MODIFIED SRF-QRD-LSL ADAPTIVE ALGORITHM WITH IMPROVED NUMERICAL ROBUSTNESS

Constantin Paleologu, Felix Albu, Andrei Alexandru Enescu, and Silviu Ciochină

Telecommunications Department, University Politehnica of Bucharest, Romania
e-mail: {pale, felix, aenescu, silviu}@comm.pub.ro

ABSTRACT

The QR-decomposition-based least-squares lattice (QRD-LSL) algorithm is one of the most attractive choices for adaptive filters applications, mainly due to its fast convergence rate and good numerical properties. In practice, the square-root-free QRD-LSL (SRF-QRD-LSL) algorithms are frequently employed, especially when fixed-point digital signal processors (DSPs) are used for implementation. In this context, there are some major limitations regarding the large dynamic range of the algorithm's cost functions. Consequently, hard scaling operations are required, which further reduce the precision of numerical representation and lead to performance degradation. In this paper we propose a SRF-QRD-LSL algorithm based on a modified update of the cost functions, which offers improved numerical robustness. Simulations performed in fixed-point and logarithmic number system (LNS) implementations support the theoretical findings. Also, in order to outline some practical aspects of this work, the proposed algorithm is tested in the context of echo cancellation. It is shown that this algorithm outperforms by far the normalized least-mean-square (NLMS) algorithm (which is the most common choice for echo cancellation), especially in terms of double-talk robustness.

***Index Terms*—** Adaptive filters, echo cancellation, fixed-point arithmetic, logarithmic number system (LNS), QR-decomposition-based least-squares lattice (QRD-LSL).

1. INTRODUCTION

The QR-decomposition-based least-squares lattice (QRD-LSL) adaptive algorithm [1]–[3] combines the good numerical properties of QR-decomposition and the desirable features of a recursive least-squares lattice. Whereas its transversal counterpart, i.e., the recursive QR-decomposition-based recursive least-squares (QRD-RLS) algorithm [2], requires a high computational load on the order of M^2 (where M is the adaptive filter order), the QRD-LSL algorithm is "fast" in the sense that the computational complexity is reduced to a linear dependence on M. This algorithm exploits the shifting property of serialized input data, i.e., the Toeplitz structure of the data matrix, to perform joint-process estimation in a fast manner. Due to its features, the QRD-LSL algorithm proves to be a very attractive choice for many applications [4]–[6].

The standard version of the QRD-LSL algorithm uses Givens rotations for implementing the QR-decomposition, which implies the use of square-root operations. In general, these operations are expensive and awkward to calculate in practice, constituting a bottleneck for overall performance. It has to be noticed that the square root operations require a large computing time, as they must be approximated by another technique, e. g., Taylor series. Consequently, the computing time increases significantly and the application areas of these algorithms become restricted. Thus, the main goal is to minimize the number of instructions within the implemented algorithm. For these reasons, square-root-free QRD-LSL (SRF-QRD-LSL) algorithms have been formulated, using special methods for performing Givens rotations without square roots [7].

Following these issues, a crucial aspect is the behavior of the adaptive algorithm in finite precision implementations. In this context, due to cost considerations, fixed-point digital signal processors (DSPs) could be preferred over floating-point ones. Quantization effects in the former result in a deviation of the adaptive filter performance from that observed in infinite precision. This deviation, which becomes more apparent as the number of representation bits is reduced, may take the form of an increased residual error after convergence (i.e., loss of precision), or more dramatically, of an unbounded accumulation of quantization errors over time (i.e., numerical instability). Moreover, there are some limitations related to the dynamic range of the algorithm's parameters. It is well known that in a fixed-point implementation context the absolute values of all involved parameters have to be smaller than one. In the case of the classical QRD-LSL algorithm, the cost functions asymptotically increase;

This work was supported by the UEFISCSU Romania under Grants PN-II-"Idei" no. 65/01.10.2007 and no. 331/01.10.2007.

they are upper bounded by the value $1/(1-\lambda)$, where λ is the exponential weighting factor (with $0 < \lambda \leq 1$). When dealing with a value of this parameter very close to one, (which is highly preferred due to stability reasons [2]) very large values of the cost functions are expected. Therefore, in order to prevent the overflow phenomena, it is necessary to perform hard scaling operations; this could lead to significant degradation of the algorithm's performance due to the decreased precision of numerical representation.

Most contemporary microprocessors perform real arithmetic using the floating-point system. Floating-point circuits are large, complex and much slower than fixed-point units; they require separate circuitry for the add/ subtract, multiply, divide, and square-root operations. All floating-point operations are liable to a maximum half-bit rounding error. A recent alternative to the floating point is the logarithmic number system (LNS). The LNS performs the real multiplication, division, and square-root at fixed-point speed [8].

In this paper, we propose a version of the SRF-QRD-LSL algorithm based on a modified update formula of the cost functions. The main issue is that the maximum value of the cost functions will be at initialization and then they will asymptotically decrease to a lower bound. Therefore, the scale factors are less critical, leading to an increased precision of numerical representation [9].

In [10] we have analyzed some versions of the QRD-LSL algorithm according to some ITU standard requirements concerning echo cancellers. The experiments indicated that the QRD-LSL algorithms fulfill by far the requirements of the ITU G.168 recommendation [11] concerning the steady-state echo return loss enhancement and convergence speed. A special interest was given to the problem of the behavior of the algorithms during the double-talk periods. Two distinct effects were identified, i.e., 1) the incomplete attenuation of the far-end signal and 2) the unwanted attenuation of the near-end signal, as a result of the near-end signal leakage to the output of the adaptive filter through the error signal. Using a value of λ very close to 1 can reduce the last effect. The experiments prove that it is possible to work with such a high value of λ by preserving in the same time a convergence speed that fulfils the requirements of the ITU recommendation. Generally, one can assert that this class of algorithms is much more robust to double-talk than the normalized least-mean-square (NLMS) type algorithms. The QRD-LSL algorithms could satisfactorily operate even in the absence of a double-talk detector (DTD). Since the proposed SRF-QRD-LSL algorithm is suitable for fixed-point implementation we present a network echo canceller based on this algorithm, implemented on a fixed-point DSP.

The rest of the paper is organized as follows. In Section 2, we briefly present the original SRF-QRD-LSL algorithm and we discuss some fixed-point and LNS implementation aspects. The proposed algorithm is developed in Section 3. Some backgrounds of echo cancellation are given in Section 4. The experimental results are given in Section 5. The simulations performed in both fixed-point DSP and LNS implementations prove the theoretical findings; also, the experiments performed in echo cancellation context outline the practical aspects. Finally, Section 6 concludes this work.

2. NUMERICAL ISSUES OF THE SRF-QRD-LSL ALGORITHM

Among the versions of the SRF-QRD-LSL algorithm presented in the literature we have chosen the most frequently used one [12]. This version is described below.

Initialization:

$J_m^f(0) = J_m^b(0) = \delta$ - positive constant

$k_m^f(0) = k_m^b(0) = k_m^c(0) = 0$

$e_0^f(n) = e_0^b(n) = x(n)$ - input signal

$e_0(n) = d(n)$ - desired signal, $\alpha_0(n) = 1$

For time moment n and filter order m compute:

- *Lattice part*:

$$J_m^f(n) = \lambda J_m^f(n-1) + \alpha_m(n-1)\left|e_m^f(n)\right|^2$$

$$c_m^f(n) = \lambda \frac{J_m^f(n-1)}{J_m^f(n)}$$

$$s_m^f(n) = \frac{\alpha_m(n-1)e_m^f(n)}{J_m^f(n)}$$

$$J_m^b(n) = \lambda J_m^b(n-1) + \alpha_m(n)\left|e_m^b(n)\right|^2$$

$$c_m^b(n) = \lambda \frac{J_m^b(n-1)}{J_m^b(n)}$$

$$s_m^b(n) = \frac{\alpha_m(n)e_m^b(n)}{J_m^b(n)}$$

$$e_{m+1}^f(n) = e_m^f(n) + k_m^{f*}(n-1)e_m^b(n-1)$$

$$k_m^f(n) = c_m^b(n-1)k_m^f(n-1) - s_m^b(n-1)e_m^{f*}(n)$$

$$e_{m+1}^b(n) = e_m^b(n-1) + k_m^{b*}(n-1)e_m^f(n)$$

$$k_m^b(n) = c_m^f(n)k_m^b(n-1) - s_m^f(n)e_m^{b*}(n-1)$$

- *Ladder part*:

$$e_{m+1}(n) = e_m(n) - k_m^{c*}(n-1)e_m^b(n)$$

$$k_m^c(n) = c_m^b(n)k_m^c(n-1) + s_m^b(n)e_m^*(n)$$

$$\alpha_{m+1}(n) = \alpha_m(n-1) - \frac{\alpha_m^2(n-1)\left|e_m^f(n)\right|^2}{J_m^f(n)}$$

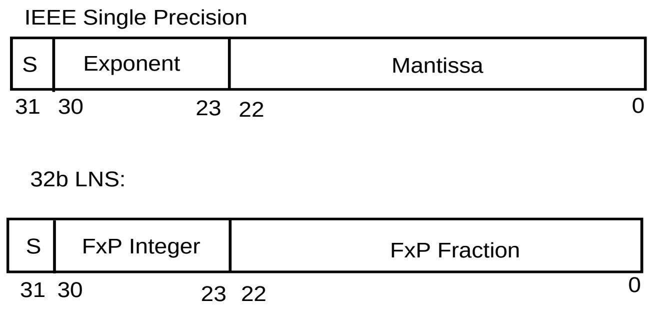

Fig. 1. IEEE standard single precision floating point representation and the 32-bit LNS format.

The superscript * denotes complex conjugation. The parameters involved in the algorithm are denoted as follows (for prediction order *m* and time index *n*):

- $J_m^b(n), J_m^f(n)$ - sum of weighted backward/forward prediction error squares (i.e., the cost functions);
- $e_m^b(n), e_m^f(n)$ - backward/forward prediction error;
- $k_m^b(n), k_m^f(n)$ - lattice structure coefficients;
- $c_m^b(n), c_m^f(n), s_m^b(n), s_m^f(n)$ - Givens parameters;
- $e_m(n)$ - joint-process a posteriori estimation error;
- $\alpha_m(n)$ - joint-process a priori estimation error;
- $k_m^c(n)$ - ladder structure coefficients;
- λ - exponential weighting factor.

Next, let us assume the context of a fixed-point DSP implementation, with a word length of *B* bits. The absolute values of the algorithm's parameters have to be less than one. For simplicity, we will take into consideration only the forward prediction part of the algorithm; the same analysis could be straightforwardly extended for the backward prediction part. According to the update equation of the cost function, the upper bound limit is

$$J_m^f(n)\Big|_{n\to\infty} = \frac{1}{1-\lambda} \tag{1}$$

For a value of λ very close to one, the cost functions reach to very large values. Therefore, a scaling operation (i.e., right shift) has to be applied, such that

$$J_m^f(\infty)2^{-Q} = 1 \tag{2}$$

The parameter *Q* is the scaling factor that is computed as

$$Q = \left\lceil -\log_2(1-\lambda) \right\rceil \tag{3}$$

where $\lceil \bullet \rceil$ denotes superior integer part.

Table 1. LNS arithmetic operations.

x + y	ADD	Lz = Lx + log(1+2^(Ly-Lx)), Sz depends on sizes of x,y
x - y	SUB	Lz = Lx + log(1-2^(Ly-Lx)), Sz depends on sizes of x,y
x * y	MUL	Lz = Lx + Ly, Sz = Sx OR Sy
x / y	DIV	Lz = Lx - Ly, Sz = Sx OR Sy
x^2	SQU	Lx << 1, Sz = Sx
x^0.5	SQRT	Lx >> 1, Sz = Sx
x^-1	RECIP	Lz = Lx, Sz = -Sx
x^-0.5	RSQRT	Lz = Lx >> 1, Sz = -Sx

For example, a value of λ = 0.99 implies *Q* = 7 bits. The numerical precision loss in this case is important. Assuming a word length *B* = 16 bits we can notice that the numerical precision representation is almost half reduced. Moreover, the small parameters from the algorithm (e.g., estimation errors) will be affected because they are "pushed" to the lowest limit 2^{-B+1}, increasing the stalling probability.

We should note that a value of λ very close to one is not unusual in practice. In general, mainly due to stability reasons, the weighting factor for this type of algorithms is greater than $\lambda = 1 - 1/3M$ [12]. Accordingly, a filter with the order *M* = 33 is sufficient to produce such a value of λ.

As an alternative to floating-point, the LNS offers the speed advantages when implementing algorithms with many multiplication, division, and square-roots; in the case of multiplication and division operations there are not rounding errors at all. These advantages are, however, offset by the problem of performing logarithmic addition and subtraction.

The 32-bit floating-point representation consists of a sign, 8-bit biased exponent, and 23-bit mantissa. The LNS format is similar in structure, as shown in Fig. 1. The 'S' bit again indicates the sign of the real value represented, with the remaining bits forming a 31-bit fixed-point word in which the size of the value is encoded as its base-2 logarithm in 2's complement format. Since it is not possible to represent the real value zero in the logarithmic domain, the 'spare' (most negative) code in the 2's complement fixed-point part is used for this purpose, which is convenient since smaller real values are represented by more negative log-domain values. The chosen format compares favorably against its floating-point counterpart, having greater range and slightly smaller representation error [8]. A 20-bit LNS format is similar. It maintains the same range as the 32-bit, but has precision reduced to 11 fractional bits. This is comparable to the 16-bit formats used on commercial DSP devices. The LNS arithmetic operations are presented in Table 1. More details about the LNS and some of its applications are available in [13], [14].

3. MODIFIED SRF-QRD-LSL ALGORITHM

In order to overcome the large dynamic range of the algorithm parameters we propose to update the cost functions in sort of "reverse" manner. Let us focus on the forward prediction part of the algorithm, where we can rewrite the cosine Givens rotation parameter as

$$c_m^f(n) = \frac{\lambda J_m^f(n-1)}{J_m^f(n)} = \frac{\lambda J_m^f(n-1)}{\lambda J_m^f(n-1) + \alpha_m(n-1)\left|e_m^f(n)\right|^2} = $$
$$= \frac{1}{1+\alpha_m(n-1)\left|e_m^f(n)\right|^2 \bar{J}_m^f(n-1)} \quad (4)$$

where

$$\bar{J}_m^f(n-1) = \frac{1}{\lambda J_m^f(n-1)} \quad (5)$$

Similarly, the sine Givens rotation parameters can be expressed as

$$s_m^f(n) = \frac{\alpha_m(n-1)e_m^f(n)}{\lambda J_m^f(n-1)\left[1+\dfrac{\alpha_m(n-1)\left|e_m^f(n)\right|^2}{\lambda J_m^f(n-1)}\right]} = $$
$$= \alpha_m(n-1)e_m^f(n)\bar{J}_m^f(n-1)c_m^f(n) \quad (6)$$

Finally, according to (4), the update of the modified cost function from (5) becomes

$$\frac{1}{\lambda J_m^f(n)} = \frac{1}{\lambda}\cdot\frac{1}{\lambda J_m^f(n-1)}\cdot\frac{1}{1+\dfrac{\alpha_m(n-1)\left|e_m^f(n)\right|^2}{\lambda J_m^f(n-1)}} \quad (7)$$

so that

$$\bar{J}_m^f(n) = \frac{1}{\lambda}\bar{J}_m^f(n-1)c_m^f(n) \quad (8)$$

The initial value of the modified cost function will be chosen as

$$\bar{J}_m^f(0) = \frac{1}{\lambda\delta} \quad (9)$$

The backward prediction part of the algorithm can be modified in a similar manner. In the ladder part of the algorithm the update of the a priori estimation error has to be rewritten as

$$\alpha_{m+1}(n) = \alpha_m(n-1) - \lambda\alpha_m^2(n-1)\left|e_m^f(n)\right|^2 \bar{J}_m^f(n) \quad (10)$$

Concluding, the first six relations from the lattice part of the original SRF-QRD-LSL algorithm described in Section 2 have to be changed as follows:

$$c_m^f(n) = \frac{1}{1+\alpha_m(n-1)\left|e_m^f(n)\right|^2 \bar{J}_m^f(n-1)}$$
$$\bar{J}_m^f(n) = \frac{1}{\lambda}\bar{J}_m^f(n-1)c_m^f(n)$$
$$s_m^f(n) = \lambda\alpha_m(n-1)e_m^f(n)\bar{J}_m^f(n)$$
$$c_m^b(n) = \frac{1}{1+\alpha_m(n)\left|e_m^b(n)\right|^2 \bar{J}_m^b(n-1)}$$
$$\bar{J}_m^b(n) = \frac{1}{\lambda}\bar{J}_m^b(n-1)c_m^b(n)$$
$$s_m^b(n) = \lambda\alpha_m(n)e_m^b(n)\bar{J}_m^b(n)$$

We may notice that a slight modification was performed in (6) according to (8). In addition, the last equation from the ladder part of the original algorithm (see Section 2) has to be replaced by (10); the initial value of the modified cost functions is given by (9). In this manner, it results a modified SRF-QRD-LSL (MSRF-QRD-LSL) algorithm.

The proposed algorithm is mathematical equivalent with the original one, so that they will have the same behaviour in infinite precision. Nevertheless, in a fixed-point arithmetic context they will behave differently, as we will demonstrate in the following.

As we have shown in Section 2, the cost functions of the original algorithm asymptotically grow to very large values, when the value of λ is close to one. Thus, hard scaling operations are required in order to avoid overflows. On the other hand, the modified cost functions of the proposed MSRF-QRD-LSL algorithm are updated in a reverse manner, so that they will asymptotically decrease to a lower bound, which can be computed as

$$\left.\bar{J}_m^f(n)\right|_{n\to\infty} = \frac{1-\lambda}{\lambda} \quad (11)$$

The maximum value of these functions will be the initial one given in (9). Consequently, we have to impose a scale factor such that

$$\frac{1}{\lambda\delta}2^{-S}=1 \tag{12}$$

which leads to

$$S=\left\lceil -\log_2\left(\lambda\delta\right)\right\rceil \tag{13}$$

The value of the initialization parameter δ slightly influences only the initial convergence of the algorithm [15], so that it is less important than the parameter λ. Typically, we can set δ = 1. Consequently, for a value of the weighting factor λ = 0.99, the value of the scale factor from (13) is $S\cong\lceil -0.015\rceil=1$ bit, which is insignificant. Thus, in practice we may set the initial value of the modified cost functions to one, so that there is almost no need for the scale factor *S* any more.

A second aspect that we have to take into account is related to the lower bound from (11). As we may notice from the update (8), the algorithm stalls when the modified cost functions decrease under the lowest limit 2^{-B+1}. In order to prevent this problem we have to impose

$$\frac{1-\lambda}{\lambda}\geq 2^{-B+1} \tag{14}$$

From (14), the condition for the maximum value of the weighting factor results as

$$\lambda\leq\frac{1}{1+2^{-B+1}} \tag{15}$$

For example, if the word length is *B* = 16 bits, we have to choose λ ≤ 0.999969, which is not a severe limitation. Concluding, if we initialize the cost functions with the value one and if we take into account condition (15), there is almost no need for scaling operations in the proposed MSRF-QRD-LSL algorithm. Consequently, its numerical robustness is improved as compared to the original SRF-QRD-LSL algorithm.

4. BACKGROUNDS OF ECHO CANCELLATION

In practice, there is a need for network echo cancellers for echo paths with long impulse response. Therefore, long FIR adaptive filters (e.g., $M\geq 256$) are required. It is well known that the longer impulse response implies slower convergence rate, thus rendering traditional algorithms like NLMS inadequate. Based on convergence performance alone, a RLS-based algorithm is clearly the algorithm of choice. However, the requirements of an echo canceller are for both rapid convergence and a low computational cost. Thus, a highly desirable algorithm is a low cost (i.e., fast) RLS algorithm. On the other hand, another consideration for this application is the algorithm stability, because it is unacceptable for the algorithm to diverge unexpectedly from the true solutions. Taking these two aspects into account, the proposed MSRF-QRD-LSL algorithm could be feasible for real-time applications, like network echo cancellation.

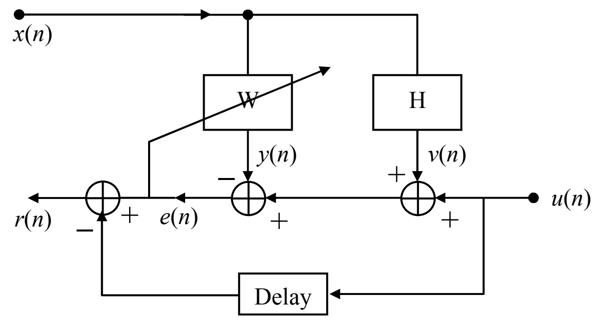

Fig. 2. Echo cancellation configuration.

Besides convergence rate and complexity issues, an important aspect of an echo canceller is its performance during double-talk. In the case of NLMS-based algorithms, the presence of near-end signal considerably disturbs the adaptive process. To eliminate the divergence of echo cancellers the standard procedure is to inhibit the weight updating during the double-talk. The presence of double-talk is detected by a DTD. A number of samples are required by the DTD to detect the double-talk presence. Nevertheless, this very small delay can generate a considerable perturbation of the echo estimate.

Let us consider the “interference cancellation” configuration from Fig. 2, which is the basis of echo cancellation. The purpose of the scheme is to extract the signal $u(n)$ from the mixture $u(n)+v(n)$. In the case of an echo canceller, $x(n)$ is the far-end signal, $u(n)$ is the near-end, H is the echo path equivalent to a FIR filter with the impulse response $\mathbf{h}(n)$, and W is an adaptive filter, having the coefficients $\mathbf{w}(n)$. As was suggested in [16], to make more apparent in results, it is convenient to subtract out the direct near-end component from the error signal $e(n)$. In this manner, the residual error $r(n)$ cumulates the undesired attenuation of the near-end signal $u(n)$ and the imperfect rejection of the echo path response $v(n)$. In a real application such a subtraction can never be done because the signal $u(n)$ is not available.

In the real case of any adaptive algorithm the coefficients $\mathbf{w}(n)$ depend on the signal $u(n)$. As a consequence, two effects appear [6], [10]:

- $\mathbf{w}(n)$ differs to $\mathbf{h}(n)$ in a certain extent and this may be viewed as a divergence of the algorithm; as a direct consequence, will result a decrease of the echo return loss enhancement (ERLE).
- $y(n)$ will contain a component proportional to $u(n)$, that will be subtracted from the received signal. This phenomenon is in fact a leakage of the $u(n)$ in $y(n)$,

through the error signal $e(n)$; the result consists of an undesired attenuation of the near-end signal.

In the case of the RLS-based algorithms, this leakage process is important for lower values of λ, where $y(n) \approx v(n) + u(n)$, and is practically absent for $\lambda \approx 1$, where $y(n) \approx v(n)$. On the other hand, when λ is very close to one, we deal with the finite-precision effects presented in Section 2. These practical aspects motivate the use of the proposed MSRF-QRD-LSL instead of the classical algorithm.

5. SIMULATION RESULTS

- *Fixed-point and LNS implementations*

For the first set of experimental results we consider a "system identification" configuration. In this class of applications an adaptive filter is used to provide a linear model that represents the best fit (in some sense) to an unknown system. The adaptive filter and the unknown system are driven by the same input. The unknown system output supplies the desired response for the adaptive filter. These two signals are used to compute the estimation error, in order to adjust the filter coefficients.

In a first experiment, the input signal is a random sequence with an uniform distribution on the interval (–1;1). The order of the adaptive filter is $M = 32$ and it is equal to the order of the system that has to be identified. We compare the original SRF-QRD-LSL algorithm presented in Section 2 with the proposed MSRF-QRD-LSL algorithm. The parameters of the algorithms were fixed to $\lambda = 0.99$ and $\delta = 1$. This set of simulations was run on a fixed-point DSP, using a word length $B = 16$ bits. The results are presented in Fig. 3. Following the discussions from Sections 2 and 3, it can be seen that the precision loss (because of the scale factors) disturbs the behavior of the SRF-QRD-LSL algorithm (Fig. 3 - upper; the error starts to grow after 3000 iterations). In the same context, the proposed MSRF-QRD-LSL algorithm achieves good performances (Fig. 3 - lower; the algorithm is stable after 3000 iterations) due to its improved numerical robustness.

A comparison of the algorithms performance on 32-bit floating point, 32-bit LNS, and 20-bit LNS implementations is performed in a similar "system identification" scheme. The input signal was generated as a first-order AR process with a correlation matrix eigenvalue spread of 20. The standard deviation of the input was 0.1 and the standard deviation of measurement noise was 0.001. The parameters of the algorithms were identical to those of the previous example. An accurate standard for comparison of the outputs of both algorithms LNS implementation was obtained by presenting the input data to their double precision versions and compute the absolute sum of errors of the 20-bit or 32-bit LNS outputs.

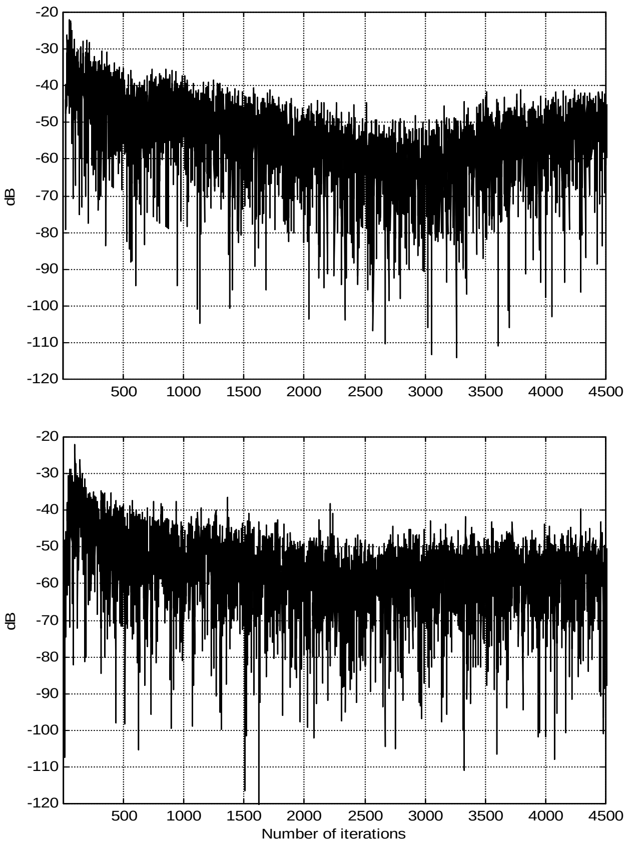

Fig. 3. Square error [dB], fixed-point $B = 16$ bits, $\lambda = 0.99$, δ=1, algorithms: (*upper*) SRF-QRD-LSL, (*lower*) MSRF-QRD-LSL.

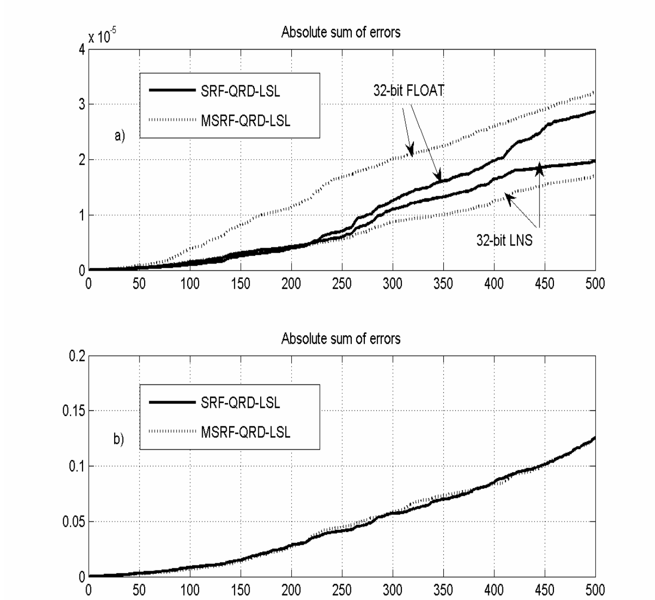

Fig. 4. (*upper*) The absolute sum of errors for 32-bit LNS and FLOAT implementations of the investigated algorithms; (*lower*) The absolute sum of errors for 20-bit LNS implementations of the investigated algorithms.

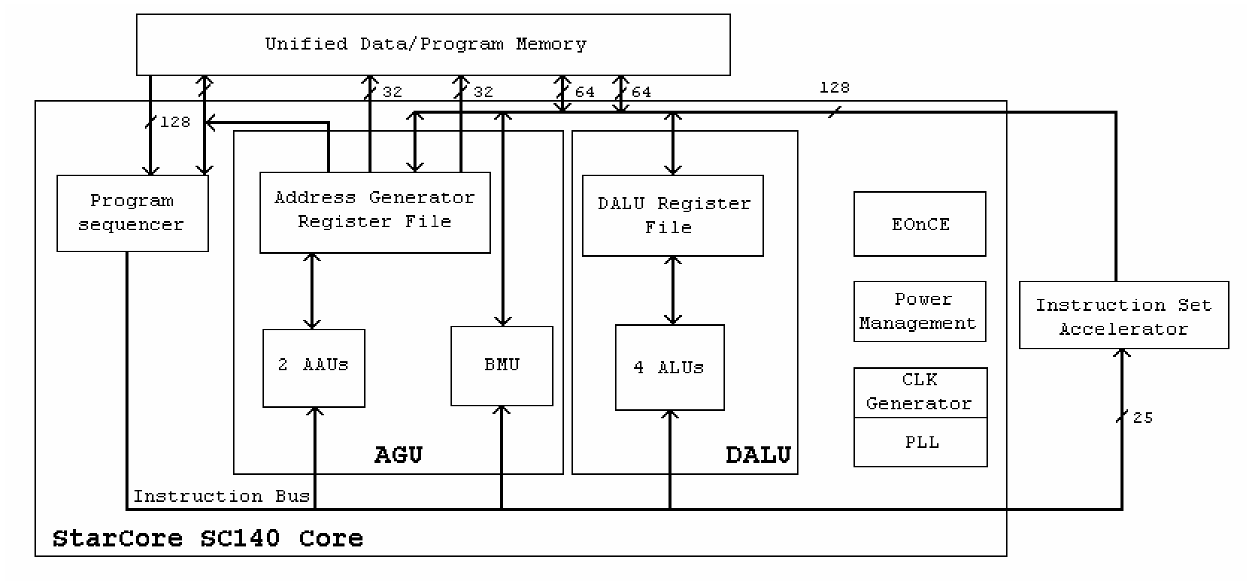

Fig. 5. Block Diagram of SC140 Core.

The 32-bit LNS and 32-bit floating-point results were virtually identical (see the amplitude of the error in Fig. 4 - upper). The absolute sum of errors for the 32-bit LNS implementation (solid line in Fig. 4 - upper) is smaller than that of the 32-bit floating-point implementation (dotted line). This is consistent with other results reported on [13] and [14] using a similar class of algorithms. It can be seen from Fig. 4 that the LNS implementations of both algorithms have only slightly different performances. As expected, the absolute sum of errors for the 20-bit LNS implementation is much higher than that of the 32-bit LNS implementation. However, for some applications that do not require much precision, the 20-bit LNS version could be an attractive alternative. As expected, the proposed algorithm does not offer advantages over the classical one when floating-point implementation is used.

- *DSP implementation of a network echo canceller*

For the practical implementation of the echo canceller we chose a well-known processor, i.e., Motorola SC140 DSP, with a large number of million instructions per second (MIPS) and a parallel architecture that allows several instructions to be executed simultaneously. In addition, we have structured the algorithm in a way to allow a high complexity algorithm to be run in real-time in a specific application. The specific features of this architecture [17] are the following (see Fig. 5):

- High level abstraction of the Application Software: applications development in C language; hardware supported integer and fractional data.
- Scalable performance: 4 Arithmetic logic Units (ALUs) and 2 Address Arithmetic Units (AAUs); 4 million multiply and accumulate operations per second (MMACS) for each megahertz of clock frequency.

The core important features are:

- Up to 10 RISC MIPS for each megahertz of clock frequency;
- A true (16*16) + 40 → 40-bit MAC unit in each ALU;
- A true 40-bit parallel barrel shifter in each ALU;
- 16 x 40-bit data registers for fractional and integer data operand storage;
- 16 x 32-bit address registers (8 can be used as 32-bit base address registers);
- 4 address offset registers and 4 modulo address registers;
- Unified data and program memory space (Harvard architecture);
- Byte addressable data memory.

However, the main feature that we have already mentioned is the C compiler and the ability to convert C source code into assembly code. The complexity of the MSRF QRD-LSL algorithm is quite large and therefore the need for flexibility is important, since programming in C code is much easier than implementing the algorithm direct in assembly code. The C compiler supports ANSI C standard and also intrinsic functions for ITU/ETSI primitives. Assembly code integration is also possible, which optimizes supplementary the code.

One of our main goals is to minimize the number of cycles needed by the algorithm per iteration, in order to lower the computational time per iteration under the sampling time of the CODEC. If we take advantage of the fact that the structure of the algorithm is symmetrical (i.e., similarities between the forward prediction structure and the backward prediction structure) then we can use two identical blocks for each lattice cell; thus, we can call twice a function in C language during one iteration. The filtering part is included in backward prediction part and is performed if a flag is set. This flag is set before the backward prediction and reset before the forward prediction. Another optimization technique accomplished using this procedure is that all the transformations are made in-place, regardless of the iteration (i.e., the moment of time), saving a large amount of memory. Choosing an appropriate level of optimization from the C compiler, i.e., Code Warrior (0 – 3), makes further optimization. As well, the proper use of intrinsic functions from C compiler can further reduce the number of cycles.

The standard ITU-T G.168 [11] recommends certain test procedures for evaluating the performance of echo cancellers. Test signals used are so-called composite source signals (CSS) that have properties similar to those of speech with both voiced and unvoiced sequence as well as pauses. Moreover, we choose a long network echo path (i.e., 64 ms) according to the same above recommendation. The impulse response and the corresponding magnitude function of the hybrid are shown in Fig. 6. The echo return loss (ERL) of this echo path is about 10 dB and it is considered typical.

The first experiment refers to the convergence rate and echo return loss enhancement. In Figs. 7 and 8, we present the convergence results obtained by the NLMS (using a normalized step-size $\mu = 1$) and the MSRF-QRD-LSL (with $\lambda = 0.9999$) algorithms. For ITU recommendation G.168 testing purposes, the method defined for measuring the level of the signals is a root mean square (RMS) method, using a 35 ms rectangular sliding window. The measurement device comprises a squaring circuit and an exponential filter (35 ms, 1-pole). The following conclusions are obvious:

- the convergence speed and ERLE are clearly superior in the case of the MSRF-QRD-LSL algorithm;
- test 1 (steady-state and residual echo level test) is by far fulfilled in the case of the MSRF-QRD-LSL algorithm;
- the requirements of the recommendation test 2B (convergence speed) are accomplished in the case of the MSRF-QRD-LSL algorithm for time far less than one second.

The second experiment was performed using speech as excitation signals in order to simulate a real-world conversation. It evaluates the performances of the echo canceller for a high level double-talk sequence (similar to test 3B). The double-talk level in this case is about the same as that of the far-end signal. The results are presented in Figs. 9 and 10. In the case of SRF QRD-LSL algorithm one can see that the near-end signal $u(n)$ is recovered in $e(n)$ with slight distortions. The NLMS based echo canceller (using Geigel DTD [18]) fails in this situation because double-talk appears during initial convergence phase so that the adaptive process is prematurely inhibited. Let us remind that we do not use any DTD in our echo canceller based on the MSRF-QRD-LSL algorithm.

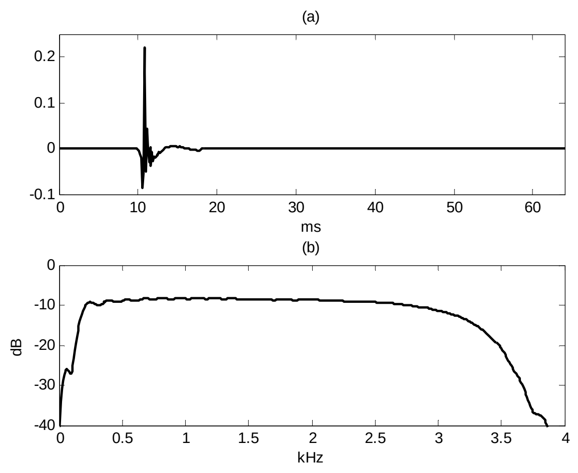

Fig. 6. Network echo path characteristics: (a) impulse response; (b) frequency response.

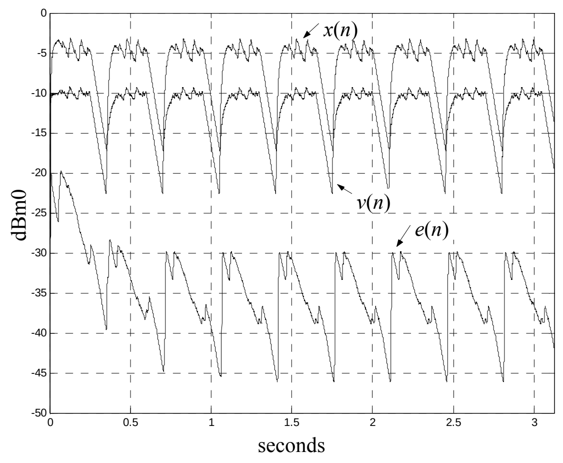

Fig. 7. Power levels [dBm0] for the far-end signal $x(n)$ (CSS), the echo signal $v(n)$, and the error signal $e(n)$, in the case of the NLMS algorithm.

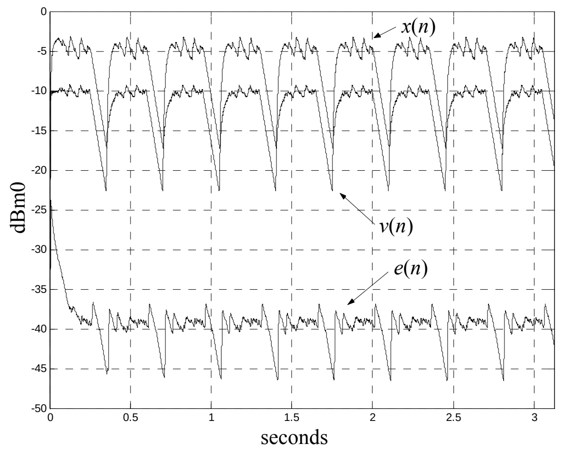

Fig. 8. Power levels [dBm0] for the far-end signal $x(n)$ (CSS), the echo signal $v(n)$, and the error signal $e(n)$, in the case of the MSRF-QRD-LSL algorithm.

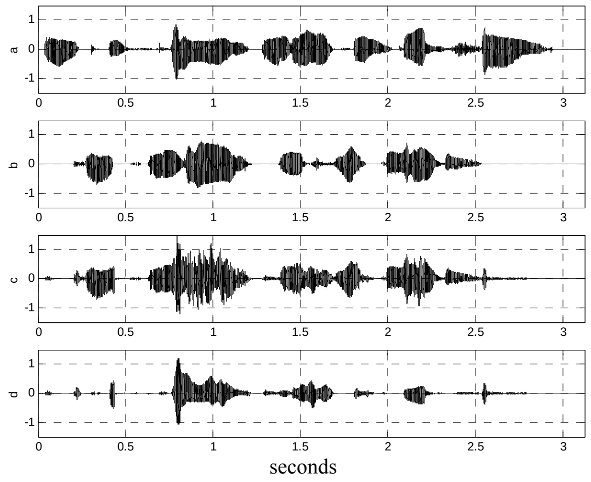

Fig. 9. Performances of the NLMS algorithm for speech signals, during double-talk: (a) far-end signal *x*(*n*); (b) near-end signal *u*(*n*); (c) recovered near-end signal in *e*(*n*); (d) residual error *r*(*n*).

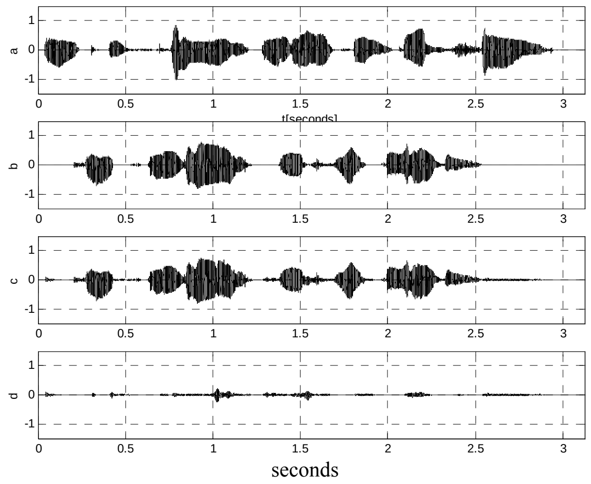

Fig. 10. Performances of the MSRF-QRD-LSL algorithm for speech signals, during double-talk: (a) far-end signal *x*(*n*); (b) near-end signal *u*(*n*); (c) recovered near-end signal in *e*(*n*); (d) residual error *r*(*n*).

Also, in our experiments, we have not used any non-linear processor (NLP) [11]. A NLP placed in the send path of the echo canceller will produce an additional attenuation of the residual echo level during the silent periods of the near-end talker, improving the overall performances.

Other experiments (which are not shown here) were performed using more "difficult" echo paths (ERL up to 6dB). It should be noted that 6 dB is a typical worst case value encountered for most networks, and most current networks have typical ERL values better than this. Nevertheless, it was obvious the superior convergence rate and double-talk robustness of the echo canceller based on the MSRF-QRD-LSL adaptive algorithm.

6. CONCLUSIONS

The SRF-QRD-LSL algorithm is an attractive choice in many adaptive systems, due to its fast convergence rate and good numerical properties. Nevertheless, it faces some limitations in fixed-point implementations, where the scaling operations required by the large values of the cost functions affect its performances.

In this paper, we have proposed a modified version of the SRF-QRD-LSL algorithm, based on a modified update of the cost functions. In our approach, these parameters are computed in a reverse manner, preventing the use of the scaling procedures. The modified cost functions decrease to a lower bound, so that the maximum value is the initial one. In this case, milder scaling conditions have to be imposed as compared to the case of the original algorithm. Using a proper initialization, the proposed algorithm does not need for scaling operations any more. Consequently, its numerical robustness is significantly improved.

The simulations performed in a fixed-point DSP context support the theoretical findings. Moreover, the proposed algorithm proved to have efficient LNS implementation as well, due to the important number of multiplications and divisions.

The proposed MSRF-QRD-LSL algorithm was tested in the context of network echo cancellation. The experimental results showed that the requirements of the ITU-T G.168 recommendation concerning the steady-state echo return loss enhancement and convergence speed are fulfilled. The most relevant result is that the MSRF-QRD-LSL algorithm could satisfactorily operate even in the absence of the DTD, which is a very important issue in echo cancellation.

7. REFERENCES

[1] I. K. Proudler, J. G. McWhirter, and T. J. Shepherd, "QRD-based lattice filter algorithms," *Proc. SPIE*, 1989, vol. 1152, pp. 56-67.

[2] S. Haykin, *Adaptive Filter Theory - Fourth Edition*, Prentice Hall, Upper Saddle River, NJ, 2002.

[3] Ph. Regalia, "Numerical stability properties of a QR-based fast least squares algorithm," *IEEE Trans. Signal Process.*, vol. 41, no. 6, pp. 2096-2109, June 1993.

[4] L. M. Davis, "Scaled and decoupled Cholesky and QR decompositions with application to spherical MIMO detection," *Proc. IEEE WCNC*, 2003, vol. 1, pp. 326-331.

[5] S. R. Muruganathan and A. B. Sesay, "A QRD-RLS-based predistortion scheme for high-power amplifier

linearization," *IEEE Trans. Circ. Syst-II*, vol. 53, no. 10, pp. 1108-1112, Oct. 2006.

[6] S. Ciochină and C. Paleologu, "On the performances of QRD-LSL adaptive algorithm in echo cancelling configuration," *Proc. IEEE ICT 2001*, Bucharest, Romania, 2001, vol.1, 563-567.

[7] E. N. Frantzeskakis and K. J. R. Liu, "A class of square root and division free algorithms and arhitectures for QRD-based adaptive signal processing," *IEEE Trans. Signal Process.*, vol. 42, no. 9, pp. 2455-2469, Sep. 1994.

[8] J. Coleman, S. Christopher, J. Kadlec, R. Matousek, M. Tichy, Z. Pohl, A. Hermanek, and N. Benschop, "The European Logarithmic Microprocessor," *IEEE Trans. Computers*, vol. 57, no. 4, pp. 532-546, Apr. 2008.

[9] C. Paleologu, F. Albu, A. A. Enescu, and S. Ciochină, "Square-root-free QRD-LSL adaptive algorithm with improved numerical robustness," *Proc. ICN 2008*, Cancun, Mexic, 2008, pp. 572-577.

[10] C. Paleologu, S. Ciochină, and A. A. Enescu, "A simplified QRD-LSL adaptive algorithm in echo cancelling configuration," *Proc. IEEE ICT 2002*, Beijing, China, 2002, vol. 1, pp. 240-244.

[11] ITU-T Recommendation G.168, *Digital Network Echo Cancellers*, 2002.

[12] D. G. Manolakis, V. K. Ingle, and S. M. Kogon, *Statistical and Adaptive Signal Processsing*, Artech House, Boston, U.S.A., 2005.

[13] F. Albu, J. Kadlec, N. Coleman, and A. Fagan, "Pipelined Implementations of the Modified EF-LSL Algorithm," *Proc. IEEE ICASSP2002*, Orlando, U.S.A, 2002, pp. 2681-2684.

[14] F. Albu, C. Paleologu, and S. Ciochină, "Analysis of LNS implementation of QRD-LSL algorithms," *Proc. IEEE CSNDSP 2002*, Stafford, U.K., 2002, pp. 364-367.

[15] S. Ciochină, C. Paleologu, and A. A. Enescu, "On the behaviour of RLS adaptive algorithm in fixed-point implementation," *Proc. IEEE ISSCS 2003*, Iaşi, Romania, 2003, vol. 1, pp. 57-60.

[16] D. L. Duttweiler, "Proportionate normalized least-mean-squares adaptation in echo cancelers," *IEEE Trans. Speech and Audio Proc.*, vol. 8, no. 5, pp. 508-518, Sept. 2000.

[17] *Motorola SC140 DSP Core*, Reference Manual, Revised 1, 6/2000.

[18] D. L. Duttweiler, "A twelve-channel digital echo canceler," *IEEE Trans. Commun.*, vol. 26, no. 5, pp. 647-653, May 1978.