Algorithm for error-free determination of the variance of all contiguous subsequences and fixed-length contiguous subsequences for a sequence of industrial measurement data

Andrzej Chmielowiec¹

Received: 29 October 2019 / Accepted: 6 March 2021 / Published online: 19 March 2021
© The Author(s) 2021

Abstract

The article presents an algorithm for fast and error-free determination of statistics such as the arithmetic mean and variance of all contiguous subsequences and fixed-length contiguous subsequences for a sequence of industrial measurement data. Additionally, it shows that both floating-point and integer representation can be used to perform this kind of statistical calculations. The author proves a theorem on the number of bits of precision that an arithmetic type must have to guarantee error-free determination of the arithmetic mean and variance. The article also presents the extension of Welford's formula for determining variance for the sliding window method—determining the variance of fixed-length contiguous subsequences. The section dedicated to implementation tests shows the running times of individual algorithms depending on the arithmetic type used. The research shows that the use of integers in calculations makes the determination of the aforementioned statistics much faster.

Keywords Variance · Arithmetic mean · Error-free variance computation · Measurement data representation

1 Introduction

The development of measurement, computational and data storage technologies provided us with huge amounts of information that can form the basis for various analyses. Statistics is one of the basic tools used to achieve this purpose. The need to process ever-increasing sets, however, means that the methods developed a few decades ago are no longer useful today due to their low efficiency. Consequently, a need arises

✉ Andrzej Chmielowiec
achmie@prz.edu.pl

¹ Faculty of Mechanics and Technology, Rzeszow University of Technology, Kwiatkowskiego 4, 37-450 Stalowa Wola, Poland

to constantly improve the existing algorithms and propose new solutions. Data from production processes of enterprises serve as a good example of an area where the amount of data collected has been growing rapidly. Currently, we are able to measure the manufactured goods and the condition of the production line at almost every stage of the process. This way, huge amounts of numerical data are collected and can be used, among others, in advanced methods of quality control and predictive maintenance. In this context, the concept of statistical process control created by Shewart (1931), Deming (1986) takes on a whole new dimension—a statistical analysis of large data sets. In general, analysis of statistical properties of a sequence of measurement data is a well-known issue. There are many proven methods to verify specific statistical hypotheses (Selvamuthu and Das 2018). They all assume, however, that we have a well-defined data sequence that we verify. The situation is completely different when we look for contiguous subsequences having specific statistical properties within a certain data sequence. The number of potential possibilities poses a problem of high computational complexity of the approach consisting in independent analysis of all possible contiguous subsequences. To illustrate this fact, we will use the example of the arithmetic mean calculated for all possible contiguous subsequences of a sequence consisting of 10^5 measurements. This task will take approximately $1.7 \cdot 10^{14}$ floating-point operations. Assuming that a single core of a typical processor is now capable of performing about $5 \cdot 10^9$ operations, it would take about 9.5 hours to make these calculations. Consequently, it is reasonable to look for such algorithmic solutions that will allow us to accelerate this type of analyses.

The question obviously arises: why set statistics for all contiguous subsequences? This is due to the fact that the analysis of statistics for the entire measurement series does not give us an answer if there were any situations during production that should be checked more closely. However, if we analyze all measurement contiguous subsequences, then we will be able to indicate the time periods in which something unexpected happened. In this way, for example, you can try to detect employee mistrust, technical problems with machines or problems with the quality of raw material. In addition, the analysis of individual contiguous subsequences gives us accurate information about what time period interesting events took place. To better present this idea, we decided to examine a series of 10^5 measurements from the production line of one of the factories. This series corresponds to about 30 days of production. The measured value was the diameter of the aluminum piston of an internal combustion engine. The measurements were carried out for each item produced. Ideally, these measurements should have a normal distribution. Therefore, we decided to test about 10^6 overlapping measuring contiguous subsequences of lengths $10^3, 2 \cdot 10^3, 3 \cdot 10^3, \dots, 10 \cdot 10^3$. Each of the sequences $(x_1, x_2, \dots, x_n)$ was subjected to a quick statistical test, which verified that all the elements of the sequence belong to a specific range $[\mu - d, \mu + d]$. The range was chosen in such a way that, assuming the normality of samples distribution, all samples were in the range with a probability of $1 - \alpha$. Verification of this property is very quick, because it comes down to checking the conditions $\min\{x_i\} < \mu - d$ and $\max\{x_i\} > \mu + d$ with $d = \sigma\sqrt{2} \operatorname{erf}^{-1}((1 - \alpha)^{1/n})$. Each time the test was rejected, we increased the counter of participating samples by 1. Each sample participated in approximately 50,000 tests. The results obtained are presented in Fig. 1.

Fig. 1 Number of test rejections in which the sample participated

It can be deduced from this plot that potentially interesting phenomena occurred around samples $0.6 \cdot 10^4$, $2.5 \cdot 10^4$, $3.2 \cdot 10^4$, $4.9 \cdot 10^4$, $5.8 \cdot 10^4$, $6.1 \cdot 10^4$ and $9 \cdot 10^4$, which corresponds to specific periods during production. Figures 2 and 3 show histograms for samples in the ranges (57500, 58500) and (90000, 91000). In addition, there is a graph showing the histogram of all tested samples (10^5 measurements) scaled by factor 0.01. The above ranges were chosen due to the fact that in the most obvious way they show evident deviations from what can be observed for entire data set. The real values taken by the samples were linearly converted into numbers in the range $[-15, 15]$. It was necessary due to the inability to show the real dimensions of the measured object. The presented histograms show an unusually large number of products with dimensions matching the design specification. This can be caused, for example, by the replacement of the machining tool with a new or deliberate action of employees who modified the settings of the measuring machine in order to achieve the appropriate effect during quality control. The search for such regularities in the production process is the main reason for the development of the calculation methods presented in this article.

It should be emphasized here that the problem of finding anomalies in time series has been the subject of intense research for the last twenty years. There are many algorithms and techniques used to search for anomalies provided that we know the details of their location or the size of the time window to be analyzed (Keogh et al. 2005, 2006; Senin et al. 2015).

The current methods of finding anomalies in time series can be divided into three basic categories in terms of the type of results obtained. We distinguish algorithms for finding point anomalies, structural anomalies and anomalous series (in the case of many series). By point anomalies we mean single, significant deviations from the values for a given time series (Hawkins 1980; Evans et al. 2015). On the other hand, structural anomalies are such contiguous time series subsequences that deviate from the values (Keogh and Lin 2005; Zhang and Meratnia 2010; Senin et al. 2015). In contrast, by anomalous series we mean whole series that are somehow different from other observed series (Hyndman et al. 2015; Laptev et al. 2015). In many cases, one of the most important element that allows us for search structural anomalies is determining the value of appropriate statistics for a large class of contiguous subsequences. Therefore, the algorithms presented in this article may be of considerable importance

Fig. 2 Histogram of samples from the interval (57500, 58500) against the histogram of all samples scaled by a factor of 0.01

Fig. 3 Histogram of samples from the interval (90000, 91000) against the histogram of all samples scaled by a factor of 0.01

for such methods. Anomaly search algorithms using artificial intelligence methods are currently very popular. The publication based on the experience gathered by the Intel Wang et al. (2018) is highly interesting from the point of view of the production process. Beside article Chalapathy and Chawla (2019) presents a comprehensive overview of anomaly detection methods using machine learning. For this type of method, it is also very important to be able to quickly determine the value of statistics and divide the data set into appropriate clusters.

Article Ding et al. (2016) introduces the division of algorithms that detect anomalies by the type of mechanisms used to detect them. The authors distinguish four

basic methods for detecting anomalies: classification, nearest neighbor, clustering and statistical. It should be emphasized that the last three methods more or less directly use functions that are either statistics or certain measures originating in the theory of probability. Quick determination of various types of statistics is therefore crucial from the point of view of the computational complexity of these algorithms.

The next sections of this article focus primarily on determining the arithmetic mean $\mu_n = \frac{1}{n} \sum_{i=1}^n x_i$ and variance $v_n = \frac{1}{n} \sum_{i=1}^n (x_i - \mu_n)^2$, but the methods proposed therein can also be used to determine other statistics. It should be emphasized that the use of the biased estimator for the variance is appropriate in this case. This is due to its computational properties that allow for more effective determination of its numerical value. When its value is determined, it is very easy to use it to calculate the value of the unbiased estimator by multiplying the result by $\frac{n}{n-1}$. This is the reason why we will consistently use the biased variance estimator later in this article. Before we get to that, however, we briefly summarise the current state of knowledge on the calculation of variance, because it is undoubtedly one of those parameters whose calculation poses a lot of numerical problems. The first algorithm, which significantly reduces the numerical error in the determination of variance, was proposed by Welford (1962). His approach is modelled on the idea put forward by Box and Hunter (1959), who proposed a similar technique for determining residual sums. The Welford formula was improved by Knuth (1981), who gave it a very elegant and computationally effective form

$$\begin{aligned}\mu_{n+1} &= \mu_n + \frac{1}{n+1}(x_{n+1} - \mu_n), \\ v_{n+1} &= v_n + (x_{n+1} - \mu_n)(x_{n+1} - \mu_{n+1}).\end{aligned}$$

Next, Neely (1966) developed the idea presented by Kahan (1965) and proposed an algorithm that needs double access to data; its main element was the correction of input data by the mean computed during the first round. This way, he developed a procedure highly resistant to numerical errors in many practical applications. In search for solutions offering the highest resistance to rounding errors, Rodden (1967) presented the concept of replacing floating-point numbers with integers. We will elaborate on his proposal later on in this article, adding estimates on the number of bits necessary for error-free computation of variance. Then, Van Reeken (1968), Youngs and Cramer (1971) extended the set of algorithms described by Neely. Ling (1974) summarised the methods described above and compared their operation for many different input data distributions. Next, Hanson (1975), West (1979) and Cotton (1975) added an option to determine weighted variance to the existing methods and increased the numerical stability of the results obtained. A substantial contribution to the estimation of numerical errors for individual methods was made by Chan and Lewis (1979), Chan et al. (1982), Chan et al. (1983). Recent years have seen a fairly intense development of algorithms and methods focused on the processing of large data sets. The two basic directions of research work are worth mentioning. The first one includes online calculation methods developed by Kreinovich et al. (2007), Pébay (2008), Villaverde and Xiang (2008) and Terriberly (2008). The second one involves parallel data processing methods proposed by Schubert et al. (2015), Schubert and Gertz (2018) and

Gouin et al. (2016). Peaby, Terriberly, Kolla and Bennett published very valuable results that combine both of these research directions. In article Pébay et al. (2016) they defined general recursive formulas for determining higher order moments. Their approach can be the basis for many parallel and numerically stable algorithms. This article focuses on computationally effective methods for determining statistics for all possible contiguous subsequences or for a specific group of contiguous subsequences.

2 Computation of the arithmetic mean and variance after conversion of measurement data into integer values

A classic algorithm for determining variance based on the following relation

$$\mu_n = \frac{1}{n} \sum_{i=1}^n x_i,$$

$$v_n = \frac{1}{n} \sum_{i=1}^n x_i^2 - \mu_n^2.$$

poses a multitude of numerical problems. In production control, calculation errors grow along with rising product quality. This is due to the fact that the expected value is calculated by adding numbers that are very similar in value. However, calculation of the variance under the aforesaid method involves deducting the square of the arithmetic mean from the root mean square, which poses a number of problems connected with calculation precision.

The arithmetic and logic unit of modern processors usually supports both integer and floating point operations (mathematical coprocessor). Nowadays, a 64-bit application processor is able to perform very fast operations on 32 and 64-bit integers, and 32 and 64-bit floating point numbers (single and double precision). Floating point numbers are given by

$$S \cdot M \cdot 2^C,$$

where S is a sign bit, M is the mantissa and C is the exponent. In the case of floating point numbers, the IEEE-754 standard (IEEE 2008) specifies that in the single-precision format, the mantissa and the exponent have 23 and 8 bits, respectively. In the double-precision format, the mantissa and the exponent have 52 and 11 bits, respectively. The speed of multiplication of numbers in particular formats depends on the type of processor. Table 1 presents the times of scalar multiplication of vectors consisting of 10^8 elements of a given type.

This table clearly shows that integer multiplication is more efficient than floating point multiplication. Therefore, by switching to integer arithmetic, we can achieve two goals: speed up calculations and eliminate rounding errors. Such an approach may raise doubts, as the measurement results are, by their nature, floating point values. However, if we assume that we know the nominal value of the measured element and assume

Table 1 Scalar Multiplication times of vectors consisting of 10^8 elements of a given type

Typ liczby	Intel Core i5-3360M 2.80GHz	Intel Xeon E3-1240V2 3.40GHz
int32	0.063 s	0.055 s
int64	0.072 s	0.063 s
single	0.092 s	0.079 s
double	0.093 s	0.082 s
long double	0.140 s	0.113 s

a specific measurement precision, then we can easily calculate the relevant statistics by using integer operations. If, for example, the measured value has a nominal value of $\delta_0 = 125.950$ [mm] and the measurement precision is $\varepsilon_0 = 0.001$ [mm], then the measurement results δ_i can be represented as integers m_i , where

$$x_i = \delta_0 + m_i \varepsilon_0$$

is the actually measured value. By determining m_i from the above formula, we get

$$m_i = \frac{x_i - \delta_0}{\varepsilon_0}.$$

The great advantage of this representation is that it allows for computing the expected value and variance without rounding errors. Some of them cannot be avoided in the case of floating point representation, because if, for example, a measuring device produces results with a precision of 0.1, then the floating point representation of number $\frac{1}{10}$ in the binary system is infinite and is given by:

$$\begin{aligned} \frac{1}{10} &= \frac{1}{2^4} + \frac{1}{2^5} + \frac{1}{2^8} + \frac{1}{2^9} + \frac{1}{2^{12}} + \frac{1}{2^{13}} + \dots \\ &= (0.0001100110011(0011))_2. \end{aligned}$$

We can avoid such problems by switching to integer representation. Its correct operation, however, is conditional on precise control of the factor by which the integers being processed had been previously multiplied. This approach will be most effective for sets that have a well-defined and appropriately small range of possible values. For example, such a set is the measurement results. If we perform the measurement using a 16-bit sensor, we can be sure that we will get no more than 65536 possible values. However, if variables can potentially take any floating point values, then in such cases the transition to an integer representation will not make much sense.

Later on in the article we focus on determining the number of bits of the mantissa necessary for correct calculation of the arithmetic mean μ_n and variance v_n . In doing so, we assume that only integer numbers are used, i.e. that the exponent has zero bits. Under these assumptions, the following theorem is true:

Theorem 1 Let δ_0 be the nominal value of the measured value and ε_0 the calculation precision. We also assume that the measuring range is set and the results fall within the range $[\delta_0 - r_0, \delta_0 + r_0]$. If n is the number of measurements, $A_n(\varepsilon_0, r_0)$ is the number of integer bits necessary to correctly determine the arithmetic mean and $V_n(\varepsilon_0, r_0)$ is the number of integer bits necessary for correct determination of the variance, then:

$$A_n(\varepsilon_0, r_0) = \left\lfloor \log_2 \left(\frac{2nr_0}{\varepsilon_0} \right) \right\rfloor + 1,$$

$$V_n(\varepsilon_0, r_0) = \left\lfloor \log_2 \left(\frac{nr_0^2}{\varepsilon_0^2} \right) \right\rfloor + 1.$$

Proof To prove the theorem it suffices to show that the largest values appearing in the calculation of μ_n and v_n are contained in bit registers with lengths $A_n(\varepsilon_0, r_0)$ and $V_n(\varepsilon_0, r_0)$, respectively. Note that if the calculation precision is ε_0 , then there are $[\delta_0 - r_0, \delta_0 + r_0]$ possible measurement results in the interval $\left[\frac{2r_0}{\varepsilon_0} \right]$. This means that the measurement results x_i stand in a one-to-one relation with integers $m_i = \frac{x_i - \delta_0}{\varepsilon_0}$ satisfying the following inequality

$$-\frac{r_0}{\varepsilon_0} \leq m_i < \frac{r_0}{\varepsilon_0}.$$

Observe that

$$\mu_n = \delta_0 + \frac{\varepsilon_0}{n} \sum_{i=1}^n m_i.$$

Therefore, to precisely calculate the arithmetic mean it is necessary to precisely compute the sum $m_1 + \dots + m_n$, satisfying the relation

$$-\frac{nr_0}{\varepsilon_0} \leq \sum_{i=1}^n m_i < \frac{nr_0}{\varepsilon_0}.$$

Remembering that the number of bits necessary to represent the nonnegative integer m is expressed by the formula $\lfloor \log_2 m \rfloor + 1$ and considering the sign bit, we obtain

$$\begin{aligned} A_n(\varepsilon_0, r_0) &= \left\lfloor 1 + \log_2 \left(\frac{nr_0}{\varepsilon_0} \right) \right\rfloor + 1 \\ &= \left\lfloor \log_2 \left(\frac{2nr_0}{\varepsilon_0} \right) \right\rfloor + 1. \end{aligned}$$

To calculate the variance, we use the fact that it is invariant to shifts by a fixed value and can be expressed as

$$v_n = \text{Var}(\{x_i\}) = \text{Var}(\{x_i - \delta_0\})$$

$$\begin{aligned}
&= \frac{1}{n} \sum_{i=1}^n (\varepsilon_0 m_i)^2 - \left(\frac{1}{n} \sum_{i=1}^n \varepsilon_0 m_i \right)^2 \\
&= \varepsilon_0^2 \left(\frac{1}{n} \sum_{i=1}^n m_i^2 - \left(\frac{1}{n} \sum_{i=1}^n m_i \right)^2 \right).
\end{aligned}$$

From Jensen's inequality (Jensen 1906) for convex function it follows that $\frac{1}{n} \sum_{i=1}^n m_i^2 \geq \left(\frac{1}{n} \sum_{i=1}^n m_i \right)^2$, therefore, to compute the variance precisely, it is necessary to accurately determine the sum of squares $m_1^2 + \dots + m_n^2$ satisfying the following relation

$$0 \leq \sum_{i=1}^n m_i < \frac{nr_0^2}{\varepsilon_0^2}.$$

Consequently, we obtain

$$V_n(\varepsilon_0, r_0) = \left\lfloor \log_2 \left(\frac{nr_0^2}{\varepsilon_0^2} \right) \right\rfloor + 1,$$

which ends the proof. $\square$

Corollary 1 *If we have a sequence of measurement data $(x_1, \dots, x_n)$ and there exist such constants δ_0, ε_0 and r_0 that*

$$x_i = \delta_0 + m_i \varepsilon_0, \quad x_i \in [\delta_0 - r_0, \delta_0 + r_0)$$

for certain integers m_i , then the arithmetic mean μ_n and variance v_n of the sequence of measurement data can be computed precisely using the formulas

$$\mu_n = \delta_0 + \frac{\varepsilon_0}{n} \sum_{i=1}^n m_i, \tag{1}$$

$$v_n = \varepsilon_0^2 \left(\frac{1}{n} \sum_{i=1}^n m_i^2 - \left(\frac{1}{n} \sum_{i=1}^n m_i \right)^2 \right), \tag{2}$$

on condition that for the purpose of computing the arithmetic mean we use the integer data type with a length of at least $A_n(\varepsilon_0, r_0)$ bits, and for the purpose of computing the variance we use the integer data type with a length of at least $V_n(\varepsilon_0, r_0)$ bits.

To illustrate the importance of the result, imagine an optical measuring system on the production line. Let's assume that it is able to measure manufactured elements with an accuracy of 2^{-20} [m] (about 1 [μm]), assuming that their dimensions do not exceed 2^{-4} [m] (about 6 [cm]). If we use the 64-bit integer type to operate on measurement data, then the Theorem 1 shows that we will be able to correctly determine the variance

for about $2 \cdot 10^9$ measured elements. As for the production process, this number is so large that it gives a real opportunity to analyze this process in a fairly long time horizon.

3 The algorithm for computing the arithmetic mean and variance of all contiguous subsequences

In this section we assume that we have a certain sequence of n observations

$$t = (x_1, x_2, \dots, x_{n-1}, x_n).$$

We define contiguous subsequence s of sequence t to be sequence $s = (x_{l+1}, x_{l+2}, \dots, x_{l+k})$ such that for every i satisfying the condition $l + 1 \leq i \leq l + k$, $x_i \in s$. We also assume that the order of elements of contiguous subsequence s is the same as in sequence t . The contiguous subsequence can therefore be treated as a kind of time window. Because the contiguous subsequence is precisely defined by the index of its first and last term, we also use the notation $s_{l+1, l+k}$ to denote contiguous subsequence $(x_{l+1}, x_{l+2}, \dots, x_{l+k})$.

Let t_k signify the number of k -long contiguous subsequences of sequence s . Note that this number is given by $t_k = n + 1 - k$. Therefore, the number of all possible contiguous subsequences T of sequence t equals

$$\begin{aligned} T &= \sum_{k=1}^n t_k = \sum_{k=1}^n (n + 1 - k) \\ &= n(n + 1) - \frac{n(n + 1)}{2} = \frac{n(n + 1)}{2}. \end{aligned}$$

Now, let us assume that we want to calculate a certain statistic for all possible contiguous subsequences of an n -element sequence. Let the statistics be, for example, the arithmetic average and variance. Note that if k -element contiguous subsequence $s = (x_{l+1}, x_{l+2}, \dots, x_{l+k})$, is given, then the time needed to compute both the arithmetic mean and variance is proportional to the number of its elements. This arises directly from the definitions of these values:

$$\begin{aligned} \mu(s) &= \frac{1}{k} (x_{l+1} + x_{l+2} + \dots + x_{l+k}), \\ \nu(s) &= \frac{1}{k} (x_{l+1}^2 + x_{l+2}^2 + \dots + x_{l+k}^2) - \mu(s)^2. \end{aligned}$$

In order to compute the arithmetic mean, one needs $k - 1$ additions and one division (k operations in total), and to compute the variance, one needs $k - 1$ additions, $k + 1$ squarings, one subtraction, one division and operations needed to determine the arithmetic mean ($3k + 2$ operations in total). We can therefore assume that the number of floating point operations needed to calculate the arithmetic mean and the variance is limited by Ck , where C is a constant and k is the number of elements of the contiguous subsequence. Consequently, the complexity of the algorithm for computing

the arithmetic mean and variance of all possible contiguous subsequences of an n -element sequence may be expressed by

$$\begin{aligned}
 R_n &= \sum_{k=1}^n x_k \cdot Ck \\
 &= C \sum_{k=1}^n (n+1-k)k \\
 &= C(n+1) \sum_{k=1}^n k - C \sum_{k=1}^n k^2 \\
 &= C \left(\frac{n(n+1)^2}{2} - \frac{n(n+1)(2n+1)}{6} \right) \\
 &= \frac{C}{6} n(n+1)(n+2) = O(n^3).
 \end{aligned}$$

Note that the operations of determining the sum and the sum of squares materially affect the complexity of computing the arithmetic mean and variance. These are the only operations that depend on the number of elements in the contiguous subsequence. The number of the remaining operations is constant and independent of the length of the processed sequence. Therefore, the only way to reduce the computational complexity of the algorithm referred to above is to speed up the calculation of sums and sums of squares. It turns out that if the contiguous subsequences are processed in the right order, it takes a constant amount of time to compute the values of the arithmetic mean and variance. It is similar for other statistics. Consequently, the whole algorithm will be presented in the most general form possible.

Let us set all contiguous subsequences in the order shown below:

$$\begin{aligned}
 &(x_1), (x_1, x_2), \dots, (x_1, \dots, x_{n-1}), (x_1, \dots, x_n), \\
 &(x_2), (x_2, x_3), \dots, (x_2, \dots, x_n), \\
 &\vdots \\
 &(x_{n-1}), (x_{n-1}, x_n), \\
 &(x_n).
 \end{aligned}$$

Note that each row begins with a contiguous subsequence of length 1 and that the difference in length between neighbouring contiguous subsequences in one row is also 1. This means that the time needed to compute the expected value and variance for the first contiguous subsequence in a row is constant. In the case of neighbouring contiguous subsequences located in one row, the expected value and variance of the next contiguous subsequence can also be computed in a constant time.

Definition 1 Let F be a function defined for every contiguous subsequence $s = (x_{l+1}, \dots, x_{l+k})$. We say that F is *forward computationally separable function*, if

it meets the condition:

$$F(x_{l+1}, \dots, x_{l+k}) = h(F(x_{l+1}, \dots, x_{l+k-1}), x_{l+k})$$

for a certain function h having constant computational complexity. Moreover, we say that F is *backward computationally separable function*, if it meets the condition:

$$F(x_{l+2}, \dots, x_{l+k}) = g(F(x_{l+1}, x_{l+2}, \dots, x_{l+k}), x_{l+1})$$

for a certain function g having constant computational complexity. If function is both forward and backward computationally separable, then it is *computationally separable function*.

Corollary 2 Functions

$$M_1(s) = M_1(x_{l+1}, \dots, x_{l+k}) = x_{l+1} + \dots + x_{l+k},$$

$$M_2(s) = M_2(x_{l+1}, \dots, x_{l+k}) = x_{l+1}^2 + \dots + x_{l+k}^2,$$

are computationally separable functions.

Proof In the case of M_1 we simply assume that $h(x, y) = x + y$ and $g(x, y) = x - y$. For M_2 we assume that $h(x, y) = x + y^2$ and $g(x, y) = x - y^2$. $\square$

Before we proceed any further, it is worth showing that there are functions that are either only forward or only backward computationally separable. We can use continued fractions as an example. If F_1 and F_2 are functions defined as follows

$$F_1(x_{l+1}, x_{l+2}, \dots, x_{l+k}) = \frac{1}{x_{l+k} + \frac{1}{\dots + \frac{1}{x_{l+2} + \frac{1}{x_{l+1}}}}},$$

$$F_2(x_{l+1}, x_{l+2}, \dots, x_{l+k}) = \frac{1}{x_{l+1} + \frac{1}{x_{l+2} + \frac{1}{\dots + \frac{1}{x_{l+k}}}}},$$

then F_1 is a forward computationally separable function and F_2 is backward computationally separable function. It results directly from the following relations

$$F_1(x_{l+1}, \dots, x_{l+k}) = \frac{1}{x_{l+k} + F_1(x_{l+1}, x_{l+2}, \dots, x_{l+k-1})},$$

$$F_2(x_{l+2}, \dots, x_{l+k}) = \frac{1}{F_2(x_{l+1}, x_{l+2}, \dots, x_{l+k})} - x_{l+1}.$$

In other words, function h for F_1 takes the form $h(x, y) = \frac{1}{y+x}$, while function g for F_2 takes the form $g(x, y) = \frac{1}{x} - y$.

Let R be any statistic such that for every contiguous subsequence $s = (x_{l+1}, x_{l+2}, \dots, x_{l+k})$ meets the condition $R(s) = f_s(F(s))$, where f_s is a function dependent on s having constant computational complexity and $F(s) = (F_1(s), \dots, F_m(s))$ is a forward computationally separable function. Since s is explicitly designated by l and k , then we can define f_s as $f_{l+1, l+k}$.

Using these designations, we can define Algorithm 1, that computes statistic R for all contiguous subsequences.

Data: Sequence $t = (x_1, \dots, x_n)$, statistic R meeting the condition

$R(s) = f_s(F(s)) = f_s(F_1(s), \dots, F_m(s))$ or F being a forward computationally separable function.

Result: Sequence of statistics $(r_{i,j} = R(s_{i,j}))$ computed for all possible contiguous subsequences t .

```

1 for  $i = 1, 2, \dots, n$  do
2    $u_{i,i} \leftarrow (F_1(s_{i,i}), \dots, F_m(s_{i,i}))$ ;
3    $r_{i,i} \leftarrow f_{i,i}(u_{i,i})$ ;
4   for  $j = i + 1, i + 2, \dots, n$  do
5     /* We compute  $u_{i,j} = (F_1(s_{i,j}), \dots, F_m(s_{i,j}))$  */
6      $u_{i,j} \leftarrow h(u_{i,j-1}, x_j)$ ;
7     /* We compute statistics  $r_{i,j} = R(u_{i,j}) = f_{i,j}(u_{i,j})$  */
8      $r_{i,j} \leftarrow f_{i,j}(u_{i,j})$ ;
9   end
10 end

```

Algorithm 1: Computation of statistics for all contiguous subsequences.

Theorem 2 If $s = (x_{l+1}, \dots, x_{l+k})$ is a contiguous subsequence of sequence $t = (x_1, \dots, x_n)$, statistic R is given by $R(s) = f_s(F(s)) = f_s(F_1(s), \dots, F_m(s))$, functions f_s, h have fixed computational complexity $O(1)$ and values $F_i((x_j))$ can also be computed in a constant time $O(1)$, then Algorithm 1 allows for computing R for all contiguous subsequences t in time $O(n^2)$.

Proof First, we prove that the algorithm operates correctly. For this purpose, it suffices to prove that $r_{i,j} = R(s_{i,j}) = R((x_i, \dots, x_j))$. Note that lines (2) and (5) of Algorithm 1 define the following recursive relation

$$\begin{aligned} u_{i,i} &= (F_1(s_{i,i}), \dots, F_m(s_{i,i})), \\ u_{i,j} &= h(u_{i,j-1}, x_j). \end{aligned}$$

We prove by induction on j that $u_{i,j}$ was computed correctly. The first induction step for every i is true because this is how $u_{i,i}$ values are initiated. In step two we assume that for $j - 1$ the following is true

$$u_{i,j-1} = (F_1(s_{i,j-1}), \dots, F_m(s_{i,j-1})).$$

Consequently, we obtain

$$u_{i,j} = h(u_{i,j-1}, x_j)$$

$$\begin{aligned}
&= h\left((F_1(s_{i,j-1}), \dots, F_m(s_{i,j-1})), x_j\right) \\
&= (F_1(s_{i,j}), \dots, F_m(s_{i,j})).
\end{aligned}$$

Therefore, R is computed correctly, which results directly from step (6) of Algorithm 1, because

$$\begin{aligned}
r_{i,j} &= f_{i,j}(u_{i,j}) = f_{i,j}(F_1(s_{i,j}), \dots, F_m(s_{i,j})) \\
&= F(s_{i,j}) = R(s_{i,j}).
\end{aligned}$$

This ends first part of the proof. In order to estimate computational complexity, note that the assumptions show that steps (2), (3), (5) and (6) of the Algorithm have constant computational complexity $O(1)$. Consequently, the determination of every $r_{i,j}$ also has constant computational complexity $O(1)$. Finally, we conclude that the computational complexity of the entire algorithm equals the number of computed $r_{i,j}$ values which is exactly the same as the number of two-element combinations from an n -element set, that is $\frac{n(n+1)}{2}$. Therefore, we proved that the algorithm has complexity $O(n^2)$, which ends the proof. $\square$

Theorem 3 *If $s = (x_{l+1}, \dots, x_{l+k})$ and $R(s) = \mu(s) = \frac{1}{k}M_1(s)$, then Algorithm 1 makes it possible to compute the arithmetic mean for all contiguous subsequences of sequence $(x_1, \dots, x_n)$ in time $O(n^2)$.*

Proof If $s_{i,j} = (x_i, x_{i+1}, \dots, x_j)$, then from definition of R it follows that $F_1(s_{i,j}) = M_1(s_{i,j}) = x_i + \dots + x_j$ and $f_{i,j}(x) = \frac{1}{j-i+1}(x)$. Using these designations, Algorithm 1 takes the form presented as Algorithm 2. First, we prove that this algorithm works correctly. For this purpose, it suffices to prove that $r_{i,j}$ contains the arithmetic mean of the elements of sequence $(x_i, x_{i+1}, \dots, x_j)$. To do this, let us note that for a given i the elements of $u_{i,j}$ meet the following recursive relation:

$$\begin{aligned}
u_{i,i} &= x_i, \\
u_{i,j} &= u_{i,j-1} + x_j.
\end{aligned}$$

By expanding this recursion, we obtain that $u_{i,j} = x_i + x_{i+1} + \dots + x_j$. But

$$r_{i,j} = \frac{1}{j-i+1}u_{i,j} = \frac{x_i + x_{i+1} + \dots + x_j}{j-i+1},$$

which proves that $r_{i,j}$ is the arithmetic mean of sequence $(x_i, x_{i+1}, \dots, x_j)$. The estimation of the computational complexity arises directly from Theorem 2, because the individual computations of $u_{i,j}$ and $r_{i,j}$ have constant computational complexity. $\square$

Theorem 4 *If $s = (x_{l+1}, \dots, x_{l+k})$ and $R(s) = v(s) = \frac{1}{k}M_2(s) - \left(\frac{1}{k}M_1(s)\right)^2$, then Algorithm 1 makes it possible to compute the variance for all contiguous subsequences of sequence $(x_1, \dots, x_n)$ in time $O(n^2)$.*

Data: Sequence $t = (x_1, \dots, x_n)$.

Result: Sequence of arithmetic means $(r_{i,j} = \mu(s_{i,j}))$ computed for all possible contiguous subsequences t .

```

1 for  $i = 1, 2, \dots, n$  do
2    $u_{i,i} \leftarrow x_i$ ;
3    $r_{i,i} \leftarrow u_{i,i}$ ;
4   for  $j = i + 1, i + 2, \dots, n$  do
5      $u_{i,j} \leftarrow u_{i,j-1} + x_j$ ;
6      $r_{i,j} \leftarrow \frac{1}{j-i+1} u_{i,j}$ ;
7   end
8 end

```

Algorithm 2: Computation of the arithmetic mean for all contiguous subsequences.

Proof If $s_{i,j} = (x_i, \dots, x_j)$, then from the definition of R it follows that

$$\begin{aligned}
 F(s_{i,j}) &= (F_1(s_{i,j}), F_2(s_{i,j})), \\
 F_1(s_{i,j}) &= M_1(s_{i,j}) = x_i + \dots + x_j, \\
 F_2(s_{i,j}) &= M_2(s_{i,j}) = x_i^2 + \dots + x_j^2, \\
 f_{i,j}(y_1, y_2) &= \frac{1}{j-i+1}(y_2) - \left(\frac{1}{j-i+1}(y_1) \right)^2.
 \end{aligned}$$

Using these designations, Algorithm 1 takes the form presented as Algorithm 3. First, we prove that this algorithm works correctly. For this purpose, it suffices to prove that $r_{i,j}$ contains the variance of the elements of sequence $(x_i, \dots, x_j)$. To do this, let us note that for a given i the elements of $u_{i,j}$ meet the following recursive relation:

$$\begin{aligned}
 u_{i,i} &= (x_i, x_i^2), \\
 u_{i,j} &= u_{i,j-1} + (x_j, x_j^2).
 \end{aligned}$$

By expanding this recursion, we obtain that $u_{i,j} = (x_i + \dots + x_j, x_i^2 + \dots + x_j^2)$. But

$$\begin{aligned}
 r_{i,j} &= f_{i,j}(x_i + \dots + x_j, x_i^2 + \dots + x_j^2), \\
 &= \frac{x_i^2 + \dots + x_j^2}{j-i+1} - \left(\frac{x_i + \dots + x_j}{j-i+1} \right)^2
 \end{aligned}$$

which is the variance of sequence $(x_i, \dots, x_j)$. The estimation of the computational complexity arises directly from Theorem 2, because the individual computations of $u_{i,j}$ and $r_{i,j}$ have constant computational complexity. $\square$

The variance computation method used in Theorem 4 is not numerically stable. That is why it should be applied only in cases where the values of functions M_1 and M_2 are determined precisely (i.e. are free from numerical errors). The conditions that must be met in this regard have been set out in Theorem 1 and Corollary 1. When numerical errors cannot be avoided, it is advisable to compute the variance using the

Data: Sequence $t = (x_1, \dots, x_n)$.

Result: Sequence of variances $(r_{i,j} = v(s_{i,j}))$ computed for all possible contiguous subsequences t .

```

1 for  $i = 1, 2, \dots, n$  do
2    $u_{i,i} \leftarrow (x_i, x_i^2)$ ;
3    $r_{i,i} \leftarrow 0$ ;
4   for  $j = i + 1, i + 2, \dots, n$  do
5      $u_{i,j} \leftarrow u_{i,j-1} + (x_j, x_j^2)$ ;
6      $r_{i,j} \leftarrow f_{i,j}(u_{i,j})$ ;
7   end
8 end

```

Algorithm 3: Computation of the variance for all contiguous subsequences.

method proposed by Welford (1962), which also defines a forward computationally separable function.

Theorem 5 If $s = (x_{l+1}, \dots, x_{l+k})$ and $R(s) = (\mu(s), v(s))$, then the Welford recursive formula and Algorithm 1 allow us to simultaneously determine the arithmetic mean and variance for all contiguous subsequences of sequence $(x_1, \dots, x_n)$ in time $O(n^2)$.

Proof If $s_{i,j} = (x_i, \dots, x_j)$, then the following relations for the arithmetic mean and variance result from the definitions of μ and v and the Welford formulas Welford (1962)

$$\mu(s_{i,j}) = \frac{1}{j-i+1} \sum_{k=i}^j x_k, \quad (3)$$

$$\mu(s_{i,j}) = \frac{j-i}{j-i+1} \mu(s_{i,j-1}) + \frac{1}{j-i+1} x_j, \quad (4)$$

$$v(s_{i,j}) = \frac{1}{j-i+1} \sum_{k=i}^j (x_k - \mu(s_{i,j}))^2, \quad (5)$$

$$v(s_{i,j}) = v(s_{i,j-1}) + \frac{j-i}{j-i+1} (x_j - \mu(s_{i,j-1}))^2. \quad (6)$$

The Welford formulas shown in Eqs. 4 and 6 confirm that $s_{i,j} \mapsto (\mu(s_{i,j}), v(s_{i,j}))$ is a forward computationally separable function. This means that Algorithm 1 takes the form presented as Algorithm 4 and proves correctness of presented method. Complexity $O(n^2)$ results from the fact that all operations in the loop have complexity $O(1)$. This ends the proof. $\square$

Data: Sequence $t = (x_1, \dots, x_n)$.

Result: Sequence of statistics $(r_{i,j} = (\mu(s_{i,j}), v(s_{i,j})))$ computed for all possible contiguous subsequences t .

```

1 for  $i = 1, 2, \dots, n$  do
2    $(u_{i,i}, v_{i,i}) \leftarrow (x_i, 0)$ ;
3    $r_{i,i} \leftarrow (u_{i,i}, v_{i,i})$ ;
4   for  $j = i + 1, i + 2, \dots, n$  do
5      $u_{i,j} \leftarrow \frac{j-i}{j-i+1}u_{i,j-1} + \frac{1}{j-i+1}x_j$ ;
6      $v_{i,j} \leftarrow v_{i,j-1} + \frac{j-i}{j-i+1}(x_j - u_{i,j-1})^2$ ;
7      $r_{i,j} \leftarrow (u_{i,j}, v_{i,j})$ ;
8   end
9 end

```

Algorithm 4: Computation of the arithmetic mean and variance for all contiguous subsequences using Welford's formula.

4 The algorithm for computing the arithmetic mean and variance for fixed-length contiguous subsequences

Theorem 2 which has been proven, is of general nature and applies to many different statistics. Theorems 3, 4 and 5, which use Algorithm 1 to compute the arithmetic mean and variance for all contiguous subsequences of a given sequence $t = (x_1, x_2, \dots, x_n)$, serve as an example. However, it is not always necessary to calculate statistics for all possible contiguous subsequences. Sometimes we just want to scan a series of observations using a window of a predefined width. If such a window covers w observations, then the analysed contiguous subsequences are arranged in the following series

$$(x_1, \dots, x_w), (x_2, \dots, x_{w+1}), \dots, (x_{n-w+1}, \dots, x_n).$$

In order to effectively compute statistics for such a set of contiguous subsequences, it is necessary to assume that a statistic is made up of certain computationally separable functions. The weaker assumption about forward computationally separability is insufficient.

Let R be any statistic such that for every contiguous subsequence $s = (x_{l+1}, x_{l+2}, \dots, x_{l+k})$ meets the condition $R(s) = f_s(F(s))$, where f_s is a function dependent on s having constant computational complexity and $F(s) = (F_1(s), \dots, F_m(s))$ is a computationally separable function. Since s is explicitly designated by l and k , then we can define f_s as $f_{l+1,l+k}$. Using these designations, we can define Algorithm 5, that computes statistic R for all contiguous subsequences of length w .

Theorem 6 If $s = (x_{l+1}, \dots, x_{l+k})$ is a contiguous subsequence of sequence $t = (x_1, \dots, x_n)$, statistic R is given by $R(s) = f_s(F(s)) = f_s(F_1(s), \dots, F_m(s))$, functions f_s, h, g have fixed computational complexity $O(1)$ and values $F_i((x_j))$ can also be computed in a constant time $O(1)$, then Algorithm 5 allows for computing statistic R for all contiguous subsequences given by $s_{l+1,l+w} = (x_{l+1}, \dots, x_{l+w})$ in time $O(n)$ for any w .

Data: Sequence $t = (x_1, \dots, x_n)$, window of width w , statistic R such that

$R(s) = f_s(F(s)) = f_s(F_1(s), \dots, F_m(s))$ for a computationally separable function F .

Result: A sequence of statistics $(r_i = R(s_{i,i+w-1}))$ set for all contiguous subsequences t of length w .

```

/* We calculate  $u_{1,1} = (F_1(s_{1,1}), \dots, F_m(s_{1,1}))$  */
1  $u_{1,1} \leftarrow (F_1(s_{1,1}), \dots, F_m(s_{1,1}))$ ;
2 for  $i = 2, 3, \dots, w$  do
    /* We calculate  $u_{1,i} = (F_1(s_{1,i}), \dots, F_m(s_{1,i}))$  */
    3  $u_{1,i} \leftarrow h(u_{1,i-1}, x_i)$ ;
4 end
/* We calculate  $u_1 = u_{1,w} = (F_1(s_{1,w}), \dots, F_m(s_{1,w}))$  */
5  $u_1 \leftarrow u_{1,w}$ ;
/* We calculate statistic  $r_1 = R(u_1) = f_{1,w}(u_1)$  */
6  $r_1 \leftarrow f_{1,w}(u_1)$ ;
7 for  $i = 2, 3, \dots, n - w + 1$  do
    /* We calculate  $v_i = (F_1(s_{i,i+w-2}), \dots, F_m(s_{i,i+w-2}))$  */
    8  $v_i \leftarrow g(u_{i-1}, x_{i-1})$ ;
    /* We calculate  $u_i = (F_1(s_{i,i+w-1}), \dots, F_m(s_{i,i+w-1}))$  */
    9  $u_i \leftarrow h(v_i, x_{i+w-1})$ ;
    /* We calculate statistics  $r_i = R(u_i) = f_{i,i+w-1}(u_i)$  */
    10  $r_i \leftarrow f_{i,i+w-1}(u_i)$ ;
11 end

```

Algorithm 5: Computation of statistics for contiguous subsequences of a given length.

Proof First, we prove that the algorithm operates correctly. For this purpose, it suffices to show that $r_i = R(s_{i,i+w-1}) = R((x_i, \dots, x_{i+w-1}))$. We prove the above by induction, which first step is to show that $r_1 = R(s_{1,w})$. Note that lines (1) and (3) of Algorithm 5 define the following recursive relation for $u_{1,i}$

$$\begin{aligned} u_{1,1} &= (F_1(s_{1,1}), \dots, F_m(s_{1,1})), \\ u_{1,i} &= h(u_{1,i-1}, x_i). \end{aligned}$$

Since $h((F_1(s_{1,i-1}), \dots, F_m(s_{1,i-1})), x_i) = ((F_1(s_{1,i}), \dots, F_m(s_{1,i})))$, then by expanding this simple recursion we conclude that

$$u_1 = u_{1,w} = (F_1(s_{1,w}), \dots, F_m(s_{1,w})).$$

Consequently, we proved that

$$r_1 = f_{1,w}(F_1(s_{1,w}), \dots, F_m(s_{1,w})) = R(s_{1,w}).$$

Now, we assume that r_{i-1} was calculated correctly. By definition of function R it follows that

$$u_{i-1} = (F_1(s_{i-1,i+w-2}), \dots, F_m(s_{i-1,i+w-2})).$$

Considering the properties of function g , we conclude that step (8) of Algorithm 5 defines

$$v_i = (F_1(s_{i,i+w-2}), \dots, F_m(s_{i,i+w-2})).$$

Then, using the properties of function h in step (9) we obtain

$$u_i = (F_1(s_{i,i+w-1}), \dots, F_m(s_{i,i+w-1})).$$

This means that

$$\begin{aligned} r_i &= f_{i,i+w-1}(u_i) \\ &= f_{i,i+w-1}(F_1(s_{i,i+w-1}), \dots, F_m(s_{i,i+w-1})) \\ &= R(s_{i,i+w-1}). \end{aligned}$$

Using inductive reasoning we proved that Algorithm 5 correctly determines statistics r_i for $i \in \{1, \dots, w\}$. In order to estimate computational complexity, note that the assumptions show that steps (1), (3), (5–6) and (8–10) of the Algorithm have constant computational complexity $O(1)$. Consequently, the total complexity of the Algorithm equals the total number of iterations of loop (2) and (7) and amounts to $O(w + (n - w)) = O(n)$, which ends the proof. $\square$

Corollary 3 If $s = (x_{l+1}, \dots, x_{l+k})$ and $R(s) = \mu(s) = \frac{1}{k} M_1(s)$, then Algorithm 5 makes it possible to compute the arithmetic mean for all contiguous subsequences $s_{l+1,l+w} = (x_{l+1}, \dots, x_{l+w})$ in time $O(n)$ for any w .

Corollary 4 If $s = (x_{l+1}, \dots, x_{l+k})$ and $R(s) = v(s) = \frac{1}{k} M_2(s) - (\frac{1}{k} M_1(s))^2$, then Algorithm 5 makes it possible to compute the variance for all contiguous subsequences $s_{l+1,l+w} = (x_{l+1}, \dots, x_{l+w})$ in time $O(n)$ for any w .

The proof of the above conclusions is very similar to the proof of Theorems 3 and 4. It must be stressed that the method presented in Corollary 4 should be used to compute the variance only when we are certain that the calculations will be free from numerical errors. Otherwise, we advise to use the method presented below.

Lemma 1 If $s = (x_1, \dots, x_m)$, and $s' = (x_2, \dots, x_{m+1})$, then the following relations are true

$$\mu(s') = \mu(s) + \frac{x_{m+1} - x_1}{m}, \quad (7)$$

$$v(s') = v(s) + \frac{x_{m+1} - x_1}{m} (x_1 + x_{m+1} - \mu(s) - \mu(s')). \quad (8)$$

Proof Equation 7 can be easily proved using the definition of arithmetic mean

$$\mu(s') = \frac{x_2 + \dots + x_m + x_{m+1}}{m}$$

$$\begin{aligned}
&= \frac{x_1 + x_2 + \cdots + x_m}{m} - \frac{x_1}{m} + \frac{x_{m+1}}{m} \\
&= \mu(s) + \frac{x_{m+1} - x_1}{m}.
\end{aligned}$$

Equation 8 is slightly more complex and so to prove it we use alternative expressions for the variance $v(s) = \frac{x_1^2 + x_2^2 + \cdots + x_m^2}{m} - \mu(s)^2$ and $v(s') = \frac{x_2^2 + \cdots + x_m^2 + x_{m+1}^2}{m} - \mu(s')^2$. Using the above relations, we can write that

$$\begin{aligned}
v(s') - v(s) &= \frac{x_2^2 + \cdots + x_m^2 + x_{m+1}^2}{m} - \mu(s')^2 \\
&\quad - \frac{x_1^2 + x_2^2 + \cdots + x_m^2}{m} + \mu(s)^2 \\
&= \frac{x_{m+1}^2 - x_1^2}{m} + \mu(s)^2 - \mu(s')^2 \\
&= \frac{x_{m+1}^2 - x_1^2}{m} \\
&\quad + (\mu(s) - \mu(s'))(\mu(s) + \mu(s')).
\end{aligned}$$

Using Eq. 7 proved above, we obtain that

$$\begin{aligned}
v(s') - v(s) &= \frac{x_{m+1}^2 - x_1^2}{m} - \frac{x_{m+1} - x_1}{m}(\mu(s) + \mu(s')) \\
&= \frac{x_{m+1} - x_1}{m} (x_1 + x_{m+1} - \mu(s) - \mu(s')).
\end{aligned}$$

This ends the proof of the lemma. $\square$

Theorem 7 If $s = (x_{l+1}, \dots, x_{l+k})$ and $R(s) = (\mu(s), v(s))$, then Algorithm 5 allows for computing the variance for all contiguous subsequences $s_{l+1, l+w} = (x_{l+1}, \dots, x_{l+w})$ in time $O(n)$ for any w .

Proof If $s_{i,j} = (x_i, \dots, x_j)$, then the Welford formulas can be used to compute the arithmetic mean and variance in loop (2)–(4) of Algorithm 5. Since in this loop we calculate $\mu(s_{1,w})$ and $v(s_{1,w})$, then the formulas are reduced to

$$\mu(s_{1,i}) = \frac{i-1}{i} \mu(s_{1,i-1}) + \frac{1}{i} x_i, \quad (9)$$

$$v(s_{1,i}) = v(s_{1,i-1}) + \frac{i-1}{i} (x_i - \mu(s_{1,i-1}))^2. \quad (10)$$

In turn, for loop (7)–(11) of Algorithm 5 we use the formulas from Lemma 1, which for a window of width w are given by

$$\mu(s_{i+1, i+w}) = \mu(s_{i, i+w-1}) + \frac{x_{i+w} - x_i}{w}, \quad (11)$$

$$v(s_{i+1,i+w}) = v(s_{i,i+w-1}) + \frac{x_{i+w} - x_i}{w} (x_i + x_{i+w} - \mu(s_{i,i+w-1}) - \mu(s_{i+1,i+w})). \quad (12)$$

The above equations prove that function $R(s) = (\mu(s), v(s))$ is determined correctly for every $s = s_{l+1,l+w} = (x_{l+1}, \dots, x_{l+w})$ using Algorithm 5, which takes the form presented as Algorithm 6. Since all operations within the two loops have the complexity $O(1)$, then by Theorem 6 we conclude that the complexity of the presented algorithm is $O(n)$, which ends the proof. $\square$

Data: Sequence $t = (x_1, \dots, x_n)$, window of width w .

Result: A sequence of statistics $(r_i = (\mu(s_{i,i+w-1}), v(s_{i,i+w-1})))$ set for all contiguous subsequences t of length w .

```

1   $(u_{1,1}, v_{1,1}) \leftarrow (x_1, 0);$ 
2  for  $i = 2, 3, \dots, w$  do
    /* Welford's formulas—Eqs. (9), (10). */
3       $u_{1,i} \leftarrow \frac{i-1}{i} u_{1,i-1} + \frac{1}{i} x_i;$ 
4       $v_{1,i} \leftarrow v_{1,i-1} + \frac{i-1}{i} (x_i - u_{1,i-1})^2;$ 
5  end
6   $(u_1, v_1) \leftarrow (u_{1,w}, v_{1,w});$ 
7   $r_1 \leftarrow (u_1, v_1);$ 
8  for  $i = 1, 2, \dots, n - w$  do
    /* Lemma 1 formulas—Eqs. (11), (12). */
9       $u_{i+1} \leftarrow u_i + \frac{x_{i+w} - x_i}{w};$ 
10      $v_{i+1} \leftarrow v_i + \frac{x_{i+w} - x_i}{w} (x_i + x_{i+w} - u_i - u_{i+1});$ 
11      $r_{i+1} \leftarrow (u_{i+1}, v_{i+1});$ 
12 end
```

Algorithm 6: Computation of arithmetic mean and variance for contiguous subsequences of a given length using Lemma 1 formula.

5 Practical implementation and its results

In order to compare the methods proposed in the previous section, in C++ we implemented Algorithm 7 calculating the minimum, maximum, arithmetic mean and variance for all contiguous subsequences of a given sequence of measurement data.

The first objective of the implementation was to determine the speed of calculation using the available data types. The following five arithmetic types were analysed:

1. 32-bit integer type `int32_t`,
2. 64-bit integer type `int64_t`,
3. 32-bit floating-point type with a 23-bit mantissa `float` (single precision),
4. 64-bit floating-point type with a 53-bit mantissa `double` (double precision),
5. 80-bit floating-point type with a 63-bit mantissa `long double` (extended precision).

Data: Sequence $t = (x_1, \dots, x_n)$, statistic R meeting the condition

$R(s) = (\min\{s\}, \max\{s\}, \mu(s), \nu(s))$ for any contiguous subsequence $s \subset t$.

Result: Sequence of statistics $(r_{i,j} = R(s_{i,j}))$ computed for all possible contiguous subsequences t .

```

1 for  $i = 1, 2, \dots, n$  do
    /*  $a_{i,j}$ —minimum of contiguous subsequence  $(x_i, \dots, x_j)$  */
    /*  $b_{i,j}$ —maximum of contiguous subsequence  $(x_i, \dots, x_j)$  */
    /*  $u_{i,j}$ —the arithmetic mean of contiguous subsequence  $(x_i, \dots, x_j)$  */
    /*  $v_{i,j}$ —the variance of contiguous subsequence  $(x_i, \dots, x_j)$  */
    /*  $c_{i,j}$ —the sum of terms of contiguous subsequence  $(x_i, \dots, x_j)$  */
    /*  $d_{i,j}$ —the sum of squares of terms of contiguous subsequence
       ( $x_i, \dots, x_j$ ) */
2    $(a_{i,i}, b_{i,i}, u_{i,i}, v_{i,i}, c_{i,i}, d_{i,i}) \leftarrow (x_i, x_i, x_i, 0, x_i, x_i^2)$ ;
3    $r_{i,i} \leftarrow (a_{i,i}, b_{i,i}, u_{i,i}, v_{i,i})$ ;
4   for  $j = i + 1, i + 2, \dots, n$  do
5        $a_{i,j} \leftarrow a_{i,j-1}$ ;
6        $b_{i,j} \leftarrow b_{i,j-1}$ ;
7        $c_{i,j} \leftarrow c_{i,j-1} + x_j$ ;
8        $d_{i,j} \leftarrow d_{i,j-1} + x_j^2$ ;
9       /* Correction of the minimum */
10      if  $x_j < a_{i,j-1}$  then
11          |  $a_{i,j} \leftarrow x_j$ ;
12      end
13      /* Correction of the maximum */
14      if  $b_{i,j-1} < x_j$  then
15          |  $b_{i,j} \leftarrow x_j$ ;
16      end
17      /* Determination of the arithmetic mean */
18       $u_{i,j} \leftarrow \frac{1}{j-i+1} c_{i,j}$ ;
19      /* Determination of the variance */
20       $v_{i,j} \leftarrow \frac{1}{j-i+1} d_{i,j} - u_{i,j}^2$ ;
21       $r_{i,j} \leftarrow (a_{i,j}, b_{i,j}, u_{i,j}, v_{i,j})$ ;
22   end
23 end

```

Algorithm 7: Computing the minimum, maximum, arithmetic mean and variance of all contiguous subsequences.

Table 2 Time needed to determine the basic statistics for a data vector consisting of 10^6 elements

Operation	Duration of an operation in [ms] for a vector consisting of 10^6 elements				
	int32_t	int64_t	float	double	long double
Min/Max	0.806	0.814	0.817	0.865	2.169
Arithmetic mean	0.416	0.476	0.803	0.805	0.954
Variance	1.078	1.155	1.610	1.618	1.980
Total	1.890	1.969	2.404	2.445	4.115

Measurements were made for individual data types

Table 3 Comparison of running times of Algorithm 7 and the algorithm determining statistics for every sequence separately (std.) for two arithmetic types: 64-bit integer type and 80-bit floating-point type with a 63-bit mantissa

Vector size	Number of contiguous subsequences	Time needed to determine statistics in [ms]			
		<code>int64_t</code>		<code>long double</code>	
		Algorithm 7	std.	Algorithm 7	std.
100	5050	0.54	0.84	0.67	1.11
200	20100	2.27	4.20	3.04	5.88
400	80200	11.49	25.40	14.08	35.00
800	320400	56.44	156.10	67.73	206.20
1600	1280800	285.00	1031.00	301.00	1407.00
3200	5121600	1344.00	7360.00	1547.00	10610.00
6400	20483200	6016.00	54706.00	6844.00	79926.00

Bearing in mind the application of Algorithm 7 to the analysis of large sets of measurement data, we focused on `int64_t` and `long double` arithmetic types. These are the only types allowing us to precisely determine the variance for samples consisting of more than 10^6 elements and represented with a 16-bit precision. However, in order to compare all options, each of the previously mentioned data types has been subjected to a performance analysis. Table 2 presents the time needed to perform the following operations: determination of extremes, determination of the arithmetic mean, determination of the variance, and determination of all these statistics simultaneously.

An analysis of running times showed that `int64_t` is more than twice as fast as `long double`. Due to the nature of our calculations (determination of extremes, expected value and variance), both types guarantee virtually identical precision. In addition, the integer type allows us to avoid rounding errors when the precision of measurements is based on the powers of 10. Table 3 presents running times of Algorithm 7 and of the algorithm determining statistics for each contiguous subsequence separately (denoted as std.). Figure 4 gives a more accurate picture of the speed of the algorithms for particular arithmetic types. It clearly shows the difference in the complexity of both algorithms. The time needed to determine the basic statistics grows quickly with the size of the analysed data. For a sequence of around 10^6 elements, the running time of Algorithm 7 is approximately 30 hours. For comparison, it would take almost 4 years to perform this analysis using a non-optimised algorithm. The basic condition for the reliability of Algorithm 7 is the selection of an arithmetic type that allows for error-free calculation of sums and sums of squares for a large number of elements. That is why we suggest using integer types instead of floating-point types.

Figure 5 compares the running times of the proposed algorithm for two types of data: `long double` and `int64_t`. Note that in terms of speed, the advantage of the integer type over the floating-point type has been significantly reduced. There are three reasons behind this:

Fig. 4 Time needed to compute extremes, expected value and variance for all possible contiguous subsequences

Fig. 5 Time needed to compute extremes, expected value and variance for all possible contiguous subsequences (only 64-bit integer type and 80-bit extended type with a 63-bit mantissa)

1. a large number of short contiguous subsequences for which the statistics are calculated,
2. the need to store a large number of calculated statistics, which translates into a large number of read and write memory operations,
3. a larger number of conditional instructions in the loops determining individual statistics (the processor spends more time on analysing conditions than on making calculations).

Fig. 6 Calculations accelerated through the use of Algorithm 7 (only 64-bit integer type and 80-bit floating-point type with a 63-bit mantissa)

Naturally a question arises how the performance time will change for sequences consisting of 10^5 or 10^6 elements. Another important issue is how to optimise access to memory or structure the algorithm so that it does not remember statistics, but only locations that should be analysed more closely. Figure 6 presents the ratio of the running time of the algorithm determining the statistics for each sequence separately to the running time of Algorithm 7 for two arithmetic types: `long double` and `int64_t`. The plot shows that the integer type guarantees a much higher calculation efficiency than the floating point type, and it can also be seen that this disproportion increases as the size of the input sequence being processed increases.

6 Conclusions

The article proposes a new algorithm for determining statistics such as the arithmetic mean and variance for all contiguous subsequences and fixed-length contiguous subsequences. Its main element is to reduce floating point data to their integer representation. As a result, an algorithm was obtained that works without a numerical error for many practical applications—especially for the analysis of industrial measurement data. The developed method significantly speeds up the calculations and gives the possibility of its effective application in machine learning and accurate statistical inference.

The first section of this article presents a brief introduction to the issues of statistical analysis of measurements in manufacturing enterprises and a description of the current state of knowledge on the determination of the arithmetic mean and variance.

The second section outlines calculation methods based on measurement data with the use of both floating-point representation and integer representation. In Theorem 1 we laid down the conditions that an arithmetic type (its mantissa) has to meet so

that it can be used to precisely determine statistics such as the expected value and variance. Additionally, this section presents a method of converting measurement data to a corresponding sequence of integers that can be used to determine the said statistics, which is expressed in Corollary 1.

The third section presents an algorithm for fast statistical analysis of all contiguous subsequences of a certain numerical sequence. Algorithm 1 has been presented in the general version using the proposed concept of *computationally separable functions*. Theorem 2 proves the correctness of the proposed method and determines its computational complexity at $O(n^2)$, where n is the number of elements of the analysed sequence. Theorems 3 and 4 prove the usefulness of Algorithm 1 for calculating the arithmetic mean and variance. It is worth stressing that these methods have been presented in a form that allows error-free calculations, provided that the assumptions of Theorem 1 are met. For numerically unstable cases, we presented Theorem 5, which uses Algorithm 1 to determine arithmetic means and variances based on the Welford formula. This theorem applies when the mantissa of a number is too small to enable the calculations to be free from numerical errors.

The fourth section deals with the problem of determining statistics for a set of fixed-length contiguous subsequences. In this case, we proposed Algorithm 5, which was proved to operate correctly in Theorem 6. The theorem also proved that if the statistics can be defined using computationally separable functions, then the determination of statistics for all w -element contiguous subsequences of an n -element sequence has complexity $O(n)$ regardless of the value of w . Next, Corollaries 3 and 4 show how to apply Algorithm 5 to determine arithmetic means and variances, avoiding numerical errors in the process. Lemma 1 proved the correctness of the new formula for calculating the arithmetic mean and variance for fixed-length contiguous subsequences. The lemma was used to prove Theorem 7, that proved that Algorithm 5 operates correctly based on the proposed relation.

The last section of the article compares the running times of the proposed algorithms for various data types. The conclusion is that by converting measurement data from floating-point to integer representation, we can not only speed up calculations, but also make them more precise. Although it is possible to achieve this goal only when the conditions of Theorem 1, are met, practice shows that it is very often feasible, especially in the case of a well-defined set of measurement data.

Acknowledgements The study was carried out with the use of apparatus purchased from funds of the project *Establishment of the Scientific and Research Inter-University Laboratory in Stalowa Wola*, realized as part of the Operational Programme Eastern Poland 2007-2013, Priority axis I *Modern Economy*, Measure 1.3 *Supporting Innovativeness* as per contract No. POPW.01.03.00-18-016/12-00.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Box G, Hunter J (1959) Condensed calculations for evolutionary operation programs. *Technometrics* 1(1):77–95
- Chalapathy R, Chawla S (2019) Deep learning for anomaly detection: a survey. *CoRR*. abs/1901.03407
- Chan T, Golub G, LeVeque R (1982) Updating formulae and a pairwise algorithm for computing sample variances. *COMPSTAT*. https://doi.org/10.1007/978-3-642-51461-6_3
- Chan T, Golub G, LeVeque R (1983) Algorithms for computing the sample variance: analysis and recommendations. *Am Stat* 37(3):242–247. <https://doi.org/10.2307/2683386>
- Chan T, Lewis J (1979) Computing standard deviations: accuracy. *Commun ACM* 22(9):526–531. <https://doi.org/10.1145/359146.359152>
- Cotton I (1975) Remark on stably updating mean and standard deviation of data. *Commun ACM* 18(8):458. <https://doi.org/10.1145/360933.360981>
- Deming W (1986) *Out of the crisis*. MIT Press, Cambridge
- Ding J, Liu Y, Zhang L, Wang J, Liu Y (2016) An anomaly detection approach for multiple monitoring data series based on latent correlation probabilistic model. *Appl Intell* 44:340–361. <https://doi.org/10.1007/s10489-015-0713-7>
- Evans K, Love T, Thurston S (2015) Outlier identification in model-based cluster analysis. *J Classif* 32:63–84. <https://doi.org/10.1007/s00357-015-9171-5>
- Gouin F, Ancourt C, Guettier C (2016) Threewise: a local variance algorithm for gpu. In: 2016 IEEE International Conference on CSE, IEEE International Conference on EUC, International Symposium DCABES, pp 257–262. <https://doi.org/10.1109/CSE-EUC-DCABES.2016.194>
- Hanson R (1975) Stably updating mean and standard deviation of data. *Commun ACM* 18(1):57–58. <https://doi.org/10.1145/360569.360662>
- Hawkins D (1980) *Identification of outliers*. Springer, Dordrecht
- Hyndman R, Wang E, Laptev N (2015) Large-scale unusual time series detection. In: 2015 IEEE international conference on data mining workshop (ICDMW), pp 1616–1619. <https://doi.org/10.1109/ICDMW.2015.104>
- IEEE: Ieee std 754-2008. Tech. rep., IEEE (2008). <https://doi.org/10.1109/ieeestd.2008.4610935>
- Jensen J (1906) Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Math* 30:175–193. <https://doi.org/10.1007/BF02418571>
- Kahan W (1965) Pracniques: further remarks on reducing truncation errors. *Commun ACM* 8(1):40. <https://doi.org/10.1145/363707.363723>
- Keogh E, Lin J (2005) Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowl Inf Syst* 8(2):154–177. <https://doi.org/10.1007/s10115-004-0172-7>
- Keogh E, Lin J, Fu A (2005) Hot sax: efficiently finding the most unusual time series subsequence. In: Fifth IEEE international conference on data mining (ICDM'05), p 8. <https://doi.org/10.1109/ICDM.2005.79>
- Keogh E, Lin J, Lee S, Herle H (2006) Finding the most unusual time series subsequence: algorithms and applications. *Knowl Inf Syst* 11(1):1–27. <https://doi.org/10.1007/s10115-006-0034-6>
- Knuth D (1981) *The art of computer programming*, vol II: seminumerical algorithms, 2nd edn. Addison-Wesley
- Kreinovich V, Nguyen H, Wu B (2007) On-line algorithms for computing mean and variance of interval data, and their use in intelligent systems. *Inf Sci* 177:3228–3238. <https://doi.org/10.1016/j.ins.2006.11.007>
- Laptev N, Amizadeh S, Flint I (2015) Generic and scalable framework for automated time-series anomaly detection. In: *KDD'15: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp 1939–1947. <https://doi.org/10.1145/2783258.2788611>
- Ling R (1974) Comparison of several algorithms for computing sample means and variances. *J Am Stat Assoc* 69(348):859–866. <https://doi.org/10.2307/2286154>
- Neely P (1966) Comparison of several algorithms for computation of means, standard deviations and correlation coefficients. *Commun ACM* 9(7):496–499. <https://doi.org/10.1145/365719.365958>
- Pébay P (2008) Formulas for robust, one-pass parallel computation of covariances and arbitrary-order statistical moments. Tech. Rep. SAND2008-6212, Sandia National Laboratories. <https://doi.org/10.2172/1028931>

- Pébay P, Terriberry T, Kolla H, Bennett J (2016) Numerically stable, scalable formulas for parallel and online computation of higher-order multivariate central moments with arbitrary weights. *Comput Stat* 31(4):1305–1325. <https://doi.org/10.1007/s00180-015-0637-z>
- Rodden B (1967) Error-free methods for statistical computations. *Commun ACM* 10(3):179–180. <https://doi.org/10.1145/363162.363205>
- Schubert E, Gertz M (2018) Numerically stable parallel computation of (co-)variance. In: *SSDBM'18*. <https://doi.org/10.1145/3221269.3223036>
- Schubert E, Koos A, Emrich T, Züfle A, Schmid K, Zimek A (2015) A framework for clustering uncertain data. *Proc VLDB Endowment* 8(12):1976–1979. <https://doi.org/10.14778/2824032.2824115>
- Selvamuthu D, Das D (2018) Introduction to statistical methods, design of experiments and statistical quality control. Springer Nature Singapore Pte Ltd., Singapore
- Senin P, Lin J, Wang X, Oates T, Gandhi S, Boedihardjo A, Chen C, Frankenstein S (2015) Time series anomaly discovery with grammar-based compression. In: *Proceedings of the 18th international conference on extending database technology, EDBT 2015*, pp 481–492. <https://doi.org/10.5441/002/edbt.2015.42>
- Shewart W (1931) *Economic control of quality manufactured product*. D. Van Nostrand, New York
- Terriberry T (2008) Computing higher-order moments online. <http://people.xiph.org/tterribe/notes/homs.html>
- Van Reeken A (1968) Letters to the editor: dealing with Neely's algorithms. *Commun ACM* 11(3):149–150. <https://doi.org/10.1145/362929.362961>
- Villaverde K, Xiang G (2008) Estimating variance under interval and fuzzy uncertainty: Parallel algorithms. In: *IEEE International Conference on Fuzzy Systems*, pp 1030–1033. <https://doi.org/10.1109/FUZZY.2008.4630496>
- Wang X, Lin J, Patel NMB (2018) Exact variable-length anomaly detection algorithm for univariate and multivariate time series. *Data Mind Kowl Discov* 32:1806–1844. <https://doi.org/10.1007/s10618-018-0569-7>
- Welford B (1962) Note on a method for calculating corrected sums of squares and products. *Technometrics* 4(3):419–420. <https://doi.org/10.2307/1266577>
- West D (1979) Updating mean and variance estimates: an improved method. *Commun ACM* 22(9):532–535. <https://doi.org/10.1145/359146.359153>
- Youngs E, Cramer E (1971) Some results relevant to choice of sum and sum-of-product algorithms. *Technometrics* 13(3):657–665. <https://doi.org/10.1080/00401706.1971.10488826>
- Zhang Y, Meratnia NPH (2010) Outlier detection techniques for wireless sensor networks: a survey. *IEEE Commun Surv Tutor* 12(2):159–170. <https://doi.org/10.1109/SURV.2010.021510.00088>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.