Chapter

Effective Algorithms for Detection Outliers and Cycle Slip Repair in GNSS Data Measurements

Igor V. Bezmenov

Abstract

The chapter describes effective algorithms that are often used in processing data measurements in Global Navigation Satellite Systems (GNSSs). Existing effective algorithm was developed for detection and elimination of outliers from GNSS data measurements. It is based on searching for a so-called optimal solution for which standard deviation and maximum absolute deviation of the measured data from mean values do not exceed specified threshold values, and the number of the detected outliers is minimal. A modification of this algorithm with complexity of $N \log_2 N$ is discussed. Generalization of the existing algorithm to the case when data series included some unknown trend will be presented. The processing trend is assumed to be described by an unknown function of time. The generalized algorithm includes the outlier detection algorithm and trend searching algorithm that has been tested using simulated data. A new algorithm will be presented for cycle slip repair using Melbourne-Wübbena linear combination formed from GNSS data measurements on two carrier frequencies. Test results for repair data in the case of multiple (cascade) cycle slips in actual observation data will also be presented in this chapter.

Keywords: global navigational satellite systems (GNSSs), GNSS measurements, outliers, data screening, optimal solution, trend function, Melbourne-Wübbena combination, cycle slips

1. Introduction

At present, five constellations of GNSS satellites are involved in the formation of observational data, which serve as a source for many applications related to navigation, geodesy, geodynamics, and in the performance of solving of many fundamental problems. These are American Global Positioning System (GPS), Russian Global Navigation Satellite System (GLONASS), European Galileo, Chinese BeiDou, and Japanese Quasi-Zenith Satellite System (QZSS). The satellites of each of the operating systems transmit signals, as a rule, on two L-band carriers, which are received by GNSS receivers. A large number of stations equipped with GNSS receivers and located around the World are part of the International GNSS Service (IGS) network. These stations generate observation data files and transmit them to international databases in real time [1, 2], after which these data become available for use by many institutions and laboratories over the World. When solving

applications, the measurement data go through various processing steps. Significant element of the data processing is the detection of rough measurements and removal them from the further processing. Despite the fact that many of the laboratories use a high-end application of the software regarding accuracy, reliability, and robustness, the presence of rough measurements in the observational data excludes the possibility of obtaining an accurate final result. In order to obtain results of unprecedented accuracy, the measurement data must be cleared of coarse measurements or outliers. It should be noted that the concept of outliers is key in the measurement processing theory [3], and there is no general definition for it. In order to distinguish outliers from the rest of the measured data, in some cases, the deviation of the data series values from some average value of the data is considered. If the deviation from the average is exceeded by a predetermined threshold value, the measured value is considered as an outlier. Such an approach has a significant disadvantage that the exact mean is generally unknown, and the estimate obtained by averaging a series may be very inaccurate due to outliers. Existing iterative procedures are also based on the idea of calculating deviation from the average and often result in the unjustified rejection of many observations. Reducing the data involved in processing may, in turn, result in a loss of accuracy of the final result.

This chapter describes the outliers cleaning algorithm for GNSS data. The proposed algorithms are based on the search for the so-called optimal solution with the minimum amount of invalidly rejected data. The algorithm for accelerated detection of outliers in a large amount of measurements has been developed, as well as an algorithm for detecting outliers in data containing an unknown trend. In conclusion, the algorithm of jump detection in the Melbourne-Wübbena combination [3–5], including the developed procedure of cleaning data from outliers, is considered.

In Section 2, the problem of searching for the so-called optimal solution is formulated. Section 3 provides a search algorithm, with a common number of arithmetic operations not exceeding $\sim N^2$. Section 4 presents the test results for actual measurements in global navigational satellite systems at two carrier frequencies. The searching of outliers was performed in the Melbourne-Wübbena combination. In Section 5, the assertions that are the mathematical prerequisites for justifying a fast outlier search algorithm are proved. In Section 6, the fast outlier detection algorithm with the number of arithmetic operations of $N\log_2 N$ is proposed. Section 7 describes the case of data with unknown trend. Iterative procedure of outlier search is proposed based on the finding of suitable trend approximation in polynomials class. The idea of excluding coarse measurements is based on finding a so-called minimizing set of measurement data of a given length. This distinguishes the proposed algorithm from known similar procedures in which outliers are detected by exceeding a preset threshold. The test results with simulated data are given. Sections 8 and 9 discuss the problem of detecting jumps in the Melbourne-Wübbena combination. An algorithm is proposed that includes the outlier cleaning procedure based on the search for the optimal solution. Section 10 shows the numerical calculations with real data for algorithms presented in Sections 3–9. Section 11 concludes the chapter.

2. Statement of the problem: a Melbourne-Wübbena combination

Often, measurements y_j taken at moments of time j can be presented in the form:

$$y_j = f_j + z + \xi_j; j = 1 \dots N, \tag{1}$$

where f_j is a trend function, depending on physical process and as a rule is unknown, the sum $z + \xi_j$ is unknown random variable imposed on the trend with unknown constant z and a centered random variable ξ_j.

Detection of outliers in data series expressed in Eq. (1) with unknown trend is uncertain since the concept of measurement or outliers itself is uncertain. In many cases, however, the trend function is known a priori.

For example, many data processing programs often use different linear combinations formed of code and phase measurement data to eliminate unknown parameters. One such combination is the Melbourne-Wübbena combination composed of both, carrier phase and code observables as described by Melbourne [4] and Wübbena [5]. This combination eliminates the effect of the ionosphere, the geometry, the clocks, and the troposphere [3], and it is often used to detect loss of carrier phase capture in the preprocessing stages. The Melbourne-Wübbena combination generated for a specific satellite-receiver pair can be presented in the form of the sum of three terms [6]. One of the terms includes the integer wide-lane ambiguity for the two carrier frequencies [3]; the second component accounts for the satellite and receiver instrumental delays; and the third component is the measurement noise, including carrier phase and code multipath. Thus, during a time interval where the integer wide-lane ambiguity does not change, the Melbourne-Wübbena combination can be written as formula (1) with $f_j = \text{const}$.

Another example is satellite clock correction values derived from navigation message data, which can also be represented as in Eq. (1) with $f_j = dj + a$, where d and a are the drift and offset parameters known from navigation data, respectively. In the case where the trend is known, the measurement data after the trend subtraction can be represented as, assuming N observations:

$$y_j = z + \xi_j; j = 1 \ldots N, \tag{2}$$

with an unknown constant z, which we cannot be determined in advance, because the random value ξ_j is not known and may contain outliers.

In Sections 2–6, we consider the case where the trend is known a priori, that is, the data can be presented as Eq. (2). A problem with an unknown trend will be discussed in Section 7. In principle, the outlier detection procedure described below is not affected by the measurement format expressed in Eq. (1) or (2); it can be applied to any set of data measurements y_j.

The preliminary processing task includes rejection of rough measurements or outliers from data series (2). In other words, it is necessary to find a set $Y_L = \left\{ y_{j_1}, \ldots, y_{j_L} \right\}$ of L elements, where L is the length for which the following conditions are satisfied:

$$\sigma_{Y_L} = \sqrt{(L-1)^{-1} \sum_{j \in \{j_1, \ldots, j_L\}} \left(y_j - z \right)^2} \leq \sigma_{max}, \tag{3}$$

$$\left| y_j - z \right| \leq 3 \cdot \sigma_{max}; y_j \in \left\{ y_{j_1}, \ldots, y_{j_L} \right\}, \tag{4}$$

$$L \geq \text{MINOBS}, \tag{5}$$

where σ_{Y_L} and σ_{max} are the standard deviation (SD) and their associated specified threshold values; MINOBS is a parameter limiting from below the length of the required set of measured values (e.g., 10), and we will assume hereafter that MINOBS < N.

The values y_j that are not included in the set Y_L are treated as coarse measurements and removed from further processing. Typically, expressions in Eqs. (3)–(5) are the only conditions considered in processing programs when screening out rough measurements. Below we will formulate the problem of searching for a solution, complementing the conditions expressed in Eqs. (3)–(5) with two extreme conditions [7].

1. First, we will require that the length of the set sought be the maximum, that is, the number of measurements deemed to be coarse is the minimum:

$$L \to \max . \tag{6}$$

Note that for the predetermined values y_j, the problem solution satisfying conditions in Eqs. (3)–(5) may not exist (e.g., when y_j includes a trend, in particular when y_j is an arithmetic progression with a step greater than σ_{max}). In the case when the solution does exist, we will denote the value L at which the maximum of Eq. (6) is reached as L_{max}. Note that the condition expressed in Eq. (6) does not ensure the uniqueness of the set because several sets of length L_{max} can be found that satisfy the conditions in Eqs. (3)–(5).

2. From all possible sets that satisfy conditions expressed in Eqs. (3)–(5) and (6), we will select the one for which the variable σ_{Y_L} receives the smallest value:

$$\sigma_{Y_L} \xrightarrow[Y_L=\left\{y_{j_1}, \ldots, y_{j_L}\right\}; L=L_{max}]{} \min . \tag{7}$$

Let us define Y_{opt} as follow:

Definition 1. *For a given sequence of values y_j, $j = 1, 2, \ldots N$, the set of values:*

$$Y_{opt} = \left\{y_{j_1}, \ldots, y_{j_{L_{max}}}\right\}, \tag{8}$$

satisfying conditions in Eqs. (3)–(7), we refer to as the optimal solution of the problem expressed in Eqs. (3)–(7). The corresponding SD value is denoted by σ_{opt}.

Thus, the problem consists in the creation of a search algorithm for the optimal solution of the problem shown in Eqs. (3)–(7).

In a practical situation, the precise value z, given conditions in Eqs. (3) and (4), is not known. We will estimate the values using the following formula:

$$z = L^{-1} \sum_{j \in \left\{j_1, \ldots, j_L\right\}} y_j . \tag{9}$$

Note that the value z depends on the required solution, which will complicate its search.

Usually, iterative methods are used to find a solution to problem expressed in Eqs. (3)–(5). For example, the algorithm implemented in the observation data smoothing program (see [3]) is designed to find a set Y satisfying the conditions given by Eqs. (3)–(5). The proposed step-by-step algorithm is based on iterations (the index number of iteration is designated by the upper index in parentheses):

Step 1: Initialization: $Y^{(0)} = \{y_1, \ldots, y_N\}$; $Level^{(0)} = 10^{20}$; k = 0.

Step 2: Checking the length of the set $Y^{(k)}$; if it is less than MINOBS, then the process comes to an end and a solution is not found.

Step 3: Calculation of the values $z^{(k)}$ and $\sigma^{(k)}$ on the available set $Y^{(k)}$ using formulas (3) and (9).

Step 4: Checking the fulfillment of the inequality $\sigma^{(k)} \le \sigma_{\max}$. If it is satisfied, the set $Y^{(k)}$ is accepted as the solution, and the search process comes to an end. Otherwise, there is a transition to Step 5.

Step 5: Definition of $Level^{(k+1)}$ for outlier detection:

$$Level^{(k+1)} = 3 \cdot \sigma^{(k)};$$

In order to prevent an infinite loop of iterations, a required verification is carried out:

$$Level^{(k+1)} < Level^{(k)}.$$

If this inequality is not satisfied, then the following is assumed:

$$Level^{(k+1)} = Level^{(k)}/2;$$

Step 6: Definition of a new set $Y^{(k+1)}$ to include those and only those y_j for which

$$\left| y_j - z^{(k)} \right| \le Level^{(k+1)}.$$

Step 7: Increasing k by 1: k++. Transition to Step 2.

Note that the optimal solution cannot be found in such a manner, as confirmed by numerical calculations (see Section 4).

3. Algorithm for solving the problem

Let us formulate a statement that is the key to the creation of an effective search algorithm for the optimal solution (Eqs. (3)–(7)).

Assertion 1. *Let the set* $Y_{opt} = \left\{ y_{j_1}, \ldots, y_{j_{L_{\max}}} \right\}$ *be optimal for a specified sequence of values* $\left\{ y_j \right\}$ *and*

$$y_{\min} = \min \left\{ y_{j_1}, \ldots, y_{j_{L_{\max}}} \right\}, y_{\max} = \max \left\{ y_{j_1}, \ldots, y_{j_{L_{\max}}} \right\},$$

then the interval $(y_{\min}, y_{\max})$ *does not contain values* y_j *that are not in the set* Y_{opt}.

Proof. In fact, let us assume the opposite: Let $y_j \notin Y_{opt}, y_{\min} < y_j < y_{\max}$ and y_k and y_l be values from the set Y_{opt} for which $y_k = y_{\min}$ and $y_l = y_{\max}$. One of these cases is possible:

a. $z < y_j$ and, therefore $0 < \left(y_j - z\right) < \left(y_l - z\right) \Rightarrow \left(y_j - z\right)^2 < \left(y_l - z\right)^2$,

b. $z \ge y_j$ and, therefore $0 \le \left(z - y_j\right) < \left(z - y_k\right) \Rightarrow \left(y_j - z\right)^2 < \left(y_k - z\right)^2$.

In the first case, Case (a), we replace the value y_l in the set Y_{opt} with y_j. In the second case, Case (b), we replace y_k with y_j. In any of the cases, we will have another set of the same length L_{max} for which conditions expressed in Eqs. (3) and (4) are satisfied, but the SD, as follows from Eq. (3), is less than σ_{opt}. Consequently, the set Y_{opt} is not optimal since requirement expressed in Eq. (7) is not satisfied. The contradiction that is reached proves Assertion 1.

Further, note that if $Y_{opt} = \left\{ y_{j_1}, \ldots, y_{j_{L_{max}}} \right\}$ is the optimal solution of the problem given by Eqs. (3)–(7), then an arbitrary permutation of the numbers $y_{j_1}, \ldots, y_{j_{L_{max}}}$ will also be the optimal solution of the problem described by Eqs. (3)–(7). Thus, the optimal solution does not depend on the arrangement of given numbers y_j. This allows us to arrange the numbers y_j in the order most suitable for searching the optimal solution. Taking advantage of this important circumstance, we arrange the values $\left\{ y_j \right\}$ in the ascending order and we will look for the optimal solution in the ordered sequence. For brevity, the ordered sequence will also be denoted by $\left\{ y_j \right\}$. Thus, $y_1 \le y_2 \le \ldots \le y_N$. Moreover, for simplicity of logic, we will assume that all y_j are different, that is,

$$y_1 < y_2 < \ldots < y_N \tag{10}$$

Note that due to the formulated Assertion 1, if Y_{opt} is the optimal set and y_{j_1} and $y_{j_{L_{max}}}$ are its smallest and greatest values, respectively, then all values of y_j from the interval $\left(y_{j_1}, y_{j_{L_{max}}} \right)$ belong to Y_{opt}. Consequently, considering Eq. (10), we have $y_{j_{L_{max}}} = y_{j_1 + L_{max} - 1}$ and

$$Y_{opt} = \left\{ y_{j_1}, y_{j_1+1}, \ldots, y_{j_1+L_{max}-1} \right\}$$

Thus, the optimal solution should be sought in the ascending sequence y_j among all possible sets $\{y_k, \ldots, y_{k+L-1}\}$ of length L with k and L satisfying the following conditions:

$$\text{MINOBS} \le L \le N, \tag{11}$$

$$1 \le k \le N - L + 1. \tag{12}$$

Hence, instead of searching for all possible sets of various length, numbering 2^N, for the solution of the problem described by Eqs. (3)–(7), it is sufficient to vary just the two parameters, k and L, associated with the conditions expressed in Eqs. (11) and (12). The number of pairs of integer numbers k and L subject to condition in Eqs. (11) and (12) is equal to:

$$(N - \text{MINOBS})(N - \text{MINOBS} + 1)/2.$$

Let $L = l - k + 1$ be the length of an arbitrary set $\left\{ y_k, \ldots, y_l \right\}$. We introduce the designations:

$$z(k; L) = \frac{1}{L} \sum_{j=k}^{k+L-1} y_j, \tag{13}$$

$$\sigma^2(k;L) = \frac{1}{L-1}\sum_{j=k}^{k+L-1}\left(y_j - z(k;L)\right)^2. \tag{14}$$

We rewrite conditions given by Eqs. (3) and (4) in the new designations:

$$\sigma^2(k;L) \le \sigma^2_{max}, \tag{15}$$

$$\begin{cases} y_{k+L-1} - z(k;L) \le 3\cdot\sigma_{max} \\ z(k;L) - y_k \le 3\cdot\sigma_{max} \end{cases}. \tag{16}$$

Note that the two last inequalities directly follow from Eq. (4), monotony of y_j, and obvious inequalities: $y_k \le z(k;L) \le y_{k+L-1}$.

Remark. *In the conditions expressed in Eqs. (15) and (16), L means the length of the set under checking, and k is the index of the smallest number included in the set. Although "k" and "L" are also encountered as indexes in the sets we use below for monotonically increasing sequences, we hope nevertheless that this will not lead to confusion.*

The following recursive relationships are available, making it possible to find $z(k;L)$ and $\sigma^2(k;L)$ through the calculated values $z(k;L+1)$ и $\sigma^2(k;L+1)$ for seven arithmetic operations:

$$z(k;L) = z(k;L+1) + A_{L+1}\cdot\left(z(k;L+1) - y_{k+L}\right), \tag{17}$$

$$\sigma^2(k;L) = B_{L+1}\cdot\sigma^2(k;L+1) - C_{L+1}\cdot\left(y_{k+L} - z(k;L+1)\right)^2, \tag{18}$$

$$A_L = \frac{1}{L-1}; B_L = \frac{L-1}{L-2}; C_L = \frac{L}{(L-1)(L-2)}. \tag{19}$$

The values of the fractions may be computed in advance as elements of a one-dimensional array. Analogously, the following formulas can make it possible to express $z(k+1;L)$ and $\sigma^2(k+1;L)$ through $z(k;L)$ and $\sigma^2(k;L)$:

$$z(k+1;L) = z(k;L) + A_{L+1}\cdot\left(y_{k+L} - y_k\right), \tag{20}$$

$$\sigma^2(k+1;L) = \sigma^2(k;L) + A_{L+1}\cdot\left(y_{k+L} - y_k\right)\left(y_{k+L} - z(k;L) + D_{L+1}\left(y_k - z(k;L)\right)\right), \tag{21}$$

$$D_L = \frac{L}{L-2}. \tag{22}$$

The algorithm described below is based on the search for all possible pairs (k, L), where L denotes the length of the set to be checked and k is the index of the smallest of the values included in the set. At that, k and L must satisfy conditions Eqs. (11) and (12). The set $\{y_k, \ldots, y_{k+L-1}\}$ corresponding to each such pair must be checked for fulfillment conditions (15) and (16).

We organize this search according to the algorithm described below, at each step of which we check the validation of Eqs. (15) and (16) for all possible sets of a certain length. We start the examine process with Step 1, where we check the set of maximum length N. Further, with each next step, we will reduce the length of the sets to be checked by 1.

Step 1: We consider the set of length N. There is only one such set: $\{y_1, \ldots, y_N\}$. We check it for fulfillment conditions expressed in Eqs. (15) and (16). If they are satisfied, this set is selected as a solution, and further search stops. Otherwise, transit to Step 2.

Step 2: We consider the sets of length $N-1$. There are two sets of length $N-1$.

$$\{y_1, \dots, y_{N-1}\} \text{ and } \{y_2, \dots, y_N\}.$$

We test each of these sets for compliance with the conditions specified by Eqs. (15) and (16). If none of them satisfies conditions (15) and (16), then we transit to the next step. Otherwise, the following options are available:

- Option 1: if only one set from them is found that satisfies conditions (15) and (16), then it will also be the solution of the stated problem; the search process stops here, where $L_{max} = N - 1$.
- Option 2: if both sets simultaneously satisfy conditions (15) and (16), we will select the set corresponding to the smallest of two values $\sigma^2(1, N-1)$ or $\sigma^2(2, N-1)$, and the search process stops here, where $L_{max} = N - 1$.

Step $N - L + 1$: We consider the sets of length L. If $L <$ MINOBS, then the search process stops, and a solution is not found. For $L \geq$ MINOBS, we examine $N - L + 1$ sets of length L:

$$\{y_1, \dots, y_L\}, \{y_2, \dots, y_{L+1}\}, \dots, \{y_{N-L+1}, \dots, y_N\}. \quad (23)$$

We check each of these sets for fulfillment of conditions (15) and (16). If any of them does not satisfy these conditions, then we transit to the next step where we consider the sets of length $(L - 1)$. Otherwise, two options are possible:

- Option 1: if only one set from (23) is found that satisfies the conditions of (15) and (16), then it will also be the solution of the stated problem with $L_{max} = L$, and the search process stops here.
- Option 2: if several sets simultaneously satisfy conditions (15) and (16), we chose the set for which the value $\sigma^2(k, L)$ appears smallest as the solution, and the search process stops here, where $L_{max} = L$.

In order to calculate the values $z(k, L)$ and $\sigma^2(k, L)$, we use the recursive formulas (17)–(22) in accordance with a search scheme shown in **Figure 1** where only the $z(k, L)$ is involved.

In accordance with the proposed arrangement, we calculate the values $z(1; N)$ and $\sigma^2(1; N)$ in the first step of the algorithm using formulas expressed in Eqs. (13) and (14), and 4 *N* arithmetic operations are required for this arrangement. In order to find $z(k, L)$ and $\sigma^2(k, L)$ on all subsequent steps, we apply the recursive formulas (17)–(22), making it possible to calculate the values of the specified pairs of

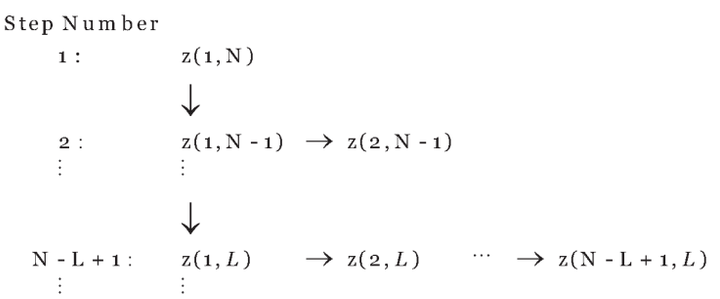

Figure 1.
Scheme of calculations when finding the optimal solution.

variables, each for the seven arithmetic operations, based on the results of the calculations of the preceding step. So, for example, on the second step, proceeding from the known values $z(1;N)$ and $\sigma^2(1;N)$, we find the values $z(1;N-1)$ and $\sigma^2(1;N-1)$ (vertical arrow on the diagram) by using formulas in Eqs. (17)–(19) and the values $\sigma^2(2;N)$ and $\sigma^2(2;N)$ (horizontal arrow on the diagram) by using Eqs. (20)–(22). In the general case, the transition to the following step in the direction of the vertical arrows (see diagram) is carried out according to formulas (17)–(19), and in the direction of horizontal arrows, according to formulas (20)–(22). The number of arithmetic operations required to find the solution should not exceed:

$$4N + 9((N - L_{max} + 2)(N - L_{max} + 1)/2 - 1) \qquad (24)$$

In the above number of computations, the computational costs of verifying the satisfaction of inequalities (15) and (16) are also considered, which comprise from 0 to 2 arithmetic operations.

4. Results of calculations using algorithm presented in Section 3

We test the algorithms discussed above on the real data obtained by the ONSA station that is a part of the IGS network [2]. These data are included in the distribution kit of the installation software package [3] and available for usage. We consider measurement data received from global positioning system (GPS) satellite with system number PRN = 12 for 2010, day 207 to check the efficiency of the proposed algorithm described in Section 3. **Figure 2** plots the values of the Melbourne-Wübbena combination over a time interval of 89.5 min (N = 180). The index numbers j of time epochs counting from the beginning of a 24-h period with a 30-second interval are plotted on the horizontal axis. The values y_j of the combination are plotted on the vertical axis and are expressed in cycles with wavelength $\lambda_5 \approx 0.86$ [3]. **Figure 3** shows the values of deviations from the mean of the data cleared of outliers using the algorithms described in Sections 2 and 3. In both cases, σ_{max} = 0.6 and MINOBS = 10. The values $y_j - z$ are plotted on the vertical axes in cycles with wavelength λ_5 and the index numbers j of epochs on the horizontal axis.

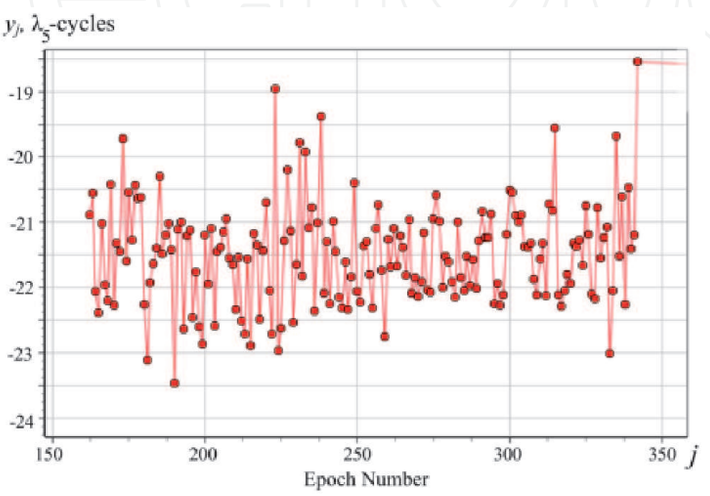

Figure 2.
Melbourne-Wübbena combination for ONSA station (GPS satellite, PRN = 12 for 2010, day 207).

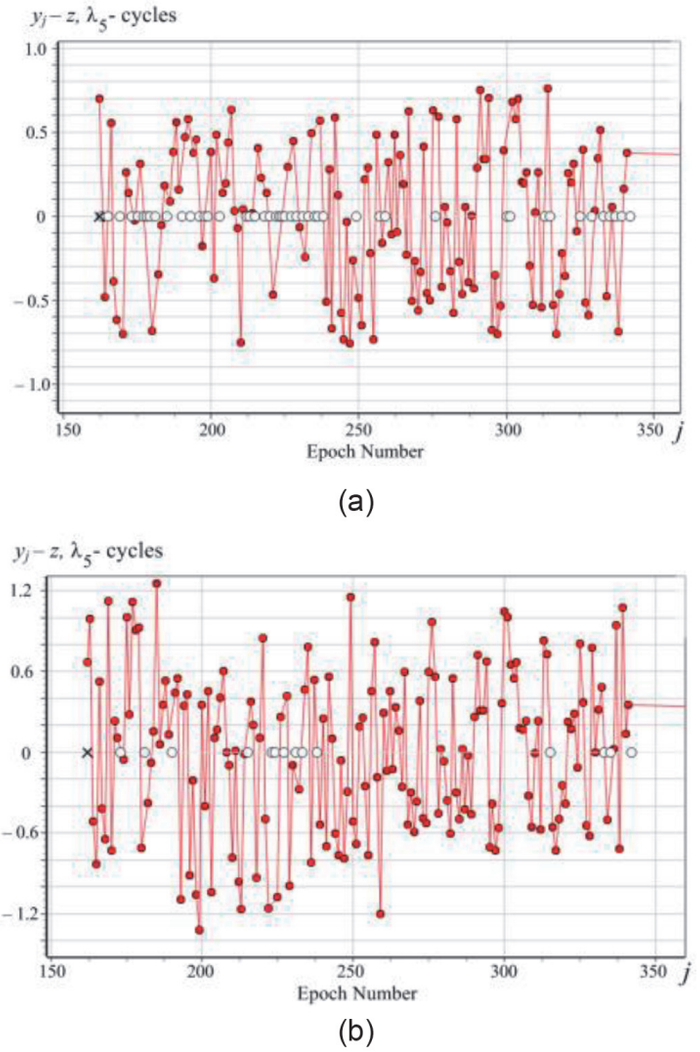

Figure 3.
(a) Deviations of values of the Melbourne-Wübbena combination from the mean value after data cleaning from outliers using the algorithm described in Section 2. (b) Deviations of values of the Melbourne-Wübbena combination from the mean value after data cleaning from outliers using the developed algorithm (see Section 3).

Epochs in which the data were rejected are designated by white circles. In the first case (see **Figure 3a**), 47 data of the measurements were rejected, which are 26.1% of the total amount of data. In the second case (see **Figure 3b**), 14 of these measurements were rejected (7.8%), which are almost 18% less than in the previous calculation.

We also provide similar results for data obtained by TLSE station, which is also included in the IGS network. We consider measurement data from GLONASS, Russia satellite with system number PRN = 1 for 2010, day 207. **Figure 4** shows the values of the Melbourne-Wübbena combination over a time interval of 65.5 min (N = 132). **Figure 5** plots the values of deviations from the mean value of the data cleared of outliers using the algorithms described in Sections 2 and 3, respectively. Parameters σ_{max} and MINOBS are the same as in the previous calculation example. In the first case (see **Figure 5a**), 41 data of the measurements were discarded, which are 31% of the total amount data. In the second case (see **Figure 5b**), 8 of these measurements were rejected (6%), which are 25% less than in the previous calculation.

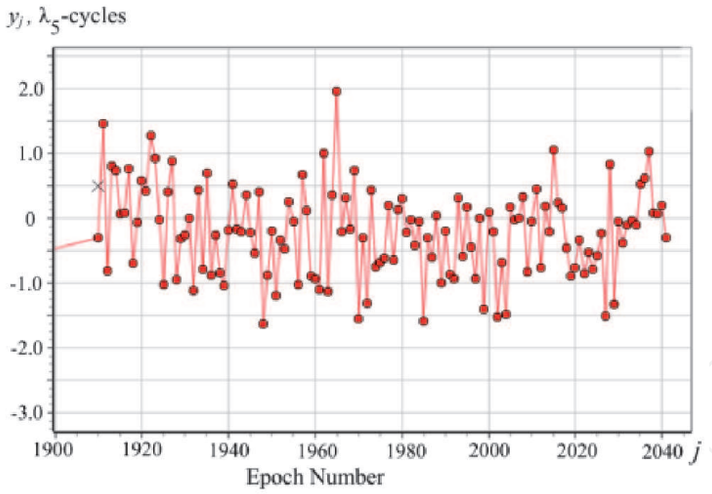

Figure 4.
Melbourne-Wübbena combination for TLSE station (GLONASS satellite, PRN = 1 for 2010, day 207).

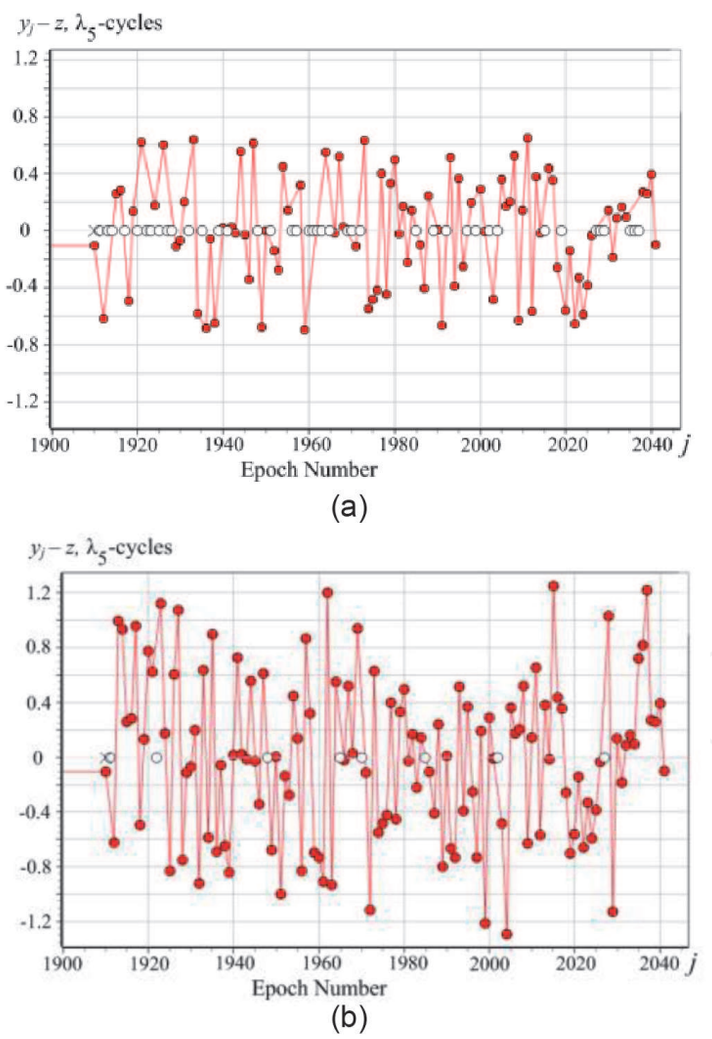

Figure 5.
(a) Deviations of values of the Melbourne-Wübbena combination from the mean value after data cleaning from outliers using the algorithm described in Section 2. (b) Deviations of values of the Melbourne-Wübbena combination from the mean value after data cleaning from outliers using the developed algorithm (see Section 3).

5. Mathematical prerequisites for modifying of existing algorithm

Note that the number of arithmetic operations required to find the optimal solution according to the algorithm described in Section 3 depends on the L_{max}, which is the length of the solution. As can be seen from Eq. (24), the smaller the length of the found solution (i.e., the larger the number of detected outliers), the more arithmetic operations are required to find it. This number of arithmetic operations is estimated to be of order $N + (N - L_{max})^2$. Note that the expression in the parentheses herein is equal to the number of outliers detected. Thus, if the number of outliers in the original data series is comparable to N, it will take $\sim N^2$ arithmetic operations to find the optimal solution. In particular, to make certain that there is no solution (e.g., in the case where the data contain a trend), $\sim N^2$ arithmetic operations will also be required. In Section 6, we will modify the existing algorithm and describe fast outlier search algorithm that requires $\sim N\log_2 N$ arithmetic operations.

The necessary preparations are given in this section. Note that in this and the next sections we are dealing with the sequence $\left\{y_j\right\}_{j=1}^{N}$ arranged in the ascending order.

Assertion 2. *Let* $\left\{y_j\right\}_{j=1}^{N}$ *be monotonically increasing sequence. The following inequality is true:*

$$\sigma^2(k; L+1) \geq \min\left\{\sigma^2(k; L),\ \ \sigma^2(k+1; L)\right\}. \tag{25}$$

Proof. From the monotonicity of the sequence y_j and the definition $z(k, L+1)$ [see Eq. (13)],

$$y_k \leq z(k, L+1) \leq y_{k+L}. \tag{26}$$

One of two cases is possible:

a. $2z(k, L+1) \leq y_{k+L} + y_k, \Rightarrow z(k, L+1) - y_k \leq y_{k+L} - z(k, L+1),$

b. $2z(k, L+1) > y_{k+L} + y_k, \Rightarrow z(k, L+1) - y_k > y_{k+L} - z(k, L+1).$

Suppose, for example, the case (a) holds. Let us show that in this case

$$\sigma^2(k; L+1) \geq \sigma^2(k; L). \tag{27}$$

At first, we will show that:

$$|y_j - z(k, L+1)| \leq y_{k+L} - z(k, L+1); j = k, \ldots, k+L. \tag{28}$$

Truly, inequalities:

$$y_k \leq y_j \leq y_{k+L}; j = \ldots, k+L,$$

and the above inequality derived in Case (a) implies:

$$z(k, L+1) - y_j \leq z(k, L+1) - y_k \leq y_{k+L} - z(k, L+1), \tag{29}$$

$$y_j - z(k, L+1) \le y_{k+L} - z(k, L+1). \tag{30}$$

These inequalities, in turn, imply Eq. (28). Next let us prove (27). This inequality is expanded as follows:

$$\frac{1}{L}\sum_{j=k}^{k+L}\Big(y_j - z(k; L+1)\Big)^2 \ge \frac{1}{L-1}\sum_{j=k}^{k+L-1}\Big(y_j - z(k; L)\Big)^2.$$

Substituting here of the expression (17) in place of $z(k; L)$ and writing for brevity, z instead of $z(k; L+1)$, we get inequality:

$$(L-1)\sum_{j=k}^{k+L}\Big(y_j - z\Big)^2 \ge L\sum_{j=k}^{k+L-1}\Big(y_j - z - \frac{z - y_{k+L}}{L}\Big)^2. \tag{31}$$

Transform the right-hand side of this inequality:

$$\begin{aligned} RHS(31) &= L\sum_{j=k}^{k+L}\Big(y_j - z + \frac{y_{k+L} - z}{L}\Big)^2 - \frac{(L+1)^2}{L}(y_{k+L} - z)^2 \\ &= L\sum_{j=k}^{k+L}\Big(y_j - z\Big)^2 + \frac{(L+1)}{L}(y_{k+L} - z)^2 - \frac{(L+1)^2}{L}(y_{k+L} - z)^2. \end{aligned}$$

Here we take into account the equality: $\sum_{j=k}^{k+L}\Big(y_j - z\Big) = 0$. Next, we have:

$$RHS(31) = L\sum_{j=k}^{k+L}\Big(y_j - z\Big)^2 - (L+1)(y_{k+L} - z)^2.$$

Substituting this expression in Eq. (31), we get inequality

$$\sum_{j=k}^{k+L}\Big(y_j - z\Big)^2 \le (L+1)(y_{k+L} - z)^2,$$

that is true due to Eq. (28). Thus, Eq. (27) is proved for case (a). Analogously, case (b) is considered.

We introduce the notation:

$$\sigma^2_{min}(L) = \min_{1 \le k \le N-L+1}\sigma^2(k, L). \tag{32}$$

Assertion 3. *The following inequalities hold:*

$$\sigma^2_{min}(N) \ge \sigma^2_{min}(N-1) \ge \ldots \ge \sigma^2_{min}(MINOBS). \tag{33}$$

That is, the sequence $\sigma^2_{min}(L)$ *monotonically decreases when L decreases from N to MINOBS.*

Proof. Assertion 2 and definition of $\sigma^2_{min}(L)$ expressed in Eq. (32) imply:

$$\sigma^2(k, L+1) \ge \min\left\{\sigma^2(k, L), \sigma^2(k+1, L)\right\} \ge \sigma^2_{min}(L).$$

Since k is chosen arbitrarily, then for all L = MINOBS, ... , N − 1 the following inequalities hold:

$$\sigma^2_{\min}(L+1) \geq \sigma^2_{\min}(L),$$

which proves Assertion 3.

Assertion 3 implies the following corollary.

Corollary 1. *If the inequality*

$$\sigma^2_{\min}(L_0) > \sigma^2_{\max}. \tag{34}$$

holds for some L_0*, then for existence of the solution* $Y_L = \{y_k, y_{k+1}, \dots, y_{k+L-1}\}$ *for the problem expressed in Eqs. (3)–(7), it is necessary that the length L of the set* Y_L *satisfies the condition* $L < L_0$.

Proof. Let us assume that $L \geq L_0$. Assertion 3 on account of monotony of $\sigma^2_{\min}(\cdot)$, expressed in Eq. (33) implies the following inequalities $\sigma^2_{\min}(L) \geq \sigma^2_{\min}(L_0) > \sigma^2_{\max}$ for all $L \geq L_0$. From this, it follows that for any set $Y_L = \{y_k, y_{k+1}, \dots, y_{k+L-1}\}$ we will have $\sigma^2(k, L) \geq \sigma^2_{\min}(L) > \sigma^2_{\max}$. Thus, any of sets Y_L of length $L \geq L_0$ does not satisfy the condition in Eq. (15) and, therefore, cannot be a solution of the problem, expressed in Eqs. (3)–(7).

In particular, we have come to the next important result. If, for example, the inequalities $\sigma^2_{\min}(\text{MINOBS}) > \sigma^2_{\max}$ are fulfilled, then the solution for the problem described in Eqs. (3)–(7) does not exist.

In the above-described procedure for solving problem (3)–(7), it takes $\sim N^2$ arithmetic operations to make certain that the solution not exists. Taking into account Assertion 3 and Corollary 1, the search procedure may begin by checking the conditions.

$$\sigma^2_{\min}(N) = \sigma^2(1, N) \leq \sigma^2_{\max} \text{ and } \sigma^2_{\min}(\text{MINOBS}) \leq \sigma^2_{\max}.$$

This will require approximately $\sim N$ arithmetic operations. If none of these conditions are fulfilled, the solution search must stop because the solution does not exist. As a result, only $\sim N$ arithmetic operations are required to ensure that there is no solution.

6. Fast outlier search algorithm

The above proposed search procedure consists in the calculating values of $z(k, L)$ and $\sigma^2(k, L)$ using recurrent formulas (17)–(22) and checking at every k and L the fulfillment of the inequalities (11) and (12). The algorithm complexity is estimated by value of $\sim(N + N^2_{\text{Outlier}})$, where N_{Outlier} is the number of outliers found. If it is known a priori that there are few outliers in the measurement data, then the search algorithm for the optimal solution that described in Section 3 can be applied. If the measurement data contain an N-comparable number of outliers, the complexity of such an algorithm will be estimated by about N^2. It is namely for such type of data we below describe a modified outlier search algorithm with complexity of about $N\log_2 N$.

First of all, note one property that is the key to the construction of a fast outlier search algorithm. Note that if the inequality (15) holds for some set of length L + 1, then there exists a set of length L for which the inequality (15) is valid too. Truly, let assume for some k the inequality $\sigma^2(k, L+1) \leq \sigma^2_{\max}$ holds. This inequality and Eq. (25) imply

$$\min\left\{\sigma^2(k;L),\ \ \sigma^2(k+1;L)\right\} \le \sigma^2_{max}.$$

From this, it follows that

$$\sigma^2(k;L) \le \sigma^2_{max} \text{ and/or } \ \sigma^2(k+1;L) \le \sigma^2_{max}. \tag{35}$$

This means that at least one of these sets $\{y_k, \ldots, y_{k+L-1}\}$ and $\{y_{k+1}, \ldots, y_{k+L}\}$ with length of L satisfies conditions expressed in Eq. (15).

However, this property is not true when checking the conditions expressed in Eq. (16). In other words, if these conditions are fulfilled for any set of length L + 1, it might happen that none of the sets of length L may satisfy them. This fact is a significant obstacle to increasing the rate of outlier detection that is necessary when processing a large amount of data with a large number of rough measurements. To overcome this obstacle, we will make the condition expressed in Eq. (16) weaker.

First of all, note that if for some set $\{y_k, \ldots, y_{k+L-1}\}$, the both conditions expressed in Eq. (16) are fulfilled, then the following inequality will hold:

$$y_{k+L-1} - y_k \le 6\sigma_{max}. \tag{36}$$

Consider a problem with condition expressed in Eq. (36) instead of conditions expressed in Eq. (16).

Remark. *Recall that in this condition L means the length of the set under checking, and k is the index of the smallest number included in the set. Although "k" and "L" are also encountered as indexes in the sets we use hereinafter, we hope nevertheless that this will not lead to confusion.*

It is easily seen that condition expressed in Eq. (36) for an arbitrary set $\{y_k, \ldots, y_{k+L}\}$ of length L + 1 implies the fulfillment this condition for each of the sets $\{y_k, \ldots, y_{k+L-1}\}$ and $\{y_{k+1}, \ldots, y_{k+L}\}$ of length L. In fact, the fulfillment condition (36) for the set $\{y_k, \ldots, y_{k+L}\}$ means the fulfillment of inequality $y_{k+L} - y_k \le 6\sigma_{max}$ from which due to monotony of y_j, the inequalities imply $y_{k+L} - y_{k+1} \le 6\sigma_{max}$ and $y_{k+L-1} - y_k \le 6\sigma_{max}$.

Thus, we have established the validity of the following assertion.

Assertion 4. *If the set $\{y_k, \ldots, y_{k+L}\}$ of length L + 1 satisfies conditions (15) and (36), then at least one of the two sets $\{y_k, \ldots, y_{k+L-1}\}$ or $\{y_{k+1}, \ldots, y_{k+L}\}$ of length L also satisfies conditions (15) and (36).*

Based on this statement, we can formulate the following:

Assertion 5. *Solution for the problem (15) + (36) can be found for $\sim N \log_2 N$ arithmetic operations.*

Proof. Let us consider the sequence of steps.

Step 0: Consider the segment $\left[N^{(0)}_{Left}, N^{(0)}_{Right}\right]$ of numerical axis, where $N^{(0)}_{Left}$ = MINOBS, $N^{(0)}_{Right}$ = N. In this step, there is one set $\{y_1, \ldots, y_N\}$ of length N. We check it for fulfillment of the conditions expressed in Eqs. (15) and (36) with k = 1, L = N. If they are satisfied, this set is the solution, and our search stops. Otherwise, we pass to considering of (N − MINOBS − 1) sets of length MINOBS. We check each of these sets for fulfillment of conditions expressed in Eqs. (15) and (36). If none of them satisfy these conditions, then we stop the search process and conclude that the solution does not exist. Otherwise, once we find the set of length MINOBS satisfying conditions (15) and (36), we transit to Step 1.

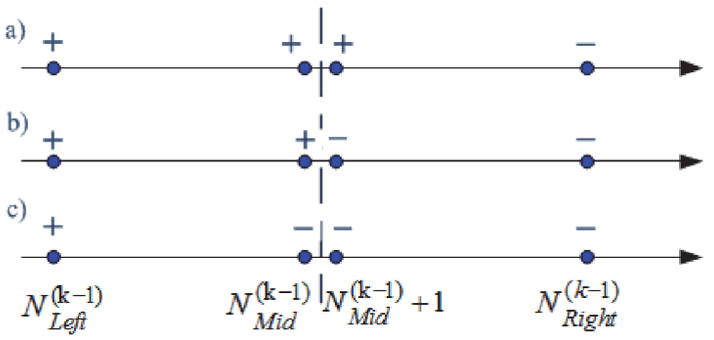

Figure 6.
Three possible cases for the proposed search. In case (a) we go to the right-hand side range (range for length of sets) to find a solution, in case (c) we go to the left-hand side range to find a solution, in case (b) we look for a solution with length $L = N_{Mid}^{(k-1)}$ *and the search ends.*

Step 1: Step 1 is the same as Step k described below for k = 1

...

Step k: On the *k*th step, where (k ≥ 1), we consider a segment $\left[N_{Left}^{(k-1)}, N_{Right}^{(k-1)}\right]$, which we halve, as a result we receive two segments $\left[N_{Left}^{(k-1)}, N_{Mid}^{(k-1)}\right]$ and $\left[N_{Mid}^{(k-1)}+1, N_{Right}^{(k-1)}\right]$, where $N_{Mid}^{(k-1)} = N_{Left}^{(k-1)} + \left[\left(N_{Right}^{(k-1)} - N_{Left}^{(k-1)}\right)/2\right]$, [·] designate integral part of a number. Next, we check the sets of length $N_{Mid}^{(k-1)}$ and $N_{Mid}^{(k-1)} + 1$ for fulfillment of conditions (15) and (36). The following three cases are possible, schematically shown in **Figure 6**. The sign "−" above the point means that for none of the sets of the corresponding length the conditions (15) + (36) are satisfied, the sign "+" vice versa, that is, there is at least one set of the corresponding length satisfying (15) + (36). In the case of (a), we set $N_{Left}^{(k)} = N_{Mid}^{(k-1)} + 1$, $N_{Right}^{(k)} = N_{Right}^{(k-1)}$ and transit to the (k + 1)-th step; in the case of (c), we set $N_{Left}^{(k)} = N_{Left}^{(k-1)}$, $N_{Right}^{(k)} = N_{Mid}^{(k-1)}$ and transit to the (k + 1)-th step; in the case of (b), we search the solution (15) + (36) with minimal value of $\sigma^2(k, L)$ with $L = N_{Mid}^{(k-1)}$; the algorithm is terminated.

The search process will continue until either case (b) or until the length of the segment $\left[N_{Left}^{(k)}, N_{Right}^{(k)}\right]$ is less than or equal to 1. In either case, the total number of steps will not exceed the number of $\log_2(N - \text{MINOBS})$. Since ~N operations are performed in each step, the search process is guaranteed to be terminated in $\sim N \log_2 N$ arithmetic operations.

7. Search an unknown trend in the power polynomial class

The need to process GNSS measurements including a trend on which noise and outliers are superimposed arises at different processing stages of the application process. As already stated above, satellite clock corrections contain a linear trend. In some cases, it may not be known, and then, one has to search for it, for example, by the least square method. The presence of outliers in the measurement data is a significant obstacle to accurate determination of drift and offset parameters of satellite clocks. Other examples are linear combinations of code and phase data on two carriers [3]. To obtain high accuracy results, it is necessary to detect outliers against an unknown trend and remove them from further processing. This is the subject of this section.

7.1 Statement of the problem

Consider the problem of outlier detecting in data presented in the form of Eq. (1), recall that:

$$y_j = f_j + z + \xi_j; j = 1 \dots N.$$

The procedure described above for finding the optimal solution in an ordered series of numbers may not produce an adequate result if applied to data containing an unknown trend. For example, there may be no solution, and all data will be defined as outliers. In order to detect outliers in a series of numbers with a trend using the algorithm described above, it is necessary to find a suitable approximation of an unknown function f_j and subtract it from the data set. Searching for this approximation is usually done by selecting functions from some functional class. The choice of the functional class depends on the task. In some cases, these may be power polynomial, in other cases, trigonometric polynomial, etc. The presence of outliers in the data measurements makes it much more difficult to find such an approximation. In this section, we will describe the general approach to solving the problem and look for suitable approximations of trend f_j in the class of power polynomial with real coefficients:

$$P_{n,j}\left(\vec{a}\right) = a_n(j/N)^n + a_{n-1}(j/N)^{n-1} + \dots + a_0, \tag{37}$$

where n is the polynomial degree, and $\vec{a} = \{a_0, \dots, a_n\}$ is vector of coefficients.

Thus, the problem consists in the creation of an algorithm for searching the trend in the class of power polynomial and detecting outliers in specified data series y_j represented in Eq. (1).

7.2 Minimizing set of specified length L

Before we turn to the trend search algorithm construction, we will define the so-called minimizing set of given length L, which plays an essential role in the trend search. In addition, in Section 7.3, we will describe a search algorithm for such set based on the recurrent formulas (17)–(19) and (20)–(22).

Let $Y_L = \left\{y_{j_1}, \dots, y_{j_L}\right\}$ be an arbitrary set of length L, composed of the values of a numerical series $\left\{y_j\right\}_{j=1}^{N}$, the monotony is not supposed. The mean and the SD values for it are denoted by z_{Y_L} and σ_{Y_L}. These values are calculated by standard formulas:

$$z_{Y_L} = L^{-1} \sum_{j \in \left\{j_1, \dots, j_L\right\}} y_j, \tag{38}$$

$$\sigma^2_{Y_L} = (L-1)^{-1} \sum_{j \in \left\{j_1, \dots, j_L\right\}} \left(y_j - z_{Y_L}\right)^2. \tag{39}$$

Definition 2. *Given L for a specified sequence of values* $\left\{y_j\right\}_{j=1}^{N}$ *the set of values:*

$$Y_{L,\min} = \left\{y_{j_1}, \dots, y_{j_L}\right\},$$

at which the minimum value of $\sigma^2_{Y_L}$ *defined in Eq. (39) is reached will be called the minimizing set of length L. The corresponding mean and SD values are denoted by* $z_{Y_{L,\min}}$ *and* $\sigma_{Y_{L,\min}}$.

According to this definition, we have

$$z_{Y_{L,\min}} = L^{-1} \sum_{j \in \{j_1, \ldots, j_L\}} y_j, \tag{40}$$

$$\sigma^2_{Y_{L,\min}} = (L-1)^{-1} \sum_{j \in \{j_1, \ldots, j_L\}} \left(y_j - z_{Y_{L,\min}}\right)^2 = \min_{Y_L} \left\{\sigma^2_{Y_L}\right\}. \tag{41}$$

Minimum in Eq. (41) is searched by all kinds of sets of length L composed of numbers of series $\left\{y_j\right\}_{j=1}^{N}$.

Note that the numbers y_j are not supposed to be in the ascending order.

Next, we will formulate and prove a statement similar to Assertion 1, which will allow us, when searching for a minimizing set, to proceed from the original series to its ordered permutation.

Assertion 6. Let $Y_{L,\min} = \left\{y_{j_1}, \ldots, y_{j_L}\right\}$ be a minimizing set of length L *for a given sequence of values* $\left\{y_j\right\}_{j=1}^{N}$ and

$$y_{\min} = \min\left\{y_{j_1}, \ldots, y_{j_L}\right\}, y_{\max} = \max\left\{y_{j_1}, \ldots, y_{j_L}\right\},$$

then the interval $(y_{\min}, y_{\max})$ *does not contain values* y_j *that are not in the set* $Y_{L,\min}$.

Proof. Let us assume the opposite: Let $y_j \notin Y_{L,\min}$, $y_{\min} < y_j < y_{\max}$. Let $y_k, \ldots, y_{k+L-1}$ is a permutation of the numbers $y_{j_1}, \ldots, y_{j_L}$ in the ascending order; then $y_k = y_{\min}$ and $y_{k+L-1} = y_{\max}$. One of these cases is possible:

a. $y_j < z_{Y_{L,\min}}$ and therefore subject to inequality $y_k < y_j$, we have

$$\left(z_{Y_{L,\min}} - y_k\right) > \left(z_{Y_{L,\min}} - y_j\right) \Rightarrow \left(z_{Y_{L,\min}} - y_k\right)^2 > \left(z_{Y_{L,\min}} - y_j\right)^2$$
$$\Rightarrow \left(z_{Y_{L,\min}} - y_j\right)^2 - \left(z_{Y_{L,\min}} - y_k\right)^2 < 0 \tag{42}$$

b. $y_j \geq z_{Y_{L,\min}}$ and therefore subject to inequality $y_{k+L-1} > y_j$, we have

$$\left(y_{k+L-1} - z_{Y_{L,\min}}\right) > \left(y_j - z_{Y_{L,\min}}\right) \Rightarrow \left(y_j - z_{Y_{L,\min}}\right)^2 - \left(y_{k+L-1} - z_{Y_{L,\min}}\right)^2 < 0 \tag{43}$$

In the first case, Case (a), we replace the value y_k in the set $\left\{y_k, \ldots, y_{k+L-1}\right\}$ with y_j. In the second case, Case (b), we replace the value y_{k+L-1} with y_j. We want to show that doing such replacement, the value $\sigma^2_{Y_{L,\min}}$ expressed in Eqs. (40) and (41) will decrease. This will mean that $Y_{L,\min}$ is not a minimizing set.

Suppose Case (a). For brevity, we will write below z instead of $z_{Y_{L,\min}}$ and σ^2 instead of $\sigma^2_{Y_{L,\min}}$. Denote $\tilde{z}$ and $\tilde{\sigma}^2$, the similar values obtained after replacement y_k with y_j. We have:

$$z = L^{-1}(y_k + \ldots + y_{k+L-1}), \tag{44}$$

$$\sigma^2 = (L-1)^{-1}\left((y_k - z)^2 + \ldots + (y_{k+L-1} - z)^2\right), \tag{45}$$

and

$$\tilde{z} = L^{-1}\left(y_{k+1} + \ldots + y_{k+L-1} + y_j\right), \tag{46}$$

$$\tilde{\sigma}^2 = (L-1)^{-1}\left((y_{k+1} - \tilde{z})^2 + \ldots + (y_{k+L-1} - \tilde{z})^2 + \left(y_j - \tilde{z}\right)^2\right). \tag{47}$$

We want to show that

$$\tilde{\sigma}^2 < \sigma^2. \tag{48}$$

Eqs. (44) and (46) imply:

$$\tilde{z} = z + L^{-1}\left(y_j - y_k\right). \tag{49}$$

Modify $\tilde{\sigma}^2$ expressed in Eq. (47) taking account of Eq. (49):

$$\tilde{\sigma}^2 = (L-1)^{-1}\left(\left(y_{k+1} - z - L^{-1}\left(y_j - y_k\right)\right)^2 + \ldots \right.$$
$$\left. + \left(y_{k+L-1} - z - L^{-1}\left(y_j - y_k\right)\right)^2 + \left(y_j - z - L^{-1}\left(y_j - y_k\right)\right)^2\right).$$

After simplification with taking into account Eq. (45), we get from here:

$$\tilde{\sigma}^2 = \sigma^2 + (L-1)^{-1}\left[\left(y_j - z\right)^2 - (y_k - z)^2 - L^{-1}\left(y_j - y_k\right)^2\right].$$

From (42), it follows (recall that we write z instead of $z_{Y_{L,\min}}$) that the expression in the square brackets is strictly less than zero. Thus, inequality (48) is proven. From this follows that the set $Y_{L,\min}$ is not minimizing one because the condition (41) is not met. Thus, we have arrived at a contradiction that proves the validity of the formulated Assertion 6. Case (b) is considered similarly.

7.3 $Y_{L,\min}$ search algorithm

Assertion 6 is much like Assertion 1, which made it possible to go from an arbitrary numerical series to an ordered one for the optimal solution search (see Section 3). Similarly, Assertion 6 makes it possible to go to an ordered number series to find the minimizing set of numbers of a given length. In our case, considerations similar to those presented in Section 3 may be made when searching for a minimizing set. Namely, if $Y_{L,\min} = \left\{y_{j_1}, \ldots, y_{j_L}\right\}$ is the minimizing set of length L, then an arbitrary permutation of the numbers $y_{j_1}, \ldots, y_{j_L}$ will also be the minimizing set. Thus, the process for searching of $Y_{L,\min}$ does not depend on the arrangement of given numbers y_j. This allows us to arrange them in the order most suitable for searching the minimizing set. We arrange the values $\left\{y_j\right\}$ in the ascending order, and we will look for the minimizing set in the ordered sequence. As in Section 3, for brevity, the ordered sequence will also be denoted by $\left\{y_j\right\}$. For simplicity of logic,

we will assume that all y_j are different, that is, Eq. (10) holds. Rewrite it for convenience:

$$y_1 < y_2 < \ldots < y_N$$

Note that due to Assertion 6, if $Y_{L,\min}$ is the minimizing set and y_{j_1} and y_{j_L} are its smallest and greatest values, respectively, then all values of y_j from the interval $\left(y_{j_1}, y_{j_L}\right)$ belong to $Y_{L,\min}$. Consequently, considering Eq. (10), we have $y_{j_L} = y_{j_1+L-1}$ and therefore

$$Y_{L,\min} = \left\{y_{j_1}, y_{j_1+1}, \ldots, y_{j_1+L-1}\right\}.$$

Thus, in order to find a minimizing set of length L, it is sufficient for us to check $N - L + 1$ sets

$$\{y_1, \ldots, y_L\}, \{y_2, \ldots, y_{L+1}\}, \ldots, \{y_{N-L+1}, \ldots, y_N\}.$$

and choose from them the one that has minimal SD value. In the minimizing set searching procedure, we use the appropriate designations $z(k, L)$ and $\sigma^2(k, L)$ expressed by Eqs. (13) and (14) to denote mean and square SD in the case of sets composed of the ascending ordered values.

The calculation of $z(k, L)$ and $\sigma^2(k, L)$ when searching $Y_{L,\min}$ can be carried out in the manner similar to that of the optimal solution according to the schematic, in which only the $\sigma^2(k, L)$ is shown:

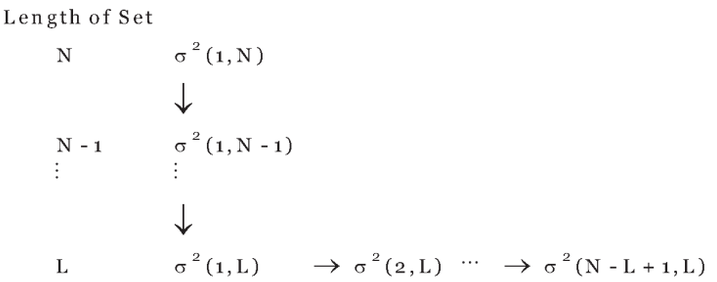

Figure 7.
Scheme of calculations when finding the minimizing set of length L.

At first, we carry out the transitions in the direction of the vertical arrows (see diagram in **Figure 7**) and calculate sequential values $\sigma^2(1, N), \sigma^2(1, N-1), \ldots, \sigma^2(1, L)$ using formula (17)–(19). Finally, we go in the direction of the horizontal arrows and calculate values of $\sigma^2(1, L), \sigma^2(2, L), \ldots, \sigma^2(N - L + 1, L)$ using formula (20)–(22) and choose from them the minimal one.

7.4 Trend search algorithm

In order to correctly detect outliers in measurement data that include an unknown trend, it is necessary to find and remove trend from the original data. The problem in determining of unknown trend is to find a suitable profile for the measurement data by adjusting the fitting parameters. This implies, in turn, the needs to select from the specified series y_j the "right" reference values $\left\{y_{j_1}, \ldots, y_{j_L}\right\}$ used for fitting. If the data do not contain coarse measurements, all N values of the original data can be treated as reference ones and used for fitting. If the data contain

rough measurements, the participation all or some of those values in the fit will result in a fatal distortion of the trend function and, as a result, an incorrect determination of the outliers. Thus, when determining the trend, it is necessary to exclude rough measurements from the number of reference values used for fitting and by which the trend approximations are built. We have the vicious circle. To determine outliers, it is necessary to find a proper trend approximation, and to find an exact trend approximation, we need to find all outliers and remove them from the values used for fitting. Another vicious circle is obtained when trying to choose the number of values for fitting. If we take it too small to minimize the possibility for outliers to be included into the subset of values for fitting, the data fit may not be accurate enough for the rest values of the data outside of the subset. If, on the contrary, we take rather large number of points for fitting, the outliers may be among them, and we will also get a bad approximation of the trend.

We describe herein a strategy for finding an unknown trend and detecting outliers. This strategy assumes that the number of outliers in the series presented by Eq. (1) does not exceed a certain value N_{maxout} known a priori. Thus, we can suppose that the number of "right" values suitable for fitting in the series y_j is not less than $N - N_{maxout}$.

Below L is supposed to be fixed and associated with the number of the reference values of the series (1) used for fitting, and $L \le N - N_{maxout}$.

Let us consider the following algorithm. It contains internal iterations, which we will denote with the upper index "s" in parentheses.

Step 0: n = 0. We set some L, satisfying the condition $L \le N - N_{maxout}$.

Step 1: n++; s = 0; $flag_j^{(0)} = 1$.

Step 2: s++. We fit polynomial to the data set and find fitting coefficients $\vec{a}^{(s)} = \left\{a_0^{(s)}, \ldots, a_n^{(s)}\right\}$:

$$\vec{a}^{(s)} = \underset{a_0, \ldots, a_{n-1}, a_n}{\arg\min} \Phi^{(s-1)}\left(\vec{a}\right), \tag{50}$$

$$\Phi^{(s-1)}\left(\vec{a}\right) \triangleq \sum_{j=1}^{N} \left(y_j - P_{n,j}\left(\vec{a}\right)\right)^2 \cdot flag_j^{(s-1)}. \tag{51}$$

Step 3: Consider the values $\hat{y}_j^{(s)}$ obtained after subtraction from y_j the polynomial values with coefficients found in Step 2:

$$\hat{y}_j^{(s)} = y_j - P_{n,j}\left(\vec{a}^{(s)}\right). \tag{52}$$

Using the algorithm described in Section 7.3, we find from the numbers $\left\{\hat{y}_j^{(s)}\right\}_{j=1}^{N}$ defined in Eq. (52) a minimizing set $Y_{L,\min}^{(s)} = \left\{\hat{y}_{j_1^{(s)}}^{(s)}, \ldots, \hat{y}_{j_L^{(s)}}^{(s)}\right\}$ of length L. For this set, we calculate the corresponding values of the mean $z_{Y_{L,\min}^{(s)}}$ and SD $\sigma_{Y_{L,\min}^{(s)}}$ according to the formulas (40) and (41) in which $\{j_1, \ldots, j_L\}$ is replaced by $\left\{j_1^{(s)}, \ldots, j_L^{(s)}\right\}$. We redefine the reference points for the next fit process (Step 2) by marking them with $flag_j^{(s)}$:

$$flag_j^{(s)} = \begin{cases} 1, & \text{if} \quad j \in \left\{j_1^{(s)}, \ldots, j_L^{(s)}\right\} \\ 0, & \text{otherwise} \end{cases}. \tag{53}$$

We transit to Step 2 and do so until the convergence of the $\sigma_{Y^{(s)}_{L,\min}}$, s = 1,2, ..., is achieved. Note that we will consider the issue of convergence further in Section 7.5. After reaching convergence of values $\sigma_{Y^{(s)}_{L,\min}}$, we transit to Step 4 for outlier detection.

Step 4: Searching the optimal solution for data set $\hat{y}_j^{(s)} = y_j - P_{n,j}\left(\vec{a}^{(s)}\right)$, j = 1, ..., N.

We find the optimal solution for values $\hat{y}_j^{(s)}$ using the algorithm of Section 3 and determine the number N_{out} of outliers detected. If it turns out that $N_{out} \le N_{maxout}$, then solution is considered found; searching process stops: $f_j = P_{n,j}\left(\vec{a}^{(s)}\right)$. Otherwise, if $N_{out} > N_{maxout}$, then we transit to Step 1.

We do this until we find a solution or until n reaches some preset value N_{max} (e.g., 10). In this case, probably, we may need to select a different functional class to search for a trend.

Example. Let us consider the data simulated in accordance with formula: $y_j = 10\sqrt{j+10} + 2\text{random}_j$, where $j = 1 \ldots N$, random_j—pseudorandom numbers, evenly distributed in the segment [0,1]. Let us set N = 150 and model 10 outliers at points $j = 6 \ldots 10$ and $j = 140 \ldots 144$ (see **Figure 8**). Further, let us set σ_{max} = 0.6, MINOBS = 10 and search for outliers as described above with L = 130. If n = 1, we get N_{out} = 88; if n = 2, we get N_{out} = 33; if n = 3, we get N_{out} = 17; if n = 4, we get N_{out} = 10. Thus, a suitable trend approximation turned out to be a fourth degree polynomial. **Figure 9** shows values for fourth degree polynomial fitted to the data set on

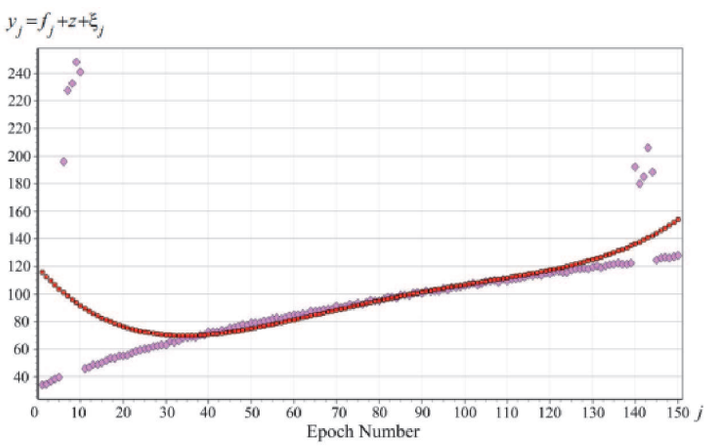

Figure 8.
Data simulated using: $y_j = 10\sqrt{j+10} + 2\text{random}_j$ and fourth degree polynomial after the first iteration.

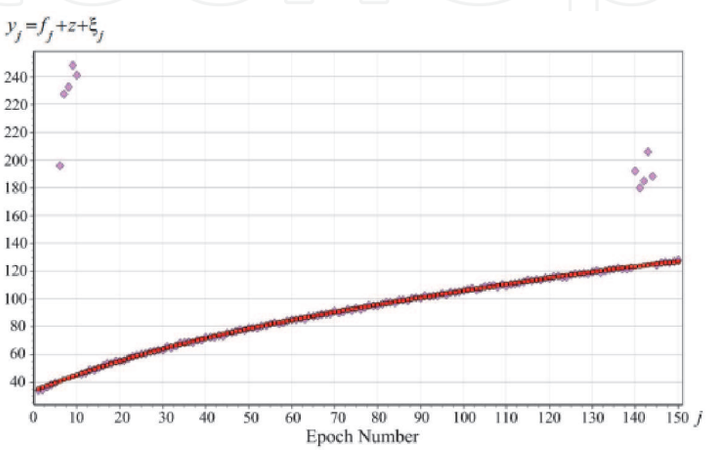

Figure 9.
Simulated data and fourth degree polynomial after the eighth iteration.

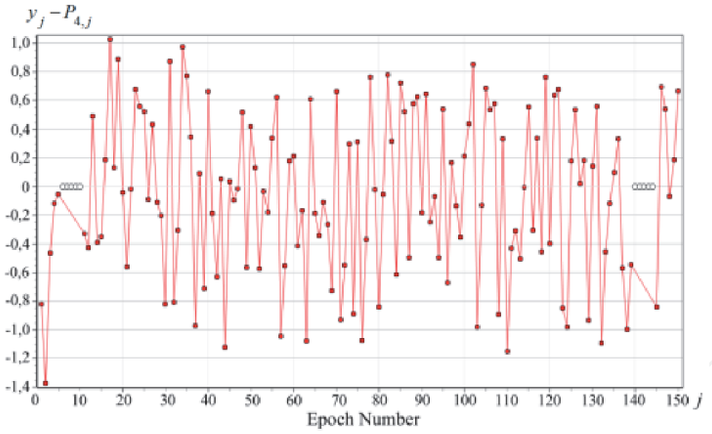

Figure 10.
The differences $\hat{y}_j = y_j - P_{4,j}\left(\vec{a}\right)$, approximation of unknown trend with fourth degree polynomial.

the eighth iteration after the convergence discussed above is reached. In **Figure 10**, the differences $\hat{y}_j = y_j - P_{4,j}\left(\vec{a}\right)$ are plotted. Positions of detected outliers are marked with white circles. Note that the trend was modeled by a function that does not belong to the class of power polynomials.

7.5 Convergence of iterations in trend search algorithm

This section explains the convergence of the iterations described in the trend search algorithm (see previous section).

Assertion 7. *The SD sequence $\sigma_{Y^{(s)}_{L,\min}}$ of values calculated in the trend search algorithm monotonically decreases at s = 1, 2, ... :*

$$\sigma_{Y^{(s)}_{L,\min}} \geq \sigma_{Y^{(s+1)}_{L,\min}} \geq \ldots \tag{54}$$

and therefore converged.

Proof. We start our consideration with Step 3 and *s*th iteration, *s* = 1, 2, ... In Step 3, for the sequence $\left\{\hat{y}^{(s)}_j\right\}_{j=1}^{N}$, we find a minimizing set of length L: $Y^{(s)}_{L,\min} = \left\{\hat{y}^{(s)}_{j^{(s)}_1}, \ldots, \hat{y}^{(s)}_{j^{(s)}_L}\right\}$. Write the expressions for the mean and the square of SD corresponding to this set. We have due to Eqs. (40) and (41):

$$z_{Y^{(s)}_{L,\min}} = L^{-1} \sum_{j \in \left\{j^{(s)}_1, \ldots, j^{(s)}_L\right\}} \hat{y}^{(s)}_j, \tag{55}$$

$$\sigma^2_{Y^{(s)}_{L,\min}} = (L-1)^{-1} \sum_{j \in \left\{j^{(s)}_1, \ldots, j^{(s)}_L\right\}} \left(\hat{y}^{(s)}_j - z_{Y^{(s)}_{L,\min}}\right)^2. \tag{56}$$

Transform the expression on the right side of Eq. (56). Substitution here instead of $\hat{y}^{(s)}_j$ expression in Eq. (52) gives:

$$\sigma^2_{Y^{(s)}_{L,\min}} = (L-1)^{-1} \sum_{j \in \left\{j^{(s)}_1, \ldots, j^{(s)}_L\right\}} \left(y_j - P_{n,j}\left(\vec{a}^{(s)}\right) - z_{Y^{(s)}_{L,\min}}\right)^2.$$

Using the $\text{flag}_j^{(s)}$ defined in Eq. (53), rewrite the last equality as follows:

$$\sigma^2_{Y^{(s)}_{L,\min}} = (L-1)^{-1}\sum_{j=1}^{N}\left(y_j - P_{n,j}\left(\vec{a}^{(s)}\right) - z_{Y^{(s)}_{L,\min}}\right)^2 \cdot \text{flag}_j^{(s)} =$$
$$= (L-1)^{-1}\sum_{j=1}^{N}\left(y_j - P_{n,j}\left(\vec{b}\right)\right)^2 \cdot \text{flag}_j^{(s)} = (L-1)^{-1}\Phi^{(s)}\left(\vec{b}\right). \quad (57)$$

Here we introduce the designation $\vec{b} = \left\{a_0^{(s)} + z_{Y^{(s)}_{L,\min}}, a_1^{(s)}, \ldots, a_n^{(s)}\right\}$ and take into account the definition of $P_{n,j}\left(\vec{a}\right)$ expressed by Eq. (37) and the definition of functional $\Phi^{(s)}\left(\vec{a}\right)$ given in Eq. (51). From Eq. (57), we get:

$$\sigma^2_{Y^{(s)}_{L,\min}} = (L-1)^{-1}\Phi^{(s)}\left(\vec{b}\right) \quad (58)$$

We transit to Step 2; *s* is incremented by 1. We find a vector $\vec{a}^{(s+1)}$ of polynomial coefficients, which minimizes the functional $\Phi^{(s)}\left(\vec{a}\right)$:

$$\vec{a}^{(s+1)} = \arg\min_{\vec{a}} \Phi^{(s)}\left(\vec{a}\right).$$

Thus,

$$\Phi^{(s)}\left(\vec{a}^{(s+1)}\right) = \min_{\vec{a}}\left\{\Phi^{(s)}\left(\vec{a}\right)\right\} \quad (59)$$

From the definition of the extremum of functional, it follows:

$$\Phi^{(s)}\left(\vec{a}^{(s+1)}\right) \le \Phi^{(s)}\left(\vec{b}\right). \quad (60)$$

Taking into account Eq. (58), we have:

$$\sigma^2_{Y^{(s)}_{L,\min}} \ge (L-1)^{-1}\Phi^{(s)}\left(\vec{a}^{(s+1)}\right). \quad (61)$$

The extremum condition (one of n + 1) of functional $\Phi^{(s)}\left(\vec{a}\right)$ is as follows:

$$\frac{\partial}{\partial a_0}\Phi^{(s)}\left(\vec{a}\right)\Big|_{\vec{a}=\vec{a}^{(s+1)}} = 0.$$

From here, we derive:

$$\sum_{j=1}^{N}\left(y_j - P_{n,j}\left(\vec{a}^{(s+1)}\right)\right)\cdot \text{flag}_j^{(s)} = 0.$$

Taking into account the designation for $\hat{y}_j^{(s+1)}$ given by Eq. (52), we can write this equality in the form.

$$\sum_{j=1}^{N}\hat{y}_j^{(s+1)}\cdot \text{flag}_j^{(s)} = 0 \text{ or } \sum_{j\in\left\{j_1^{(s)},\ldots,j_L^{(s)}\right\}}\hat{y}_j^{(s+1)} = 0. \quad (62)$$

Consider the set $Y_L^{(s+1)} = \left\{\hat{y}_{j_1^{(s)}}^{(s+1)}, \ldots, \hat{y}_{j_L^{(s)}}^{(s+1)}\right\}$. The mean and SD values for it are calculated using formulas (38) and (39). Taking into account Eq. (62), we get:

$$z_{Y_L^{(s+1)}} = L^{-1} \sum_{j \in \left\{j_1^{(s)}, \ldots, j_L^{(s)}\right\}} \hat{y}_j^{(s+1)} = 0,$$

and

$$\sigma^2_{Y_L^{(s+1)}} = (L-1)^{-1} \sum_{j \in \left\{j_1^{(s)}, \ldots, j_L^{(s)}\right\}} \left(\hat{y}_j^{(s+1)} - z_{Y_L^{(s+1)}}\right)^2 = (L-1)^{-1} \sum_{j \in \left\{j_1^{(s)}, \ldots, j_L^{(s)}\right\}} \left(\hat{y}_j^{(s+1)}\right)^2 =$$
$$= (L-1)^{-1} \sum_{j=1}^{N} \left(y_j - P_{n,j}\left(\vec{a}^{(s+1)}\right)\right)^2 \cdot flag_j^{(s)} = (L-1)^{-1} \Phi^{(s)}\left(\vec{a}^{(s+1)}\right).$$

Thus,

$$\sigma^2_{Y_L^{(s+1)}} = (L-1)^{-1} \Phi^{(s)}\left(\vec{a}^{(s+1)}\right). \tag{63}$$

We transit to Step 3. In this step, for sequence $\left\{\hat{y}_j^{(s+1)}\right\}_{j=1}^{N}$, we find a minimizing set of length L: $Y_{L,\min}^{(s+1)} = \left\{\hat{y}_{j_1^{(s+1)}}^{(s+1)}, \ldots, \hat{y}_{j_L^{(s+1)}}^{(s+1)}\right\}$. The SD square $\sigma^2_{Y_{L,\min}^{(s+1)}}$ for this set does not exceed the same magnitude for any other set, particularly for $Y_L^{(s+1)} = \left\{\hat{y}_{j_1^{(s)}}^{(s+1)}, \ldots, \hat{y}_{j_L^{(s)}}^{(s+1)}\right\}$:

$$\sigma^2_{Y_L^{(s+1)}} \geq \sigma^2_{Y_{L,\min}^{(s+1)}}. \tag{64}$$

Finally, from Eqs. (61) and (63) and the last inequality (64), we get

$$\sigma^2_{Y_{L,\min}^{(s)}} \geq (L-1)^{-1} \Phi^{(s)}\left(\vec{a}^{(s+1)}\right) = \sigma^2_{Y_L^{(s+1)}} \geq \sigma^2_{Y_{L,\min}^{(s+1)}}$$

that is, $\sigma^2_{Y_{L,\min}^{(s)}} \geq \sigma^2_{Y_{L,\min}^{(s+1)}}$, that proves Assertion 7.

8. Detection and repair cycle slips in the Melbourne-Wübbena combination algorithm

In this and the following section, we describe cycle slip repair algorithm for observers represented in the form of Melbourne-Wübbena combination, which is often used in modern GNSS measurement data processing programs. Loss by the receiver of the carrier phase capture results in jumps in the code and phase measurement data. In the absence of jumps, as we already discussed in Section 2, the values of the combination consist of measurement noise superimposed on an unknown constant value dependent on a specified satellite-receiver pair.

In case of temporary loss of carrier phase capture by the receiver, jumps occur in a series of values representing the Melbourne-Wübbena combination. The procedure of detecting jumps and eliminating them from the values of the combination,

called cycle slip repair, is one of the most important steps of preprocessing GNSS data. The main difficulty in detecting jumps is that neither the exact size of jumps nor their epochs are known. A number of algorithms, descriptions of which can be found, for example, in Refs. [3, 6, 8] are proposed for the detection of jumps. Although differing in detail, they are based on a common idea, that is, comparison of the SDs of the time series of measurement data obtained from one of the bounds of the time interval to an arbitrary moment, an epoch. If the differences of the SDs corresponding to two adjacent epochs exceed a predetermined threshold value, then a jump is declared in one of these two epochs. A drawback of similar algorithms is the frequent false detection of jumps during epochs containing rough measurements (outliers) since the values of outliers can exceed the sizes of a jump itself. On the other hand, an attempt to increase the threshold value leads to the opposite effect, an inability to recognize jumps that are small in magnitude.

Below, we propose a robust cycle slip repair algorithm that allows, more reliably than similar known algorithms, to detect jumps and determine their sizes. The proposed algorithm is based on search for so-called clusters consisting of epochs, in which the values of the combination are grouped about corresponding predefined values. Besides, this algorithm implements the above-described method of cleaning data from outliers based on the search for the optimal solution. This method, combined with Springer's algorithm used in Ref. [3], allows for the reliable determination of multiple (cascade) jumps in the Melbourne-Wübbena combination.

The Melbourne-Wübbena combination L_{M-W} can be presented in the form of the total of three terms [6]:

$$L_{M-W} = \lambda_5 n_5 + B + \nu, \tag{65}$$

where λ_5 is the formal wavelength (for all GPS satellites λ_5 = 86.16 cm and for GLONASS satellites λ_5 = 84.0 ÷ 84.36 cm); n_5 is an unknown integer, the so-called wide lane ambiguity [3]; B is the residual systematic error caused by instrumental delays relative to the specific receiver-satellite pair and, as assumed, not time-dependent; and ν is a random component. Both parts of the equality in (65) are expressed in meters. At the same time, the L_{M-W} combination is often expressed in cycles of wavelength λ_5. Let us divide both parts of Eq. (65) by λ_5. We introduce the designations $y = L_{M-W}/\lambda_5$, $\beta = B/\lambda_5$, and $\xi = \nu/\lambda_5$ and derive for each of the epochs associated with indexes $j = 1, \ldots, N$:

$$y_j = n_{5j} + \beta + \xi_j. \tag{66}$$

where n_{5j} is an unknown piecewise-constant function of a discrete argument, regarding which it is only known that it takes integer values and can undergo integer jumps; β is an unknown constant value; and ξ_j is a random variable. The problem consists in the development of an algorithm for automatic processing of the informational data y_j of form of Eq. (66), making it possible to effectively determine integer-valued jumps against the noise component and outliers.

9. Description of the proposed algorithm

Let us present the proposed algorithm as the following sequence of steps.

Step 0*:* We introduce parameter Δ by specifying its value as $\Delta = 2 \cdot \sigma_{\max}$. We mark the indexes of epochs with function $flag_j$ as nulls: $flag_j = 0$; j = 1, ..., N.

Arrange the array y_j in the ascending order and renumber it by placing the indexes in the ascending order. Denote the resulting array as Y_j. For simplicity of

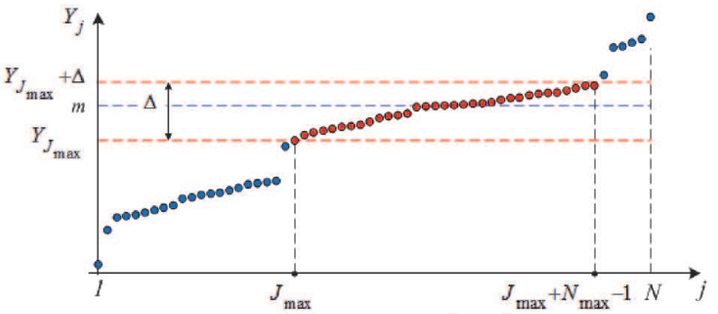

Figure 11.
Searching for the maximum of the density of values of the array Y_j.

logic, we suppose that all Y_j are different. Therefore, we have: $Y_1 < Y_2 < \ldots < Y_N$. Denote the corresponding permutation of the values of function flag_j as FLAG_j.

Step 1: On this step, we are looking for the maximum density of array values y_j. We consider the segments $[Y_j, Y_j + \Delta]$ of length Δ and look for the one that contains the largest number of Y_i values. Here, only those indexes i and j from the range of $1 \le i, j \le N$ are considered for which $FLAG_i = FLAG_j = 0$. The purpose of the ordering of the original array consists in improving the effectiveness of the maximum density search procedure. It can be shown that the number of comparisons required when searching for the above segment does not exceed 2 N in the case of the ordered array. For an unordered array, the number of comparisons is evaluated as N^2.

Step 2: Let $[Y_{J_{max}}, Y_{J_{max}} + \Delta]$ be the segment found in the previous step, containing N_{max} values of Y_j (see **Figure 11**). We calculate the mean m of these numbers Y_j.

$$m = N_{max}^{-1} \cdot \sum_{j=J_{max}}^{J_{max}+N_{max}-1} Y_j, \tag{67}$$

$$Y_j \in [Y_{J_{max}}, Y_{J_{max}} + \Delta]. \tag{68}$$

Step 3: Cluster searching. The values y_j that are clustered about the mean m found in the previous step can pertain to the epochs scattered along the time axis in chaotic order. The task of this step is to define an accumulation of such points, a so-called cluster, which we define as follows:

Definition 3. *We designate as an (m, Δ) cluster the set of points (epochs) of the time axis, the indexes j of which belong to the segment [k,l] $(1 \le k < l \le N)$ and for which the following requirements are satisfied:*

a. all points of the segment [k, l] are marked by the same value: $flag_k = \ldots = flag_l$.

b. at the points j= k, l of the left and right boundaries of the segment, this inequality is satisfied:

$$\left| y_j - m \right| \le \Delta. \tag{69}$$

c. the amount of points at which (69) is satisfied is no less than the present value of MINOBS;

d. the number of consecutive points in which (69) is not satisfied does not exceed the predefined value MAXGAP (e.g., 5);

e. the value of k cannot be reduced while maintaining the requirements of a–d; and

f. the value of l cannot be incremented while maintaining the requirements of a–d.

This definition illustrates in **Figure 12**.

It is understood that for specified values of m and Δ, there might be one, several, or no (m, Δ) clusters. Searching for clusters is performed by sequential checking of the satisfaction of inequality (69) for all $j = 1, \ldots, N$ for which $\text{flag}_j = 0$. Note that inequality (69) is satisfied for at least $N_{\max}$ values of j. In fact, from expressions (67) and (68), we derive the following equations:

$$Y_{J_{\max}} \le m \le Y_{J_{\max}} + \Delta, \tag{70}$$

and since the arrays y_j and Y_j differ only by permutation, it follows from Eq. (68) that the inequalities

$$Y_{J_{\max}} \le y_j \le Y_{J_{\max}} + \Delta. \tag{71}$$

are fulfilled for $N_{\max}$ index j. Inequality (69) is satisfied also for all these indexes, as follows from Eqs. (70) and (71).

If cluster was found, then we mark all points of it as 1, and then, we repeat the cluster search procedure. If even just one cluster has been found at this step, we transfer to Step 1. If not even one cluster has been found, then the search for clusters is complete and we transfer to Step 4.

Step 4: If even just one cluster has been found, we transfer to Step 5, and otherwise: (a) all points of the segment $[1; N]$ are marked as outliers and removed from further processing and (b) the operation of the algorithm terminates.

Step 5: Search for individual jumps in clusters. Let us assume that $n \ge 1$ clusters have been found:

1. $[k_1, l_1]$ is the (m_1, Δ) cluster

...

n. $[k_n, l_n]$ is the (m_n, Δ) cluster.

In each of the clusters that are found, outliers and 1-size jumps are possible. This follows immediately from the inequalities in (69) and the preestablished value $\Delta = 1.2$.

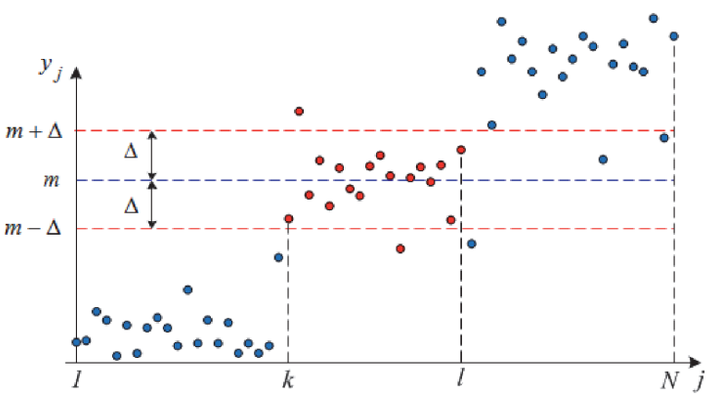

Figure 12.
Epochs with indexes $k \le j \le l$ form (m, Δ) cluster.

Substep 5.1: In detecting a 1-size jump in a cluster, we use modified Springer algorithm (see Refs. [9, 10]) combined with the proposed in Section 3 algorithm that executes a search for the optimal solution with a minimum quantity of defective data.

Substep 5.2: We find all epochs J_p and values $\Delta n_{5,J_p} \triangleq n_{5,J_p} - n_{5,J_p-1}$ ($p = 1, \ldots, n$) of the jumps inside the clusters. The possible values for $\Delta n_{5,J_p}$ are 0, ± 1.

Substep 5.3: We repair the data by the value of each found jump, using the formula

$$y_j^{(p)} = \begin{cases} y_j^{(p-1)}; \ j < J_p \\ y_j^{(p-1)} - \Delta n_{5,J_p}; \ J_p \le j < N \end{cases}; \quad p = 1, .., n, \tag{72}$$

where $y_j^{(0)} = y_j$.

Substep 5.4: We rename: $y_j = y_j^{(n)}$.

Step 6: Marking of points outside clusters. All points outside of the found clusters (if any) are marked as outliers.

Step 7: Ordering of clusters. We renumber the clusters so that they are placed left to right on the time axis. For the ordered set of clusters $[k_p, l_p]$, $p = 1, \ldots, n$, the conditions $1 \le k_1 < l_1 < k_2 < l_2 < \ldots < k_n < l_n \le N$ are satisfied.

Step 8: Data screening within clusters and improving the mean values of y_j.

Substep 8.1: In accordance with the algorithm proposed in Section 3, we perform screening from outliers in each of the n clusters.

Substep 8.2: For each of the clusters cleaned of outliers, we determine the modified mean values m_p^*.

Step 9: Jumps between clusters. It follows from the description presented above that the remaining jumps in the data y_j are on boundaries between clusters. If only one cluster is found, the algorithm is terminated: no additional jumps are detected, and the data are cleaned of outliers. If more than one cluster is found ($n > 1$), then the epochs $j_1, \ldots, j_{n-1}$ of the jumps will be the coordinates of the left-hand boundaries of clusters, beginning from the second: $j_1 = k_2, \ldots, j_{n-1} = k_n$. The values of jumps $\Delta n_{5,j_p}$ are found in this case as rounded to the nearest integer of the difference of mean values of adjacent clusters:

$$\Delta n_{5,j_p} = NINT\left(m_{p+1}^* - m_p^*\right), p = 1, \ldots, n-1. \tag{73}$$

Step 10: Repair data. We delete the jumps between clusters using formula analogous to (72):

$$y_j^{(p)} = \begin{cases} y_j^{(p-1)}; \ j < j_p \\ y_j^{(p-1)} - \Delta n_{5,J_p}; \ j_p \le j < N \end{cases}; \quad p = 1, \ldots, n-1, \tag{74}$$

where $y_j^{(0)} = y_j$.

We rename: $y_j = y_j^{(n-1)}$. The algorithm is terminated.

10. Numerical calculations using algorithms presented in Sections 8 and 9

We present here the results of testing the proposed algorithm using real data obtained by the JOZ2 station, which is part of the IGS network [2]. These data are

included in the distribution set of the installation software package [3]. Testing was carried out for data obtained from GPS satellite with number PRN = 13 for 2010, day 207. **Figure 13** shows the Melbourne-Wübbena combination values in the

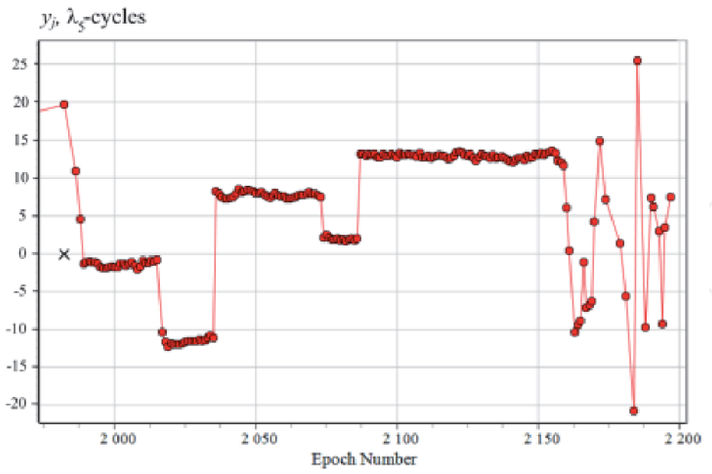

Figure 13.
*Values of the Melbourne-Wübbena combination for the JOZ2 station (*PRN = *13 for 2010, day of year = 207).*

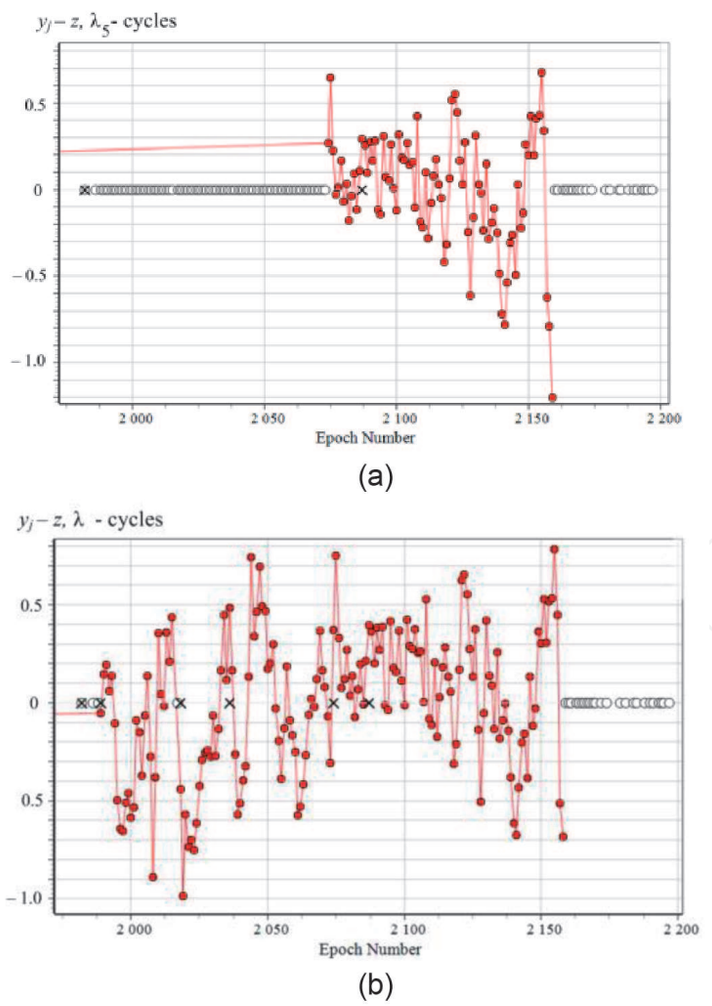

Figure 14.
(a) Deviations of the values of the Melbourne-Wübbena combination from the mean after detection and elimination of jumps from the data using the algorithm from Ref. [3]. (b) Deviations of the values of the Melbourne-Wübbena combination from the mean after detection and elimination of jumps from the data using the proposed algorithm in Section 9.

107 min time interval (N = 215). The number j of time epochs counted from the beginning of 24-h day with interval of 30 seconds is plotted along the horizontal axis. The combination values y_j expressed in cycles with wavelength λ_5 are plotted along the vertical axis.

Figure 14a and **b** presents the values of the deviations from the mean value z of data after cycle slip repair procedure, by using the algorithm applied in Ref. [3] (see **Figure 14a**) and the proposed algorithm (**Figure 14b**). The values y_j–z in cycles of λ_5 are plotted along the vertical axes, and the number j of epochs is plotted along the horizontal axis. Epochs in which the measurement data were rejected are marked by light circles. In the first case (see **Figure 14a**), 111 of the conducted measurements or 51% of the total number of data was discarded. In the second case (see **Figure 14b**), 29 of these measurements were rejected (13%), which are almost 38% less than in the previous computation. The epochs of detected jumps are marked by daggers.

11. Conclusion

This chapter presents several effective and stable algorithms for processing data received from GNSS receivers. These data form the basis of almost all engineering applications in the field of computational geo-dynamics and navigation and cadastral survey and in numerous fundamental research works as well. The accuracy of the results obtained is significantly influenced by the quality of the data used in the calculations. In particular, the presence of rough measurements (outliers) in the observation data can significantly reduce the accuracy of the calculations carried out. One of the tasks at the preliminary stage of data processing is reliable detection and removal of rough measurements from the series of measured data with minimum amount of rejected data. The so-called optimal solution, introduced in the chapter, made it possible to detect and eliminate outliers from observed data minimizing the number of rejected measurements. In addition, it is assumed that the data may contain a trend as an unknown function of time. The strategy for determining of the trend is depending on the physical process in question under an assumption that the trend is a continuous function of time. The efficiency of the search is definitely influenced by the choice of the function class from which the trend is searched. In this chapter, we considered the class of power polynomials, but in some cases, this choice may not lead to the expected result. It may require, for example, a class of trigonometric functions to find a suitable trend. The automatic search for the best functional class, together with the strategy of effectively finding an unknown trend, against the background of random noise and outliers, is a complex task for future research.

IntechOpen

Author details

Igor V. Bezmenov
"VNIIFTRI", Mendeleevo, Moscow region, Russian Federation

*Address all correspondence to: bezmenov@vniiftri.ru

IntechOpen

References

[1] Perov AI, Kharisov VN. GLONASS: Principles of Construction and Operation. 4th ed. Moscow: Radiotekhnika; 2010

[2] International GNSS Service. Available from: wwv.igs.org/network [Accessed: 4 April 2018]

[3] Dach R, Andritsch F, Arnold D, et al. In: Dach R et al, editors. Bernese GPS Software, Ver. 5.2. Astronomical Institute, University of Bern; 2015. DOI: 10.7892/boris.72297. Available from: www.bernese.unibe.ch [Accessed: 20 May 2020]

[4] Melbourne WG. The case for ranging in GPS based geodetic systems. In: Goad C, editor. Proceedings of the 1st International Symposium on Precise Positioning with the Global Positioning System. Rockville, Maryland: US Department of Commerce; 1985. pp. 373-386

[5] Wubbena G. Software developments for geodetic positioning with GPS using TI 4100 code and carrier measurements. In: Goad C, editor. Proceedings First International Symposium on Precise Positioning with the Global Positioning System. Rockville, Maryland: US Department of Commerce; 1985. pp. 403-412

[6] Sanz Subirana J, Juan Zornoza M, Hernández-Pajares M. Detector Based in Code and Carrier Phase Data: The Melburne–Wübbena Combination. ESA Navipedia. Available from: https://gssc.esa.int/navipedia/index.php/Detector_based_in_code_and_carrier_phase_data:_The_Melboume-Wübbena_combination [Accessed: 24 January 2019]

[7] Bezmenov IV, Naumov AV, Pasynok SL. An effective algorithm for elimination of outliers from data measurements of global navigation satellite systems. Measurement Techniques. 2018;**61**:878-884. DOI: 10.1007/s11018-018-1518-y

[8] Blewitt G. An automatic editing algorithms for GPS data. Geophysical Research Letters. 1990;**17**(3):199-202

[9] Springer TA. Modeling and validating orbits and clocks using the global positioning system. In: Geodätisch-Geophysikalische Arbeiten in der Schweiz. Vol. 60. Zurich: Institute. Geodesy and Photogrammetry, ETH Zurich; 2000

[10] Bezmenov IV, Blinov IY, Naumov AV, et al. An algorithm for cycle-slip detection in a Melbourne–Wübbena combination formed of code and carrier phase GNSS measurements. Measurement Techniques. 2019;**62**: 415-421. DOI: 10.1007/s11018-019-01639-5