined by the power of 2 (i.e., 0, 1, 2, 4, 8, 16 and so on). The length of the transform is the same as the original signal. The transform representation consists of one average value and the detail elements at various levels. The number of detail levels is determined by the power (exponent) of the length of the original signal. E.g., in the case of signal length of 8, there are 3 levels of details (i.e., $8 = 2^3$), and the level of details is indexed from 0.

Fig. 4 shows an example of a signal in blue being transformed into a set of decomposed green signals (transform). This is done by using the HWT given in (5) that involves a repeat process of computing the mean of pair sum and mean pair of difference at various levels to finally produce the filtered signal in the green cell.

In Fig. 4, the cells in red are the sum of pair averages while the cells in yellow are the difference of pair averages from the block above them.

E.g. 8 – 4 – 8 – 4 is derived from

$$\frac{9+7}{2} \quad \frac{3+5}{2} \quad \frac{6+10}{2} \quad \frac{2+6}{2}$$

while 1 – 1 – 2 – 2 is derived from

$$\frac{9-7}{2} \quad \frac{3-5}{2} \quad \frac{6-10}{2} \quad \frac{2-6}{2}$$

Mathematically, the transformation (T) could be carried out by applying the dot product of Haar Matrix (H) with the signal (S) (Zhang, 2019).

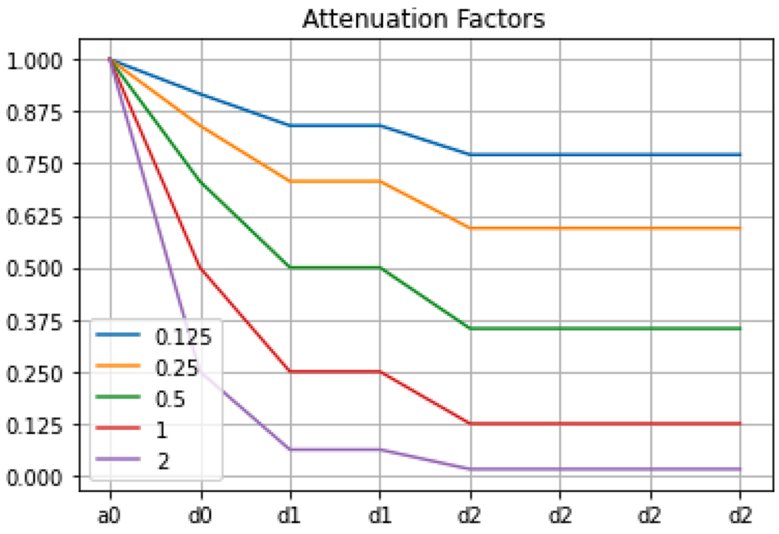


Fig. 5. Comparison of different attenuation factors from (6).

$$H = \begin{bmatrix} 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 \\ 1/8 & 1/8 & 1/8 & 1/8 & -1/8 & -1/8 & -1/8 & -1/8 \\ 1/4 & 1/4 & -1/4 & -1/4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/4 & 1/4 & -1/4 & -1/4 \\ 1/2 & -1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & -1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & -1/2 \end{bmatrix}, S = \begin{bmatrix} 9 \\ 7 \\ 3 \\ 5 \\ 6 \\ 10 \\ 2 \\ 6 \end{bmatrix}$$

$$T = H \cdot S = [\ 6 \quad 0 \quad 2 \quad 2 \quad 1 \quad -1 \quad -2 \quad -2\ ]^T$$

The last level of detail component (d2 green cell in Fig. 4) is exclusively the mean of difference. Hence, the last level of detail component (d2 in this case) which makes up the half end of the transform output, holds the most information on the fluctuation/noise in the signal. On the other hand, the other detail levels have a combination of the mean of sum and the mean of difference. The closer the detail element is to the average element (a0), the least information of fluctuation it holds.

Hence, to control the fluctuation, an Attenuation Matrix (A) given in (6) is applied to the transformed signal. By tuning the attenuation factor (f), the aggressiveness of attenuation of noise is made possible. Fig. 5 shows the values of A at various attenuation factors (f). The idea is to apply the highest reducing factor to the last level of detail component, and decrease towards the average. Note that the average component should not be altered as it holds the essential trend information.

$$A = [\ 1/2^{0f} \quad 1/2^{1f} \quad 1/2^{2f} \quad 1/2^{2f} \quad 1/2^{3f} \quad 1/2^{3f} \quad 1/2^{3f} \quad 1/2^{3f}\ ]^T \tag{6}$$

A dot product of the Haar matrix (H) with the signal (S) produces the transform (T). The transform is attenuated by applying the element-wise product with A (Horn, 1990). Finally, it is inverse transformed back (dot product with H^{-1}) to the signal domain as a denoised signal (S_A).

$$S_A = H^{-1} \cdot (T \circ A) = H^{-1} \cdot [(H \cdot S) \circ A] \tag{7}$$

Fig. 6(a) shows the comparison of T for example in Fig. 4, with and without the application of A. Fig. 6(b) compares S with S_A. The higher the attenuation factor f, the closer to average (a0) the S_A is.

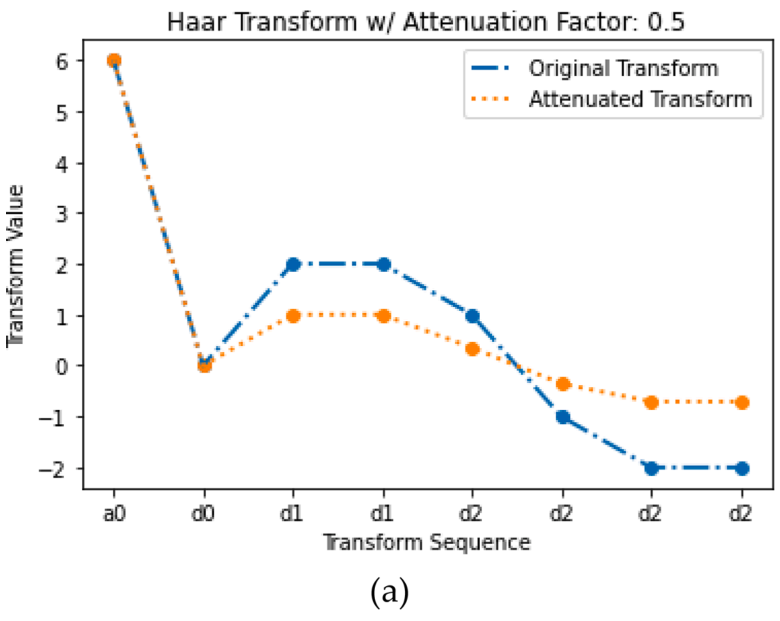
It is to note that the example shown in Figs. 5, 6 and 7 uses a signal that is of 8 elements length. In the Results and Discussion section (Section 4), the signal length is referred to as the subset window size (subset size) at various lengths.

3.2. Kalman filter

Kalman filter (KF) uses a series of iterations to perform the filtering. Each data point requires one iteration prior to moving to a subsequent data point for another iteration. The subsequent iteration requires information in form of covariance coefficients as well as a prior from the previous one – a method used by KF to gain insight from the previous iteration to denoise the current one.

Each iteration consists of three stages: prediction, estimation and update. Fig. 7 shows the Kalman filter (KF) algorithm diagram (R. Labbe, 2018) for one iteration, where the objective is to reach a new estimation (x_t).

Prediction:



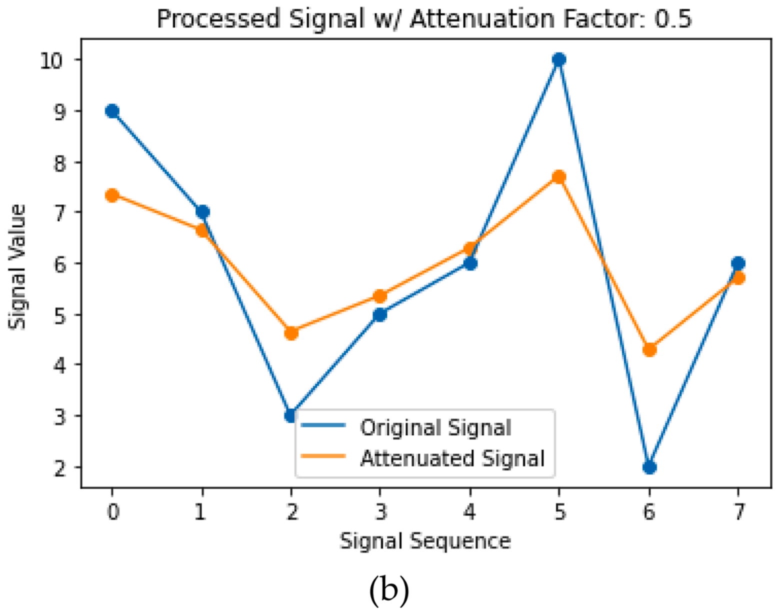


Fig. 6. Comparisons of (a) Original vs attenuated transform, (b) Original vs denoised/attenuated signal.

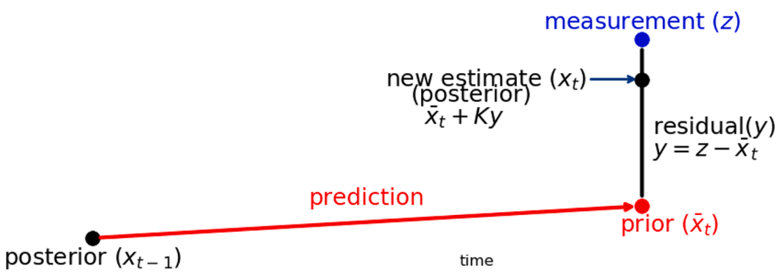


Fig. 7. Kalman filter algorithm diagram.

$$\bar{x}_t = F \cdot x_{t-1} \tag{8}$$

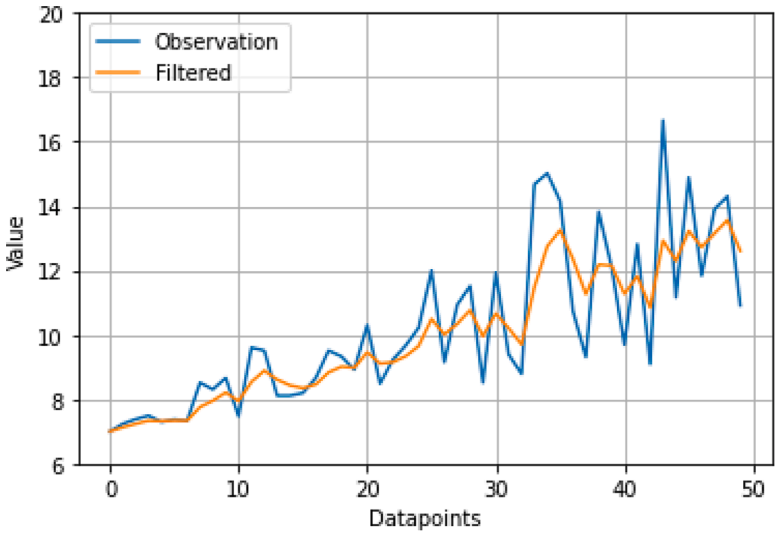


Fig. 8. Kalman filter on test data.

$$P_t = P_{t-1} + Q \tag{9}$$

<u>Estimation:</u>

$$K = P_t \cdot (P_t + R)^{-1} \tag{10}$$

$$y = z - \overline{x}_t \tag{11}$$

$$x_t = \overline{x}_t + K \cdot y \tag{12}$$

<u>Update:</u>

$$P_{t+1} = P_t \cdot (1 - K) \tag{13}$$

KF algorithm starts with prediction to generate a prior ($\overline{x}_t$) from state transition (F) and the previous posterior (x_{t-1}), as also denoted by (8). F is a linear function, usually represented in a matrix. However, in this example, F is statically set as a scalar of 1.0, i.e., there is a 100% likelihood that the current state (prediction) does not change from the previous state. Concurrently in the prediction stage, the covariance (P_t) is calculated from the previous covariance (P_{t-1}) and process uncertainty (Q), as shown in (9).

Both $\overline{x}_t$ and P_t from the prediction stage are used in the estimation stage to generate Kalman gain/scaling (K) and residual (y). In (10), K is in a factor of P_t, and is derived from measurement uncertainty (R). In the case where R is a positive number, K scales P_t down, and vice versa. y according to (11) is a simple difference between measurement/observation (z) and $\overline{x}_t$ which was derived from prediction. Eventually, the new estimate (x_t) is a linear function of y, with K being the gradient and $\overline{x}_t$ being the offset as shown in (12).

Finally, a new covariance coefficient (P_{t+1}) is calculated. In (13), P_{t+1} is calculated from P_t that is factored by $(1 - K)$. The next iteration starting from the prediction stage will take place for as long as there is a new observation (z). P_{t+1} will be used as P_t and x_t will be used as x_{t-1} in the next iteration. In a typical implementation, the Kalman filter uses a measurement function (H) which is a linear function to convert the state into a measurement (Laaraiedh, 2012). For simplicity, this paper drops H by asserting an assumption that H is a static scalar of 1.

Fig. 8 shows the series of observations that has a characteristic of an upward trend and ripples riding on top of the trend. KF algorithm manages to reduce the ripples to a less erratic extent while maintaining the trend. This is similar to the smoothening effect. The smoothening effect could be used to denoise noisy signals (that have noise in form of spikes as shown in Fig. 2). This paper uses the `filterpy` software with the Kalman filter sub-module to process the denoising (R. Labbe, 2018).

3.3. *Fourier transform*

Similar to the Haar wavelet transform, the Fourier transform allows the filtering of noisy signals from the raw data by altering the transform. Fig. 9 shows sample segment #2 with the spikes representing the high-frequency noisy signal. It is not immediately obvious at which high frequencies the noises are. However, the high-frequency elements can be discriminately reduced in a controlled way. A window could be applied to the frequency response. The two example types of windows that could be used to reduce high-frequency signals while keeping the low-frequency signals are shown in Fig. 10 (Huang et al., 2011). Also shown in Fig. 10 is the various Gaussian and exponential windows with varying width, i.e., standard deviation (σ or std) for Gaussian window $w_g(n)$ and tau or τ for Exponential

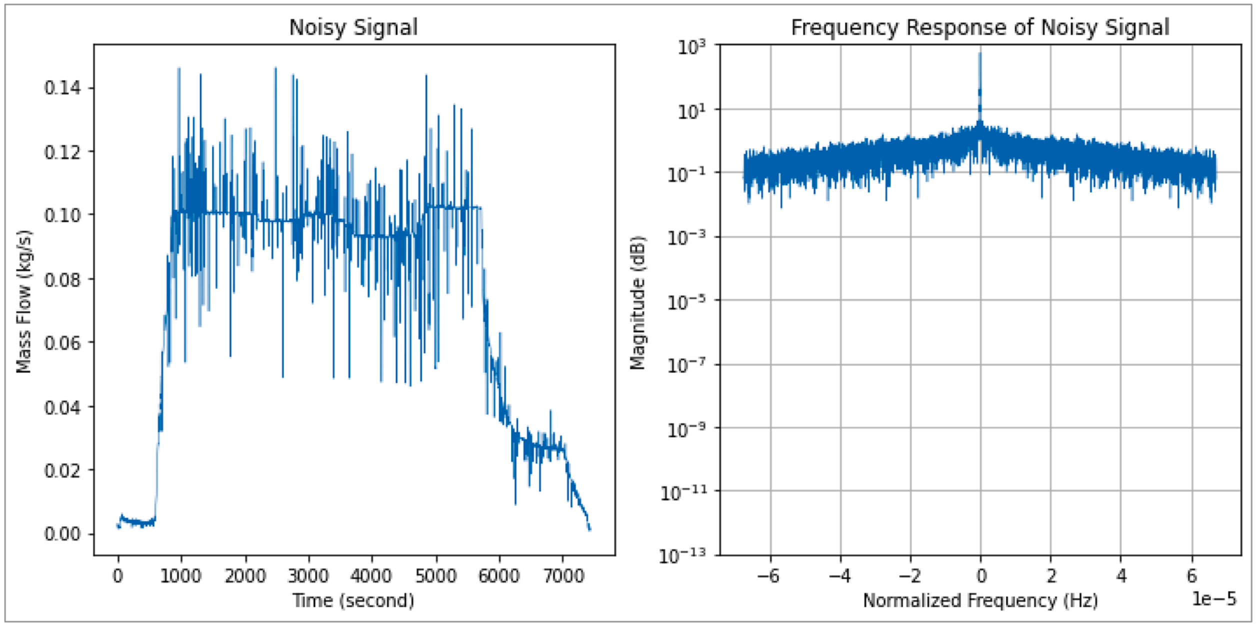


Fig. 9. Application of FFT to time-series noisy signal (left) to obtain its frequency response (right).

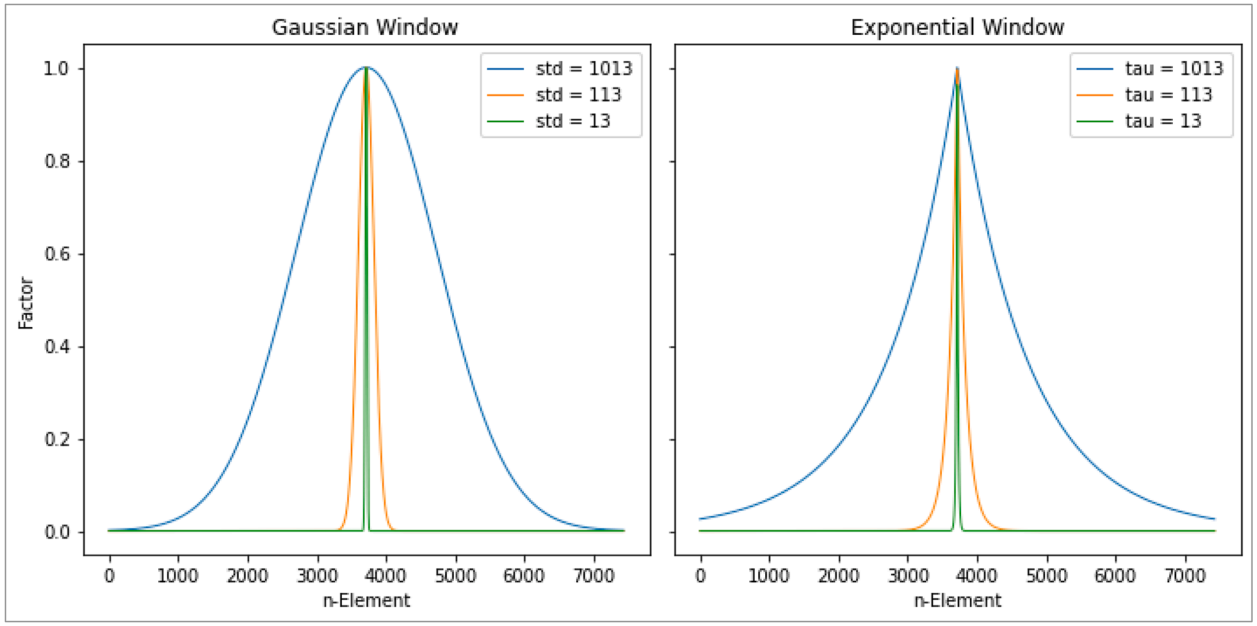


Fig. 10. Comparison of Gaussian and Exponential windows for FFT.

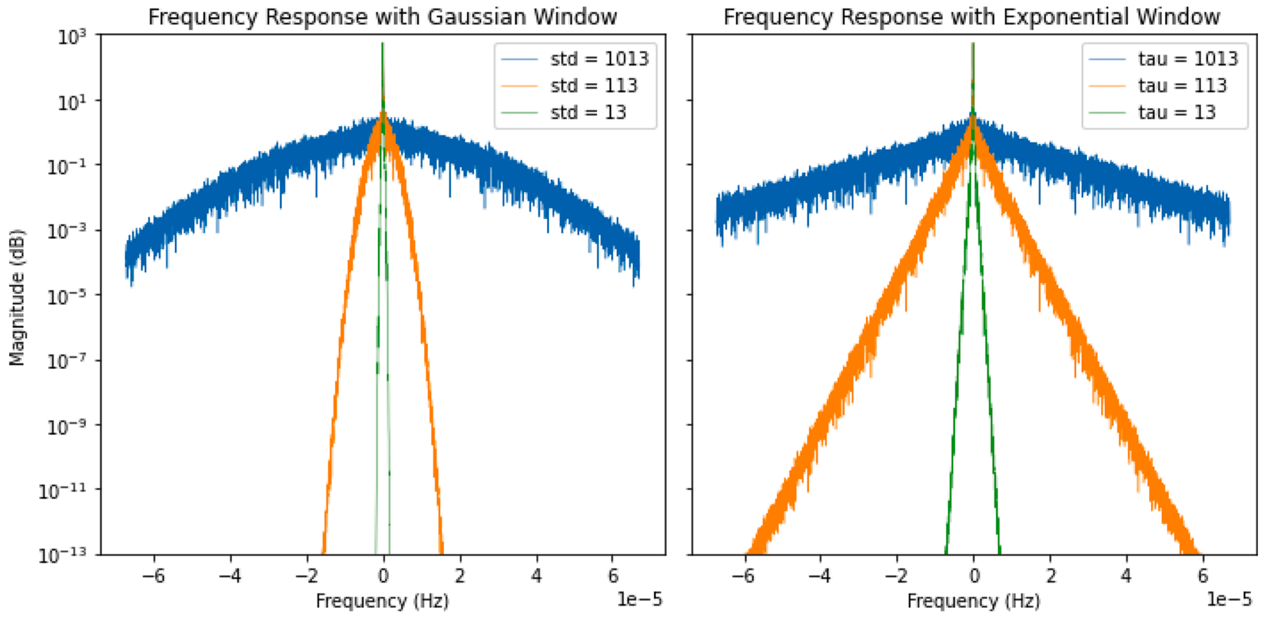


Fig. 11. Frequency response comparison after applying windows.

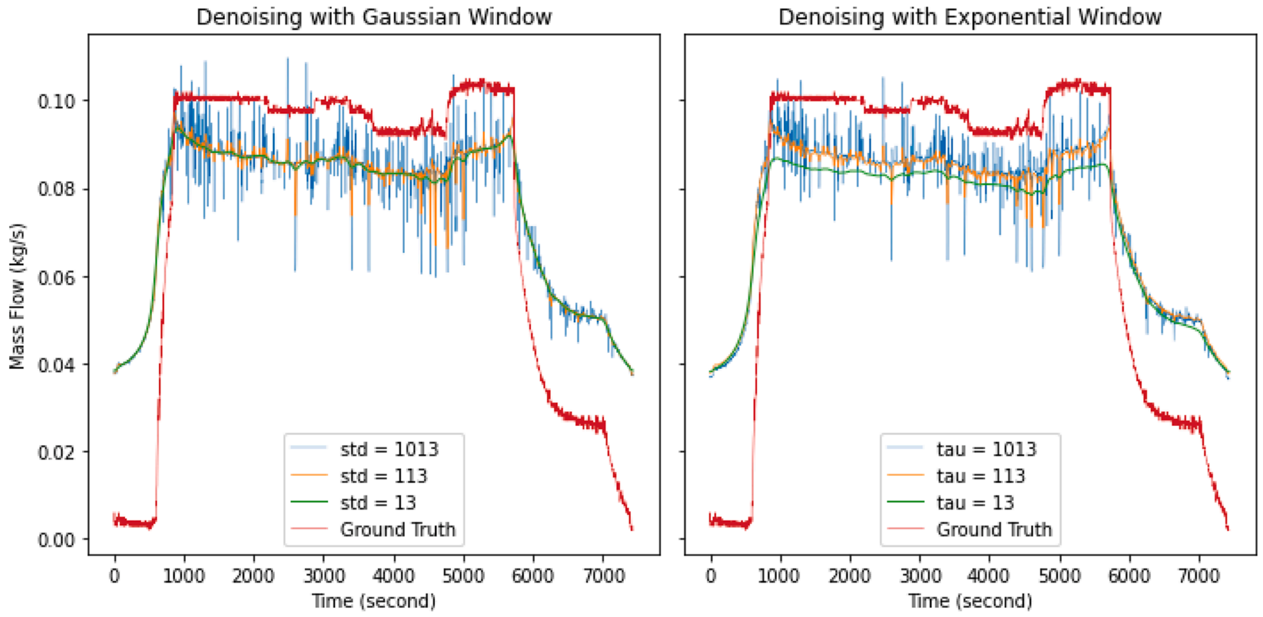
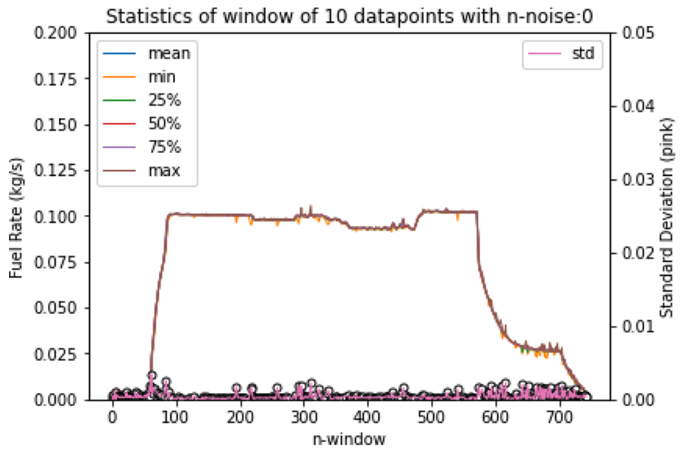
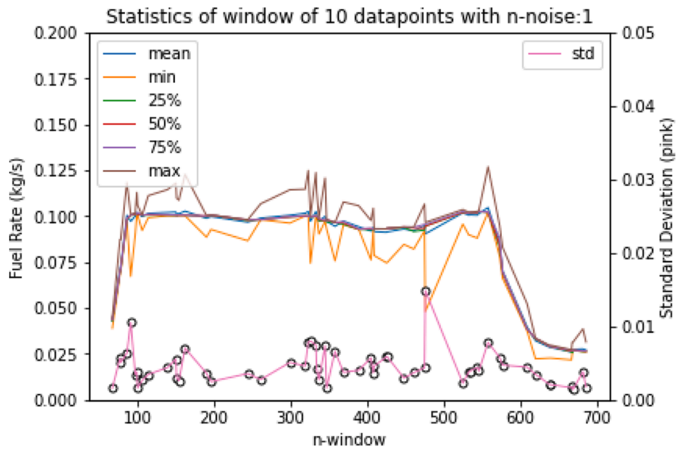


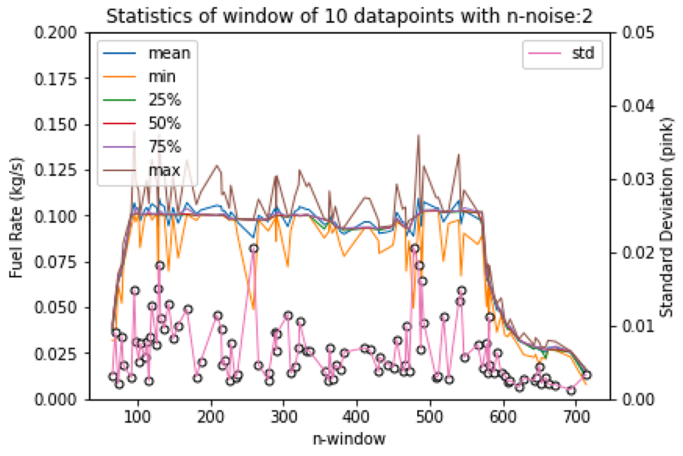
Fig. 12. Comparison of denoising from different configurations from two window types.



(a)



(b)



(c)

Fig. 13. Comparison of window statistics by number of noises. Relative tolerance = 5%, and absolute tolerance = 0.005.

window $w_e(n)$.

$$w_g(n) = e^{-\frac{1}{2}\left(\frac{n}{\sigma}\right)^2} \tag{14}$$

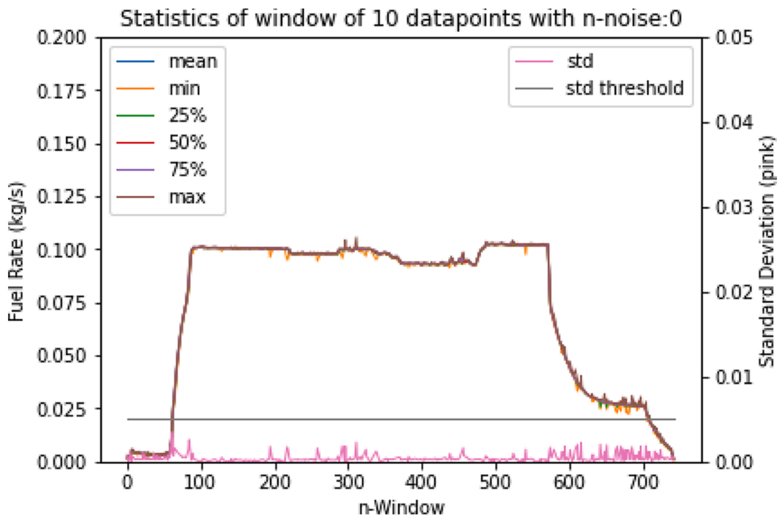


Fig. 14. The zero noise window statistics with threshold.

$$w_e(n) = e^{-|n-center|/\tau} \tag{15}$$

Eqs. (14) and (15) are the equations for Gaussian and exponential windows respectively, where the narrowness of each window's center region is determined by σ and τ. The small σ and τ (value = 13) produce narrower center regions if compared with (value = 1013). Fig. 11 shows windows of various σ and τ. The windowing application multiplies the frequency response of Fourier transform to a window element-wise. Afterwards, by performing the inverse transform, the frequency response is transformed back to the time-series response. As shown in Fig. 12, narrow Gaussian and exponential windows remove more noise. However, applying the Fourier transform scales the entire signal. This is due to the period of the transformation being set as one whole segment. It is noted that Fig. 12 is for demonstration purposes only. In the Results and Discussion section (Section 4), the period is reduced to a much smaller size and is referred to as subset size.

3.4. *Descriptive statistics denoising*

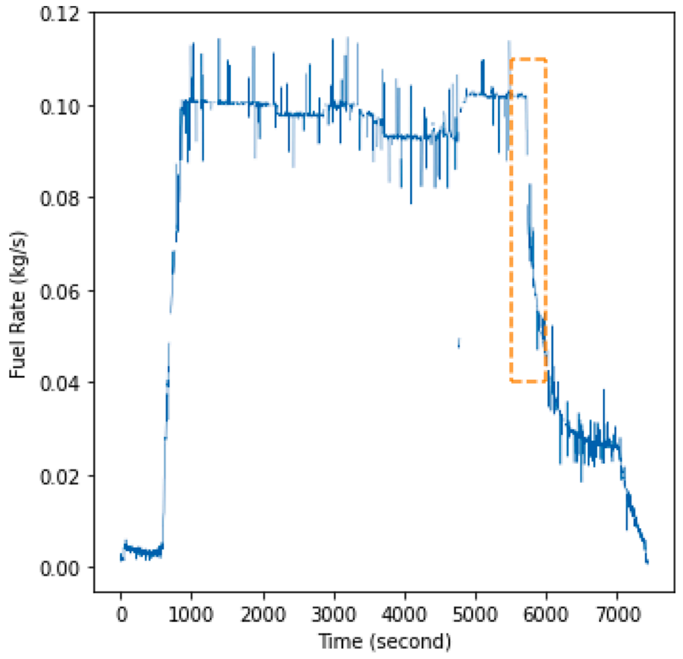
Descriptive statistics summarize the dataset into a set of new measures. One measure indicates central tendency, such as mean and median, while another indicates dispersion or spread, such as percentiles and standard deviation. Descriptive statistics denoising (DSD) aims to utilize these descriptive measures (Fisher and Marshall, 2009). The descriptive measures help make anomalies stand out in a pattern. A simple condition could be determined and applied to rule out the anomalies.

One set of descriptive measures over the entire dataset could become an oversimplification. By doing so, the entire dataset is reduced to one set of descriptive measures such as mean, median, minimum, maximum, standard deviation, etc. which is not too useful as far as denoising is concerned. Descriptive measures for a subset (a window) of the dataset at an adaptive interval could produce more useful information (Suoranta and Estola, 1991). The information is in the form of a sequence of descriptive measures sets instead of just one set of values. This paper, nonetheless, focuses on a regular window at regular intervals (or stride) instead of an adaptive one. The sequence of descriptive measures could be separated into different tables by the number of suspected noises.

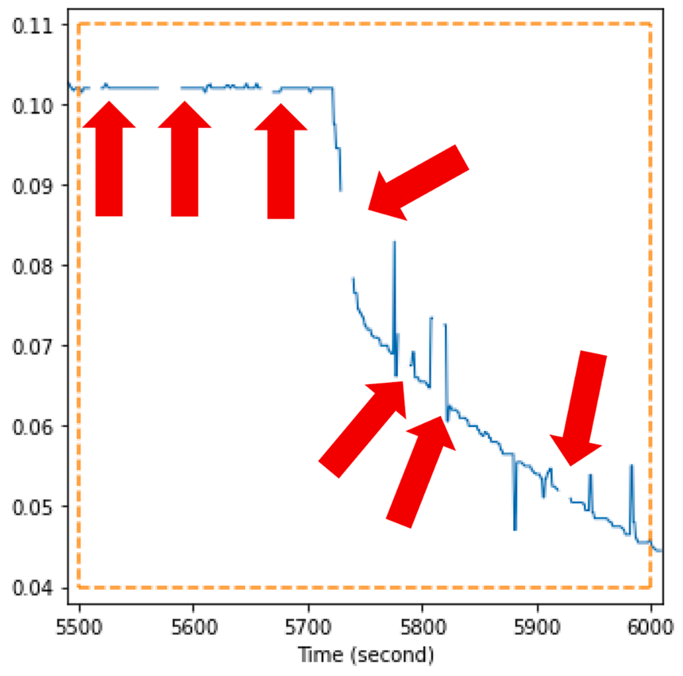
The suspected noise in the subset (or window) is detected by the use of tolerance. To locate a suspected noise, each data point in a particular window is compared with the next one for its difference, represented as a percentage. The percentage is qualified by a relative tolerance. In addition to relative tolerance, absolute tolerance must also be used. It is because, for a point whose value is zero, any number relative to zero is the number itself. Hence, by using relative tolerance alone, the tolerance must be set near 100% which is not very effective. These tolerances aim to determine if the subsequent data point is a discontinuity. Once discontinuity is determined, it is suspected as noise.

The subsets (windows) are grouped by the number of suspected noises. Fig. 13 plots the groups by n-noise into the series of n-window (x-axis) and the descriptive measures/statistics, i.e., mean, percentiles (25%, 75%), median (50% percentile), minimum (min), maximum (max), and standard deviation (std). To produce Fig. 13, the relative tolerance is 5% and the absolute tolerance is 0.005. In Fig. 13(a), which shows that the suspected noise is zero, the mean and median (50%) are very close to the ground truth. This indicates that both relative and absolute tolerances values are doing a good job in separating good data points from noisy data points. Otherwise, if relatively high fluctuation appears at n-noise = 0, more stringent tolerances must be applied. However, if the number of data points at n-noise = 0 is too sparse (data points are further apart as shown in Fig. 13(b)), the tolerances could have been set too stringent. It has resulted in non-noise data points being suspected as noise. The determination of tolerance requires an intuitive judgement or experience.

With insight from Fig. 13, a conditional threshold on one or more statistics can be set. The conditional threshold functions as a rule that accepts windows in which the number of suspected noises is zero. Fig. 14 uses the results from Fig. 13(a) where the *std threshold* is



(a)



(b)

Fig. 15. Results prior to mitigation and interpolation. (a) Denoised signal before mitigation and interpolation. (b) The zoomed-in perspective of (a) in orange dashed rectangle. Red arrows point to the gaps of discarded windows.

set at 0.005. The statistic which is used as a conditional threshold is not constricted to standard deviation only. Other statistics, such as median, could be used (Justusson, 1981). Nonetheless, in the sample case, the *std threshold* is the most convenient way to apply a conditional threshold. Refer to Fig. 14. All the std (pink) values fall under the *std threshold* of 0.005.

All windows in the signal (or noisy data) are defined as S_w by (16), with v being the total number of windows. The data points in any window are defined as S_i by (17), with j being the total number of data points in any window. Hence, each of S_i is a member of S_w. In (18), a particular window is accepted as is, if the standard deviation in the particular window σ_w satisfies the condition by the threshold

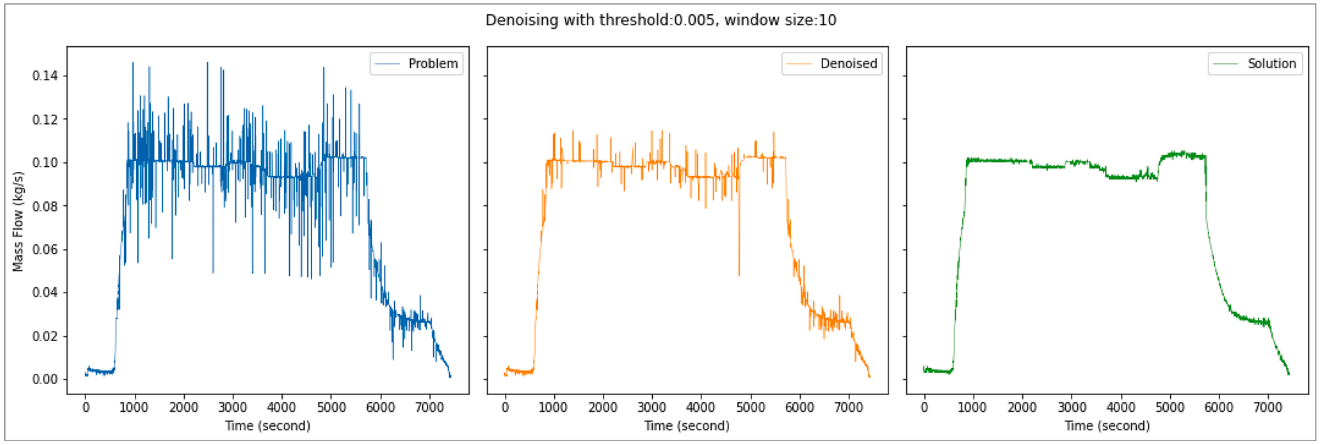


Fig. 16. The comparison between problem, denoised, and solution.

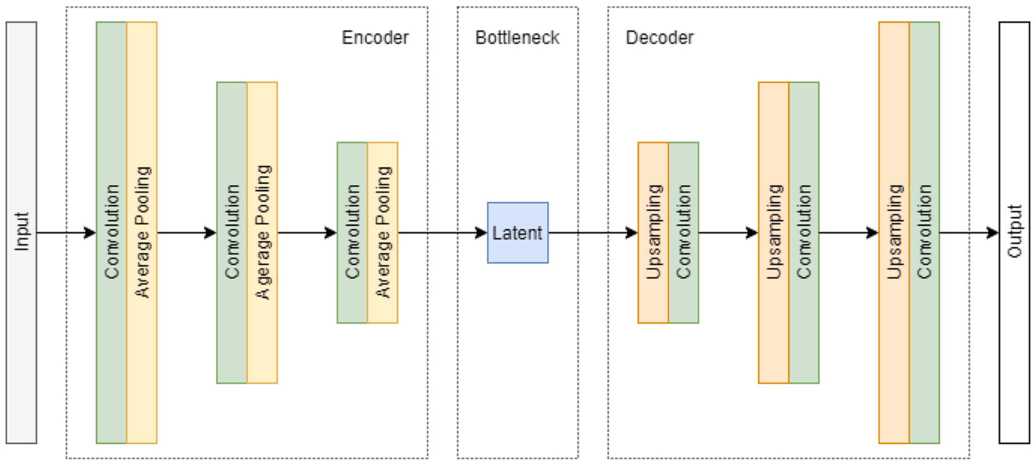


Fig. 17. Autoencoder architecture.

γ. Otherwise, the window is discarded altogether, creating an undefined period in the denoised signal.

$$(S_w)_{w=1}^{v} = (S_1, S_2, S_3, \ldots, S_v) \tag{16}$$

$$(S_i)_{i=1}^{j} = (S_1, S_2, S_3, \ldots, S_i) \tag{17}$$

$$S_i = \begin{cases} S_i, & \sigma_w \le \gamma \\ undefined, & \sigma_w > \gamma \end{cases} \tag{18}$$

The effect of undefined periods is discarded windows appearing as gaps (x-axis), as shown in Fig. 15. To mitigate these gaps, the DSD is to perform point-wise qualification (as opposed to window-wise qualification up to this point). In (19) and (20), α and β are the lower and higher limits respectively and are set by either S_{-i} or S_{+i}. S_{-i} is the last data point from the previous window, while S_{+i} is the first data point of the next window of any discarded S_i (except the first and the last in S_w). In (21), S_k is each data point in the discarded S_i. It is qualified/accepted to fill the points in the discarded window if the value is between α and β. Otherwise, the value remains undefined. The idea is to perform a second screening in every discarded window, to accept values that do not exceed the values of adjacent windows. Finally, a simple linear interpolation is used to fill the remaining undefined data points which may originally be the noise.

$$\alpha = \begin{cases} S_{-w}, & S_{-w} < S_{+w} \\ S_{+w}, & otherwise \end{cases} \tag{19}$$

$$\beta = \begin{cases} S_{-w}, & S_{-w} \ge S_{+w} \\ S_{+w}, & otherwise \end{cases} \tag{20}$$

$$S_k = \begin{cases} S_k, (S_k \ge \alpha)\Lambda(S_k \le \beta) \\ undefined, otherwise \end{cases} \tag{21}$$

Fig. 16 shows the comparison of the original problem versus the denoised and expected solution. It is obvious from Fig. 16 that there are still some spikes in the denoised signal.

3.5. Artificial neural network

The Artificial Neural Network (ANN) mimics the natural working of biological neural networks. The network comprises neuron that is arranged in layers. The layers may be multi-dimensional and are arranged as tensors. The tensor holds the coefficient of a function, also known as the activation function. During training, the algorithm feeds training inputs from both ends of the network (similar to a pipeline). Given an *X* value, the pipeline is to predict the *y* value. The algorithm iterates over the training inputs to update values in the tensors so that the predicted *y* is as close as possible to the original *y*. The evaluation of the predicted and the original is done using a loss function. A well-designed and well-trained ANN could produce a very good approximation to a function for a solution (also known as ground truth). The most common application of ANN is a regression model where the regression model tries to approximate a function (Jain et al., 1996).

The one specific design of artificial neural network (ANN) that this paper uses is the autoencoder (Bank et al., 2020). It features an encoder and decoder in the same network. The encoder compresses autoencoder input to smaller dimensions of the original information. Afterwards, it is immediately decompressed by a decoder. The decoding is a reconstruction attempt to get back the original

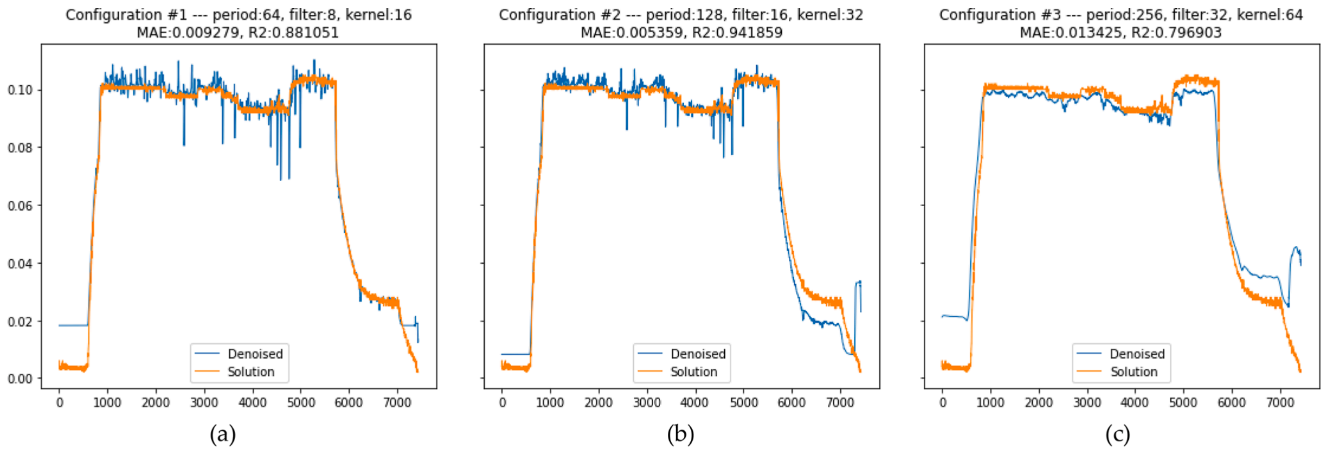


Fig. 18. Results with different combinations of configuration.

Table 2
Haar wavelet transform configuration comparison.

No.	Subset Size	Stride	Factor	*MAE*	*R*2	Processing Time (s)
1	8 (2^3)	0.25	0.125	0.002300	0.720	31
2	8 (2^3)	0.25	8	0.002288	0.761	31
3	8 (2^3)	1	0.125	0.002304	0.713	8
4	8 (2^3)	1	8	0.002294	0.756	8
5	64 (2^6)	0.25	0.125	0.002231	0.754	48
6	**64 (2^6)**	**0.25**	**8**	**0.002267**	**0.820**	**48**
7	64 (2^6)	1	0.125	0.002236	0.750	12
8	64 (2^6)	1	8	0.002231	0.806	12

information. By using the autoencoder, the variant of original information could be expected. It is almost impossible to get an identical result to the original information due to the loss of information during compression and decompression. Hence, it is possible to train an autoencoder to purposely *lose* a piece of certain information (noise).

A technique that uses a pattern filter to scan a presence of a pattern in data is called a convolution (O'Shea and Nash, 2015). The convolution also makes the pattern in the data more prominent to become a feature. Usually, the convolution is followed by pooling where only the average, minimum or maximum value is retained at a certain interval. Multiple filters are used to extract certain multiple features (multiple filters may also be referred to as channels). Combining ANN, autoencoder architecture, and convolution, a denoising model could be built to take out spikes (Gondara, 2016).

From Fig. 18, three sample results are using a certain configuration of autoencoder models. Period is the size of the input (number of data points) to the autoencoder. It indicates the size of the first convolution layer. The size is reduced by half after each stage by the average pooling layer (three stages according to Fig. 17). The filter is the number of filter channels of the first layer of the convolution layer while the kernel is the length of the convolution filter. Both filter and kernel also get halved in each stage towards the bottleneck (latent). From the latent towards the output, the period, filter, and kernel get doubled towards the output (reconstruction). The output dimension conforms to the input dimension.

For each model combination, training must take place. The training uses a dataset of 100 segments (segment #2 is one of them). After training, the model is used to make prediction presented by the denoised result in Fig. 18(b). It is immediately apparent that the spikes disappeared by configuration #3 of Fig. 18(c). In configuration #2, although there are still a few spikes, it is not as many as in configuration #1 of Fig. 18(a). More details of the results are discussed in Section 4.

4. Results and discussion

This section compares and discusses the performance of each denoising method with multiple configurations. The metrics (*MAE* and *R*2) are generated by using data from 100 segments. Sections 4.1 to 4.5 discuss each of the five filtering methods in various configurations. Finally, the comparison among the five filtering methods is presented in Section 4.6 using segment #2 as a demonstrating example.

4.1. Haar wavelet transform

Table 2 compares a few combinations of subset size, stride, and attenuation factor (factor) using *MAE* and *R*2 (the metrics). When the subset size is larger, both the metrics tend to be better than the smaller ones. This is because the larger subset takes more data points, hence more information into account. The (attenuation) factor is also a strong variable. The larger the value indicates the more aggressive the denoising process, thus the better the metrics. A larger factor number exponentially suppresses the spikes than a smaller factor number, as formulated in (6) and (7).

Out of eight combinations of variable configuration, configuration number 6 in Table 2 can produce a relatively better outcome which will be compared with other denoising methods in Section 4.6. A smaller stride does a little better than a larger stride at the expense of a longer processing time. Stride determines the interval at which the subset starts along the signal to be denoised. For example, stride 0.25 with subset size 64 equals 16 ($0.25 \times 64 = 16$). This means that one subset (of 64 data points) is created at an interval of every 16 data points. After applying (7) to a subset to get a set of results, the first 0.25 of the result is sequenced back to form the final denoising result. The overlap of 0.75 towards the end is discarded. Fig. 19 shows the comparison of results with varying strides on one segment (#2).

Fig. 19(a) explains the effect of a larger stride. The results appear *digitized* or of low resolution at a longer stride. In Figs. 19(b), (c), and (d), the results are increasingly becoming higher resolution. Ultimately, the smallest stride of one data point would have the highest resolution. Nevertheless, the smaller stride or higher resolution does not necessarily yield better results, as shown by the *MAE* and *R*2 metrics. In addition, a shorter stride requires a longer processing time.

4.2. Kalman filter

Fig. 20 compares the effect of different values of process uncertainty (*Q*), covariance (*P*), and measurement uncertainty (*R*) using

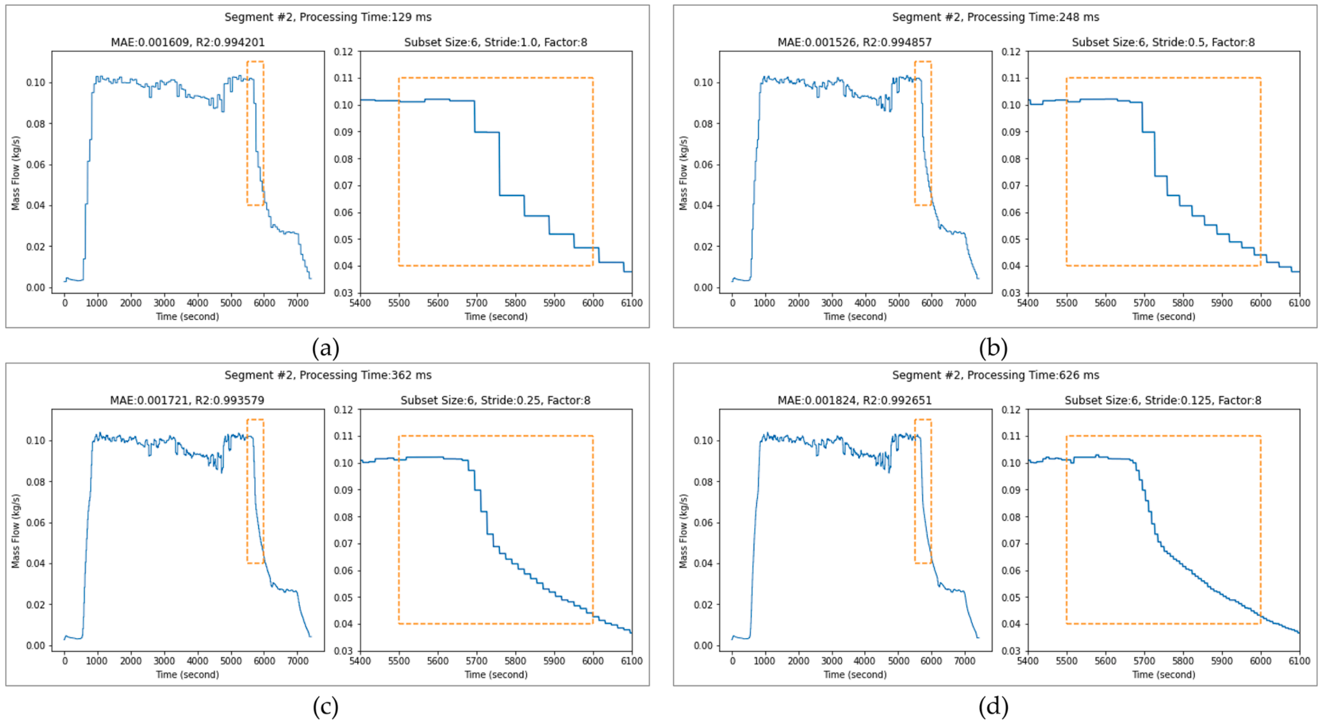
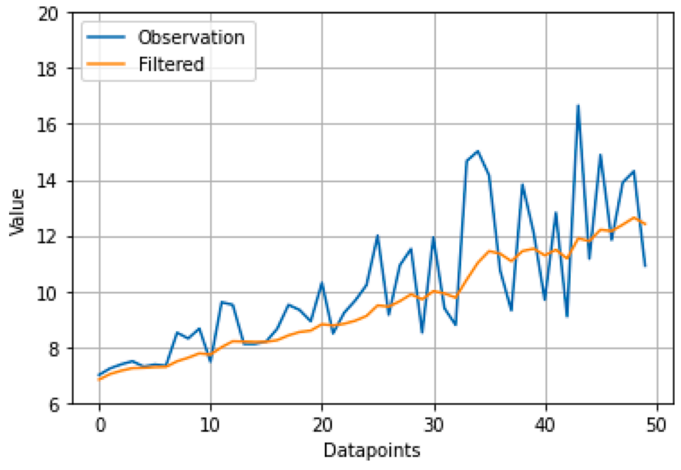
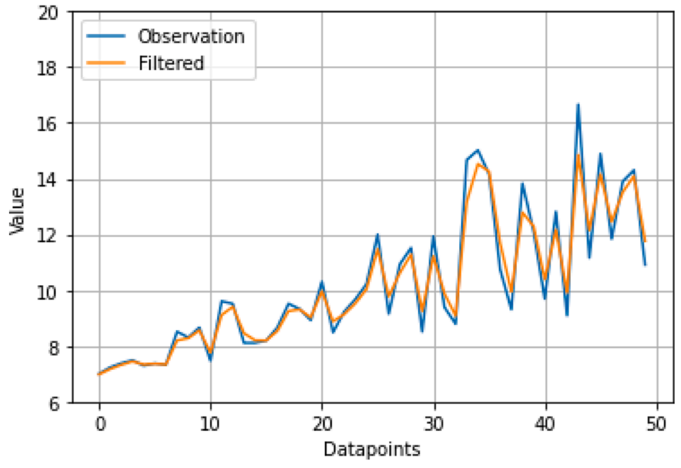


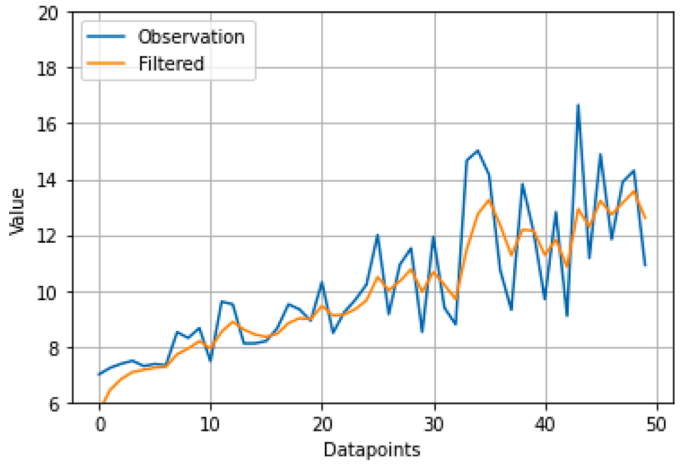
Fig. 19. Comparison of results from one segment at various stride values. Each subfigure is a pair of full segments denoised results and its zoomed-in portion (in orange). (a) Stride = 1.0; (b) Stride = 0.5; (c) Stride = 0.25; (d) Stride = 0.125.



(a)



(b)



(c)

Fig. 20. Kalman filter comparison with test data. (a) $Q = 0.005$, $P = 10$, $R = 0.25$. (b) $Q = 0.05$, $P = 10$, $R = 0.025$. (c) $Q = 0.05$, $P = 1$, $R = 0.25$.

Table 3
Kalman filter results with several configurations.

No.	*Q*	*P*	*R*	*MAE*	*R2*	Processing Time (s)
1	0.0005	0	0.25	0.002295	0.748	60
2	0.0005	0	1	0.002436	0.698	64
3	**0.0005**	**100**	**0.25**	**0.002282**	**0.793**	**60**
4	0.0005	100	1	0.002405	0.783	61
5	1	0	0.25	0.002322	0.715	61
6	1	0	1	0.002313	0.729	64
7	1	100	0.25	0.002322	0.715	65
8	1	100	1	0.002313	0.730	61

Table 4
Kalman filter results with extended configurations.

No.	*Q*	*P*	*R*	*MAE*	*R2*	Processing Time (s)
1	**0.00005**	**100**	**0.025**	**0.002282**	**0.793**	**60**
2	0.00005	100	0.25	0.002580	0.764	61
3	0.00005	100	2.5	0.003571	0.637	65
4	0.0005	100	0.025	0.002263	0.779	63
5	**0.0005**	**100**	**0.25**	**0.002282**	**0.793**	**60**
6	0.0005	100	2.5	0.002580	0.764	59
7	0.005	100	0.025	0.002298	0.751	58
8	0.005	100	0.25	0.002263	0.779	59
9	**0.005**	**100**	**2.5**	**0.002282**	**0.793**	**58**

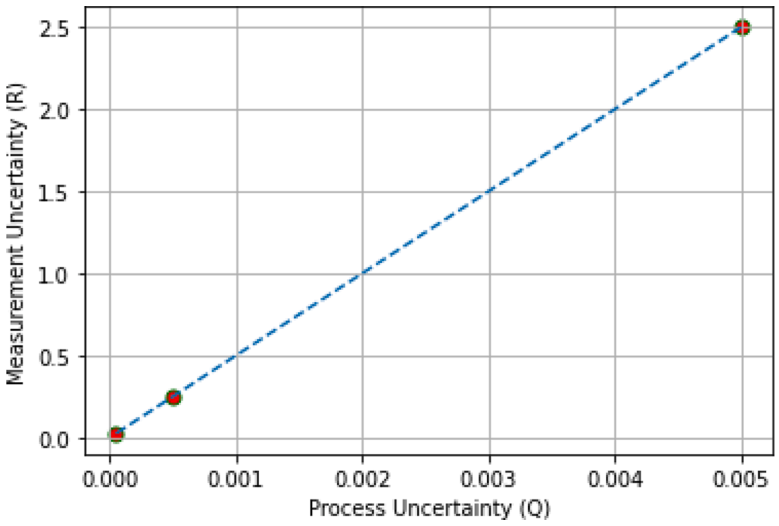


Fig. 21. Results with different combinations of configuration (combo).

toy data. It is to note that Fig. 20 uses the same test data as Fig. 8. In Fig. 20(b), *R* is reduced to 10 times that of Figs. 20(a) and (c). Reducing *R* indicates low uncertainty (high confidence) in the measurement/observation (z). Therefore, it *trusts* the measurement/observation, and the filtered values tend to be closer to the observation. In contrast, *Q* in Fig. 20(a) is reduced by 10 times that of Figs. 20(b) and (c). Reducing *Q* indicates that there is a low uncertainty (high confidence) in prediction ($\overline{x}_t$), and less confidence in z. This causes the filtered value to fit a trend, creating a smoothening effect.

P in Fig. 20(c) is reduced by 10 times that of Figs. 20(b) and (c). The effect is only obvious in the early part of the filtered value. There is an initial wide separation at data point 0. Setting *P* to be near zero or negative causes the correlation between filtered and observation to be very low or the inverse. On the other hand, if the *P* is set to be a large positive value, the correlation between observation and the filtered is very strong, which means that the filtered value is to be very close to observation, to begin with. This conforms to the general intuition of covariance. If the correlation is assumed to be weak in the beginning, the algorithm requires several iterations (or data points) to *break* the initial momentum before being able to follow the general trend. This is due to (13) that updates *P* closer to the relevant value at the end of each iteration.

Using data from Fig. 2 as well as 99 other segments, several combinations of configurations are used to generate results shown in

Table 5
Fourier transform configuration comparison.

No.	Subset Size	Stride	Window	*MAE*	*R*2	Processing Time (ms)
1	100	0.25	Exponential, tau = 5	0.004015	0.624	1832
2	100	0.25	Exponential, tau = 113	0.002314	0.742	1821
3	**100**	**0.25**	**Gaussian, std = 5**	**0.002250**	**0.816**	**1872**
4	100	0.25	Gaussian, std = 113	0.002335	0.702	1839
5	100	1	Exponential, tau = 5	0.003996	0.615	588
6	100	1	Exponential, tau = 113	0.002312	0.737	603
7	100	1	Gaussian, std = 5	0.002220	0.810	598
8	100	1	Gaussian, std = 113	0.002337	0.699	605
9	500	0.25	Exponential, tau = 5	0.004483	0.538	645
10	500	0.25	Exponential, tau = 113	0.002689	0.772	650
11	500	0.25	Gaussian, std = 5	0.002757	0.784	722
12	500	0.25	Gaussian, std = 113	0.002702	0.765	655
13	500	1	Exponential, tau = 5	0.004373	0.528	288
14	500	1	Exponential, tau = 113	0.002562	0.768	286
15	500	1	Gaussian, std = 5	0.002587	0.786	319
16	500	1	Gaussian, std = 113	0.002572	0.761	291

Table 6
Fourier transform denoising extended configuration comparison.

No.	Subset Size	Stride	Window	*MAE*	*R*2	Processing Time (ms)
1	100	0.25	exponential, tau = 5	0.004015	0.624	1858
2	100	0.25	exponential, tau = 29	0.002423	0.789	1856
3	100	0.25	exponential, tau = 53	0.002338	0.772	1899
4	100	0.25	exponential, tau = 87	0.002317	0.752	1863
5	100	0.25	exponential, tau = 113	0.002314	0.742	1866
6	**100**	**0.25**	**gaussian, std = 5**	**0.002250**	**0.816**	**1978**
7	100	0.25	gaussian, std = 29	0.002262	0.786	1912
8	100	0.25	gaussian, std = 53	0.002299	0.738	1933
9	100	0.25	gaussian, std = 87	0.002326	0.710	1885
10	100	0.25	gaussian, std = 113	0.002335	0.702	1962

Table 3. Setting P to an arbitrarily large positive number yields better results, as it quickly assumes that there is a close correlation between the observation and filtered values. This is demonstrated by results number 1 and 2 versus results number 3 and 4 for $Q = 0.0005$.

Setting both Q and R requires a balancing act to achieve the optimum level. From Table 3, it is noticeable that lower Q and R produce the best result as indicated by result number 3. However, there is a limit to how extreme both Q and R values can be, before the uncertainties are underestimated or overestimated. Table 4 attempts to explain the underestimation by extending values of Q and R combination around result number 3 in Table 3. Table 4 shows optimum combinations of Q and R as result numbers 1, 5 and 9. The correlation between Q and R is formulated as (22). The optimum ratio of the confidence level of observation over the confidence level of prediction is approximately 500 times. Fig. 21 plots Eq. (22) showing a regression line.

$$R \cong 500 \times Q \tag{22}$$

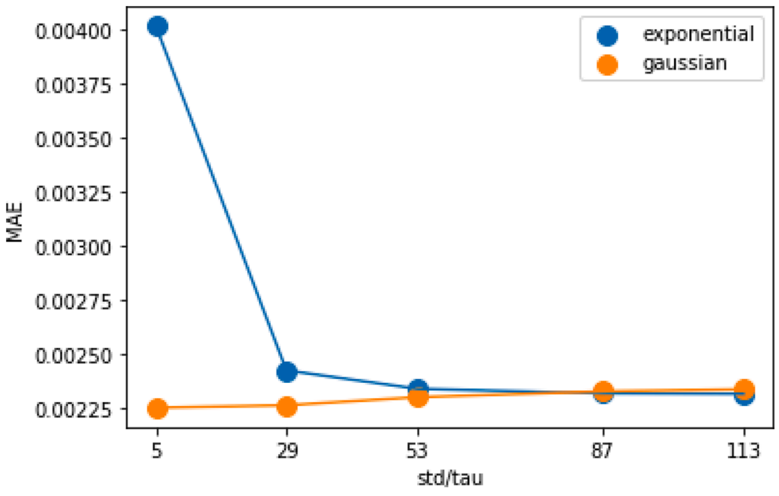
4.3. Fourier transform

In Table 5, the combination of smaller subset size and smaller stride requires longer processing time due to a higher number of iterations. The term stride in this methodology has the same idea as the stride used in HWT. In general, a smaller subset size (period) yields a better set of results.

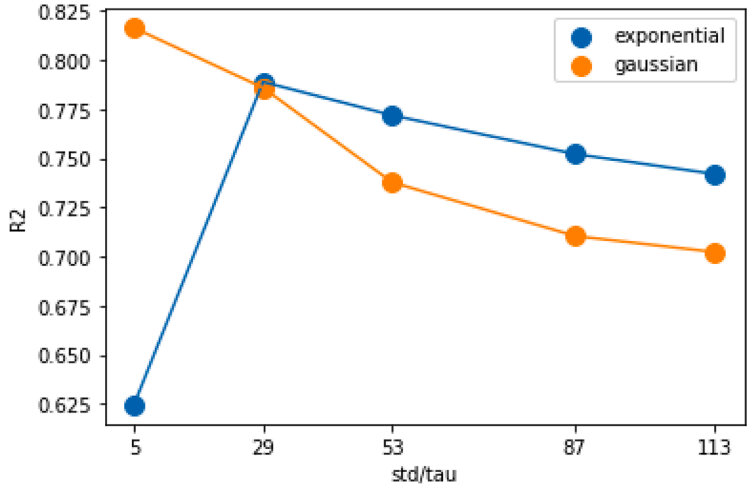
The Gaussian window does better than the exponential window, especially the narrow one (std = 5 versus tau = 5). However, wider exponential window (tau = 113) performs better than wide Gaussian window (std = 113). This demonstrates that not only the narrowness of the window is important, but also the type of window. Out of Table 5, result number 3 produces the best result. The selection as the best result is emphasized by Table 6 and Fig. 22 that the Gaussian window with std = 5 has both the best *MAE* and *R*2.

4.4. Descriptive statistics denoising

Table 7 shows that similar to other methods discussed so far, a smaller subset size and stride require longer processing time to process more iterations. The term stride in this methodology has the same idea as stride used in HWT and FFT. Results for numbers 1 to 4 have a smaller subset size advantage. Subsequently, the gap size (explained in Fig. 15) tends to be smaller as well. Therefore, interpolation over a smaller gap has a smaller error than interpolation over a larger gap. Despite yielding an inferior set of *R*2 scores, a smaller subset size manages to achieve a better set of *MAE* metrics over results number 5 to 8. This emphasizes the different functions of



(a)



(b)

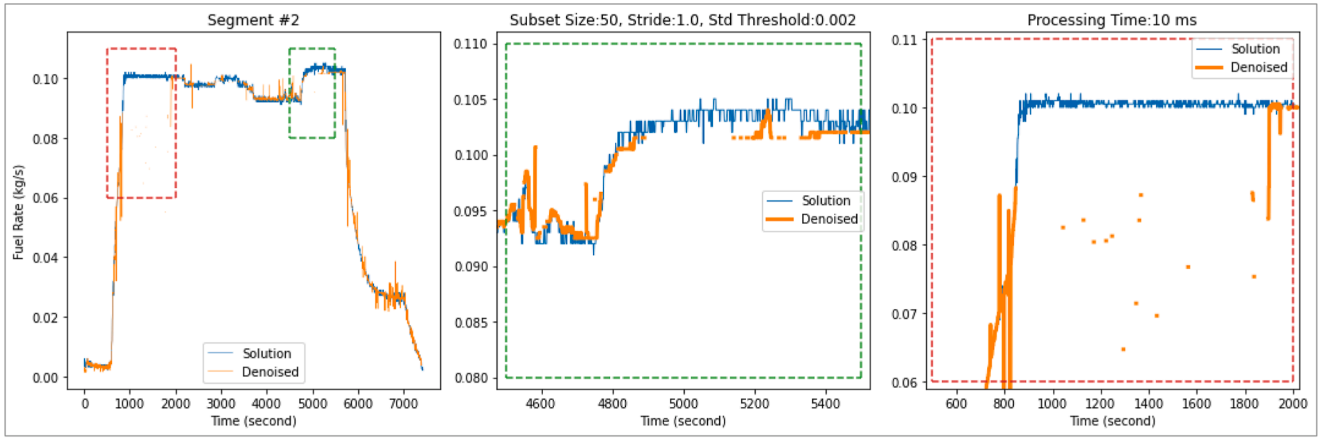
Fig. 22. . *AE* and *R*2 comparison for extended configuration. (a) *MAE* at extended std/tau settings. Lower is better. (b) *R*2 at extended std/tau settings. Higher is better.

Table 7
Statistical denoising configuration comparison.

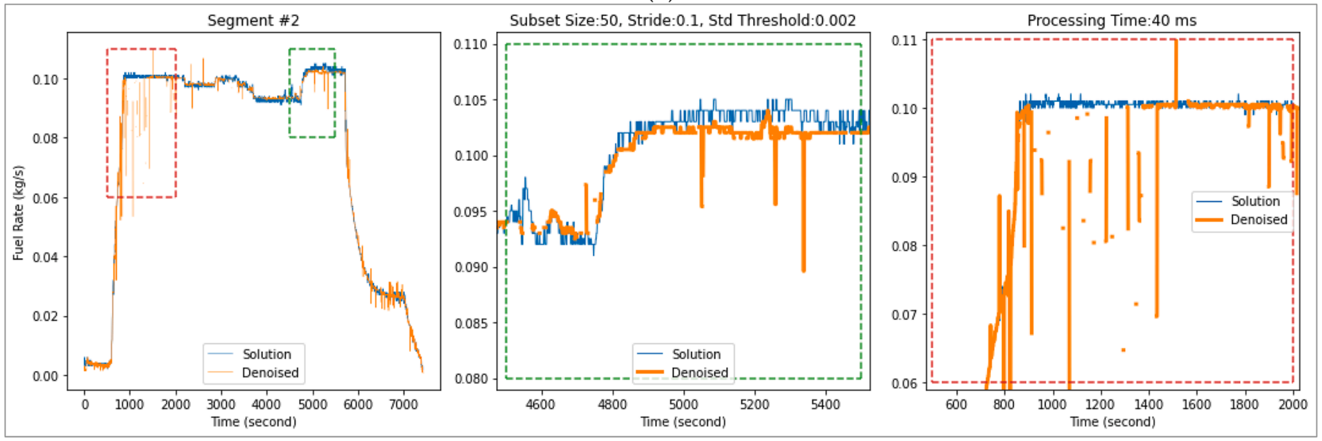
No.	Subset Size	Stride	Threshold	*MAE*	*R*2	Processing Time (ms)
1	**10**	**0.5**	**0.001**	**0.001938**	**0.796**	**6425**
2	10	0.5	0.005	0.002044	0.780	6075
3	10	1	0.001	0.001935	0.792	3627
4	10	1	0.005	0.002036	0.780	3188
5	50	0.5	0.001	0.002240	0.819	1703
6	50	0.5	0.005	0.002041	0.834	1584
7	50	1	0.001	0.002341	0.800	1064
8	50	1	0.005	0.002049	0.831	1068

MAE and *R*2 metrics. While *R*2 is a good metric to evaluate the similarity between *Solution* and *Denoised* based on the trend, *MAE* is a good metric to evaluate dissimilarity.

Fig. 23 shows the comparison of the results using different strides. In Fig. 23(a) region in the green rectangle, the suspected noise data points are discarded and appear as gaps in the orange curve (Denoised). These gaps are filled with linear interpolation (not shown in Fig. 23). Although there is a large gap in Fig. 23(a) green rectangle, the error from interpolation is relatively small as it nears the blue curve (Solution). The disadvantage of the larger gaps is shown in Fig. 23(a), as indicated in the region in the red rectangle. Once the



(a)



(b)

Fig. 23. Comparison of longer vs shorter strides prior to interpolation. Each subfigure has a zoomed-out perspective of a denoised result from one segment (left), and respective zoomed-in regions (center and right) as indicated by coloured rectangles (green and red). (a) Longer stride, Stride = 1.0; (b) Shorter stride, Stride = 0.1.

Table 8
ANN autoencoder configuration.

No.	Period	Filter	Kernel	*MAE*	*R*2	Training Time (s)
1	64	8	16	0.009279	0.881	121
2	**128**	**16**	**32**	**0.005359**	**0.942**	**122**
3	256	32	64	0.013425	0.797	175

interpolation is applied, the result will be worse than the red region in Fig. 23(b). This is because there are too many discarded data points to adequately follow the Solution (blue plot) closely. This disadvantage will be reflected in higher MAE errors.

4.5. *Artificial neural network (Autoencoder)*

Out of numerous ANN architectures, this paper only focuses on autoencoder with a one-dimensional convolutional layer. Table 8 shows the comparison of three ANN configurations. The processing time is the training time required to train a model and to make a prediction. As ANN is non-deterministic, the performance of a model may vary quite considerably by the model of the same configuration trained at different times. However, one configuration should perform better than the other on average.

In Table 8, result #2 also attempts to describe that the middle ground of not having too small or too large of convolution (period), filter and kernel is a good configuration. Apart from the configuration in Table 8, there are more possibilities to tune and improve this denoising method. One way is to enrich the training dataset by augmenting existing ones (DeVries and Taylor, 2017). Another way is to use other types of layers (e.g., dense, or RNN/LSTM layer) (Sainath et al., 2015; Hochreiter and Schmidhuber, 1997).

4.6. *Comparison of different methodologies*

Fig. 24 shows the comparison of the five methods discussed in this paper tested with one segment (segment #2). Visually, it is obvious that all of them produce acceptable results with the spikes mostly removed. The result from descriptive statistic denoising in Fig. 24(d) leaves out two or three big spikes. The Fourier transform in Fig. 24(b) denoising leaves out many smaller spikes. In the artificial neural network in Fig. 24(e), there is a downward shift by the Denoised result throughout. In addition, the tail end of the Denoised is starting to separate. Results from the Haar wavelet transform in Fig. 24(a) and Kalman filter in Fig. 24(b) produce competitive results, albeit the Haar wavelet performs slightly better.

Table 9 shows the KPI metrics comparison of the denoising methods. The best performing according to the *R*2 score is the statistical denoising, despite leaving out two big spikes (see Fig. 24(d)). It is due to the nature of this method that it does not alter original raw noisy data. It either accepts or rejects data points based on a statistical threshold. Only when it rejects suspected noise, thus creating a missing data point, it artificially fills the missing data point by linear interpolation. Hence, most of the valid (not noisy) data points are accepted as is, resulting in a very good *MAE* value. The next best performing is the Haar wavelet transform, with the best *R*2 score. Therefore, it indicates that it has the closest correlation or similarity to the solution. It is due to the nature of Haar wavelet transform denoising that tends to preserve the *shape* of the original signal. It only reduces fluctuation/peaks while maintaining the trend. In other words, the spikes do not go away, only mitigated.

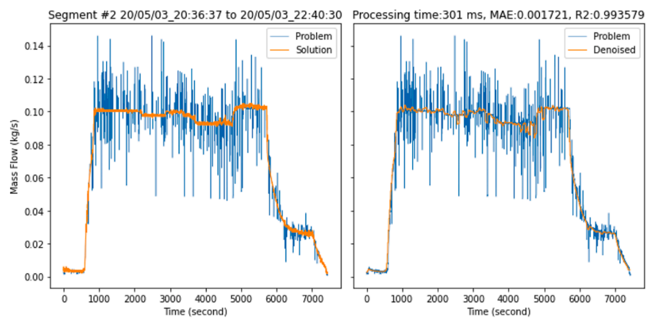
Fourier transform assumes that the noise's frequency is at the high end of the frequency spectrum. Given this assumption, it discriminately reduces the high-frequency components according to the windowing factor. Since it discriminatively treats the higher frequency range as noise, some parts of the high frequency that may be the valid data points are also affected. More analyses could be carried out to identify the noise frequencies, and to perform a targeted reduction of noise instead (Slimane and Zaid, 2021). Fourier transform implementation uses optimized software (`scipy`) (Virtanen et al., 2019). Therefore, its processing time is the best. There is an opportunity to optimize the other two methods (Haar wavelet transform and statistical denoising) to reduce processing time.

Similar to Fourier transform, Kalman filter and autoencoder also use readily available software frameworks (`filterpy` and `tensorflow`) (Abadi et al., 2016). Both Kalman Filter and autoencoder use a predictive approach (Haykin, 2004). Autoencoder requires training an ANN model. Training requires the same data points to be input to the model multiple times (also known as training epochs), hence requiring a longer processing time than the Kalman filter. Kalman filter immediately predicts the next data point in each iteration based on the previous iteration.

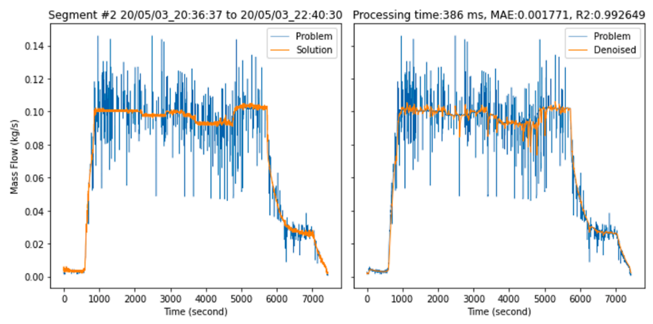
This paper aims to solve a problem that is of one type, i.e., to clean a signal whose noise is a very narrow period of discontinuity away from the true signal at random intervals and values. This noise type appears as spikes over the true valid signal. All five denoising methods could be *tricked* by a set of raw data whose true signals appear to have the same characteristics as noise. This is shown in Fig. 25 with statistical denoising. The earlier part of the true (Solution) is noise-like, and all cleaning methodology struggles to distinguish it. Therefore, it is *denoised* by all denoising methods.

Fig. 26 discusses further the case in which the valid data points are noise-like, using a sample case from segment #4. It can be observed in Fig. 26(b) that the filtering by KF is over-zealous. In contrast, as shown in Fig. 26(c), the filtering by DSD is performing relatively well. However, there is a limit to how well the DSD can perform in such a situation. Fig. 27 discusses this further using a sample case from another segment, #9. It is especially apparent from the later half of the curve, that the fluctuation by the valid data points (not noise) is overly reduced. Nevertheless, using any filtering method discussed in this paper, the general trend of the denoised signal is still maintained with the general trend of the solution, at the expense of the precision of point-by-point precision.

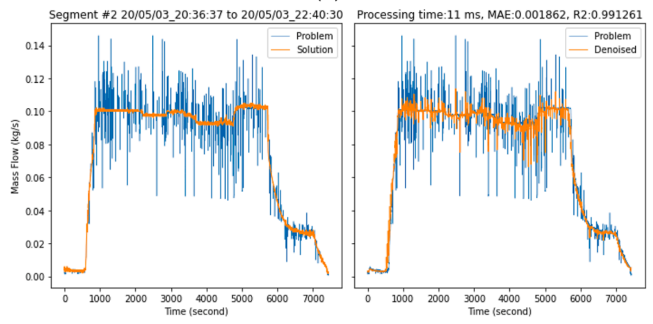
Despite its limitations, the methods discussed in the paper could be tuned to target to reject/denoise certain levels of spikes while



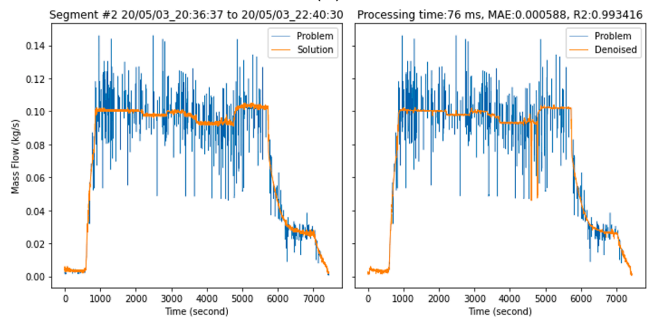
(a)



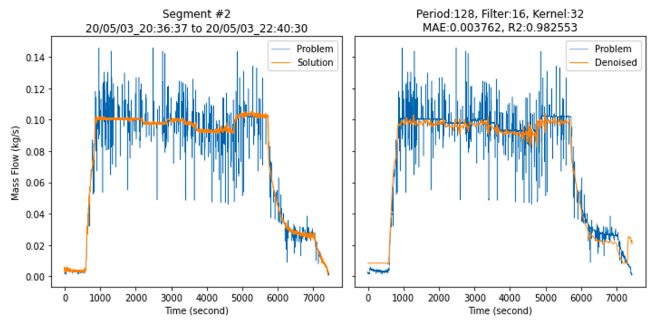
(b)



(c)



(d)



(e)

Fig. 24. Results comparison: (a) Haar wavelet transform, subset size = 2^6 (64), stride = 0.25, f = 8, (b) Kalman Filter, Q = 0.0005, R = 0.25, P = 100. (c) Fourier transform, subset size = 100, stride = 0.25, window type is Gaussian (std = 5). (d) Descriptive statistic denoising, subset size = 10, Stride = 0.25, standard deviation threshold = 0.001, and (e) ANN autoencoder, period = 128, filter = 16, kernel = 32.

Table 9
Final KPI metrics comparison among denoising methods on segment #2 only.

Denoising Method	*MAE*	*R*2	Processing Time[1]
Haar Wavelet Transform	0.001721	0.993579	301 ms
Kalman Filter	0.001771	0.992649	345 ms
Fourier Transform	0.001862	0.991261	11 ms
Descriptive Statistics Denoising	0.000588	0.993416	76 ms
ANN Autoencoder	0.003762	0.982553	122 s [2]

[1] Processing time for processing segment #2 only.
[2] Including model training time.

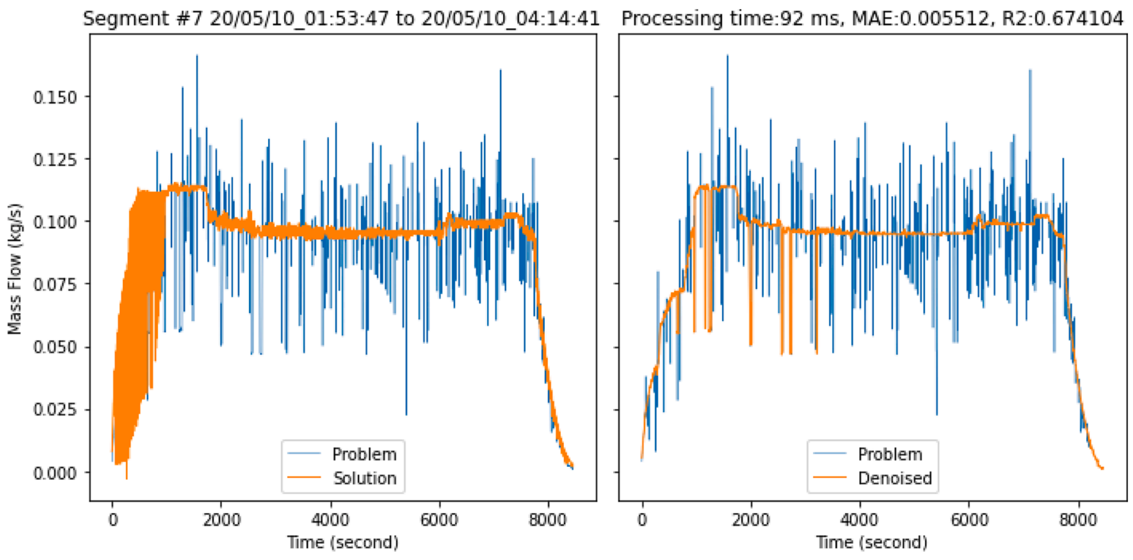


Fig. 25. Raw data with noise (Problem) whose true signal (Solution) appear to be *noisy*.

accepting others, especially the neural network method. It is especially true for HWT and DSD. The filtered data could be used in enhancing the training efficiency of the machine learning system in predicting fuel consumption.

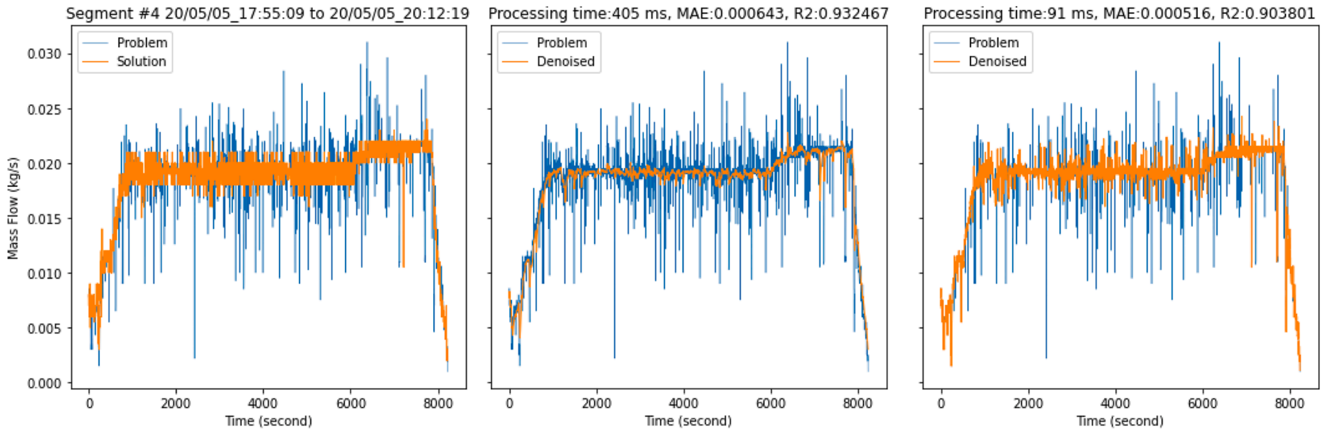
5. Conclusion

This paper aims to discuss characteristics of filtering methods from five topics, i.e., wavelet transforms, optimal estimation, frequency decomposition, descriptive statistics and artificial neural network (autoencoder) using actual noisy mass flowmeter data. As explained in the introduction, the use of a mass flowmeter on a tugboat to measure actual fuel usage is uncommon in the Singapore region. Hence, the problem presented (and its solution) is unconventional and requires an unconventional solution as well.

Given the characteristics of each filtering method, there is an opportunity to tailor the solution for applications other than those discussed in this paper. For instance, Fourier transform could easily filter data in which the unwanted frequency is known. However, the Fourier transform struggles when the target frequency is discriminatively selected or unknown. The same could not be said of descriptive statistics denoising, as it does not have such a characteristic. What descriptive statistics denoising is good at, though, is when the noise is of a certain discontinuity level which can be isolated by a set of conditional thresholds.

Haar wavelet transforms distinctive characteristic is that it processes by smoothening (or sharpening by tuning f in Eq. (6)) data in batches of power of two. Hence, it is suitable to filter digital data. Similar to the Haar wavelet transform, the Kalman filter also has the characteristic of smoothening effect. This effect creates the illusion that the data point sequence has *momentum* whose *mass* and *velocity* are analogous to Q and R. Kalman filter is already a popular filtering method to filter sensor measurement (i.e., GPS which could be quite sporadic). However, unlike the Haar wavelet transform, the Kalman filter uses prediction similar to the ANN. Nevertheless, unlike the Kalman filter, the ANN (autoencoder) requires a large amount of dataset to train, and quite considerable efforts to tune right. Once, optimally done, the autoencoder could become a useful method to filter data with a unique or peculiar pattern due to the convolutional feature.

As also explained in the introduction, the imperfection in raw mass flow data due to noise is a problem that better be solved using software correction instead of hardware correction. Although all methods discussed in the paper have done a relatively good job in denoising the spikes, some methods perform better than others as far as filtering mass flow data is concerned. The Haar wavelet transform, Kalman filter, and descriptive statistical denoising are proven to be accurate and lightweight (requiring no significant



(a) (b) (c)

Fig. 26. Filtering of a noise-like valid signal (or data-points). (a) Observation. (b) Over-zealous denoising by KF. (c) Denoising by DSD.

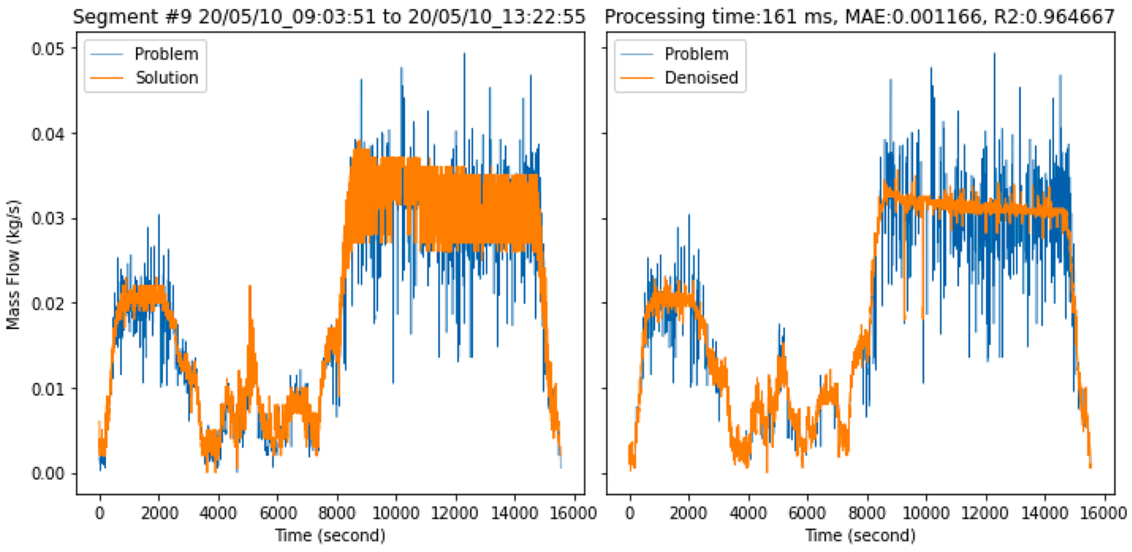


Fig. 27. An example of noise-like signal to showcase the limit to which DSD can filter.

computing requirement) to filter out spikes in the mass flow data.

Author contributions

Tay, ZY, funding acquisition, project supervision, conceptualization, review and editing; Konovessis, D, project supervision; Hadi, J, methodology, validation, original draft preparation, visualization

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The Authors wished to acknowledge the resources supported by SMI (R-SMI-A403–0001) and MOE (R-MOE-A403-C002/ MOE2018-TIF-1-G-008).

References

Miller, G.T.; Spoolman, S.*Living In The Environment*; 16th Editi.Brooks/Cole, 2009 ISBN 978-0495556718.
Shaftel, H.; Callery, S.; Jackson, R.; Bailey, D. Evidence | facts - climate change: vital signs of the planet. Available online: https://climate.nasa.gov/evidence/ (accessed on 4 November 2021).
of Transportation, U.S.E.P.A.O.; Quality, A. Greenhouse gas emissions from a typical passenger vehicle. *https://www.epa.gov/greenvehicles/greenhouse-gas-emissions-typical-passenger-vehicle* 2018.
Leong, S.; Hargreaves, C.; Singhal, P.; Yuan, J.*Estimation of CO2 emission from marine traffic in singapore straits using automatic identification systems data*; 2014.
Bialystocki, N., Konovessis, D., 2016. On the estimation of ship's fuel consumption and speed curve: a statistical approach. J. Ocean Eng. Sci. 1, 157–166. https://doi.org/10.1016/j.joes.2016.02.001.
Raszillier, H., Durst, F., 1991. Coriolis-effect in mass flowmetering. Arch. Appl. Mechan. 61, 192–214. https://doi.org/10.1007/BF00788053.
Chiang, C.; Su, C.-.M.; Ho, Y.-.L.; Kao, Y.-.H. Effects of vibration and flow pattern on coriolis flowmeter; December 2015.
Kaur, M.; Kakar, S.; Mandal, D. Electromagnetic interference. in proceedings of the 2011 3rd international conference on electronics computer technology; 2011; Vol. 4, 1–5.
Ehrentreich, F., Sümmchen, L., 2001. Spike removal and denoising of Raman spectra by wavelet transform methods. Anal. Chem. 73, 4364–4373. https://doi.org/10.1021/ac0013756.
Kim, Y.; Bang, H. Introduction to Kalman filter and its applications. *Introduction and Implementations of the Kalman Filter, F.Govaers, Ed. IntechOpen*. 2019.
Nussbaumer, H.J., 1981. The fast fourier transform. Fast Fourier Transform and Convolution Algorithms 80–111.
Mann, P., Lacke, C., 2020. Organizing and graphing data. Introductory Statistics 28–78.
Tay, Z.Y.; Hadi, J.; Konovessis, D.; Loh, D.J.; Tan, D.K.H.; Chen, X. Efficient harbor craft monitoring system: time-series data analytics and machine learning tools to achieve fuel efficiency by operational scoring system. in proceedings of the volume 6: ocean engineering; American society of mechanical engineers, June 21 2021.
Tay, Z.Y., Hadi, J., Chow, F., Loh, D.J., Konovessis, D., 2021b. Big data analytics and machine learning of harbour craft vessels to achieve fuel efficiency: a review. J. Mar. Sci. Eng. 9 https://doi.org/10.3390/jmse9121351.
Chai, T., Draxler, R., 2014. Root mean square error (RMSE) or mean absolute error (MAE)? Geosci. Model Dev. 7 https://doi.org/10.5194/gmdd-7-1525-2014.
Mittlböck, M., Heinzl, H., 2001. A note on R2 measures for Poisson and logistic regression models when both models are applicable. J. Clin. Epidemiol. 54, 99–103. https://doi.org/10.1016/S0895-4356(00)00292-4.

Stanković, R.S., Falkowski, B.J., 2003. The Haar wavelet transform: its status and achievements. Comput. Electrical Eng. 29, 25–44. https://doi.org/10.1016/S0045-7906(01)00011-8.

Luisier, F., Vonesch, C., Blu, T., Unser, M., 2009. Fast Haar-wavelet denoising of multidimensional fluorescence microscopy data. In: Proceedings of the 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro. IEEE.

Zhang, D., 2019. Wavelet transform. Fundamentals of Image Data Mining: Analysis, Features, Classification and Retrieval 35–44.

Horn, R.A. The Hadamard Product. In Proceedings of the Proc. Symp. Appl. Math; 1990; Vol. 40, 87–169.

Labbe, R. Kalman and Bayesian filters in python. https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python 2018.

Laaraiedh, M., 2012. Implementation of Kalman filter with python language. CoRR. *abs/1204.0375*.

Labbe, R. Filterpy 2018.

Huang, G.; Meng, J.; Zhang, D.; Zhu, X. Window Function for EEG Power Density Estimation and Its Application in SSVEP Based BCIs. In Proceedings of the Intelligent Robotics and Applications; Jeschke, S., Liu, H., Schilberg, D., Eds.; Springer Berlin Heidelberg, 2011; 135–144.

Fisher, M.J., Marshall, A.P., 2009. Understanding descriptive statistics. Australian Critical Care 22, 93–97. https://doi.org/10.1016/j.aucc.2008.11.003.

Suoranta, R.; Estola, K.-. Robust Median Filter with Adaptive Window Length. In Proceedings of the 1991., IEEE International Sympoisum on Circuits and Systems; 1991; 108–111 vol. 1.

Justusson, B.I., 1981. Median filtering: statistical properties. Two-Dimensional Digital Signal Prcessing II: Transforms and Median Filters 161–196.

Jain, A.K., Mao, J., Mohiuddin, K.M., 1996. Artificial neural networks: a tutorial. Computer (Long Beach Calif) 29, 31–44. https://doi.org/10.1109/2.485891.

Bank, D., Koenigstein, N., Giryes, R., 2020. Autoencoders. CoRR. *abs/2003.05991*.

O'Shea, K., Nash, R., 2015. An introduction to convolutional neural networks. CoRR. *abs/1511.08458*.

Gondara, L., 2016. Medical image denoising using convolutional denoising autoencoders. In: Proceedings of the 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), pp. 241–246.

DeVries, T.; Taylor, G.W. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538*. 2017.

Sainath, T.N., Vinyals, O., Senior, A., Sak, H.Convolutional, 2015. Long short-term memory, fully connected deep neural networks. In: Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4580–4584.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9, 1735–1780.

Slimane, A., Zaid, A., 2021. Real-time fast fourier transform-based notch filter for single-frequency noise cancellation: application to electrocardiogram signal denoising. J. Med. Signals & Sensors 11. https://doi.org/10.4103/jmss.JMSS_3_20.

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al., 2019. SciPy 1.0-fundamental algorithms for scientific computing in python. CoRR. *abs/1907.10121*.

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al., 2016. TensorFlow: large-scale machine learning on heterogeneous distributed systems. CoRR abs/1603.04467.

Haykin, S., 2004. Kalman Filtering and Neural Networks, 47. John Wiley & Sons.