Regular Paper

A Fast Online Algorithm for Analyzing Magnitude Fluctuation of Time Series

Makoto Imamura* Junji Tsuda* Daniel Nikovski ** Masato Tsuru ***

* School of Information and Telecommunication Engineering, Tokai University, Japan
** Mitsubishi Electric Research Laboratories, USA
*** Department of Computer Science and Electronics, Kyushu Institute of Technology, Japan
Imamura@tsc.u-tokai.ac.jp, 7bjnm017@mail.u-tokai.ac.jp
nikovski@merl.com, tsuru@cse.kyutech.ac.jp

Abstract - Equipment condition monitoring (ECM) has attracted much attention recently in industrial domains, especially as the Internet of Things (IoT) has been emerging and growing rapidly. Monitoring the fluctuations of sensor data generated by industrial equipment is an important issue when trying to detect equipment anomalies. This paper proposes a new fast online algorithm for analyzing a novel magnitude fluctuation feature computed from unsteady time series. The magnitude fluctuation is defined by a convex-shaped pattern which consists of an upward trend leg and a downward trend leg. This definition enables the extraction of anomalous spikes and operational regimes in sensor data of equipment by using the amplitude and duration of an extracted convex-shaped pattern. We also show that the computational complexity of our proposed algorithm is O(n), where n is the length of the input time series; this complexity enables real-time sensor data processing with a sampling period at the microsecond level.

Keywords: Magnitude Fluctuation Analysis, Anomaly Detection, Feature Extraction, Time Series Datamining, Equipment Condition Monitoring, Online Algorithm

1 INTRODUCTION

As the Internet of Things (IoT) [1] has been emerging and growing, sensor big data that is streamed from various kinds of equipment in power plants, industrial facilities, and buildings can be made available for monitoring, diagnosis, energy-saving, productivity improvement, quality management, and marketing. As a result, industry has paid much attention to the use of big sensor data generated from equipment or facilities in order to create a smart society.

Equipment Condition Monitoring (ECM) is a commonly used service based on sensor big data, and data mining techniques are key components in making ECM smarter [2]. This paper proposes a new magnitude fluctuation feature for unsteady and random time-series as a tool for a data mining technique, and also describes an efficient online algorithm for computing it from sensor big data.

After mechanical equipment has been operated for a long time, convex-shaped spikes are often observed in sensor data such as the torque current of motors or the pressure inside a pipe, because of frictional wear, adhesion of foreign substances, etc. Therefore, extracting convex-shaped spikes in sensor data is useful for detecting anomaly or degradation of equipment. However, convex-shaped patterns occur in sensor data not only as symptoms of degradation but also of controlled operating patterns or random noise (Figure 1). In most cases, the heights of convex-shaped patterns are different depending on whether it is a control operating pattern, a degradation symptom, or noise. As shown in Figure 1, the height of an operational pattern is often larger than that of a degradation symptom pattern. In its turn, the height of a degradation symptom pattern is often larger than that of noise. In this work, we define *extended maximal convex curves* to represent a convex-shaped pattern in a time series, along with its height, which we call *amplitude*. Furthermore, we propose a fast online algorithm to extract extended maximal convex curves from a time series, by introducing a novel operation "leg reduction", which will be explained later in Section 2.2. Online algorithms are typically a requirement for realizing real-time equipment condition monitoring.

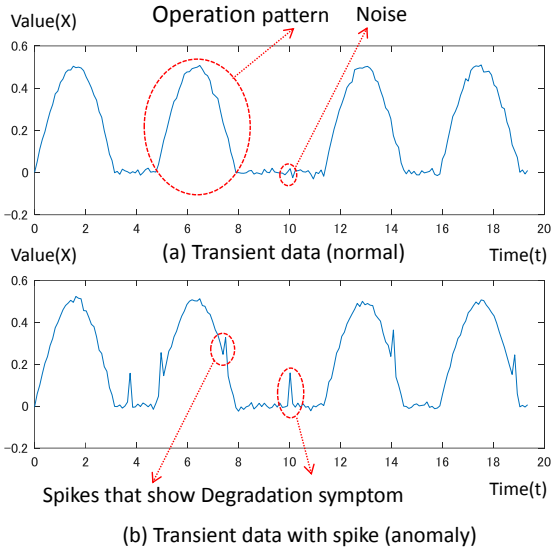

Figure 1: Transient data with spikes

Magnitude fluctuation for unsteady data has been studied from the perspective of data mining by Fink et al [3]. They proposed the concept of a leg, and its associated search method to find a global trend in a time-series including small variations such as noise. The dotted lines in Figure 2 are examples of legs. Both lines show the global upward trend that includes local up-down segments. However, their method treats only single legs, finding an upward or downward trend,

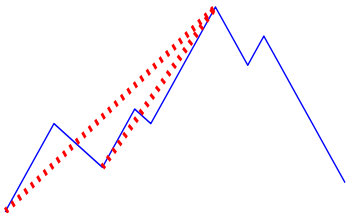

Figure 2: Leg

but can't determine the magnitude of fluctuations. A convex curve could be defined as the continuous occurrence of upward and downward trends. However, in the case of a trapezoidal subsequence with noise, it is non-trivial to select convex curves from several candidates. This is the reason for the need of a new notion of an "extended maximal convex curve", and it will be discussed later in Figure 6 in section 2.1. Furthermore, a naïve application of Fink's algorithm to extract a convex curve needs a computation proportional to $n \times l$, where n is the length of a given time series and l is the average length of legs. In contrast, the computational complexity of our proposed algorithm is $O(n)$, and it doesn't depend on the length of the convex curve.

There are also related works on time series processing with legs [4] [5]. These previous works of ours proposed the computation of the frequency of fluctuations in time series where upward trends and downward trends appear alternately and iteratively. A leg frequency is defined for a given window size and an amplitude. Regarding the difference between our previous works and this research, whereas the leg frequency has window size as a parameter that should be optimally selected by users, the amplitude of a convex curve in this research has no parameter. This means a higher usability of the amplitude of a convex curve. This paper is the extended version of our earlier work [6]. The main difference with [6] is that in this paper we show the proof and the evaluation of our proposed algorithm, while [6] only suggested its possibility.

Related works on extracting pattern from time series include motif discovery [7][8], discord discovery [9] [10] and autoregression [11]. The difference between these existing works is whether an algorithm has window size as a parameter or not. This means that our work does not need to decide an appropriate window size.

Related work on finding a subsequence that includes a distinctive pattern in an online setting is online segmentation [12]. The difference between that existing work and our work is that the former is exclusive segmentation, but the latter is overlap segmentation. Our segmentation problem is how to extract all of the convex curves included in a time series, while a convex curve may include some other convex curves. When a larger fluctuation includes smaller fluctuations as shown in the bottom graph in Figure 1, the smaller convex curve is included by the upward or downward trend in the larger convex curve.

The rest of our paper is organized as follows. Section 2 describes the definition of maximal convex curve and its mathematical properties. Section 3 shows a maximal convex curve amplitude calculating algorithm, and analyzes its order of complexity. Section 4 evaluates our proposed algorithm empirically. First, we show that it can extract convex-shaped spikes in transient data by experimental means. Second, we show that the execution time of our algorithm is linear in *n*. Section 5 provides conclusions and directions for future work.

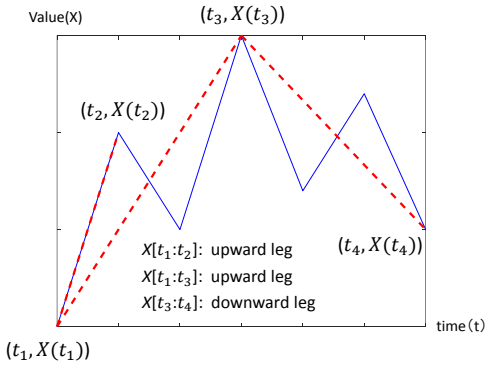

Figure 3: Upward and downward legs

2 MAXIMAL CONVEX CURVE

This section defines the amplitude of the maximal convex curve at each time of a given time series as a feature which shows the degree of magnitude fluctuation of the time series. The merit of our proposed feature is that it is parameter free. It means that it is not necessary to tune parameters when we use this feature. On the other hand, most of the features of time series proposed in the existing studies depend on at least the window size, so how to select an optimal window size is often a problem.

2.1 Definition of a Maximal Convex Curve

The maximal convex curve at each time *t* is defined as a pair of the maximal leg from t toward left and that toward right. However, a naive definition of a maximal leg makes its amplitude unstable. Therefore, we introduce an extended maximal leg from t toward left or right to obtain a robust definition.

Definition: time series X, subsequences X[p:q]

A *Time Series* $X=[x_1, \cdots, x_m]$ is a continuous sequence of real values. The value of the i-th time point is denoted by

$$X[i] = x_i.$$

A *subsequence* $S = [x_p, x_{p+1}, \ldots, x_q] = X[p{:}q]$ is a continuous subsequence of X starting at position p and ending at position q. We denote the length of a subsequence S by len:

$$len(S) \equiv q - p + 1$$

Definition: Leg

Let X be a time series. We define a leg by a subsequence $L = X[l{:}r]$ that satisfies the conditions below.

$$\forall i.\ l < i < r \quad (X[r] - X[i])(X[i] - X[l]) > 0$$

That is, a subsequence $X[l{:}r]$ has a maximum and a minimum at the terminal points *l*, *r*.

If $X[r] - X[l] > 0$, a leg L is called an upward leg.
If $X[r] - X[l] < 0$, a leg L is called a downward leg.

Figure 3 shows examples of upward and downward legs.

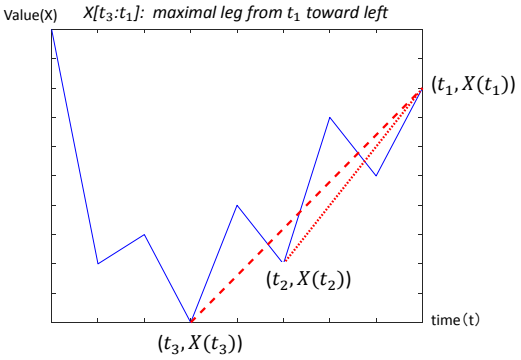

Figure 4: Maximal leg from t toward left

Definition: Sign and amplitude of a leg

We define the *sign* and *amplitude* of a leg $L = X[l:r]$ by the functions below. We denote them by amp and $sign$ respectively:

$$amp\ (L) = abs\ (X[r] - X[l]).$$

The absolute function $abs(a)$ means the absolute value of a.

$$\begin{aligned} sign\ (L) &= 1, && \text{if } (X[r] - X[l]) > 0 \\ &= 0, && \text{if } (X[r] - X[l]) = 0 \\ &= -1, && \text{if } (X[r] - X[l]) < 0 \end{aligned}$$

By the above definition, the sign of an upward leg is plus and the sign of a downward leg is minus.

Definition: A Maximal Leg from t toward left

Let *X* be a time series and *t* be a time point in X.

We define *a Maximal Leg from t toward left* by a leg $L = X[l:t]$ that satisfies the following condition.

For any $l' < l$, $X[l':t]$ is not a leg the amplitude of which is larger than that of $X[l:t]$.

That is,

for any l' such that $l' < l$ and

$sign(L)\ (X[t] - X[l]) \leq sign(L)\ (X[t] - X[l'])$

some j such that $l' \leq j < l$ exists and j satisfies

$sign(L)\ (X[t] - X[j]) \leq 0$

Figure 4 shows an example of a maximal leg from t_1 toward left. $X[t_3:t_1]$ is a maximal leg from t_1 toward left. On the other hand, $X[t_2:t_1]$ is a leg, but not a maximal leg from t_1 toward left.

Definition: A Maximal Leg from t toward right

We define *a Maximal Leg from t toward right* by a leg $L = X[t:r]$ that satisfies the following, with everything else is the same as "*Maximal Leg from t toward left*":

For any $r < r'$, $X[t:r']$ is not a leg the amplitude of which is larger than that of $X[t:r]$.

Definition: Convex curve at t in X

Let *X* be a time series and *t* be a time point in X.

We define *a convex curve at t in X* by a subsequence $S = X[l:r]$ that satisfies the following conditions.

(i) $l < t < r$

(ii) $X[l:t]$ is a leg.

(iii) $X[t:r]$ is a leg.

(iv) $\text{sign}(X[l:t])\ \text{sign}(X[t:r]) < 0$

We denote it by $X[l:t:r]$, and call *t* the *vertex* of the convex curve. l and r are called *left terminal* and *right terminal* of the convex curve, respectively. We call an interval $[l:r]$ *support* of the convex curve.

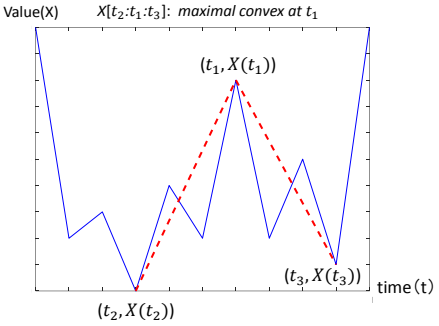

Figure 5: Maximal convex curve

Definition: Maximal Convex Curve at t in X

We define *a maximal convex curve at t in X* by a subsequence $S = X[l:r]$ that satisfies the following conditions.

(i) $l < t < r$

(ii) $X[l:t]$ is a maximal leg from t toward left.

(iii) $X[t:r]$ is a maximal leg from t toward right.

(iv) $\text{sign}(X[l:t])\ \text{sign}(X[t:r]) < 0$

Figure 5 shows the example of a maximal convex curve. $X[t_2:t_1:t_3]$ is a maximal convex curve at t_1.

Definition: *Signed Amplitude of a maximal convex curve*

Let $C = X[l:t:r]$ be a maximal convex curve.

We define *the amplitude* $amp_c(X,t)$, *sign* $sign_c(X,t)$ *and signed amplitude* $signedAmp_c(X,t)$ *of a maximal convex curve at t in X*, respectively, by the functions below:

$amp_c(X,t) \equiv \min(amp(X[l:t]), amp(X[t:r]))$

$sign_c(X,t) \equiv sign(X[l:t])$

$signedAmp_c(X,t) \equiv sign_c(X,t) \times amp_c(X,t)$

If *t* is not a vertex of a maximal convex curve, we define the amplitude at *t* to be zero.

The above definition of a maximal convex curve is not robust in the sense that even a small change of the value at the vertex of a convex curve can lead to a large change of amplitude value. For example, in Figure 6, suppose that $X(t_2) = X(t_4)$ and $X(t_1) = X(t_5)$. If $X(t_2) = X(t_3) = X(t_4)$, a maximal convex curve at t_2 is $X(t_1:t_2:t_5)$ and its amplitude is $(X(t_2) - X(t_1))$. If $X(t_3)$ is just a little less than $X(t_2)$, a maximal convex curve at t_2 is $X(t_1:t_2:t_3)$, and a maximal convex curve at t_4 is $X(t_3:t_4:t_5)$. And, the amplitude of each convex curve is $\big(X(t_2) - X(t_3)\big)$. That is, just a small change of X can make a great change of the amplitude.

In real sensor data, even real-valued data may have discrete values by the limitation of a sensor or a measuring device, such that they often might have the same values. Therefore, the above is not an atypical contrived case.

We introduce the concept of an extended leg in order to obtain a robust definition of a maximal convex curve.

Definition: Extended Leg

We define an *Extended Leg* by the sequence $L = X[l:r]$ that satisfies the condition below.

$$\forall i.\ l < i < r \quad (X[r] - X[i])(X[i] - X[l]) > 0 \\ \vee\ X[i] = X[r]$$

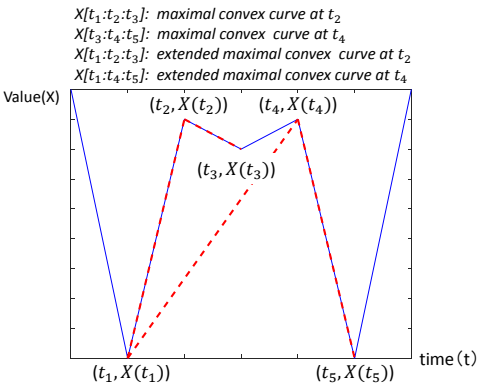

Figure 6: Extended maximal convex curve

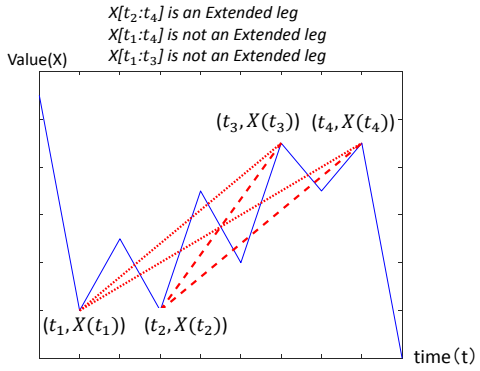

Figure 7: Examples of extended legs

Figure 7 shows the example of an extended leg. When we assume that $X[t_1] = X[t_2]$ and $X[t_3] = X[t_4]$, $X[t_2{:}\,t_3]$ and $X[t_2{:}\,t_4]$ are extended legs, but $X[t_1{:}\,t_4]$ and $X[t_1{:}\,t_3]$ are not extended legs.

"Maximal Extended Leg from t toward left and right" can be defined by the same way as *"a Maximal Leg from t toward left and right"*, respectively.

Definition: Extended Maximal Convex Curve at t in X
Let X be a time series and t be a time point in X.
We define an extended maximal convex curve at t in X by a subsequence $C = X[l{:}\,r]$ that satisfies the following conditions. We denote it by $X[l{:}\,t{:}\,r]$
(i) $l < t < r$
(ii) $X[l{:}\,t]$ is a maximal extended leg from t toward left.
(iii) $X[t{:}\,r]$ is a maximal leg from t toward right.
The *amplitude*, *sign* and *signed amplitude* of an extended maximal convex curve at t are defined similarly to those of a maximal convex curve.

Please note that the left side is extended but the right side is not extended in the above definition, so that we do not count the larger amplitude twice. For example, in Figure 6,$X[t_1{:}\,t_2{:}\,t_3]$ and $X[t_1{:}\,t_4{:}\,t_5]$ are extended maximal convex curves, but $X[t_1{:}\,t_2{:}\,t_5]$ is not an extended maximal convex curve.

Definition: Amplitude function, Positive amplitude function
Let X be a time series. *Amplitude function* $amp_X(t)$ is a function from each t in X to the signed amplitude of the extended maximal convex curve at t. If t is not vertex, its amplitude is defined to be 0. *Positive amplitude function* is defined to be $max(amp_X(t), 0)$. *Negative amplitude function* is defined to be $min(amp_X(t), 0)$.

2.2 Properties of Maximal Convex Curves

This section discusses the amplitude property of a maximal convex curve for deriving a fast online algorithm to calculate a maximal convex curve at each time t. A maximal convex curve can be defined at the point at which X is a locally maximal or minimal value. Hereafter, we assume that X is a locally maximal time series that consists of only locally maximal or minimal values.

Overlap segmentation makes it difficult to extract a maximal convex curve. The reason is that an online algorithm needs to know the time when the maximal amplitude is decided for each point, while reading data in order. In worst case, the convex curve that has the largest amplitude might not be decided until the last data is read. We introduce a novel operation "*leg reduction*" for searching the time when the maximal convex curve is decided. Leg reduction decides the convex curve and simplifies time series by removing the points which are the vertex of decided convex curves. Leg reduction is classified into three types, which are middle, left and right leg reductions, depending on the positions where maximal convex curves are decided. This section describes the definition and mathematical property of leg reductions.

2.2.1 Local Maximal Preserving Transformation

Before describing leg reductions, we introduce *local maximal preserving transformation* to reduce the problem simpler. Local maximal preserving transformation is an operation to remove the points that are not the vertex of convex curves from given time series. We note that the amplitude of the point that is not the vertex of a convex cure is defined to be zero. We will define *locally maximal time series* and *positioned time series* for the preparation to define local maximal preserving transformation.

Definition: Locally maximal time series
For any t, X is a locally maximal time series if it satisfies the following condition:

$$\forall t.\ (\mathrm{X[t+2]} - \mathrm{X[t+1]})\,(X[\mathrm{t+1}] - \mathrm{X[t]}) < 0$$

Locally maximal time series can be obtained from a given time series by removing the points that are neither a local maximum nor a local minimum (Figure 8).

Definition: Positioned time series
Positioned time series is a two-dimensional array $\mathrm{Y} = [\mathrm{X}, \mathrm{P}]$, which consists of time series $\mathrm{X} = [x_1, \ldots, x_i, \ldots, x_n]$ and position series $\mathrm{P} = [1, \ldots, i, \ldots, n]$. And we call $[\mathrm{X}, \mathrm{P}]$ *a positioned time series generated from* X.

Definition: Local maximum preserving transformation
Let X be a time series, and $[\mathrm{X}, \mathrm{P}]$ be a positioned time series generated from X. A local maximal preserving transformation is defined as the repeated below procedures from $\mathrm{E} = [\mathrm{X}, \mathrm{P}]$ until the following procedure from E_1 to E_2 can no longer be applied.

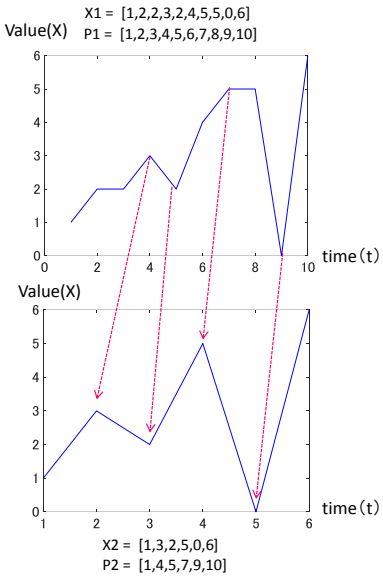

Figure 8: Local maximum preserving trans

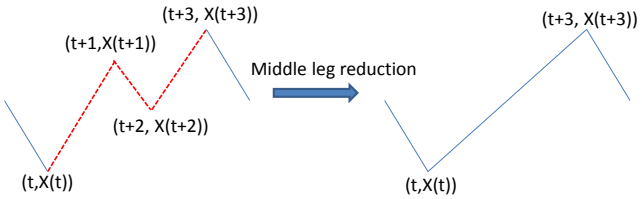

Figure 9: Middle leg reduction

Let $E_1 = [X_1, P_1]$ be a positioned time series.
If $X_1[t] = X_1[t+1]$ or
$(X_1[t+1] - X_1[t])(X_1[t+2] - X_1[t+1]) > 0$,
then we get $E_2 = [X_2, P_2]$
by removing $X_1[t+1]$ and $P_1[t+1]$ from X_1 and P_1, respectively, by the following.
For $0 \le i < t$, $X_2[i] := X_1[i]$, $P_2[i] := P_1[i]$
For $t \le i \le len(X_1) - 1$, $X_2[i] := X_1[i+1]$, $P_2[i] := P_1[i+1]$

Proposition 1: Conservation of convex amplitude in local maximum preserving transformation
Let X_1 be a time series, and $E_2 = [X_2, P_2]$ be a positioned locally maximal time series obtained from X_1 by a local maximum preserving transformation. If a maximal convex curve C_1 is defined at i in X_1, then there is a maximal convex curve C_2 that is defined at j in X_2 such that $P_2[j] = i$, with the same amplitude of C, that is, $amp_c(X_2, j) = amp_c(X_1, i)$, and vice versa.
[Proof] A maximal convex is only defined at a local maximal or minimal point. And the amplitude of a leg is not changed by a local maximum preserving transformation. Therefore, a maximal convex curve in X_1 has the corresponding convex curve in X_2, and vice versa.

2.2.2 Middle Leg Reduction

Definition: Middle leg reduction (Figure 9)
Let $t, t+1, t+2, t+3$ be time points at a locally maximal time series X. If $X[t: t+3]$ satisfies the following conditions (we call them *middle leg conditions*), the procedure to remove $X[t+1]$ and $X[t+2]$ from X is called a *middle leg reduction*.

$$abs(X[t+1] - X[t]) > abs(X[t+2] - X[t+1]) \quad \ldots(1)$$

$$abs(X[t+3] - X[t+2]) \ge abs(X[t+2] - X[t+1]) \quad \ldots(2)$$

More concretely, a middle leg reduction from $E_1 = [X_1, P_1]$ to $E_2 = [X_2, P_2]$ is described by the following.
Let $E_1 = [X_1, P_1]$ be a positioned time series, and X_1 be a locally maximal one that satisfies middle leg conditions. For
$1 \le i \le t$, $X_2[i] := X_1[i]$, $P_2[i] := P_1[i]$
For $t+1 \le i \le len(X_1) - 2$,
$X_2[i] := X_1[i+2]$, $P_2[i] := P_1[i+2]$

We note that inequality (1) does not contain an equal sign, whereas the inequality (2) contains an equal sign. It corresponds to the definition of extended maximal convex curve.

Proposition 2: *Conservation of convex amplitude in a middle leg reduction*
Let $E_1 = [X_1, P_1]$ be a positioned local maximal time series, and $E_2 = [X_2, P_2]$ be a positioned time series obtained from E_1 by a middle leg reduction.
(i) For $i \le t$, $amp_c(X_2, i) = amp_c(X_1, i)$
For $i \ge t+1$ $amp_c(X_2, i) = amp_c(X_1, i+2)$
(ii) $amp_c(X_2, t+1) = amp_c(X_1, t+2)$
$= abs(X_1[t+2] - X_1[t+1])$
[Proof]
(i) When $X_1[t+1]$ and $X_1[t+2]$ are removed from X_1, a subsequence $X_1[t: t+3]$ is a leg the sign of which is the same as $X_1[t: t+1]$ and $X_1[t+2: t+3]$. $X_1[t]$ and $X_1[t+3]$ keep being local maxima. Therefore, E_2 keeps being a positioned local maximal time series. Therefore, the amplitude of every convex curve at a time point t in X_1 except for $t+1$ and $t+2$ is not changed after a middle leg reduction, because of the definition of a maximal leg from t.
(ii) If $X_1[t: t+3]$ satisfies the middle leg conditions, a leg $X_1[t: t+1]$ is a maximal leg from $t+1$ toward left and $X_1[t+1: t+2]$ is a maximal leg from t+1 toward right. Therefore,
$amp_c(X_2, t+1)$
$= \min(amp(X_1[t: t+1]), amp(X1[t+1: t+2])$
$= abs(X_1[t+2] - X_1[t+1])$
$amp_c(X_2, t+2) = abs(X_1[t+2] - X_1[t+1])$ is proved similarly.

2.2.3 Left Leg Reduction

Definition Left leg reduction (Figure 10)
Let 1, 2,3 be time points at a locally maximal time series X. If $X[1: 3]$ satisfies the following conditions, a procedure to remove $X[1]$ from X is called a *left leg reduction*.

$$abs(X[2] - X[3]) \ge abs(X[2] - X[1])$$

The above condition is called a *left leg condition*.

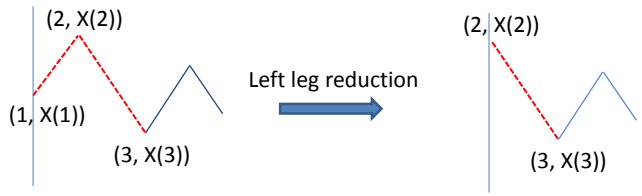

Figure 10: Left leg reduction

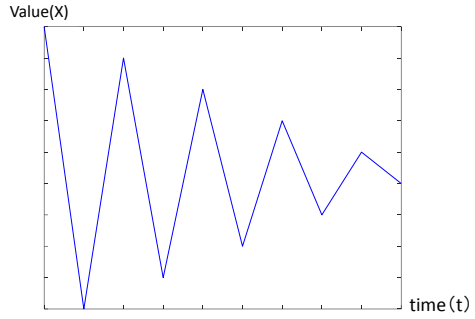

Figure 11: Shrinking time series

More concretely, a left leg reduction from $E_1 = [X_1, P_1]$ to $E_2 = [X_2, P_2]$ is described by the following:
Let End_1=len(X_1).
For i $\leq End_1 - 1$, X_2 [i] := X_1 [i+1], P_2 [i] := P_1 [i+1]

Proposition 3: Conservation of convex amplitude in left leg reduction
Let X be a time series, $E_1 = [X_1, P_1]$ be a positioned local maximal time series generated from X, and $E_2 = [X_2, P_2]$ be a positioned time series obtained from E1 by a left leg reduction.
(i) For i $\leq$ len(X_1) $- 1$, $amp_c(X_2, i) = amp_c(X_1, i+1)$
(ii) $amp_c(X_2, 2) =$ abs($X_1[2] - X_1[1]$)
[Proof] The proof for Proposition 3 is similar to that of Proposition 2.

We define shrinking time series for describing proposition 4.
Definition: Shrinking time series (Figure 11)
The locally maximal time series X is called shrinking if it satisfies the following condition.
$\forall t$. abs(X(t+1) – X(t)) > abs(X(t+2) – X(t+1)) ...(3)

Proposition 4: By repeating middle and left leg reductions until they can no longer be applied, we get a shrinking time series.
[Proof]
Let *R* be the time series that is gotten by repeating middle and left leg reductions until they can no longer be applied.
The first 3 points of *R* satisfy the below inequality, because left reduction cannot be applied.
$abs(R[2] - R[1]) > abs(R[3] - R[2])$
Therefore, the first three points satisfy inequality (3) in the definition of shrinking time series.
The next three points satisfy the below inequality, because middle leg reduction cannot be applied to the first 4 points of *R*.
$abs(R[3] - R[2]) > abs(R[4] - R[3])$
Therefore, the first four points satisfy inequality (3) in the definition of shrinking time series.
By repeating the same operation until the end of time series *R*, we get that all the points in time series *R* satisfy inequality (3) based on mathematical induction.
Note that if the sign is " $\geq$ " in inequality (1) of middle leg reduction as with that in inequality (2), this proposition is not valid. This shows that an extended maximal curve is necessary not only for robust definition but also for fast algorithm.

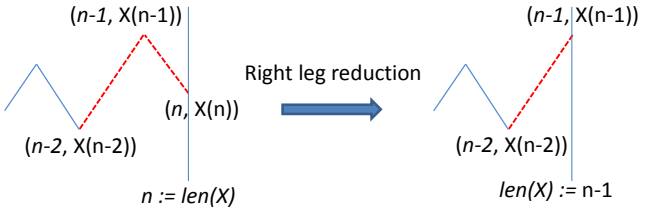

Figure 12: Right leg reduction

2.2.4 Right Leg Reduction

Definition: Right leg reduction (Figure 12)
Let "End" be $len(X)$. If $X[\text{End} - 2 : \text{End}]$ satisfies the following condition,
$abs(X[End] - X[End-1]) \geq abs(X[End-1] - X[End-2])$
A procedure to remove $X[End]$ from X is called a *right leg reduction*. The above condition is called *right leg condition*.
More concretely, a right leg reduction from $E_1 = [X_1, P_1]$ to $E_2 = [X_2, P_2]$ is described by the following:

For i $\leq$ len(X_1) $- 1$, X_2[i] := X_1 [i] , P_2[i] := P_1 [i]

Proposition5: *Conservation of convex curve amplitude in right leg reduction*
If a positioned local maximal time series E = [X, P] is shrinking,
(i) $amp_c(X, End-1) =$ abs($X[End] - X[End-1]$)
where $End = len(X)$.
[Proof] It follows directly from the definitions of a locally maximal time series and a shrinking time series.

2.2.5 Main Theorem

Theorem: Let X be a time series. The signed amplitude and length of an extended maximal convex curve at *t* in X can be calculated by middle, left and right leg reductions. In other words, the amplitude function for X can be also calculated.
[Proof]
If we apply middle and left leg reductions repeatedly until they can no longer be applied, we get a shrinking time series by proposition 4. For all the local maxima and minima that are removed by middle or left leg reduction, their amplitudes are calculated by proposition 2 and 3. For the remaining local maxima and minima, their amplitudes are calculated by proposition 5. Therefore, all the amplitudes of maximal convex curves in X are calculated by middle, left and right leg reductions.

3 MAXIMAL CONVEX CURVE AMPLITUDE CALCULATING ALGORITHM

This section shows an online algorithm to calculate the amplitude function for a given time series. The computational complexity of a naive algorithm by the definition of a maximal convex curve is $O(n^2)$, but that of our proposed one is $O(n)$ based on the results in the preceding section 2.2.
The values of an amplitude function do not depend on the order of the applications of a middle leg reduction and a left leg reduction, by virtue of the propositions in section 2.2. Therefore, we can get an online algorithm by the repetition of

executable reductions while scanning the values from the beginning.

Figure 13 shows an algorithm that computes an amplitude function. In Figure 13, "X" is an input time series and "A" is the values of the amplitude function at time *t* for "X". Both are implemented as a one-dimensional array.

Line 1-4 initializes the output "A", variable "S" and "F". "S" is a stack that stores the vertexes that are used for leg reductions. "F" is a flag that decides when the while-loop execution terminates.

Line 6-32 is a main loop that terminates when all the lines are scanned. Line 7 pushes a local maximum or minimum to the top of stack "S". The first and the last points at "X" are treated as local maximum or minimum.

Line 8-29 is a while-loop that executes middle and left leg reductions until those cannot be applied any more. Line 10-14 corresponds to a middle leg reduction. If a middle leg reduction is executed, then a flag "F" is set to be 1 at line 15, else "F" is set to be 0 at line 17. Line 20-22 corresponds to a left leg reduction. If a left reduction is executed, then a flag "F" is set to be 1 at line 23, else "F" is set to be 0 at line 25. If neither a middle leg reduction nor a left leg reduction is executed, "F" is set to be 0 at line 28. If "F" equals 0, the while-loop terminates. Then a function "getNextLocal-Maximum" will search the next maximum or minimum point at "X" and set it to "i" in line 30. If "i" is the last point at "X", main for-loop ends.

In line 10-14, line 10-11 checks the middle leg coditions. Line 12 and 13 correspond to the equations (ii) in the proposition 2. Line 14 corresponds to the operation obtaining X_2 from X_1 by a middle leg reduction. A function "pop(S, [2,3])" pops the second and the third values of a stack "S" and fills them with the values followed the fourth and fifth value in order.

In line 20-22, line 20 checks a left leg condition. Line 21 corresponds to the equation (ii) in the proposition 3. Line 22 corresponds to the operation obtaining X_2 from X_1 by a left leg reduction. A function "pop(S, [3])" pops the third value of a stack "S" and fill them with the values followed the fourth value in order.

Line 34-37 is the repeated execution of right leg reductions. The main theorem ensures that the remaining values in stack "S" can be reduced to "S" whose size is 2 by repeated application of right leg reductions. Line 36 corresponds to the equation (i) in the proposition 5. A function "pop(S, [1])" pops the first value of a stack "S" and fill them with the followed values in order.

When all the values of "X" are scanned and the size of stack "S" becomes 2, "maximalConvexAmplitude" ends and return output values "A" in line 38.

Lastly, we will show the computational complexity of this algorithm. Let *n* be the length of time series "X". The lines that depend on *n* are for-loop (line 6-32), the first while-loop(line 8-29), "getNextLocalMaximum" (line 30) and the second while-loop (line 34-37).

The repeat count of the for-loop starting at line 6 equals to the number of the local maxima and minima of "X" (We call the number *f*). And the total number of reductions in the first while-loop starting at line 8 is also smaller than the number *f*.

Algorithm: maximalConvexAmplitude *(X)*

[Input] X: time series
[Variable] S:Stack, F: Flag, i: Maximum or minimum point at X
[Output] A: the values of amplitude function for X

```
01  // (1) Initilization
02  A:= zeros(len(X));     // All the value of A is zero.
03  S : = [];              // Initialize a stack
04  F := 1;                // Flag that decides while-loop execution
05  // (2) Main loop
06  for i:=1 to len(X)     //  len(X) is the length of X
07    S := push(S,X[i]) ;
08    while F == 1
09      if len(S) >= 4      // Prop. 2: Middle leg reduction
10        if abs(X(S[3] – X(S[4])) >abs(X(S[2]) – X(S[3]) and
11           abs(X(S[1])–X(S[2])) ≥ abs(X(S[2]) – X(S[3]))
12           A(S[3]) := X(S[3]) – X(S[2]);
13           A(S[2]) := X(S[2]) – X(S[3]);
14           S := pop(S, [2,3]);
15           F := 1;
16        else
17           F:=0;
18        end if
19      elseif len(S) >= 3    //Prop. 3: Left leg reduction
20        if abs(X(S[1]) – X(S[2])) ≥abs(X(S[2]) – X(S[3]))
21          A(S[2]) := X(S[2]) – X(S[3])
22          S := pop(S, [3]);
23          F := 1;
24        else
25          F:=0;
26        end if
27      else
28        F:=0;
29    end while
30    i := getNextLocalMax (X,i) //Prop.1:Local maximal transformation
31    F: = 1;
32  end for
33  // (3) Post process
34  while len(S) >= 3       // Prop.4 and 5: Right leg reduction
35     A(S[2]) := X(S[2])  – X(S[1]);
36     S := pop(S, [1]);
37  end while
38  return A;
```

Figure 13: Maximal convex amplitude calculation

Therefore, the computational complexity of the first while-loop is order O(*f*), that is, at most order O(*n*). We note that *f* is smaller than *n*. Furthermore, the computational complexity of the total execution of "getNext-LocalMaximum" is order O(*n*), because it scans the values of "X" just once. As a result, the computational complexity of the for-loop is O(*n*).

Next, the total repeat count of the second while-loop is also smaller than *f*. In conclusion, we get the result that the computational complexity of "maximalConvex-Amplitude" is order O(*n*).

4 EVALUATION

The preceding section showed that our proposed algorithm to calculate "amplitude function (the signed amplitude of a maximal convex curve at a time)" is online and its computational complexity is order O(*n*).

This section shows that our proposed algorithm can extract convex-shaped spikes in transient data and it enables real-time data processing with a sampling period at the microsecond level by the experiment with simulated data and real data [13].

4.1 Convex-Shaped Pattern Extraction

First, we show that our algorithm can extract convex-shaped spikes from noisy sine data. Second, we confirm that it can extract spikes in transient data shown in Fig. 1. Last, we show that it can extract various convex patterns by giving the levels of amplitudes for space shuttle Marotta Valve data.

(1) Noisy sine wave with spike

In Figure 14, the top graph is a sine curve with noise, and the bottom graph shows positive amplitude function. This figure shows that the positive amplitude at a local maximal point represents the magnitude fluctuation of convex-shaped patterns even during the transient period. Furthermore, amplitude function can distinguish noisy convex patterns than from main convex patterns whose amplitude is approximately 2.5.

In Figure 15, the top graph is an anomalous transient time series mixed with convex-shaped spikes, and the bottom graph is the positive amplitude function. It shows that our algorithm can extract not only an operational patterns with 0.5 scale amplitude, but also convex-shaped spikes with 0.1 scale amplitude.

(3) Space telemetry: space shuttle Marotta Valve

Figure 16 shows an example of a Space Shuttle Marotta Valve time series that are annotated as normal [11]. Marotta Valve is a fuel supply valve for airplane or rocket.

In Figure 17, the top graph is a Space shuttle Marotta Valve time series that contains 5 cycles. The bottom graph shows the amplitudes that are larger than 3. Red circles in the top graph mean the vertexes of convex-shaped patterns. Dotted lines in the top graph are left or right terminals of the convex-shaped patterns whose vertexes are red circles. This shows that our algorithm can extract normal operation patters by giving an amplitude as a value that is a little smaller than a maximum of one normal cycle.

Figure 18 is an enlarged view of Figure 17 during times from 351 to 390. It shows that the vertexes at around 390 are seen as one vertex in Figure 17, but there are two convex-shaped patterns whose amplitudes are larger than 5 in them.

In Figure 19, the top graph shows amplitudes that are larger than 1.2 and smaller than 3. The red circles and red dotted lines have the same meaning as in Figure 17. Each up and down spike in energizing phases is extracted from each cycle, but the first and the second cycles have other convex-

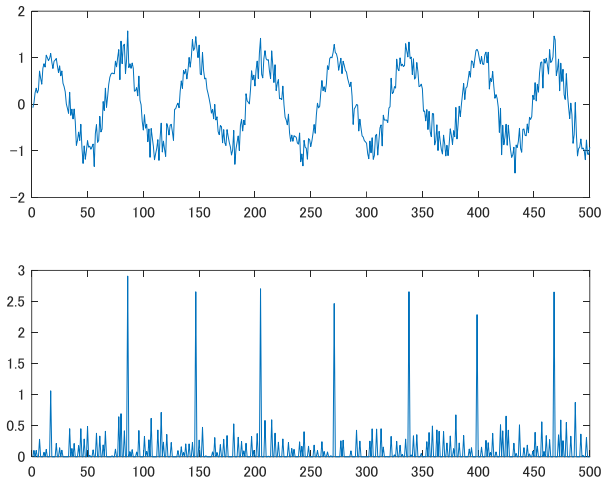

Figure 14: Convex amplitude of noisy sine wave

(2) Transient data with spike

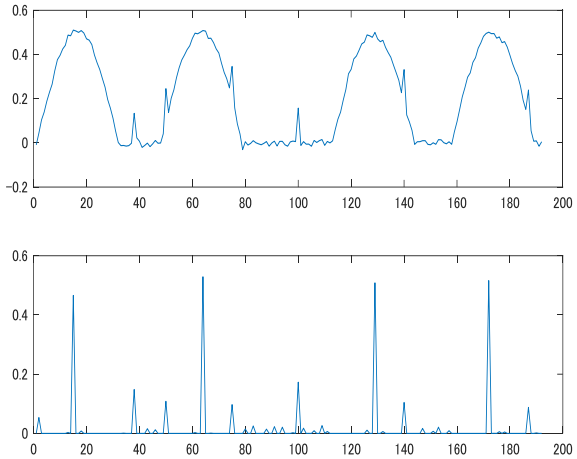

Figure 15: Transient data with spike

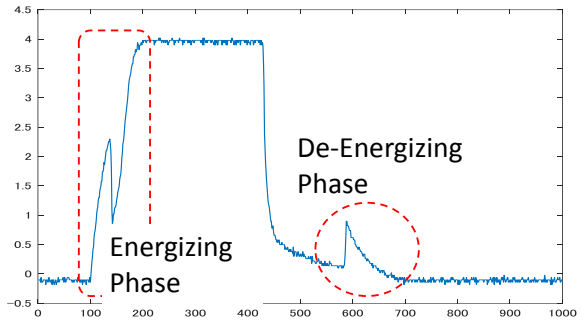

Figure 16: An example of a Space Shuttle Marotta Valve time series that are annotated as normal

shaped patterns except for a normal energizing phase. Figure 20 is an enlarged view of Figure 19 during times from 95 to 180. Similarly, Figure 21 is an enlarged view of Figure 19 during times from 3101 to 3186. The pattern shown by Figure 19 is a continuously convex-shaped so that it is different from normal energizing phase pattern such as Figure 21.

In Figure 22, the top graph shows amplitudes that are larger than 0.46 and smaller than 0.9. The red circles and red dotted lines have the same meaning as in Figure 17. Each

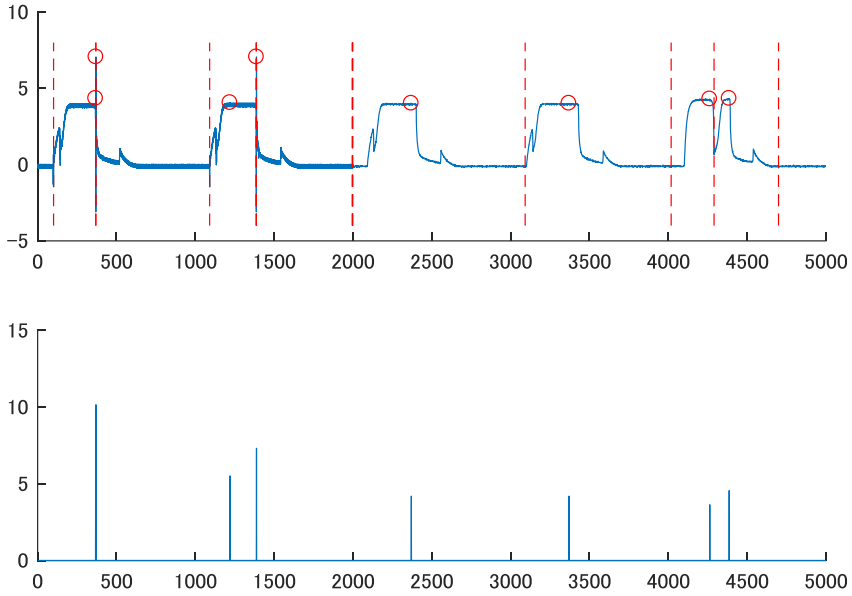

Figure 17: Convex-shaped patterns whose amplitudes are larger than 3

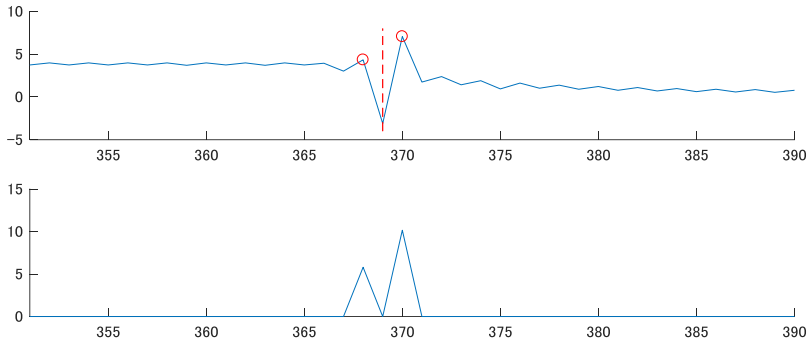

Figure 18: Enlarged view from 351 to 390 in Figure 17

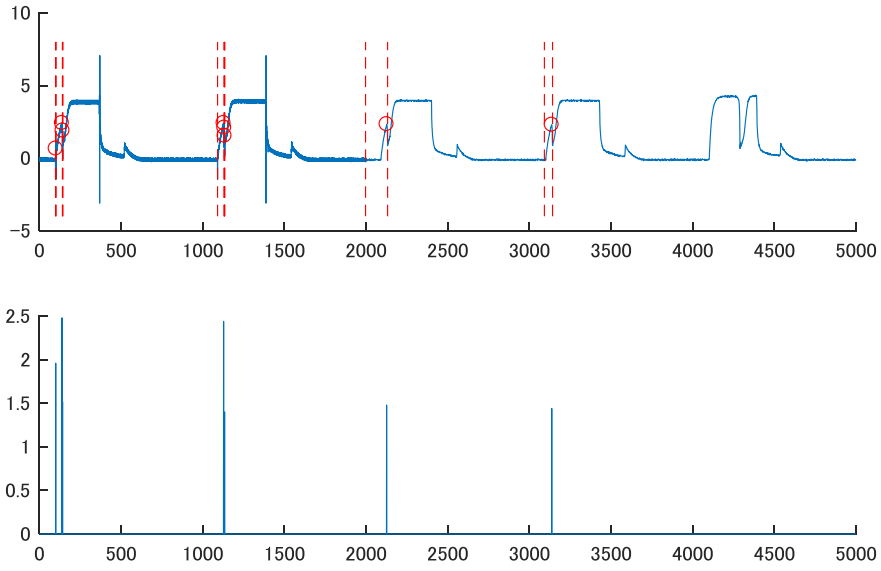

Figure 19: Convex-shaped patterns whose amplitudes are larger than 1.2 and smaller than 3.0

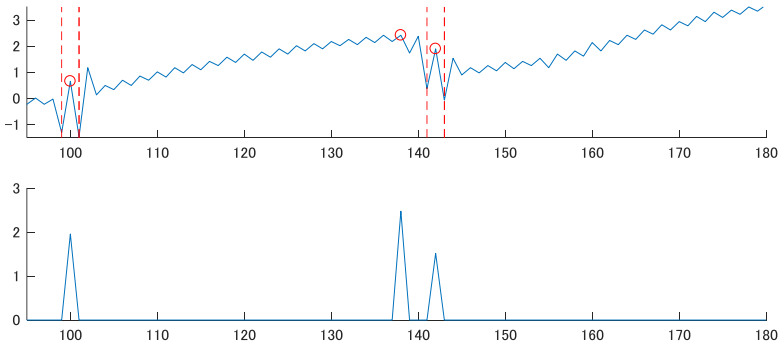

Figure 20: Enlarged view from 95 to 180 in Figure 19 (abnormal)

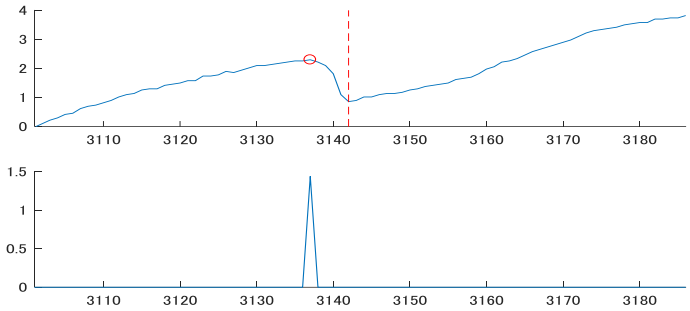

Figure 21: Enlarged view from 3101 to 3186 in Figure 19 (normal)

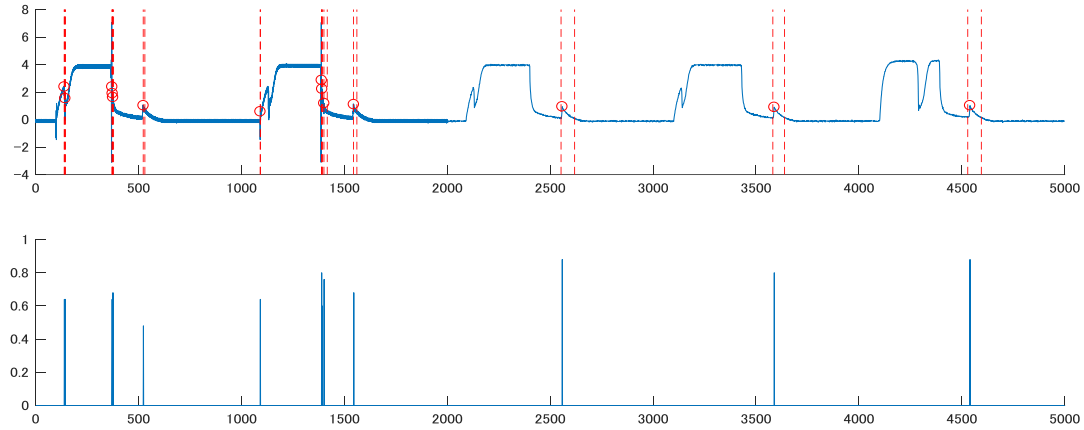

Figure 22: Convex-shaped patterns whose amplitudes are larger than 0.46 and smaller than 0.9

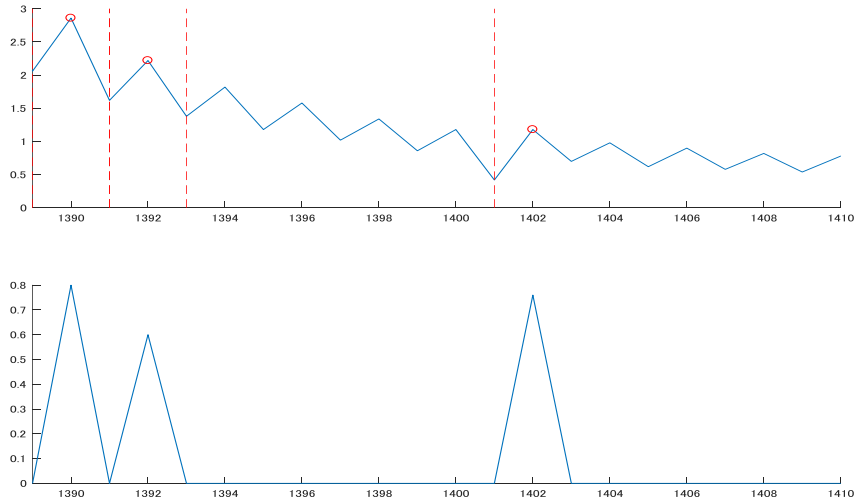

Figure 23: Enlarged view from 1389 to 1410 in Figure 22 (abnormal)

Table 1: Environment for evaluation

CPU	Intel® Core™ i5-6600 CPU 3.30GHz
Memory(RAM)	32GB®
OS	Windows 7 Professional
Language	MATLAB

de-energizing phase patterns is extracted from each cycle, but the first and second cycles have other convex-shaped patterns except in de-energizing phase patterns. Fig. 23 is an enlarged view of Fig. 17 during times from 1389 to 1410. It shows that there are 3 convex-shaped patterns whose amplitudes are the same as a normal de-energizing phase pattern.

Those experimental results above showed that amplitude function can extract anomalous convex-shaped patterns by giving the amplitude value that we wanted to find.

4.2 Performance

This section shows the dependency of the computational time of our algorithm on the length of time series, for various sensor data sets. Table 1 shows the environment for evaluation.

Figure 24 shows the trend graphs of experimental time series. Data labels "ECG", "Power", "Respiration" and "Valve" are electrocardiogram qtdb/se102, Dutch power demand dataset, a patient's respiration nprs44, and space shuttle Marotta Valve TEK16 in the UCR time series classification archive [13], respectively. NoisySine is the simulated time series shown in Figure 14.

The lengths of experimental time series are between 0 and 20000, at a step increase of 2000. The length of TEK16 is shorter than 20000, so we obtained a time series of length 20000 by concatenating the original time series multiple times.

Figure 25 shows the execution times for the 5 time series data sets. Each time is an average of the times of 10 trials, in order to reduce the effect of variance. The figure shows that the execution times depend on the behavior of data, but they are linear in n, where n is the length of the time series. The lengths of those time series are from 0 to 20000 at an increment of 2000 samples. The execution times for the length of 10000 are between 0.01 and 0.04 sec. It means that our algorithm can process one data point per between 10^{-6} and 4×10^{-6}. In other words, our algorithm enables real-time processing of time series with sampling periods between 1 and 4 microseconds.

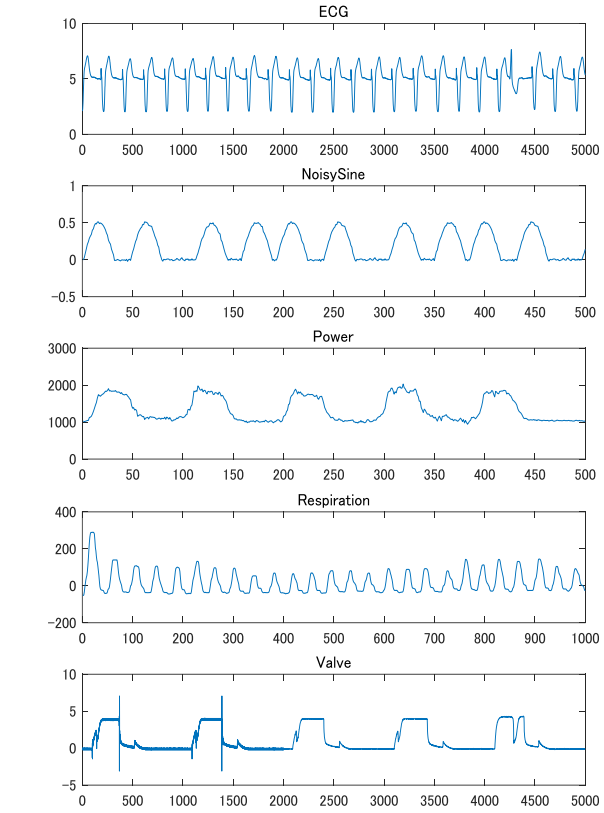

Figure 24: The trend graphs of experimental time series

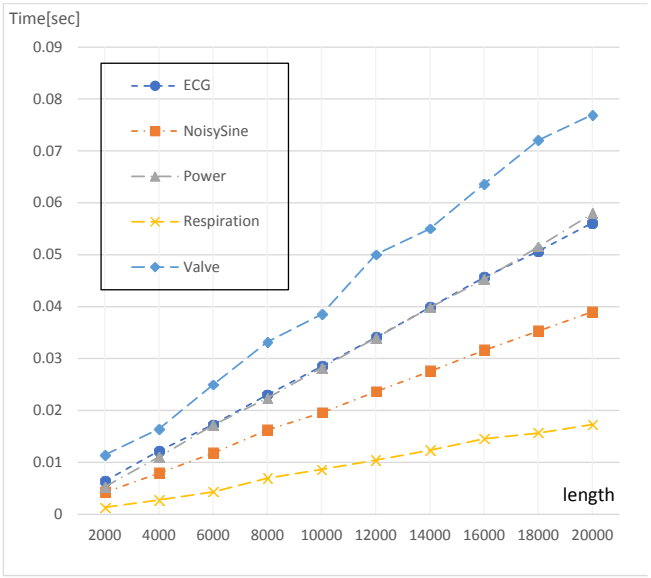

Figure 25: The execution times for experimental time series

5 CONCLUSIONS

We have proposed a new parameter-free online algorithm that calculates the amplitude of a maximal convex curve for extracting convex-shaped spikes in transient sensor data. We also showed that the computational complexity of our algorithm is O(n), where n is the length of input time series, and it enables real-time processing with sampling period ranging from 1 to 4 microseconds.

In future work, we will apply our algorithm to the following problems:

- Segmenting time series in order to identify operational regimes of equipment.
- Anomaly detection for equipment condition monitoring.

This work is supported by JSPS KAKENHI Grant Number 17K00161.

REFERENCES

[1] J. Zheng, D. Simplot-Ryl, C. Bisdikian, H. T. Mouftah: “The Internet of Things [Guest Editorial]”, Communications Magazine, IEEE, Vol.49, No.11, pp.30-31 (2011).

[2] M. Imamura, D. Nikovski, Z. Sahinoglu, M. Jones: "A Survey on Machine Learning for Equipment Condition Monitoring Using Sensor Big Data", IIEEJ Transactions on Image Electronics and Visual Computing Vol.2 No.2, pp. 112-121 (2014).

[3] E. Fink, B. P. Kevin: "Indexing of Compressed Time series", DATA MINING IN TIME SERIES DATABASES, World Scientific, pp. 43-65 (2004).

[4] M. Imamura, T. Nakamura, H. Shibata, N. Hirai, S. Kitagami, T. Munaka: "Leg Vibration Analysis for Time Series", IPSJ Journal, vol. 57, No.4, pp.1303-1318 (2016). (in Japanese).

[5] M. Imamura, D. Nikovski, M. Jones: "An Anomaly Detection System for Equipment Condition Monitoring", International Journal of Informatics Society (IJIS) VOL.8, NO.3, pp. 161-169 (2016).

[6] M. Imamura, H. Watanabe, D. Nikovski, A. Farahmand: "Online magnitude fluctuation analysis for anomaly detection", 10th International Workshop on Informatics (IWIN), p.269-275 (2017).

[7] P. Patel, E. Keogh, J. Lin and S. Lonardi, "Mining motifs in massive time series databases," 2002 IEEE International Conference on Data Mining, 2002. Proceedings., 2002, pp. 370-377.

[8] Y. Zhu et al., "Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins," 2016 IEEE 16th International Conference on Data Mining (ICDM), Barcelona, 2016, pp. 739-748.

[9] E. Keogh, J. Lin, A. Fu: "HOT SAX: finding the most unusual time series subsequence: algorithms and applications". The Fifth IEEE international conference on data mining, pp. 226–233, www.cs.ucr.edu/eamonn/discords/ (2005).

[10] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh : "Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures", VLDB 2008, pp.1542-1552 (2008).

[11] G. Box, P. Edward and G. Jenkins: Time Series Analysis, Forecasting and Control (fifth edition), WILEY, p. 54-67. (2016).

[12] E. Keogh, S. Chu, D. Hart and M. Pazzani, "An online algorithm for segmenting time series," Proceedings 2001 IEEE International Conference on Data Mining, San Jose, CA, 2001, pp. 289-296.

[13] Y. Chen, E. Keogh, B. Hu, and N. Begum, A. Bagnall, A. Mueen, and G. Batista: The UCR Time Series Classification Archive, (2015). <available from (http://www.cs.ucr.edu/~eamonn/time_series_data/>

(Received December 15, 2017)
(Revised February ,27 2018)

Makoto Imamura He received the M.E. degree from Kyoto University of Applied Mathematics and Physics in 1986 and the Ph.D. degree from Osaka University of the Information Science and Technology in 2008. From 1986 to 2016, he worked for Mitsubishi Electric Corp. In April 2016, he has moved to the school of Information and Telecommunication Engineering at Tokai University as a Professor. His research interests include machine learning, model-based design and their applications in prognostics and health management and cyber-physical system.

Junji Tsuda He received a B.E. degree from Tokai University, Japan in 2017. He is a master course student of the school of Information and Telecommunication Engineering at Tokai University. His research interests include IoT systems and data analytics.

Daniel Nikovski He received a PhD in robotics from Carnegie Mellon University in 2002, and is presently a senior member of research staff and group manager of the Data Analytics group at Mitsubishi Electric Research Laboratories. He has worked on probabilistic methods for reasoning, learning, planning, and scheduling, and their applications to hard industrial problems. He is a member of IEEE.

Masato Tsuru He received M.E. degree from Kyoto University, Japan in 1985, and then received his D.E. degree from Kyushu Institute of Technology, Japan in 2002. He worked at Oki Electric Industry Co., Ltd., Nagasaki University, and Japan Telecom Information Service Co., Ltd. In 2003, he moved to the Department of Computer Science and Electronics, Kyushu Institute of Technology as an Associate Professor, and then has been a Professor in the same department since April 2006. His research interests include performance measurement, modeling, and management of computer communication networks. He is a member of the ACM, IEEE, IEICE, and IPSJ.