IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 16, 2021, accepted May 28, 2021, date of publication June 3, 2021, date of current version June 15, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3085704

Technique to Adjust Adaptive Digital Filter Coefficients in Residue Number System Based Filters

DMITRII KAPLUN[1], (Member, IEEE), ALEXANDER VOZNESENSKY[1], ALEXANDER V. VELIGOSHA[2], IGOR A. KALMYKOV[3], AND KANDARPA KUMAR SARMA[4], (Senior Member, IEEE)
[1]Department of Automation and Control Processes, Saint Petersburg Electrotechnical University "LETI," 197376 Saint Petersburg, Russia
[2]Department 52, Military Academy of the Strategic Missile Forces named after Peter the Great, 142210 Serpukhov, Russia
[3]Department of Information Security of Automated Systems, North-Caucasus Federal University, 355009 Stavropol, Russia
[4]Department of Electronics and Communication Engineering, Gauhati University, Guwahati 781014, India

Corresponding author: Alexander Voznesensky (asvoznesensky@etu.ru)

This work was supported by the Russian Science Foundation under Grant 19-19-00566.

ABSTRACT The paper discusses adaptive filtering using Least Mean Square (LMS) and Recursive Least Square (RLS) algorithms. An algorithm for adjusting the coefficients of an adaptive digital filter in the Residue Number System and a procedure of developed algorithm applying depending on filter length and signal length are proposed. Mathematical modeling of the considered algorithms is performed. Examples are presented to demonstrate how the proposed technique can help the designer in the adjustment of the filter coefficients without the need for extensive trial-and-error procedures. The analysis of the denoising quality and computational complexity is made. Synthetic and real data (earthquake recording) were used while testing. The proposed algorithm surpasses the existing ones like LMS and RLS, and their modifications in a number of parameters: adaptation (denoising) quality, ease of implementation, execution time. The main difference between the developed algorithm is the sequential adaptation of each coefficient with zero error. In the known algorithms, the entire vector of coefficients is iteratively adapted, with some specified accuracy. The iterations (steps) number is determined by the input signal length for all algorithms.

INDEX TERMS Adaptive digital filter, residue number system, least mean square algorithm, recursive least square algorithm, denoising quality, computational complexity.

I. INTRODUCTION

Using adaptive digital signal processing (DSP) techniques in modern communication systems, the enhancement of the primary quality functioning indicators [1], [2] is achieved.

This is especially evident with the digital filtering of processed signals. The primary objective of filtering is to decompose the complex structure of the input signal into components to remove an unnecessary element that does not carry useful information.

In many applications that use modern methods and algorithms for digital signal processing, adaptive digital filters (ADF) are widely used [3], [4], including a digital filter and an adaptation system.

The associate editor coordinating the review of this manuscript and approving it for publication was Tianhua Xu.

The most widely used in ADFs are digital filters with a finite impulse response (FIR) [5]. Adaptive filtering is used in acoustic systems [6], image processing [7], equalizers designed to level the characteristics of communication channels [8], systems, using adaptive antenna arrays [9].

The efficiency of the ADF is largely determined by the algorithm, which allows adaptation of the coefficients of the FIR-filter to obtain the reference signal. At the same time, the algorithm for adjusting the coefficients vector of the FIR filter must, at a given step, implement the adjustment procedure with minimal software and hardware costs [5].

In order to increase the speed of signal processing in different applications, the use of residue number system (RNS) is justified [10]. The use of the RNS enables the enhancement of digital signal processing systems (including digital filters) performance and reducing hardware costs [11] due to the parallel and independent processing of small bit-width residues when performing arithmetic operations like addition, subtraction, and multiplication [12], [13]. The disadvantage of the RNS is the high computational complexity when performing non-modular operations, which include division, sign detection and numbers comparison [14], [15]. These limitations exist because the RNS is a non-positional number system, and numbers magnitude comparison in the RNS form is impossible, so the division operation consists of a magnitude comparison operation that is also a problematic operation. Attempts to solve this problem are still being undertaken by various scientists in different directions [16]–[18], but there is still no universal solution suitable for any tasks. As a result, there are currently no coefficients adjustment algorithms for ADF that are implemented in the RNS.

Therefore, the development of a new adaptation algorithm (filter coefficients adjustment) using the RNS and providing specified requirements to adaptation quality and rapidity indicators is a critical task in digital signal processing.

It is known that most coefficient adjustment algorithms of ADF use the optimal Wiener filtering theory. In [19], an algorithm is presented that is based on the search for the minimum of the target function based on the application of the steepest descent method. In this case, the squared error of the ADF output signal is used as the target function. Therefore, for the ADF, it is necessary to find such coefficients values that would make it possible to obtain the value of the output signal with a given accuracy. In other words, the selected filter coefficients should contribute to the maximum approximation of the digital filter output signal to the reference signal.

When applying the Least Mean Square (LMS) algorithm, the adaptive digital filter coefficients are updated recursively. The main advantage of the LMS algorithm is its extreme computational simplicity. To adjust the coefficients at each step, $N + 1$ pairs of addition-multiplication operations are needed to be performed (N is the filter order). The price for simplicity is the slow convergence and high error variance in the steady state [19]–[21].

The Recursive Least Square (RLS) adaptation algorithm allows to eliminate these shortcomings. The basis of this adaptative algorithm is a generalized least square method. In this case, digital filtering is a deterministic analogue of Wiener filtering. The RLS adaptation algorithm implementation is performed in [20], [21]. To adjust the coefficients at each step, $3N^2 + 4N$ pairs of addition-multiplication operations are needed to be performed (N is the filter order). The advantage is fast convergence.

Given the advantages and disadvantages of the considered adaptation algorithms, we can conclude that RNS is suitable to improve performance when adjusting the coefficients and reduce the computational complexity of the tuning procedure [22]. These results can be used for building effective parallel computational systems [13] based on computers with parallel structure like FPGA [17], [18] and GPU [23], [24].

The disadvantage of the ADF operating in the RNS is the need to perform the conversion from the RNS to the traditional binary number system (BNS) at each iteration. This is due to the fact that in order to perform the comparison operation it is necessary to go to the BNS. When processing in a large range, it is also necessary to scale the calculations intermediate results in a modular digital filter in order to exclude going beyond the dynamic range [10], [25].

First attempts to provide the ADF operating in the RNS were presented in [26], [27] but their attention was mainly paid to the efficient conversions BNS to RNS and vice versa, as well as the efficient hardware implementation of this conversion.

A sequential algorithm for adjusting the filter coefficients operating in the RNS is presented in this paper. To adjust each coefficient, it is necessary to perform one subtraction operation, one multiplication operation and one addition operation modulo the RNS, i.e., recalculation time is proportional to the filter order. A fundamentally new technique (Fig. 1) is provided that surpasses the existing ones like LMS and RLS and their modifications in a number of parameters: adaptation (denoising) quality, ease of implementation, execution time etc. The idea of the algorithm firstly appeared in [28] but it was a minor part of the work. Since that time, the algorithm has been corrected, updated and extended to the complete technique and is presented in this work.

The rest of the paper is organized as follows. Section II provides a detailed description of the developed algorithm. Section III presents an algorithm operation example and some comments. Section IV provides proposed algorithm simulation and adaptation algorithms comparison in terms of denoising quality and execution time. Discussion is presented in Section V. The conclusion of the paper is reported in Section VI.

II. DEVELOPING THE ADAPTATION ALGORITHM

The paper presents an adaptation algorithm using RNS without a reverse conversion. This algorithm (Fig. 1) is based on an iterative algorithm for sequentially calculating filter coefficients. This algorithm contains $N + 1$ steps, whose number coincides with the length of the digital filter L.

Let the RNS moduli set be $\{m_1, m_2, \ldots, m_i, \ldots m_k\}$.

Then the response of the FIR-filter operating in the modular code when processing n-th sample of the input signal is determined by:

$$y_i(n) = \left(\sum_{s=0}^{N} w_i(s)x_i(n-s)\right) \bmod m_i. \tag{1}$$

Here $x = [x(0),\ x(1), \ldots, x(N)]$ is the input signal, $w = [w(0),\ w(1), \ldots, w(N)]$ are the filter coefficients, $y_i(n) \equiv y(n) \bmod m_i$, $x_i(n-s) \equiv x(n-s) \bmod m_i$ and $w_i(s) \equiv w(s) \bmod m_i$; $i = 1, 2, \ldots, k$.

The digital filter response will differ from the reference signal $d(n)$ by the error value. In this case, the filter response

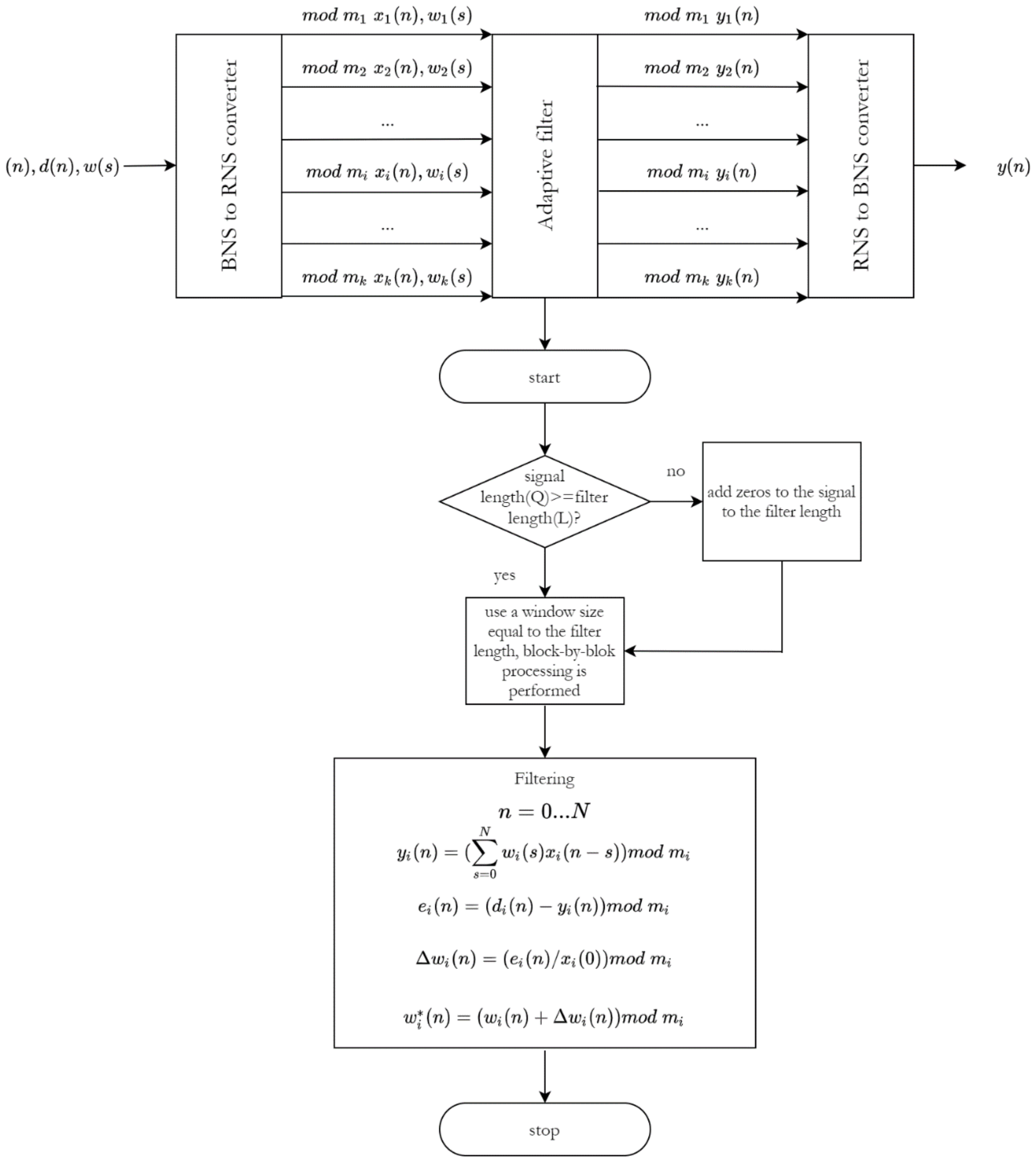
$(n), d(n), w(s)$
BNS to RNS converter
$mod\ m_1\ x_1(n), w_1(s)$
$mod\ m_2\ x_2(n), w_2(s)$
...
$mod\ m_i\ x_i(n), w_i(s)$
...
$mod\ m_k\ x_k(n), w_k(s)$
Adaptive filter
$mod\ m_1\ y_1(n)$
$mod\ m_2\ y_2(n)$
...
$mod\ m_i\ y_i(n)$
...
$mod\ m_k\ y_k(n)$
RNS to BNS converter
$y(n)$
start
signal length(Q)>=filter length(L)?
no
add zeros to the signal to the filter length
yes
use a window size equal to the filter length, block-by-blok processing is performed
Filtering
$n = 0...N$
$y_i(n) = (\sum_{s=0}^{N} w_i(s)x_i(n-s))mod\ m_i$
$e_i(n) = (d_i(n) - y_i(n))mod\ m_i$
$\Delta w_i(n) = (e_i(n)/x_i(0))mod\ m_i$
$w_i^*(n) = (w_i(n) + \Delta w_i(n))mod\ m_i$
stop

FIGURE 1. Algorithm flowchart.

error will be:

$$e_i(n) = (d_i(n) - y_i(n)) \bmod m_i. \tag{2}$$

Here $d_i(n) \equiv d(n) \bmod m_i$; $i = 1, 2, \dots, k$.

The following sections provide an outline of the steps involved.

The first step $n = 0$.

In this case, the digital filter response will be determined from:

$$y_i(0) = \left(\sum_{s=0}^{N} w_i(s)x_i(0-s)\right) \bmod m_i. \tag{3}$$

Then the adaptive digital filter response error is expressed as:

$$e_i(0) = (d_i(0) - \sum_{s=0}^{N} w_i(s)x_i(0-s)) \bmod m_i; \quad i = 1\dots k; \tag{4}$$

where $w\,(0)$ represents ADF random zero coefficient.

To adapt the ADF coefficients, the convergence accuracy of the output signal $y(n)$ and the reference signal $d(n)$ is set. This operation is quite easily implemented in the BNS. For modular codes, it is necessary to perform a non-modular operation, for example, to convert numbers from the RNS into a BNS and compare the calculated error of the digital filter response with a specified limit. However, this requires significant hardware and software costs.

To eliminate this drawback, it is proposed to use an approximate algorithm for comparing numbers represented in the RNS [29], [30]. An approximate method for comparing modular numbers is based on the use of relative values $\frac{A}{M}$ of the analyzed numbers A to the full dynamic range $M = \prod_{i=1}^{k} m_i$ of the RNS. Then the relative value $\frac{A}{M}$ is defined as:

$$\frac{A}{M} = \left| \sum_{i=1}^{k} \frac{\left|M_i^{-1}\right|_{m_i}}{m_i} \alpha_i \right|_M = \left| \sum_{i=1}^{k} k_i \alpha_i \right|_M , \tag{5}$$

$k_i = \frac{\left|M_i^{-1}\right|_{m_i}}{m_i}$ – the RNS constants.

Comparing two numbers A and B according to (6) in this case is performed as follows:

Reverse conversion

RNS $\rightarrow$ BNS $A = (\alpha_1, \alpha_2, \ldots, \alpha_k) \rightarrow \frac{A}{M}$.

Reverse conversion

RNS $\rightarrow$ BNS $B = (\beta_1, \beta_2, \ldots, \beta_k) \rightarrow \frac{B}{M}$.

Comparison $\frac{A}{M} ? \frac{B}{M}$.

Note that the expression value does not exceed 1, since the number A represented in the RNS cannot be greater than the range M. Moreover, the integer part of the expression $\sum_{i=1}^{k} \frac{\left|M_i^{-1}\right|_{m_i}}{m_i} \alpha_i$ is the rank of the number A. The rank of the number A is a non-positional characteristic, showing how many times the dynamic range M has been covered during the reverse conversion $(\alpha_1, \alpha_2, \ldots, \alpha_k)$ from the RNS to its binary representation.

From the RNS moduli sets we choose the lower moduli set, which product will determine a dynamic range of the RNS:

$$M_{acc} = \prod_{i=1}^{z} m_i. \tag{6}$$

Then the remaining $k - z$ moduli sets will relate to the RNS control moduli sets [19], [20]. If the adaptive digital filter response error value, presented in the RNS $e(n) = (e_1(n),\ e_2(n), \ldots,\ e_i(n), \ldots, e_k(n))$, is less than the specified accuracy, that is $e(n) < M_{acc}$, it can be said that this code belongs to the accuracy range M_{acc}. This situation means that the calculated ADF response error can be considered acceptable.

Otherwise, when $e(n) \geq M_{acc}$ the calculated digital filter response error has not reached the required accuracy. Therefore, it is necessary to fine-tune coefficients. To perform an algorithm for approximate the RNS numbers comparison, calculate the constant value from:

$$\frac{M_{acc}}{M} = \left| \sum_{i=1}^{k} \frac{\left|M_i^{-1}\right|_{m_i}}{m_i} b_i \right|_{m_i} . \tag{7}$$

The obtained value of the adaptive digital filter response error, presented in the RNS as:

$e(n) = (e_1(n),\ e_2(n), \ldots,\ e_i(n), \ldots, e_k(n))$, is used to calculate the approximate value given by the expression

$$\frac{e(n)}{M} = \left| \sum_{i=1}^{k} \frac{\left|M_i^{-1}\right|_{m_i}}{m_i} e_i(n) \right|_{m_i} . \tag{8}$$

If the inequality is true:

$$\frac{e(n)}{M} < \frac{M_{acc}}{M}, \tag{9}$$

then the required coefficients adaptation accuracy is achieved. Otherwise, coefficients adjustment procedure continues.

Obviously, the ADF response error is determined by the incorrect value $w(0)$. We calculate the correction value of the zero coefficient according to the expressions:

$$\Delta w_i(0) = \left(e_i(0) \big/ x_i(0)\right) \bmod m_i; \quad i = 1 \ldots k. \tag{10}$$

Then coefficient value after adjustment will be determined:

$$w_i^*(0) = (w_i(0) + \Delta w_i(0)) \bmod m_i. \quad i = 1 \ldots k. \tag{11}$$

Thus, as a result of the first stage of the proposed iterative sequential algorithm, the true value of zero coefficient is obtained, which is presented in the RNS as:

$$w^*(0) = (w_1^*(0),\ w_2^*(0), \ldots, w_i^*(0), \ldots, w_k^*(0)).$$

The second step $n = 1$.

The second stage ($n = 1$) includes similar steps for estimating the error value, calculating the correction value of the first coefficient and its value. As a result of the second stage, we obtained the true value of the first coefficient, which is represented in the RNS $w^*(1) = (w_1^*(1),\ w_2^*(1), \ldots, w_i^*(1), \ldots, w_k^*(1))$. Thus, as a result of two stages of coefficient adjustments, two adapted coefficients $w^*(0)$ and a filter $w^*(1)$ were obtained.

The last step $n = N$.

In a similar way, the values of all the remaining coefficients are corrected; at the $N + 1$ – stage, the last coefficient is adapted.

In this case, the digital filter response will be determined:

$$y_i(N) = \left(\sum_{s=0}^{N-1} w_i^*(s) x_i(N-s) + w_i(N) x_i(0) \right) \bmod m_i. \tag{12}$$

Then the adaptive digital filter response error is:

$$e_i(N) = \left[d_i(N) - \left(\sum_{s=0}^{N-1} w_i^*(s) x_i(N-s) + w_i(N) x_i(0) \right) \right] \times \bmod m_i; \quad i = 1 \ldots k. \tag{13}$$

We calculate the correction value of the N-th coefficient according to:

$$\Delta w_i(N) = \left(e_i(N) \big/ x_i(0)\right) \bmod m_i; \quad i = 1 \ldots k. \tag{14}$$

TABLE 1. Modular source data codes.

Modules	Input samples x(s)			Coefficients w(s)			Reference signal d(s)		
	61	41	29	-150	311	1601	-10401	12935	107610
2	1	1	1	0	1	1	1	1	0
5	1	1	4	0	1	1	4	0	0
7	5	6	1	4	3	5	1	6	6
11	6	8	7	4	3	6	5	10	8
13	9	2	3	6	12	2	12	0	9
19	4	3	10	2	7	5	11	15	13
23	15	18	6	11	12	14	18	9	16
29	3	12	0	24	21	6	10	1	20
31	30	10	29	5	1	20	15	8	9
37	24	4	29	35	15	10	33	22	14

Then the value of the N-th coefficient after adjustment will be determined:

$$w_i^*(N) = (w_i(N) + \Delta w_i(N)) \bmod m_i. \quad i = 1 \ldots k. \tag{15}$$

Here: $w(N)$ is the random coefficient of the ADF.

III. MODELING THE OPERATION OF THE ALGORITHM IN THE RNS

Let the RNS moduli set be $m_1 = 2, m_2 = 5, m_3 = 7, m_4 = 11, m_5 = 13, m_6 = 19, m_7 = 23, m_8 = 29, m_9 = 31, m_{10} = 37$.

This moduli set has been proposed for the implementation of a modular digital filter with 16-bit coefficients that processes 16-bit input data. Since when the operation of multiplying two 16-bit operands is performed, the result becomes 32-bit, and the rounding or truncation operation is not applied to the intermediate results. So, an extended moduli set has been proposed to obtain the correct resulting response. The dynamic range of this moduli set is equal $M = 145504669310$.

Choose the ADF order $N = 2$. Then the adaptive digital filter response operating in the RNS is determined by the expression:

$$y_i(n) = \left(\sum_{s=0}^{2} w_i(s)x_i(n-s)\right) \bmod m_i, \tag{16}$$

where $n = 0, 1, 2; x = \{x(0), x(1), x(2)\}$ – the input vector; $w = \{w(0), w(1), w(2)\}$ represents the digital filter coefficients; $y_i(n) \equiv y(n) \bmod m_i$; $x_i(n-s) \equiv x(n-s) \bmod m_i$; $w_i(s) \equiv w(s) \bmod m_i$; $i = 1, 2, \ldots, k$. Let the input vector be given by samples $x = \{61, 41, 29\}$. The reference signal is determined by a set of samples $d = \{-10401, 12935, 107610\}$. As the coefficients select the following values $w = \{-150, 311, 1601\}$.

In the example, the impulse response has both positive and negative coefficients. To represent negative numbers in the RNS, it was proposed to divide the range into two parts. Positive numbers will be contained in the first half (from 0 to $P/2 - 1$), and negative numbers will be located in the second half of the range (from $P/2$ to $P - 1$). So, the number in the selected moduli set is represented as: $-150 = P - 150 = 145504669160 = (0, 0, 4, 4, 6, 2, 11, 24, 5, 35)$.

Table 1 presents the code combinations of the selected data.

The first step $n = 0$.

We calculate the response of the FIR-filter operating in the RNS $i = 1, \ldots, k$ using:

$$y_i(0) = \left(\sum_{s=0}^{2} w_i(s)x_i(0-s)\right) \bmod m_i$$
$$= x_i(0)\, w_i(0) \bmod m_i.$$

Since the ADF coefficients are chosen arbitrarily, the response error of the digital filter $y(0)$ from the reference signal $d(0)$ is:

$$e_i(0) = (d_i(0) - y_i(0)) \bmod m_i.$$

We calculate the correction value of the zero coefficient according to:

$$\Delta w_i(0) = \left(e_i(0) \big/ x_i(0)\right) \bmod m_i.$$

Then the adapted value of the zero coefficient will be equal to:

$$w_i^*(0) = (w_i(0) + \Delta w_i(0)) \bmod m_i.$$

Consider the execution of an algorithm for a module $m_7 = 23$. In this case, the digital filter response is equal to:

$$y_7(0) = x_7(0)w_7(0) \bmod 23 = (15 \cdot 11) \bmod 23 = 4.$$

The digital filter response error $y_7(0)$ from the reference signal $d_7(0)$ is:

$$e_7(0) = (d_7(0) - y_7(0)) \bmod 23 = (18 - 4) \bmod 23 = 14.$$

Inverse element value is given as:

$$x_7^{-1}(0) \bmod 23 = 20.$$

TABLE 2. **Zero ADF coefficient adaptation.**

Modules	x(0)	w(0)	y(0)	d(0)	e(0)	$x^{-1}(0)$	Δw(0)	w*(0)	y*(0)
2	1	0	0	1	1	1	1	1	1
5	1	0	0	4	4	1	4	4	4
7	5	4	6	1	2	3	6	3	1
11	6	4	2	5	3	2	6	10	5
13	9	6	2	12	10	3	4	10	12
19	4	2	8	11	3	5	15	17	11
23	15	11	4	18	14	20	4	15	18
29	3	24	14	10	25	10	18	13	10
31	30	5	26	15	20	30	11	16	15
37	24	35	26	33	7	17	8	6	33

We calculate the correction value of the zero coefficient according to the expression:

$$\Delta w_7(0) = \left(e_7(0)\big/x_7(0)\right) \bmod 23 = (14 \cdot 20) \bmod 23 = 4.$$

Then the adapted value of the zero coefficient will be equal to:

$$\begin{aligned} w_7^*(0) &= (w_7(0) + \Delta w_7(0)) \bmod m_7 \\ &= (11 + 4) \bmod 23 = 15. \end{aligned}$$

We calculate the adapted response as:

$$y_7^*(0) = x_7(0)w_7^*(0) \bmod 23 = (15 \cdot 15) \bmod 23 = 18.$$

The results of coefficient $w\,(0)$ adaptation in the RNS are presented in Table 2.

The last column of the Table 2 presents the results of the digital filter response with the adjusted value $w^*\,(0)$.

The second step $n = 1$.

At this stage, we have a set of coefficients like $w = \{w^*\,(0)\,, 311, 1601\}$ among which there is an adapted set given as $w^*\,(0) = (1, 4, 3, 10, 10, 17, 15, 13, 16, 6)$.

Let us calculate the response of the FIR filter implemented in the RNS: $i = 1, \ldots, k$:

$$\begin{aligned} y_i\,(1) &= \left(\sum_{s=0}^{2} w_i(s)x_i(1-s)\right) \bmod m_i \\ &= \left(w_i^*\,(0)\,x_i\,(1) + w_i\,(1)\,x_i\,(0)\right) \bmod m_i. \end{aligned}$$

Since the ADF coefficients are chosen arbitrarily, the digital filter response error $y\,(1)$ from the reference signal $d\,(1)$ is:

$$e_i(1) = (d_i(1) - y_i(1)) \bmod m_i.$$

We calculate the correction value of the first coefficient according to:

$$\Delta w_i(1) = \left(e_i(1)\big/x_i(0)\right) \bmod m_i.$$

Then the adapted value of the first coefficient will be equal to:

$$w_i^*(1) = (w_i(1) + \Delta w_i(1)) \bmod m_i.$$

Consider the execution of an algorithm for a module $m_7 = 23$.

In this case, the digital filter response is equal to:

$$\begin{aligned} y_7(1) &= (w_7^*(0)x_7(1) + w_7(1)x_7(0)) \bmod m_7 = \\ &= (15 \cdot 18 + 12 \cdot 15) \bmod 23 = 13. \end{aligned}$$

The digital filter response error $y_7\,(1)$ from the reference signal $d_7\,(1)$ is:

$$e_7(1) = (d_7(1) - y_7(1)) \bmod 23 = (9 - 13) \bmod 23 = 19.$$

Inverse element value is given as:

$$x_7^{-1}(0) \bmod 23 = 20.$$

We calculate the correction value of the first coefficient according to:

$$\Delta w_7(1) = \left(e_7(1)\big/x_7(0)\right) \bmod 23 = (19 \cdot 20) \bmod 23 = 12.$$

Then the adapted value of the first coefficient will be equal to:

$$\begin{aligned} w_7^*(1) &= (w_7(1) + \Delta w_7(1)) \bmod m_7 \\ &= (12 + 12) \bmod 23 = 1. \end{aligned}$$

We calculate the adapted response as:

$$\begin{aligned} y_7^*(1) &= (w_7^*(0)x_7(1) + w_7^*(1)x_7(0)) \bmod m_7 = \\ &= (15 \cdot 18 + 1 \cdot 15) \bmod 23 = 9. \end{aligned}$$

The results of coefficient $w\,(1)$ adaptation in the RNS are presented in Table 3.

The third step $n = 2$.

At this stage, we have a set of coefficients

$$w = \left\{w^*\,(0)\,, w^*\,(1)\,, 1601\right\}.$$

TABLE 3. First ADF coefficient adaptation.

Modules	x(1)	w*(0)	x(0)	w(1)	y(1)	d(1)	e(1)	x-1(0)	Δw(1)	w*(1)
2	1	1	1	1	0	1	1	1	1	0
5	1	4	1	1	0	0	0	1	0	1
7	6	3	5	3	5	6	1	3	3	6
11	8	10	6	3	10	10	0	2	0	3
13	2	10	9	12	11	0	2	3	6	5
19	3	17	4	7	3	15	12	5	3	10
23	18	15	15	12	13	9	19	20	12	1
29	12	13	3	21	16	1	14	10	24	16
31	10	16	30	1	4	8	4	30	27	28
37	4	6	24	15	14	22	8	17	25	3

where in:

$$w^*(0) = (1, 4, 3, 10, 10, 17, 15, 13, 16, 6)$$

and

$$w^*(1) = (0, 1, 6, 3, 5, 10, 1, 16, 28, 3),$$
$$w(2) = 1601 = (1, 1, 5, 6, 2, 5, 14, 6, 20, 10).$$

Let us calculate the response of the FIR-filter, operating in the RNS, $i = 1, \ldots, k$:

$$y_i(2) = \left(\sum_{s=0}^{2} w_i(s)x_i(2-s)\right) \bmod m_i =$$
$$= (w_i^*(0)x_i(2) + w_i^*(1)x_i(1) + w_i(2)x_i(0)) \bmod m_i.$$

Since the ADF coefficients are chosen arbitrarily, the digital filter response error $y(2)$ from the reference signal $d(2)$ is:

$$e_i(2) = (d_i(2) - y_i(2)) \bmod m_i.$$

We calculate the correction value of the second coefficient according to:

$$\Delta w_i(2) = \left(e_i(2)\big/x_i(0)\right) \bmod m_i.$$

Then the adapted value of the second coefficient will be equal to:

$$w_i^*(2) = (w_i(2) + \Delta w_i(2)) \bmod m_i.$$

Consider the execution of an algorithm for a module $m_7 = 23$.

In this case, the digital filter response is equal to:

$$y_7(1) = (w_7^*(0)x_7(2) + w_7^*(1)x_7(1) + w_7(2)x_7(0)) \bmod m_7$$
$$= (15 \cdot 6 + 1 \cdot 18 + 14 \cdot 15) \bmod 23 = 19.$$

The digital filter response error $y_7(2)$ from the reference signal is:

$$e_7(2) = (d_7(2) - y_7(2)) \bmod 23 = (16 - 19) \bmod 23 = 20.$$

Inverse element value is given as:

$$x_7^{-1}(0) \bmod 23 = 20.$$

We calculate the correction value of the second coefficient according to:

$$\Delta w_7(2) = \left(e_7(2)\big/x_7(0)\right) \bmod 23 = (20 \cdot 20) \bmod 23 = 9.$$

Then the adapted value of the first coefficient will be equal to:

$$w_7^*(2) = (w_7(2) + \Delta w_7(2)) \bmod m_7$$
$$= (14 + 9) \bmod 23 = 0.$$

We calculate the adapted response:

$$y_7^*(2) = (w_7^*(0)x_7(2) + w_7^*(1)x_7(1) + w_7^*(2)x_7(0)) \bmod m_7$$
$$= (15 \cdot 6 + 1 \cdot 18 + 0 \cdot 15) \bmod 23 = 16.$$

The results of coefficient $w(2)$ adaptation in the RNS are presented in the Table 4.

Thus, after the third stage of the proposed algorithm, the following adapted coefficients, presented in the RNS, were obtained:

- Coefficient $w^*(0) = (1, 4, 3, 10, 10, 17, 15, 13, 16, 6)$.
- Coefficient $w^*(1) = (0, 1, 6, 3, 5, 10, 1, 16, 28, 3)$.
- Coefficient $w^*(2) = (1, 3, 6, 4, 11, 15, 0, 20, 22, 36)$.

Using these coefficients, we have the ADF response in the RNS:

- Response $y^*(0) = (1, 4, 1, 5, 12, 11, 18, 10, 15, 33) = -10401$.
- Response $y^*(1) = (1, 0, 6, 10, 0, 15, 9, 1, 8, 22) = 12935$.
- Response $y^*(2) = (0, 0, 6, 8, 9, 13, 16, 20, 9, 14) = 107610$.

As a result of the application of the developed adaptation algorithm of a digital filter operating in the RNS, samples of a given reference signal were obtained $d = \{-10401, 12935, 107610\}$.

Comments. In the considered example, the case is shown when the signal length Q is equal to the filter length L. If we

TABLE 4. **Second ADF coefficient adaptation.**

Modules	x(2)	w*(0)	x(1)	w*(1)	x(0)	w(2)	y(2)	d(2)	e(2)	$x^{-1}(0)$	Δw(2)	w*(2)
2	1	1	1	0	1	1	0	0	0	1	0	1
5	4	4	1	1	1	1	3	0	2	1	2	3
7	1	3	6	6	5	5	1	6	5	3	1	6
11	7	10	8	3	6	6	9	8	10	2	9	4
13	3	10	2	5	9	2	6	9	3	3	9	11
19	10	17	3	10	4	5	11	13	2	5	10	15
23	6	15	18	1	15	14	19	16	20	20	9	0
29	0	13	12	16	3	6	7	20	13	10	14	20
31	29	16	10	28	30	20	11	9	29	30	2	22
37	29	6	4	3	24	10	19	14	32	17	26	36

use the classical approach [29], [30] to calculate the adaptive filter coefficients when the signal length is longer than the filter length, then we will need to compare the received response of the digital filter with the reference signal. Since we have extended the work into RNS, for this we will have to produce the reverse conversion of the RNS-BNS, do a comparison, and then, if the required error is not achieved, generate further coefficients' adaptation.

As a result, all the RNS advantages in speed will be lost (exp.s (5) – (9)).

To eliminate this drawback, the following steps are proposed (see Adaptive Filter block in Figure.1). Let the filter length L = 15, and the signal length Q = 60, that is $x(0)\ldots x(59)$. We divide the input vector into 4 groups of L = 15 samples.

We have the following groups:

- the first group contains the input signal samples $x\,(0)\ldots x\,(14)$;
- the second group contains the input signal samples $x\,(15)\ldots x\,(29)$;
- the third group contains the input signal samples $x\,(30)\ldots x\,(44)$;
- the fourth group contains the input signal samples $x\,(45)\ldots x\,(59)$.

We take the first group of samples $x\,(0)\ldots x\,(14)$. Using the developed algorithm, we obtain the first group of adaptive filter coefficients $w(0),\ w(1),\ldots,\ w(14)$. These coefficients are stored in memory. The filter has been executed for 15 cycles.

We zero out ADF coefficients and filter content. Next the second group is presented to the input $x\,(15)\ldots x\,(29)$. Using the developed algorithm for this set of samples, we obtain the second group of coefficients $w(0),\ w(1),\ldots,\ w(14)$. These coefficients are stored in memory. The filter has been executed for 15 cycles.

Again, we zero out ADF coefficients and filter content. Next the third group is presented to the input $x\,(30)\ldots x\,(44)$. Using the developed algorithm for this set of samples, we obtain the third group of coefficients $w(0),\ w(1),\ldots,\ w(14)$. These coefficients are stored in memory. The filter has been executed for 15 cycles.

Again, we zero out ADF coefficients and filter content. Next the fourth group is presented to the input $x\,(45)\ldots x\,(59)$. Using the developed algorithm for this set of samples, we obtain the fourth group of coefficients $w(0),\ w(1),\ldots,\ w(14)$. These coefficients are stored in memory. The filter is configured to run for 15 cycles.

As a result, we get four times more filter coefficients. However, we do not have to perform the operation of comparing the output response using the RNS coefficients and the ideal response. In addition, to obtain all the coefficients, we needed only 60 clock cycles (in a conventional ADF this lasts much longer).

The filter works as follows.

The first 15 samples $x\,(0)\ldots x\,(14)$ of the processed signal are fed to the input of the filter. At the same time, the first group of coefficients $w(0),\ w(1),\ldots,\ w(14)$ is selected.

After 15 clock cycles, we get 15 output samples $y\,(0)\ldots y\,(14)$. In this case, we get a truncated linear convolution. (The length of the filter output signal for the first group should be $15+15-1=29$. We leave only the first 15 output samples). The filter content is reset.

Then the filter coefficients are replaced by the second group $w(0),\ w(1),\ldots,\ w(14)$. The second group of 15 samples $x\,(15)\ldots x\,(29)$ of the processed signal is sequentially fed to the input. After 15 clock cycles, we get the second group of 15 output samples $y\,(15)\ldots y\,(29)$ at the output. The filter content is reset.

Then the filter coefficients are replaced by the third group $w(0),\ w(1),\ldots,\ w(14)$. The third group of 15 samples $x\,(30)\ldots x\,(44)$ of the processed signal is sequentially fed to the input. After 15 clock cycles, we get the third group of 15 output samples $y\,(30)\ldots y\,(44)$ at the output. The filter content is reset.

Then the filter coefficients are replaced by the fourth group $w(0),\ w(1),\ldots,\ w(14)$. The fourth group of 15 samples $x\,(45)\ldots x\,(59)$ of the processed signal is sequentially fed to the input. After 15 clock cycles, we get the fourth group of 15 output samples $y(45)\ldots y(59)$ at the output. The filter content is reset.

Thus, we use a window size equal to the filter length.

Disadvantages:

The memory circuit costs for storing the ADF coefficients for each group increase. However, now there is a lot of memory and it is very cheap.

There is a division operation (exp. (10), (14)). Moreover, the fraction denominator, generally, can be equal to zero. To avoid dividing by zero, it is proposed to shift the input signal and the reference signal:

$$x_{sh} = x_i + \max(abs(x_i)) + C,$$
$$d_{sh} = d_i + \max(abs(x_i)) + C.$$

Here: x_i – the input signal, d_i – the reference signal, x_{sh} – the shifted input signal, d_{sh} – the shifted reference signal, $C - const$.

Advantages:

1. Using the developed algorithm, it is possible to obtain ADF coefficients for each group without errors. This will improve the signal processing accuracy.
2. There is no comparison operation with the reference signal.
3. Increases the speed of digital filtering through using of the RNS.

Using of the adaptation algorithm with the least square method in the ADF leads to a change in the coefficients at each iteration, such that with a large input signal length, the deviation of the ADF output signal from the reference signal will be minimal.

In the proposed method for implementing the ADF in the RNS by dividing the input vector into blocks by samples L, L sets of coefficients will be obtained, the use of which will provide zero error from the reference signal. In the developed algorithm, at each step, only one ADF coefficient is adapted, and not all coefficients at once as in the LMS.

To adjust all filter coefficients, we need to perform L pairs of addition/multiplication operations, L– the filter length. So, we need to perform 60 pairs of addition/multiplication operations to adjust all filter coefficients (zero error, not iterative process). In case of using LMS we need to perform 15 pairs of addition/multiplication operations per iteration to adjust all filter coefficients (not zero error, iterative process). In case of using RLS we need to perform 644 pairs of addition/multiplication operations per iteration to adjust all filter coefficients (not zero error, iterative process).

IV. SIMULATION DETAILS

The denoising quality estimates of the considered algorithms for various test non-stationary signals (Fig. 2) and various noise levels ($Q = 4096$, $L = 32$, $SNR = -6\ldots6$ dB) are shown in Fig. 3-8. The developed algorithm is hereinafter referred to as adaptive filtering – AF.

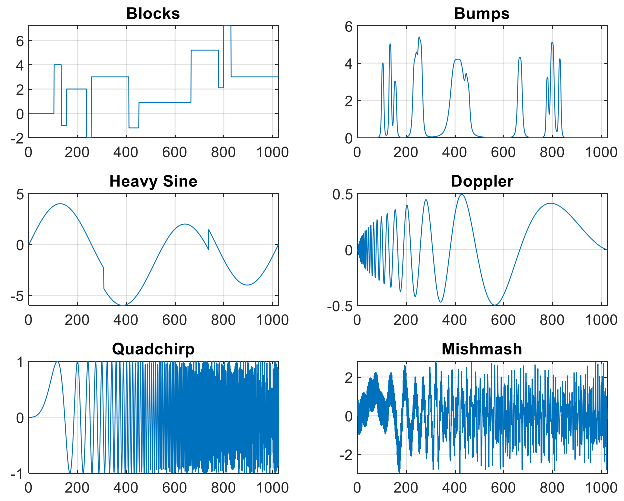

FIGURE 2. Test signals.

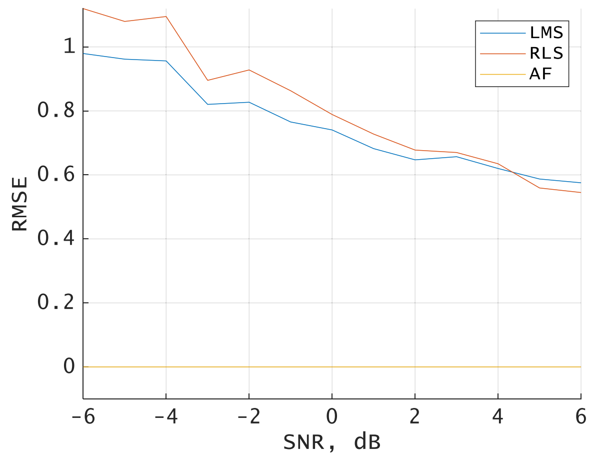

FIGURE 3. LMS vs RLS vs AF: denoising quality. Signal "Blocks".

We will use the standard RMSE metric (Root Mean Square Error) to estimate the filtering quality:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{Q}(x_i - \tilde{x}_i)^2}{Q}}, \tag{17}$$

where x_i– initial signal; $\tilde{x}_i$– noised signal; Q – number of samples.

One can see (Fig. 3-5) that LMS converges more slowly than RLS. AF always provides zero error. The dependences are preserved in all the experiments conducted for different non-stationary signals and different SNR values.

However, sometimes (Fig. 6-8) LMS doesn't converge at all. RLS still converges. AF still provides zero error. The dependences are preserved in all the experiments conducted for different non-stationary signals and different SNR values.

The denoising quality estimates of the considered algorithms for the earthquake recording (MATLAB dataset) are shown in Fig. 9. The developed algorithm allows one

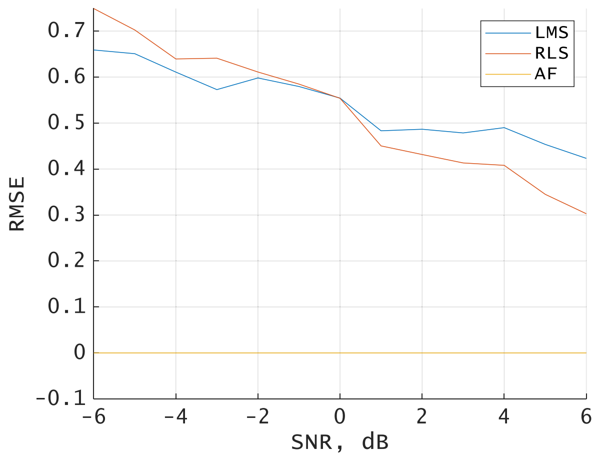

FIGURE 4. LMS vs RLS vs AF: denoising quality. Signal "Bumps".

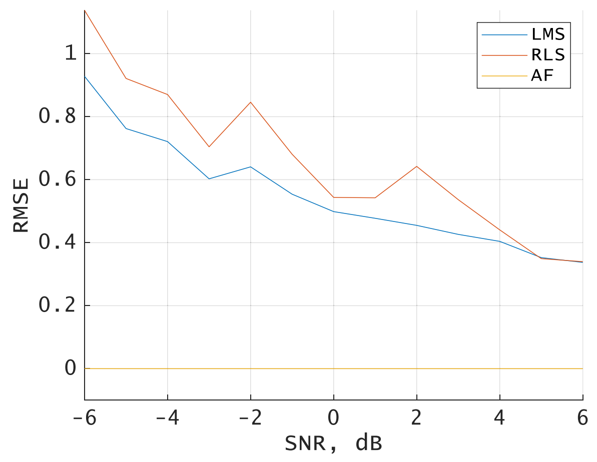

FIGURE 5. LMS vs RLS vs AF: denoising quality. Signal "Heavy Sine".

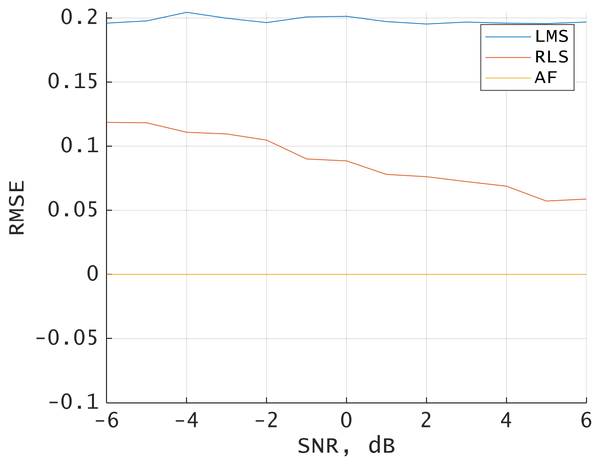

FIGURE 6. LMS vs RLS vs AF: denoising quality. Signal "Doppler".

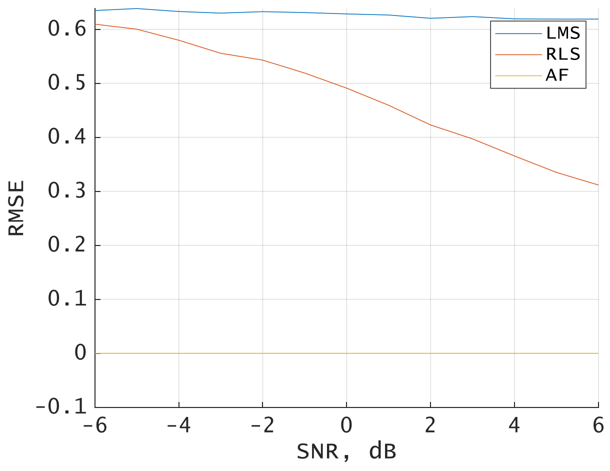

FIGURE 7. LMS vs RLS vs AF: denoising quality. Signal "Quadchirp".

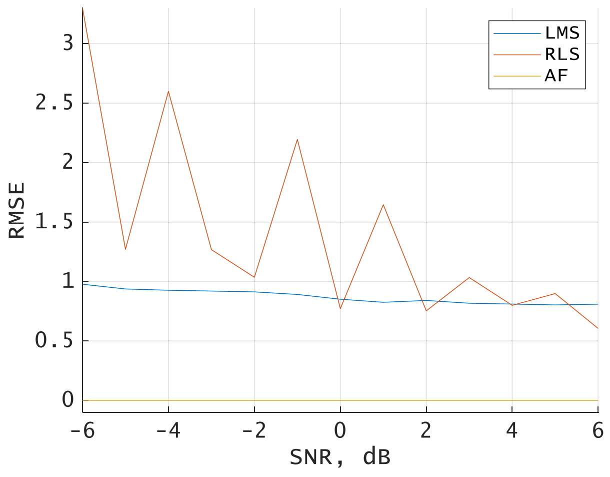

FIGURE 8. LMS vs RLS vs AF: denoising quality. Signal "Mishmash".

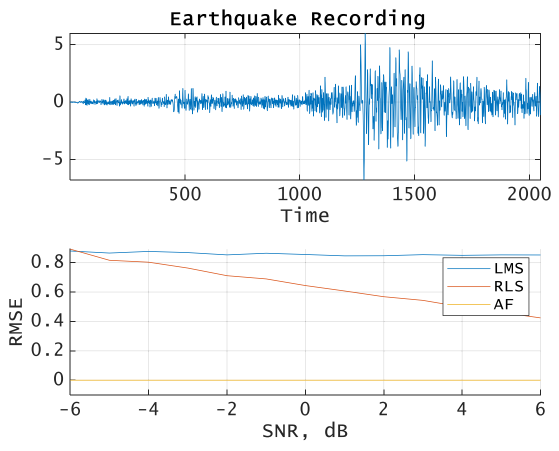

FIGURE 9. LMS vs RLS vs AF: denoising quality. Earthquake recording.

to perform denoising without losing quality, which can improve the results of further classification (e.g., earthquake/explosion).

Fig. 10 shows the dependence of the algorithm execution time on the input signal length (filter length = const = 32).

Fig. 11 shows the dependence of the algorithm execution time on the filter length (signal length = const = 4096).

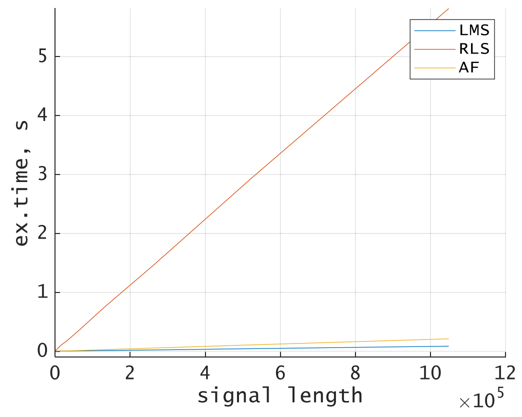

FIGURE 10. LMS vs RLS vs AF: execution time (filter length = const = 32).

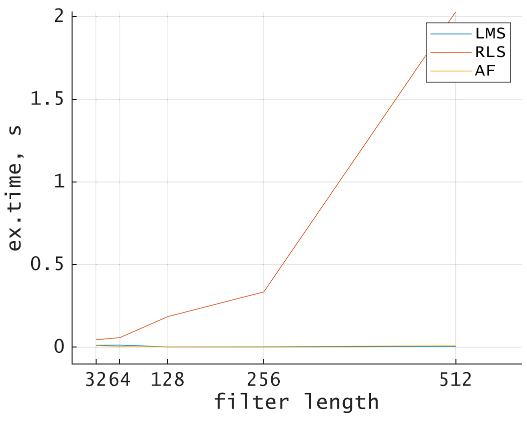

FIGURE 11. LMS vs RLS vs AF: execution time (signal length = const = 4096.

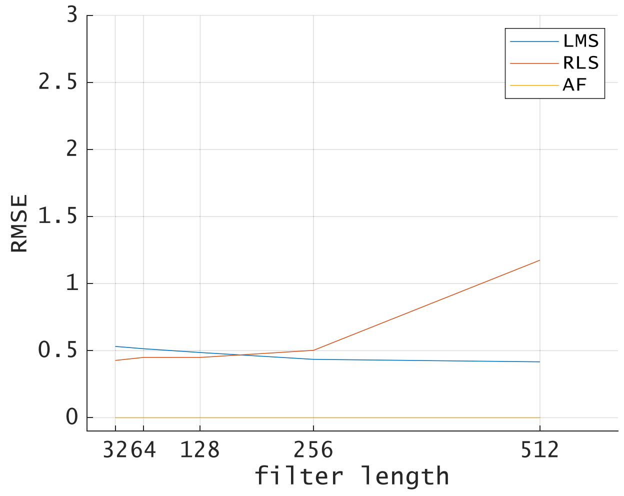

FIGURE 12. LMS vs RLS vs AF: RMSE (signal length = const = 4096).

One can see from Fig. 10-11 that RLS is very expensive in both cases, LMS and proposed algorithm AF are very fast and computationally stable (they are very close, but the upward trend continues).

TABLE 5. LMS vs RLS vs AF.

LMS	RLS	AF (proposed)
Simple and can be easily applied.	Increased complexity and computational cost.	Simple and can be easily applied.
Takes longer to converge.	Faster convergence.	N/A
Adaptation is based on gradient based approach which updates filter weights in a manner to converge to the optimum filter weights.	Adaptation is based on recursive approach that finds the filter coefficients which minimize a weighted linear least square cost function relating to the input signals.	Adaptation is performed one coefficient per step
Larger steady state error with respect to unknown system compared to the RLS algorithm.	Lower steady state error with respect to unknown system.	Zero error
No account for the past data.	Accounts for past data from the beginning to the current data point.	No account for the past data.
Objective is to minimize the current mean square error between the desired signal and the output.	Objective is to minimize the total weighted squared error between the desired signal and the output.	N/A
No memory involved. Older error values play no role in the total error considered.	Has infinite memory. All error data is considered in the total error. Using the forgetting factor, the older data can be less emphasized compared to the newer data. Since $0 \leq \lambda < 1$, applying the factor is equivalent to weighting the older error.	No memory involved. Older error values play no role in the total error considered.
To adjust the coefficients at each step, $N+1$ pairs of addition-multiplication operations are needed to be performed N is the filter order).	To adjust the coefficients at each step, $3N^2+4N$ pairs of addition-multiplication operations are needed to be performed (N is the filter order).	To adjust all filter coefficients, you need to perform N pairs of addition/multiplication operations (N is the filter order).

Fig. 12 shows the dependence of the RMSE on the filter length (signal length = const = 4096).

Obviously, the filter length does not have a significant effect on the RMSE.

The amplitude frequency responses and impulse responses of adaptive filters (LMS, RLS and AF) at the end of the adaptation process (last iteration or last block) are shown below (Fig. 13). The difference between LMS/RLS and AF amplitude frequency responses and impulse responses is due to block processing.

Fig. 14-15 illustrates the denoising quality using algorithms LMS, RLS and AF with a fixed signal-to-noise ratio for the earthquake recording.

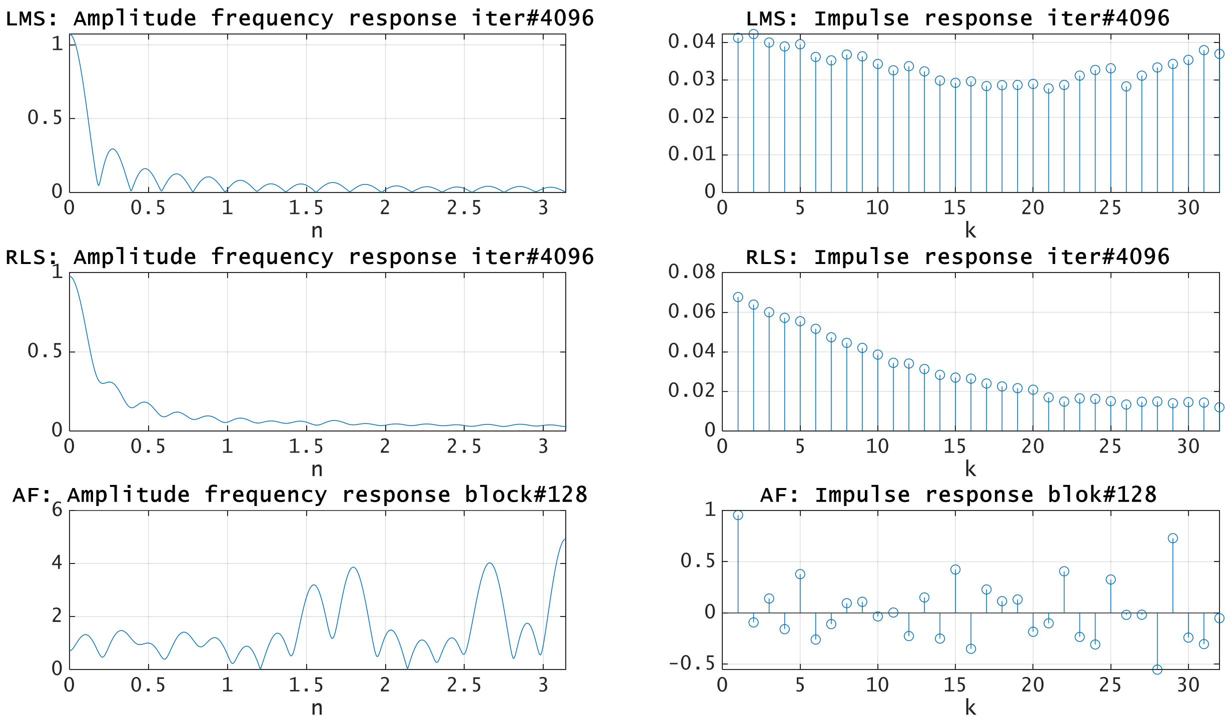

FIGURE 13. LMS vs RLS vs AF: Amplitude frequency response (left) and Impulse response (right).

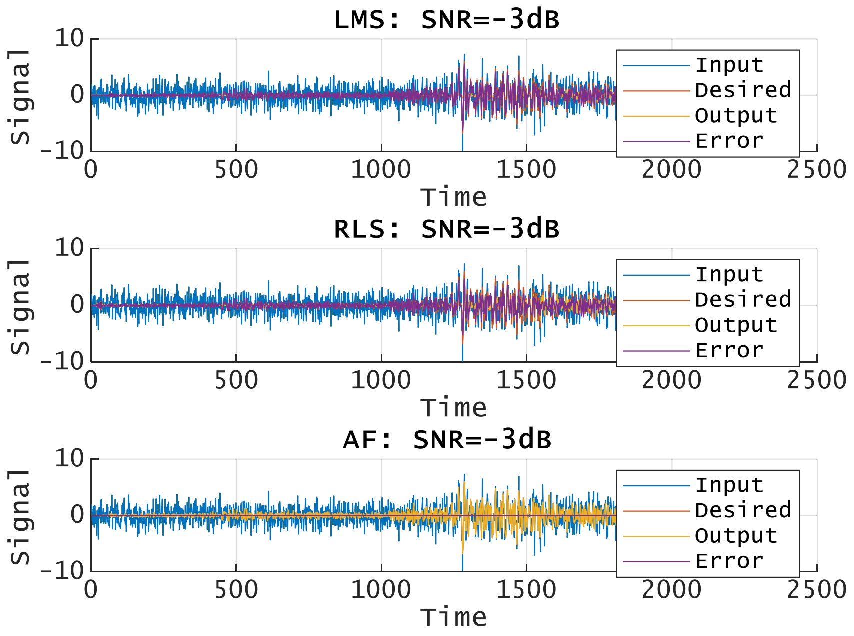

FIGURE 14. LMS vs RLS vs AF: Denoising quality for earthquake recording with SNR = −3 dB.

All experiments were performed using real and synthetic data. The simulation was performed on a PC with the following architecture: OS Win 10 64-bit, CPU Intel Core i7 Skylake 4.0 GHz, RAM Kingston HyperX Fury 64 GB 2.4 GHz DDR4, NVIDIA GeForce GTX 1080 1.7 GHz DDR5 GPU 8 GB 10 GHz, 2560 CUDA Cores, MATLAB R2020b 64-bit.

V. DISCUSSION

Studies have shown that well-known algorithms (LMS, RLS) [31] are not advisable to use when constructing digital filters operating in the RNS, due to the complexity of the implementation. These filters are found in only a few studies and for quite some time [26], [27].

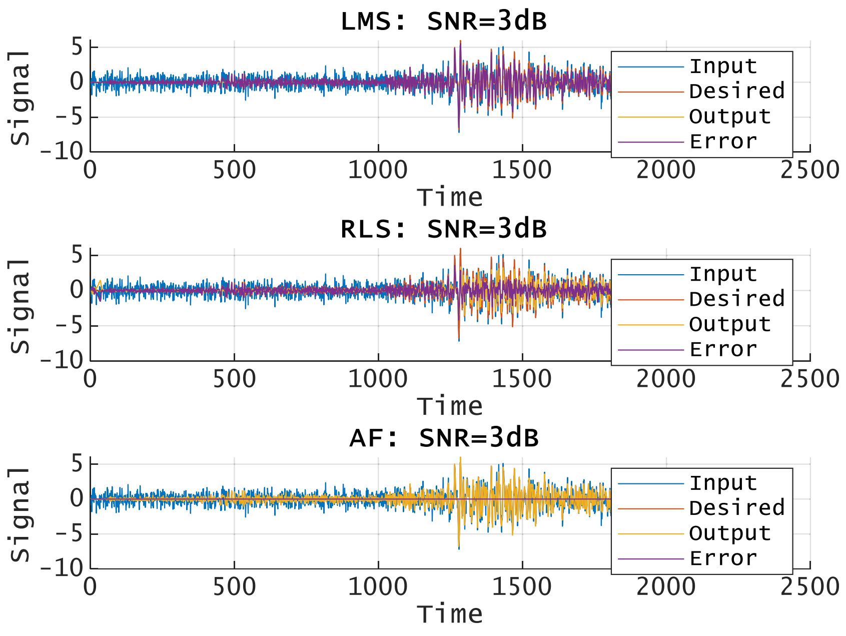

FIGURE 15. LMS vs RLS vs AF: Denoising quality for earthquake recording with SNR = 3 dB.

In [32], a hybrid Residue Number System (RNS) implementation of an adaptive FIR filter based on the LMS adaptation algorithm is presented. The advantages in terms of area and speed of the hybrid RNS adaptive filter with respect to the two's complement one have been evaluated for a standard cells' implementation showing savings in area and power dissipation of about 50%.

In [33] the authors investigated the implementation of a low-power adaptive filter, based on the use of binary components for the LMS algorithm and on RNS components for the variable filter.

A block LMS (BLMS) implementation [34] was preferred for the update of the adaptive FIR filter coefficients. RNS-FPL merged filters demonstrated its superiority when compared to 2C (two's complement) filters, being about 65% faster and requiring fewer logic elements for most study cases.

A sequential algorithm for adjusting the filter coefficients operating in the RNS is presented. To adjust each coefficient, it is necessary to perform one subtraction operation, one multiplication operation and one addition operation modulo the RNS, i.e., recalculation time is proportional to the filter order.

We provide a fundamentally new algorithm that surpasses the existing ones like LMS and RLS, and their modifications [26]–[28], [32]–[34] (Table 5) in a number of parameters: adaptation (denoising) quality (zero error, Fig. 3-8), ease of implementation, execution time (Fig. 10-11). The main difference between the developed algorithm is the sequential adaptation of each individual coefficient with zero error. In the known algorithms, the entire vector of coefficients is iteratively adapted, with some specified accuracy.

Thus, it is obvious that the use of the developed algorithm can significantly reduce the time spent on adjusting the filter coefficients in comparison with previously known adaptation algorithms. At the same time, the coefficients adjustment can be carried out simultaneously for the selected modules/groups and does not require the operation of converting the RNS into a BNS for comparing the obtained value of the output sample with its reference value.

The results obtained allow us to conclude that the use of the RNS provides an increase in the efficiency of the adaptive digital filter algorithm. In this case, it is necessary to take into account the fact that the filter coefficients adjustment in any case is carried out with an allowable (specified) error and the filtering algorithm (LMS, RLS) with the adjusted coefficients introduces an error into the value of the filter output sample. Therefore, the use of the RNS allows one to reduce the value of this error, which confirms the expediency of their application in digital signal processing algorithms.

Further research will be related with fast FPGA implementation of the ADF based on the proposed technique. This technique will be used to develop faster methods of digital signal processing, cryptography, machine learning.

VI. CONCLUSION

The paper provides a mathematical description of the known algorithms (LMS, RLS), as well as a new algorithm for adjusting the adaptive digital filter coefficients in the RNS. The computational complexity and speed of these algorithms are estimated. The speed and accuracy of adaptation were evaluated.

The proposed algorithm surpasses the existing ones like LMS and RLS, and their modifications in a number of parameters: adaptation (denoising) quality, ease of implementation, execution time.

REFERENCES

[1] A. Voznesensky and D. Kaplun, "Adaptive signal processing algorithms based on EMD and ITD," *IEEE Access*, vol. 7, pp. 171313–171321, 2019.

[2] D. M. Klionskiy, D. I. Kaplun, and S. A. Romanov, "Adaptive techniques of signal processing," in *Proc. IEE 6th Forum Strategic Partnership Universities Enterprises Hi-Tech Branches (Sci., Educ., Innov.) (SPUE)*, St. Petersburg, Nov. 2017, pp. 218–219.

[3] N. J. Bershad and J. C. M. Bermudez, "A switched variable step size NLMS adaptive filter," *Digit. Signal Process.*, vol. 101, Jun. 2020, Art. no. 102730, doi: 10.1016/j.dsp.2020.102730.

[4] D. Kari, A. H. Mirza, F. Khan, H. Ozkan, and S. S. Kozat, "Boosted adaptive filters," *Digit. Signal Process.*, vol. 81, pp. 61–78, Oct. 2018, doi: 10.1016/j.dsp.2018.07.012.

[5] P. S. R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation*. New York, NY, USA: Springer, 2008, p. 627.

[6] M. Tufail, S. Ahmed, M. Rehan, and M. T. Akhtar, "A two adaptive filters-based method for reducing effects of acoustic feedback in single-channel feedforward ANC systems," *Digit. Signal Process.*, vol. 90, pp. 18–27, Jul. 2019, doi: 10.1016/j.dsp.2019.03.016.

[7] P. A. Lyakhov, A. R. Orazaev, N. I. Chervyakov, and D. I. Kaplun, "A new method for adaptive median filtering of images," in *Proc. IEEE Conf. Russian Young Researchers Electr. Electron. Eng. (EIConRus)*, Jan. 2019, pp. 1197–1201.

[8] D. Butusov, T. Karimov, A. Voznesenskiy, D. Kaplun, V. Andreev, and V. Ostrovskii, "Filtering techniques for chaotic signal processing," *Electronics*, vol. 7, no. 12, p. 450, Dec. 2018.

[9] D. M. Klionskiy, D. I. Kaplun, A. M. Golubkov, and V. V. Gulvanskiy, "Adaptive algorithm for hydroacoustic signal processing," in *Proc. IEEE Conf. Russian Young Researchers Electr. Electron. Eng. (EIConRus)*, 2017, pp. 686–689.

[10] A. R. Omondi and A. B. Premkumar, *Residue Number System: Theory and Implementation*. London, U.K.: Imperial College Press, 2007, p. 312.

[11] G. Jullien, W. Miller, J. Soltis, A. Baraniecka, and B. Tseng, "Hardware realization of digital signal processing elements using the residue number system," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Hartford, CT, USA, May 1977, pp. 506–510, doi: 10.1109/ICASSP.1977.1170347.

[12] P. V. A. Mohan, *Residue Number Systems Theory and Applications*. Basel, Switzerland: Birkhäuser, 2016, p. 351.

[13] I. A. Kalmykov, A. V. Veligosha, D. I. Kaplun, D. M. Klionskiy, and V. V. Gulvanskiy, "Parallel-pipeline implementation of digital signal processing techniques based on modular codes," in *Proc. 19th IEEE Int. Conf. Soft Comput. Meas. (SCM)*, May 2016, pp. 213–214.

[14] C.-C. Chang, W.-K. Lee, Y. Liu, B.-M. Goi, and R. C.-W. Phan, "Signature gateway: Offloading signature generation to IoT gateway accelerated by GPU," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4448–4461, Jun. 2019.

[15] N. I. Chervyakov, M. G. Babenko, P. A. Lyakhov, and I. N. Lavrinenko, "An approximate method for comparing modular numbers and its application to the division of numbers in residue number systems," *Cybern. Syst. Anal.*, vol. 50, no. 6, pp. 977–984, Nov. 2014.

[16] G. Dimauro, S. Impedovo, and G. Pirlo, "A new technique for fast number comparison in the residue number system," *IEEE Trans. Comput.*, vol. 42, no. 5, pp. 608–612, May 1993.

[17] M. Valueva, G. Valuev, N. Semyonova, P. Lyakhov, N. Chervyakov, D. Kaplun, and D. Bogaevskiy, "Construction of residue number system using hardware efficient diagonal function," *Electronics*, vol. 8, no. 6, p. 694, Jun. 2019.

[18] P. Boyvalenkov, N. I. Chervyakov, P. Lyakhov, N. Semyonova, A. Nazarov, M. Valueva, G. Boyvalenkov, D. Bogaevskiy, and D. Kaplun, "Classification of moduli sets for residue number system with special diagonal functions," *IEEE Access*, vol. 8, pp. 156104–156116, 2020, doi: 10.1109/ACCESS.2020.3019452.

[19] S. Haykin and B. Widrow, *Least-Mean-Square Adaptive Filters*. Hoboken, NJ, USA: Wiley, 2003, p. 512.

[20] A. H. Sayed, *Adaptive Filters*. Hoboken, NJ, USA: Wiley, 2011, p. 824.

[21] S. Haykin, *Adaptive Filter Theory*, 5th ed. London, U.K.: Pearson Education, 2014, p. 912.

[22] M. A. Soderstrand, W. Jenkins, G. Jullien, and F. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*. New York, NY, USA: IEEE Press, 1986, p. 418.

[23] D. I. Kaplun, A. S. Voznesenskiy, V. V. Gulvanskii, and D. V. Bogaevskiy, "Application of non-positional codes for FIR-filter implementation using computers with CUDA technology," in *Proc. IEEE Conf. Russian Young Researchers Electr. Electron. Eng. (EIConRus)*, Jan. 2018, pp. 1086–1089.

[24] K. Isupov, "Using floating-point intervals for non-modular computations in residue number system," *IEEE Access*, vol. 8, pp. 58603–58619, 2020.

[25] A. D. Poularikas, *Adaptive Filtering: Fundamentals of Least Mean Squares with MATLAB*. Boca Raton, FL, USA: CRC Press, 2017, p. 363.

[26] D. Miller and J. Polky, "An implementation of the LMS algorithm in the residue number system," *IEEE Trans. Circuits Syst.*, vol. CAS-31, no. 5, pp. 452–461, May 1984.

[27] D. D. Miller and J. N. Polky, "A residue number system implementation of the LMS algorithm using optical waveguide circuits," *IEEE Trans. Comput.*, vol. C-32, no. 11, pp. 1013–1028, Nov. 1983.

[28] A. V. Veligosha, D. I. Kaplun, D. V. Bogaevskiy, V. V. Gulvanskiy, A. S. Voznesenskiy, and I. A. Kalmykov, "Adjustment of adaptive digital filter coefficients in modular codes," in *Proc. IEEE Conf. Russian Young Researchers Electr. Electron. Eng. (EIConRus)*, Jan. 2018, pp. 1167–1170.

[29] P. V. A. Mohan, *Residue Number Systems: Algorithms and Architectures*. New York, NY, USA: Springer, 2012, p. 253.

[30] A. S. Molahosseini, L. S. de Sousa, and C.-H. Chang, *Embedded Systems Design With Special Arithmetic and Number Systems*. Cham, Switzerland: Springer, 2017, p. 389.

[31] S. Dixit and D. Nagaria, "LMS adaptive filters for noise cancellation: A review," *Int. J. Electr. Comput. Eng.*, vol. 7, no. 5, pp. 2520–2529, Oct. 2017.

[32] G. L. Bernocchi, G. C. Cardarilli, A. Del Re, A. Nannarelli, and M. Re, "A hybrid RNS adaptive filter for channel equalization," in *Proc. 40th Asilomar Conf. Signals, Syst. Comput.*, Oct. 2006, pp. 1706–1710.

[33] G. L. Bernocchi, G. C. Cardarilli, A. Del Re, A. Nannarelli, and M. Re, "Low-power adaptive filter based on RNS components," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2007, pp. 3211–3214.

[34] J. Ramírez, A. Meyer-Bäse, A. Garcia, and A. Lloris-Ruíz, "Design and implementation of RNS-based adaptive filters," in *Proc. 13th Int. Conf. Field Program. Log. Appl. (FPL)*, Lisbon, Portugal, Sep. 2003, pp. 1135–1138.

DMITRII KAPLUN (Member, IEEE) received the Ph.D. degree from Saint Petersburg Electrotechnical University "LETI," Saint Petersburg, Russia, in 2009. He was an Associate Professor with Saint Petersburg Electrotechnical University "LETI" in 2015, where he was a Senior Researcher. In 2009, he defended the Ph.D. Thesis in digital signal processing with Saint Petersburg Electrotechnical University "LETI." The current research and academic work are concerned with digital signal and image processing, radio monitoring and hydroacoustic monitoring applications, embedded and reconfigurable systems, computer vision, and machine learning. He regularly takes part in different collaborative projects connected with signal processing, artificial neural networks, and software-hardware implementation of digital signal processing algorithms. The most substantial results are in the fields of digital signal and image processing, embedded systems, artificial neural networks, and machine learning. He is the author of more than 50 articles in journals and conference proceedings.

ALEXANDER VOZNESENSKY was born in Saint-Petersburg, Russia, in 1991. He received the B.S. and M.S. degrees in applied mathematics and informatics from Saint Petersburg Electrotechnical University "LETI," in 2012 and 2014, respectively. He is currently working on the Ph.D. Thesis. Since 2012, he has been as a Data Scientist with the Faculty of Computer Science and Technology, Department of Automation and Control Processes, Saint Petersburg Electrotechnical University "LETI." He is the author of more than 25 articles and three patents. His research interests include MATLAB, mathematical modeling, signal processing, image processing, video processing, neural networks, CNN, C/C++, CUDA, parallel computing, data mining, data science, and data analysis.

ALEXANDER V. VELIGOSHA was born in Kislovodsk, in May 1959. He graduated from the Kiev Higher Military Engineering School of Communications. In 1990, he graduated as a S.M. Budyonny from the Military Academy of Communications. He received the Ph.D. degree from the Stavropol Higher Military Engineering School of Communications, in1996. From 2010 to 2016, he worked as an Associate Professor with the Department, North Caucasus Federal University. Since 2016, he has been an Associate Professor with the Department of the Branch, Military Academy after Peter the Great in Serpukhov. His research interests include modular arithmetic and its application in digital signal processing (digital filtering algorithms) and in processing various types of video data, intelligent digital signal processing algorithms, and satellite communication systems (antenna platforms for communication repeaters). He received the award for his Ph.D. degree, in June 1997, and the academic title of Associate Professor, in May 1998.

IGOR A. KALMYKOV was born in January 1967. He graduated from the Stavropol Higher Military Engineering School of Communications, in 1989, the Ph.D. degree from the Stavropol Higher Military Engineering School of Communications, in December 1995, and the Doctor of Technical Sciences degree, in February 2007. Since 2010, he has been working as a Professor with the Department of Information Security of Automated Systems, North Caucasus Federal University. His research interests include modular arithmetic in Galois fields, corrective codes, fault-tolerant non-positional special processors and computing devices for digital signal processing, neuroinformatics, and adaptive tools of protecting information from unauthorized access. He received the academic title of an assistant professor in the department, in 2000, the academic title of a professor in the department, in February 2009, and the title of Honorary Worker of Higher Professional Education, in 2010. In 2006, he became a Laureate of the Potanin Prize-the Best Teacher of a Military University.

KANDARPA KUMAR SARMA (Senior Member, IEEE) received the M.Tech. degree in signal processing from the Indian Institute of Technology Guwahati, India, in 2005, and the Ph.D. degree in the mobile communication from the Indian Institute of Technology Guwahati. He currently serves as a Professor and the Head for the Department of Electronics and Communication Engineering, Gauhati University, India. His research interests include deep learning, mobile communication, speech processing, antenna designs, and electronic warfare.