Outlier Detection and Removal Signal Processing for Wearable Transcutaneous Oxygen Sensor

Abigail Leonardi[1*], Ciara Murphy[1*], Sydney Hobson[1*], Vanshika Rohera[1*], and Ulkuhan Guler[1]

2023 IEEE MIT Undergraduate Research Technology Conference (URTC) | 979-8-3503-0965-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/URTC60662.2023.10534887

***Abstract*— Respiratory diseases impact millions of people worldwide. In the smart and connected health era, there is a pressing need for reliable medical devices for remote monitoring of patients with respiratory disorders and patients with chronic diseases in general. For respiratory monitoring, there exist widely-used miniaturized photoplethysmography devices to measure peripheral blood oxygen saturation. More recently, transcutaneous oxygen monitors have been designed in a small form factor to be used as wearable devices to enable remote and continuous measurement of transcutaneous oxygen. In this study, we focus on processing the raw data collected from a noninvasive transcutaneous blood gas monitor that is in the development phase. Post-processing should be applied to remove the outliers in the measurements and smooth out the measurement noise. Various filters to remove outliers were tested in MATLAB, including the Hampel, rmoutliers, filloutliers, and moving average filters. The Hampel filter yielded the best results while maintaining the integrity of the data. The fillmissing function in MATLAB using the previous or linear interpolation methods is used to replace outlier data. The comparison of the raw data and the result of the data processing is presented.**

I. Introduction

Respiratory illnesses are a significant concern in the medical field, and their prevalence has only grown since the COVID-19 pandemic [1]. The effectiveness of how well oxygen can move from the lungs to arteries is assessed with the arterial partial pressure of oxygen (PaO_2), an indicator of an individual's oxygenation [2]. There is a pressing need for a device that is noninvasive and miniaturized and also allows for remote and continuous monitoring of PaO_2. A recent wearable presented in [3], [4] measures transcutaneous partial pressure of oxygen ($PtcO_2$) levels diffused through the skin, which highly correlates with PaO_2, through a platinum porphyrin luminescent oxygen-sensing film.

As a case study in this paper, we focus on a condition where the rush of oxygen from the ambient air into the sensor causes misleading data. This results in the rapid decline of lifetime value, yielding a false reading. An example of this disrupted data is illustrated in Fig. 1a. The goal of signal processing is to remove prominent outliers and smooth deviations in the data without losing the integrity of the initial signal. For example, Fig. 1b depicts a signal with small increases and decreases in decay time. It is important that the algorithm removes outliers while simultaneously preserving these original data trends. These small fluctuations indicate important details about a patient's health status. This information can help healthcare professionals determine more accurate conclusions regarding a patient's health.

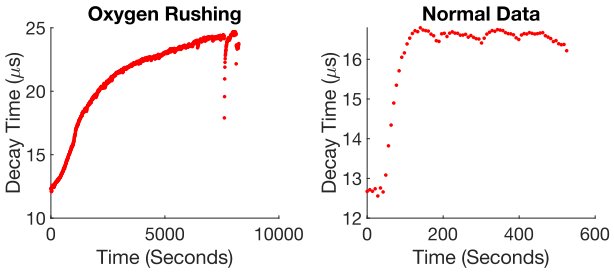

Fig. 1: (a) Example experimental data, displaying O_2 rushing and (b) example experimental data, displaying reasonable increases and decreases in decay time.

In this study, first, we decide the type of erroneous data and the data that should be preserved. Following that, we explored various signal processing algorithms to determine the outliers and replace them with interpolated data without losing the integrity of the original data. This paper is organized as follows. In Section II, we present the data processing research for outlier detection and signal interpolation using MATLAB. Section III demonstrates the rationale for our final signal processing algorithm and the results of the chosen signal processing techniques. Finally, concluding remarks are given in Section IV.

II. Methods

In this section, we will explore the wide variety of filters and functions available in MATLAB libraries. These tools play a critical role in detecting and removing inaccurate data. It is important to pinpoint and replace outliers with trustworthy and relevant information. The subsequent subsections will provide a thorough explanation of this crucial procedure.

1) Outlier Detection: At first, we examined various filter techniques, including Hampel Filter, Rmoutliers, Filloutliers, Medfilt1, and moving average filters, for identifying, eliminating, and substituting outliers.

a) Hampel Filter: The Hampel filter utilizes a property known as the Hampel Identifier, an adaptation of the three-sigma rule [5]. The three-sigma rule is also known as the empirical rule or the $68-95-99.7$ rule, where almost all the data lies within three standard deviations from the mean.

*These authors contributed equally to this work.

[1]Authors are with Electrical and Computer Engineering Department, Worcester Polytechnic Institute, Worcester, MA 01609, USA (e-mail: {awleonardi, cmurphy5, sahobson, vrohera, uguler}@wpi.edu).

This material is based upon work supported in part by the National Science Foundation (NSF) under Grant OAC-2203827. Corresponding author: Ulkuhan Guler.

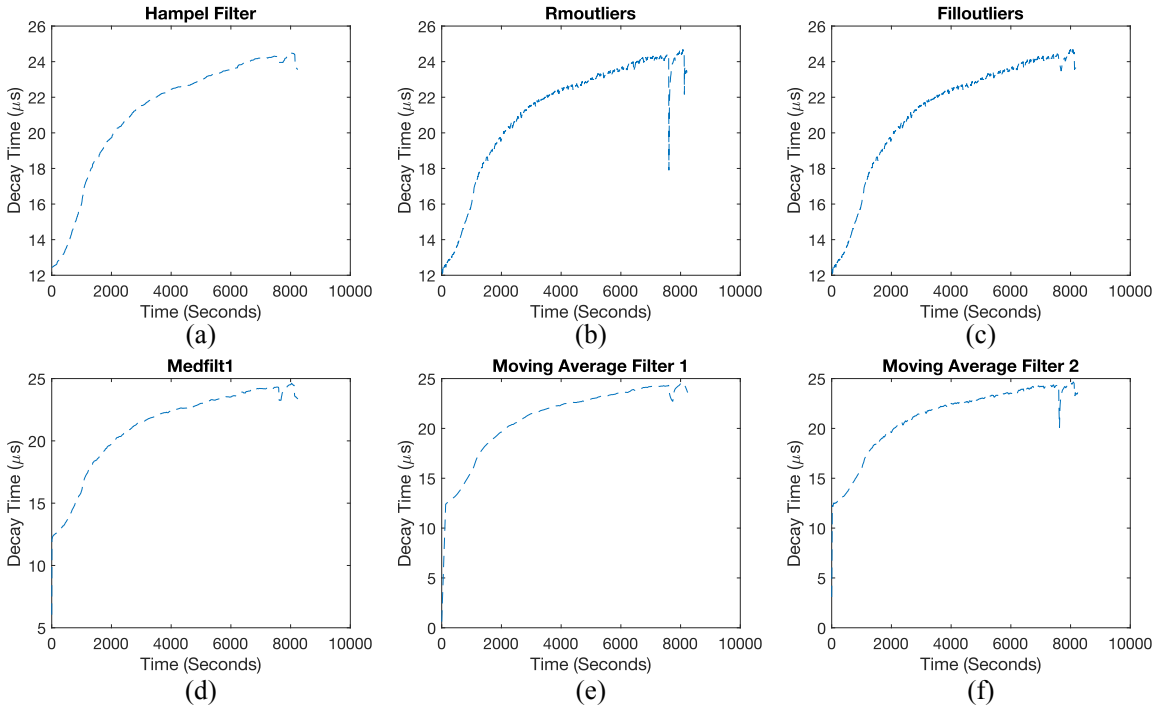

Fig. 2: Effect of filters and functions on outlier data from Fig. 1a using the methods (a) Hampel filter, (b) rmoutliers, (c) filloutliers, (d) medfilt1, (e) moving average filter 1, and (f) moving average filter 2.

For each sample, the filter computes a window of the sample taking in the current sample value and $(k-1)/2$ adjacent samples. The filter replaces a sample with the median if the sample differs from the median greater than the threshold value multiplied by the standard deviation [5].

The basic structure of a Hampel filter is as follows: [y, j, xmd, xsd] = hampel(x, k, nsigma) [6], where "y" is the output filtered data, "j" is a logical vector that returns true at all the points of the original vector that are outliers, "xmd" represents the local medians for each element of x, "xsd" represents the estimated standard deviations for each element of x, "x" is the input vector, "k" specifies the number of neighbors on both sides of each sample of x, "nsigma" represents the number of standard deviations that the sample of x must vary from the local median for it to be replaced with the local median [6]. An example of the Hampel filter applied to experimental data, displayed in Fig. 1a, is in Fig. 2a.

b) Rmoutliers: The basic structure used for rmoutliers is as follows: [B,TFrm] = rmoutliers(A, movmethod, window) [7], where "B" is the output filtered data,"TFrm" is a logical vector representing the data removed from A, "A" is the input vector, "window" is window length, and "movmethod" has two possible inputs: "movmedian" and "movmean," where "movmedian" defines outliers as more than three local scaled median absolute deviations from the local median over a specific window, while "movmean" defines outliers as samples that differ more than three local standard deviations from the local mean over a specific window. To remove outliers this filter uses either the movmean or movmedian movmethod [7]. For example, rmoutliers(A, "movmean", 7) removes data points that are more than three local standard deviations from the local mean within a seven-element window. An example of rmoutliers applied to experimental data, displayed in Fig. 1a, is in Fig. 2b.

c) Filloutliers: The basic structure used for filloutliers is as follows: [B,TF] = filloutliers(A, fillmethod, movmethod, window) [8], where "B" is the output filtered data, "TF" is a logical array that represents the position of elements of B that were previously outliers, "A" is the input vector, "window" is window length, and "fillmethod" has nine possible inputs: "center," "clip," "previous," "next," "nearest," "linear," "spline," "pchip," and "makima." These methods all represent different ways that the outliers can be filled. "movmethod" has two possible inputs: "movmedian" and "movmean." "movmedian" defines outliers as more than three local scaled median absolute deviations from the local median over a specific window. "movmean" defines outliers as samples that differ more than three local standard deviations from the local mean over a specific window. To remove outliers, this filter uses the movmethod to identify outliers and fillmethod to replace outliers [8]. For example, filloutliers(A, "next", "movmean", 7) removes data points that are more than three local standard deviations from the local mean within a seven-element window and replaces them with the next non-outlier value in the data set. An example of filloutliers applied to experimental data, displayed in Fig. 1a, is in Fig. 2c.

d) Medfilt1: The basic structure for medfilt1 is as follows: y = medfilt1(x,n) [9], where "y" is the output filtered data, "x" is the input vector, and "n" is an n^{th}-

order one-dimensional median filter. To remove outliers this filter applies an n^{th}-order filter to the input vector x. For example, this medfilt1(x, 7) smooths the data with a 7th-ordered median filter [9]. An example of medfilt1 applied to experimental data, displayed in Fig. 1a, is in Fig. 2d.

e) Moving Average Filters: There are multiple ways to implement moving average filters in MATLAB. The first method utilizes the 1-D digital filter function. The basic structure for the filter function is as follows: y = filter(b,a,x) [10]. Using numerator and denominator coefficients b and a, a rational transfer function filters the input data x. We utilized the following equation in [9] to create a moving average filter:

$$y(n) = \frac{1}{windowSize} * [x(n)+x(n-1)+...+x(n-(windowSize-1))]. \quad (1)$$

An example of this moving average filter applied to experimental data, displayed in Fig. 1a, is in Fig. 2e.

The second method also uses the filter function [11]. However, we used this equation to create a moving average filter:

$$a(1)y(n) = b(1)x(n) + b(2)x(n-1)+...+b(N_b)x(n-N_b+1) \; -a(2)y(n-1)-...-a(N_a)y(n-N_a+1). \quad (2)$$

An example of this moving average filter applied to experimental data, displayed in Fig. 1a, is in Fig. 2f.

2) Replacing Outliers: The "fillmissing" function in MATLAB replaces outlier data. The basic structure for fillmissing is as follows: F = fillmissing(A,method) [12], where "F" is the output data, "A" is the input vector, and "Method" is the fill method that replaces the outlier data. "Method" has seven possible inputs: "previous," "next," "nearest," "linear," "spline," "pchip," and "makima." Specifically, this function fills missing data (or data classified as not a number (NaN)) with the identified interpolation method [12]. To use this function with outlier data, replace the outlier values with NaN.

To conclude these efforts, we compare the success of each filter method by plotting the original and filtered data on top of each other. Among all the filter techniques, the Hampel filter stands out as the most effective in eliminating outliers while preserving the accuracy of the main signal. Then, the "fillmissing" function was used with the "previous" interpolation method to replace the outliers. Moreover, we tested these filters and functions on sample data from initial experiments to validate the algorithms' functionality on diverse data sets, discussed in the next section.

III. Results and Discussion

The Hampel filter was the most accurate at smoothing data compared to other filtering methods, as shown in Fig. 2. However, it is challenging to standardize one set of parameters to apply to diverse data sets. Instead, the team took a two-step approach. First, the Hampel filter was used to identify and remove the outliers. Then, the fillmissing function replaced the outliers with a specified interpolation method.

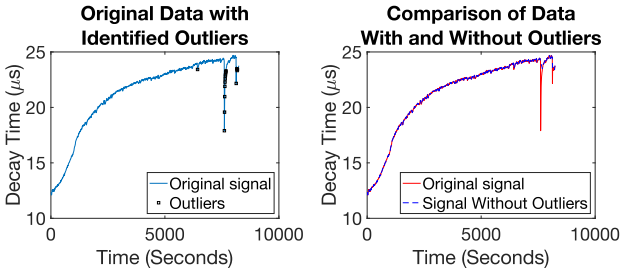

Fig. 3: Outlier detection and replacement algorithm successfully applied to outlier case of O_2 rushing depicted in Fig. 1. (a) Shows outlier detection method and (b) shows outlier removal process.

In more detail, the input parameters of the Hampel filter were fine-tuned to accommodate a more diverse data set. We utilized a k-value of 35 and a nsigma-value of 3.5. The empirical rule states that with data having a normal distribution, 99.7$ of data should be within 3 standard deviations. Since the goal is to detect data that deviates from expected outcomes, the sigma of 3.5 was standard. We chose a high k-value of 35 as it is important when dealing with biomedical data to preserve overall data trends. This data can give critical information about the patient's medical status. Fig. 1a represents a unique outlier case of O_2 rushing. As can be seen in Fig. 3a, the Hampel filter successfully detected outliers in the data. The outliers are identified with square boxes. Fig. 3b shows how the fillmissing function replaces the outliers using the previous interpolation method. To use the fillmissing function in MATLAB, the outliers identified by the Hampel Filter were cast as NaN. The "previous" interpolation method replaced outlier data with the previous nonmissing value.

We then applied this algorithm to test data from our previous experiments [13]. Fig. 4a portrays data with small increases and decreases in the decay time values. Maintaining the integrity of these fluctuations can be important for informing decisions about a patient's health. The algorithm successfully identified outliers while preserving the original data trends, as seen in Fig. 4d.

Next, an algorithm was applied to a data set that stabilized normally, as portrayed in Fig. 4b. The algorithm was able to identify that there were no outliers, as seen in Fig. 4e. This further validates that our algorithm works on diverse data. Further, we applied this algorithm to data with a system error (values reaching the negatives). Fig. 4c portrays data with a large decrease in decay time to negative values at around 42 seconds. It is important that this dip was recognized as an outlier, as negative decay values in this application indicate a system error. The algorithm was able to remove the outlier as shown in Fig. 4f.

The previous and linear interpolation methods can both be used with this algorithm. However, the validity of four common methods: previous, linear, spline, and pchip were tested on a variety of data sets. We overlaid processed signals on top of each other and noticed the subtle differences in the data, as seen in Fig. 5b. Then, we tested other data sets to

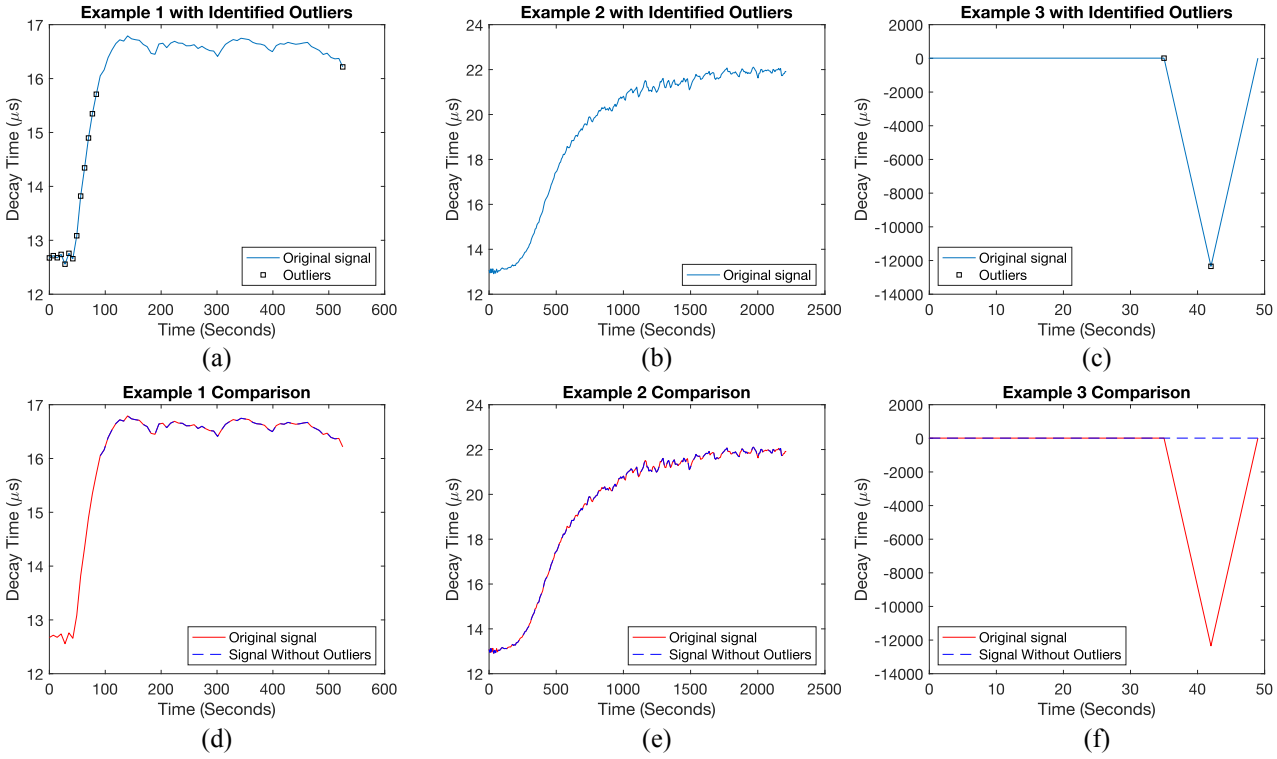

Fig. 4: (a-c) Example data with outliers (d-f) Example data without outliers.

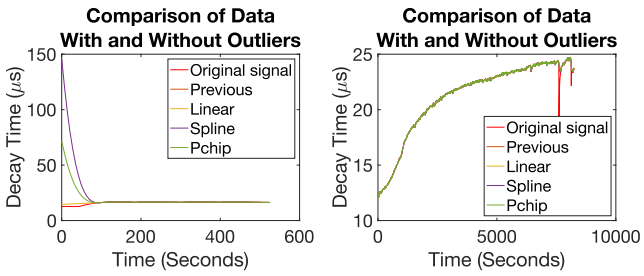

Fig. 5: Effect of previous, linear, spline, and pchip interpolation methods on data (a) from Fig. 1b and (b) from Fig. 1a.

examine if these interpolation methods had a different effect on other data sets. We found these methods had drastically different effects, as seen in Fig. 5a. The previous and linear methods successfully replaced the outliers, whereas the spline and pchip methods are not as accurate. Since the previous and linear interpolation methods had the most consistent and successful results, either method can be used in this outlier detection and replacement algorithm. For the sake of standardizing the process, one method, the previous interpolation method, was used to produce all results.

IV. Conclusion

The goal of signal processing is to remove prominent outliers and smooth deviations in the data without losing the signal's integrity. We explored multiple filtering methods and MATLAB functions to remove and replace outliers. This research focuses on testing these methods on our experimental data. We concluded that the Hampel filter was most successful at detecting outliers, and the fillmissing function with the previous interpolation method was the most successful at replacing the outliers.

References

[1] *World Health Organization*: Coronavirus (COVID-19) Dashboard, [Online]. Available: https://covid19.who.int. [Accessed Jan.26, 2023].

[2] I. Costanzo, D. Sen, L. Rhein, and U. Guler, "Respiratory Monitoring: Current State of the Art and Future Roads," *IEEE Reviews in Biomedical Engineering*, vol. 15, pp. 103–121, 2022.

[3] B. Kahraman et al., "A miniaturized prototype for continuous non-invasive transcutaneous oxygen monitoring," *2022 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pp. 486–490, 2022.

[4] V. Vakhter, B. Kahraman, G. Bu, F. Foroozan, and U. Guler, "A prototype wearable device for noninvasive monitoring of transcutaneous oxygen," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 17, no. 2, pp. 323–335, 2023.

[5] "Filter outliers using hampel identifier." [Online]. Available: https://www.mathworks.com/help/dsp/ref/hampelfilter.html

[6] "Outlier removal using hampel identifier." [Online]. Available: https://www.mathworks.com/help/signal/ref/hampel.html

[7] "Detect and remove outliers in data." [Online]. Available: https://www.mathworks.com/help/matlab/ref/rmoutliers.html#d124e1317148

[8] "Detect and replace outliers in data." [Online]. Available: https://www.mathworks.com/help/matlab/ref/filloutliers.html#d124e462722

[9] "1-d median filtering." [Online]. Available: https://www.mathworks.com/help/signal/ref/medfilt1.html

[10] "1-d digital filter." [Online]. Available: https://www.mathworks.com/help/matlab/ref/filter.html

[11] "Filter data." [Online]. Available: https://www.mathworks.com/help/matlab/data˙analysis/filtering-data.html

[12] "Fill missing entries." [Online]. Available: https://www.mathworks.com/help/matlab/ref/fillmissing.html

[13] V. Rohera, L. Abigail, M. Ciara, S. Hobson, V. Vakhter, L. Rhein, B. Kahraman, G. Bu, F. Foroozan, and U. Guler, "Optimizing transcutaneous oxygen measurement sites on humans," in *International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, July 2023, pp. 1–5.