Performance Analysis of Multichannel Lattice Equalization in Coherent Underwater Communications

João Gomes
Instituto Superior Técnico
Instituto de Sistemas e Robótica
Av. Rovisco Pais, 1049–001 Lisboa, Portugal
jpg@isr.ist.utl.pt

António Silva, Sérgio Jesus
SiPLAB — Universidade do Algarve
Campus de Gambelas, 8005–139 Faro, Portugal
asilva@ualg.pt, sjesus@ualg.pt

***Abstract*—This work examines the numerical fixed-point performance of a new multichannel lattice RLS filtering algorithm using data from two underwater acoustic communication experiments. The algorithm may be an appealing choice for underwater equalization due to its robust numerical behavior and linear scaling of the computational complexity with filter order. Simple modifications to widely-used methods for carrier/timing synchronization and symbol slicing in transversal equalizers are proposed. Experimental results show that the algorithm is as accurate as the similarly array-based QR-RLS, tolerating word lengths as low as 16-20 bits with minor degradation relative to floating-point benchmarks. These features, coupled with a very modular and regular structure, are highly desirable in energy-efficient hardware or embedded implementations.**

I. Introduction

Recently, the implementation of signal processing algorithms in dedicated hardware has gained new momentum due to the availability of (*i*) large-capacity FPGAs that incorporate hardware multipliers and other useful features for mathematical computation, and (*ii*) tools that can automatically generate reasonably-efficient HDL code from block diagrams or other high-level descriptions. Such power-efficent and potentially highly-parallel solutions are certainly appealing for equalization of underwater channels, where the number of adjustable parameters in the adaptive filter may be large due to long impulse responses and the use of multiple hydrophones at the receiver for improved diversity.

Attaining fast convergence and tracking with high filter orders requires efficient adaptive algorithms such as recursive least-squares (RLS) or adaptive step-size least-mean-squares (SLMS) [1]. Currently, the latter seems to be more popular in underwater communications due to its lower computational complexity with only a modest degradation in performance relative to RLS. In fact, the complexity scales linearly with filter order in SLMS, whereas it increases quadratically in basic versions of RLS. Moreover, the input covariance matrix in plain RLS tends to be ill-conditioned when fractional sampling or multiple receivers are used, especially with high filter orders, and in turn this leads to high dynamical ranges in its internal variables. Accomodating such numerical excursions in fixed-point arithmetic requires large word lengths, and hence more FPGA real estate and power consumption.

Research into efficient RLS algorithms has a long history, and numerous fast variants are known where the complexity scales linearly rather than quadratically [1], [2]. Many of these, particularly fast transversal forms, tend to exhibit even poorer numerical behavior than plain RLS under moderate-to-low numerical precision. Within the class of lattice forms, however, there exist efficient algorithms that exhibit much more favorable performance. While they require more operations per computed output than transversal forms do, their robustness and simple modular structure make them attractive for hardware or embedded implementations. Arguably, the most stable lattice algorithms currently available are those where the time/order recursions in each module are written in square-root array form and updated using unitary transformations (typically, Givens rotations). Actually, square-root formulations via Givens-based QR decomposition are not restricted to lattice forms; the QR-RLS algorithm derives from the general RLS problem through the same approach, and its numerical properties are similar to those of QR-based lattice filters.

The main goal of this paper is to examine the behavior of a recently-proposed multichannel RLS lattice algorithm [3], [4] in the context of coherent underwater communications. The algorithm is based on a modular decomposition approach that enables unequal channel orders to be specified and leads to a filter that is structured as an array of interconnected scalar units which operate similarly to those found in conventional single-channel lattice filters. This avoids cumbersome matrix operations that are difficult to implement in hardware and not easily parallelizable. Each scalar unit autonomously performs input-output mapping and internal updating, so that a fully-parallel implementation may be obtained by adding pipeline registers between units.

The input/output mapping of lattice filters is somewhat more complex than the inner products found in transversal filters, which makes it difficult to apply some popular methods for timing and carrier and recovery based on joint MMSE

0-933957-35-1

optimization of equalizer and synchronization parameters [5]. This work examines alternative strategies and, in particular, proposes a technique for implementing the rotating slicer that is found in the quasi-standard (transversal) DFE+PLL architecture. A low-complexity method for slicing symbols in decision-directed mode (i.e., when operating beyond the training period) is also presented.

The fixed-point performance of the array-based multichannel lattice is illustrated using data from two underwater communication experiments:

- The MREA'04 (Maritime Rapid Environmental Assessment) sea trial was conducted in the continental shelf off the west coast of Portugal in April 2004. Symbol rates of 200 and 400 baud were used, and data transmitted over ranges up to $2\mathrm{km}$, resulting in relatively mild intersymbol interference (ISI).
- The MAKAI'05 experiment took place off the island of Kauai, Hawaii, in September/October 2005. Data were transmitted over ranges of 1 to $4\,\mathrm{km}$ at a symbol rate of $2000\mathrm{baud}$. For a multipath structure comparable to that of MREA'04, this leads to longer ISI and more challenging equalization.

The proposed lattice algorithm is shown to exhibit robust behavior under fixed-point arithmetic, even for word lengths as short as 16–20 bits. Its performance is on a par with a generic array-based formulation of RLS (QR-RLS), but for a given number of input channels the complexity increases linearly with the filter order, rather than quadratically. Plain RLS requires far greater precision than both of these algorithms to avoid numerical instability.

II. Multichannel Filtering

Throughout, vectors and matrices will be represented by lowercase boldface and uppercase boldface letters, respectively. The notations $(\cdot)^T$ and $(\cdot)^*$ stand for transpose and complex conjugate transpose (hermitian).

In the multichannel setup L input channels convey discrete-time signals $u^{(i)}$, $1 \le i \le L$, that are observed over a period of time and linearly combined to approximate a desired output (reference signal) d at time n as $\hat{d}(n) = \mathbf{w}^*\mathbf{u}(n)$, where the input sample vector $\mathbf{u}(n)$ is given by

$$\mathbf{u}(n) = \begin{bmatrix} \mathbf{u}^{(1)}(n) \\ \cdots \\ \mathbf{u}^{(L)}(n) \end{bmatrix}, \quad \mathbf{u}^{(i)}(n) = \begin{bmatrix} u^{(i)}(n) \\ \cdots \\ u^{(i)}(n-m_i+1) \end{bmatrix}, \quad (1)$$

and the filter order m_i need not be the same for all channels. The optimal coefficient vector $\mathbf{w}$ minimizes the exponentially-weighted least-squares (LS) cost function

$$J(n) = \left\| \mathbf{d}(n) - \mathbf{U}(n)\mathbf{w} \right\|^2 , \quad (2)$$

with

$$\mathbf{d}(n) = \mathbf{\Lambda}^{\frac{1}{2}}(n) \begin{bmatrix} d^*(0) \\ \cdots \\ d^*(n) \end{bmatrix}, \quad \mathbf{U}(n) = \mathbf{\Lambda}^{\frac{1}{2}}(n) \begin{bmatrix} \mathbf{u}^*(0) \\ \cdots \\ \mathbf{u}^*(n) \end{bmatrix}, \quad (3)$$

and $\mathbf{\Lambda}(n) = \mathrm{diag}(\lambda^n, \ldots \lambda^0)$. RLS algorithms take advantage of the structure of $\mathbf{U}(n)$ to incrementally adjust $\mathbf{w}$ at each time step.

III. Modular Filter Structure

The multichannel filter structure has been presented in [3], [4], and will only be briefly reviewed here. In the scalar case unit m in the lattice addresses the LS problem $J_m(n) = \left\| \mathbf{d}(n) - \mathbf{U}_m(n)\mathbf{w}_m \right\|^2$ for input vectors containing the m most recent samples at each time instant. Order recursions are derived by partitioning the input matrix $\mathbf{U}_m$ in different ways to separate its columns containing the most recent or oldest samples. The desired LS projection of $\mathbf{d}$ is then expressed in terms of the one performed by unit $m-1$, as well as forward and backward linear prediction residuals of the leftmost/rightmost columns of the input matrix [2].

The same reasoning can be directly extended to multiple input channels by considering multichannel linear prediction. When formulated using QR decomposition this approach is known to yield array algorithms that bear strong similarities to the one described here [6], [7], including the same block layout (see Fig. 1a). The present work follows the alternative modular decomposition approach of [8], whereby a sequence of L parallel *chains* of scalar units interact over lattice blocks comprising L *stages*. In each stage the prediction order is increased by 1 in one of the input channels in such a way that after L stages the overall filter order is increased in all channels. Fig. 1b depicts the structure of one of these multichannel lattice blocks for $L = 3$. The interconnections between these elementary units follow directly from the decomposition, as explained next.

Let $\mathbf{u}_{i,0}$ denote the input vector corresponding to the LS problem solved at the output of the previous lattice block in the i-th chain. The actual number of samples in any of the L input channels that make up $\mathbf{u}_{i,0}$ is unimportant for deriving the lattice recursions. This vector is assumed to be updated up to time n in channels $1, \ldots i$, but only up to time $n-1$ in the remaining ones,

$$\mathbf{u}_{i,0}(n) = \Big[\mathbf{u}^{(1)\,T}(n) \ \ldots \ \mathbf{u}^{(i)\,T}(n) \ \mathbf{u}^{(i+1)\,T}(n-1) \ \ldots \ \mathbf{u}^{(L)\,T}(n-1) \Big]^T . \quad (4)$$

Unit $(i+1, 1)$ in the first stage of the current lattice block solves a LS problem whose input vector $\mathbf{u}_{i+1,1}(n)$ is obtained from $\mathbf{u}_{i,0}(n)$ by incorporating the most recent sample $u^{(i+1)}(n)$, which increases the filter order by 1 in channel $i+1$. The same holds in general when going from $\mathbf{u}_{i,l-1}(n)$ to $\mathbf{u}_{i+1,l}(n)$ for any stage l of the lattice block. As in a scalar lattice, update recursions for cell (i, l) are based on forward and backward linear prediction of selected elements of $\mathbf{u}_{i,l}(n)$ from $\mathbf{u}_{i-1,l-1}(n)$ and $\mathbf{u}_{i,l-1}(n)$. Specifically, the forward prediction (f), backward prediction (b) and reference

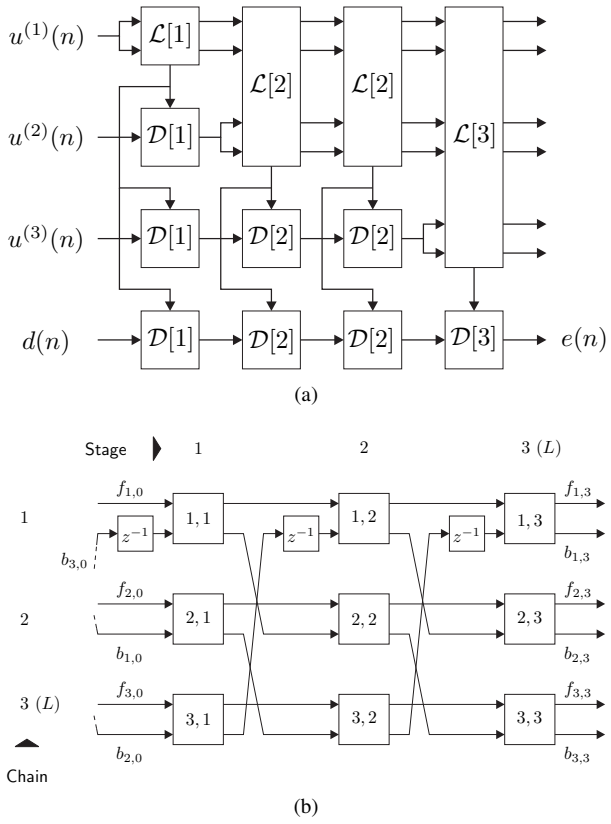

Fig. 1: Modular decomposition-based multichannel lattice filter (a) Filter structure for channel orders $m_1 = 4$, $m_2 = 3$, $m_3 = 1$ (b) Internal connections of a lattice block for $L = 3$ ($\mathcal{L}[3]$)

filtering (e) *a posteriori* errors turn out to satisfy the recursions

$$b_{i,l}(n) = b_{i-1,l-1}(n) + \kappa_{i,l}^{b\,*}(n) f_{i,l-1}(n)\,, \tag{5}$$

$$f_{i,l}(n) = f_{i,l-1}(n) + \kappa_{i,l}^{f\,*}(n) b_{i-1,l-1}(n)\,, \tag{6}$$

$$e_l(n) = e_{l-1}(n) - \kappa_l^*(n) b_{L,l-1}(n)\,, \tag{7}$$

where the various κ denote reflection coefficients. The following conclusions can be drawn from an analysis of (5)–(7) and a set of time update expressions:

1) The modular decomposition approach dictates which variables appear in each of the above equations, and hence how a lattice block is structured into scalar units and how these are interconnected. As mentioned previously, Fig. 1b follows from (5)–(6).
2) The generic update recursions carried out by a single unit turn out to be algebraically equivalent to those in a single-channel lattice unit. This implies that any of several known sets of equivalent update recursions may be selected. In [8] a mixed *a priori/a posteriori* scheme with error feedback was proposed, whereas [3] opted for QR-array-based adaptation which, similarly to the single-channel case, turns out to provide superior numerical performance.

This kind of separation between filter structure and update relations has been noted in [9] under fairly general conditions.

Regarding the overall structure shown in Fig. 1a, it assumes that input channels are sorted in descending filter order, such that the total number of lattice blocks equals the order of the first one. Lower orders in other channels are attained by incorporating them later in the processing chain; if order m_i is required in channel i, then it should pass through a total of m_i blocks. For seamless integration of an input channel into the lattice at a given point of the chain the signal must be uncorrelated with the underlying LS data matrix at the output of previous blocks. As it turns out, this can be achieved by actually incorporating a prediction residual that is calculated in exactly the same manner as the e_l for the reference signal. Accordingly, at each point of the cascade Fig. 1a shows that the reference (bottom) and all yet-unmerged channels are filtered by a set of identical ladder blocks $\mathcal{D}[\cdot]$.

A. Array Algorithm

The update recursions for individual units are written in array form, i.e., forward or backward (2×2) prearrays are built from each unit's internal variables, a Givens rotation is applied to create a postarray where a specific element is zeroed-out, and time-updated variables are read from other entries in the array. The array algorithm uses so-called angle-normalized errors $\epsilon_{i,l}^f$, $\epsilon_{i,l}^b$, ϵ_l, that equal their *a posteriori* counterparts divided by the square-root of the associated conversion factors [2]. In the forward case, for example, $\epsilon_{i,l}^f(n) = \gamma_{i-1,l}^{-1/2}(n) f_{i,l}(n)$. Reflection coefficients are similarly normalized, yielding variables $p_{i,l}^f$, $p_{i,l}^b$ and p_l. The backward array of the first chain is augmented with an additional line that propagates the conversion factor needed to regenerate *a priori* or *a posteriori* errors throughout the reference chain. Table I summarizes the array-based modular QR-RLS algorithm (abbreviated as MQR below), where F, B represent forward/backward prediction error energies, and $\mathbf{\Theta}$ denote 2×2 complex Givens matrices. Explicitly computing the Givens matrices of Tab. I requires evaluating inverse square-roots [2]. In fixed-point arithmetic this can be accomplished by the Newton algorithm, which typically converges in about 5 iterations [3]. Alternatively, CORDIC processors can carry out Givens rotations without explicitly computing the sine/cosine parameters [10]. The CORDIC algorithm is well suited to simple hardware implementations, as it enables Givens rotations to be iteratively performed using only basic add/shift operations. The extension of CORDIC to complex data used here is described in [4].

Finally, note that for a least-squares problem with L channels and M adjustable coefficients per channel the filter complexity is proportional to ML^2, the number of scalar lattice units, rather than $(ML)^2$ as in plain RLS.

IV. Lattice-based Equalization

As in other feedforward lattice filters all intermediate estimation errors are readily available along the block cascade,

TABLE I: Summary of the modular multichannel QR-RLS algorithm in array form (MQR)

Initialization: Determine lattice/ladder block sizes L_s according to Sec. III and choose $0 < \lambda \leq 1$. For unit (i,l) in lattice block s set

$$p^{f\,*}_{i,l}(-1) = p^{b\,*}_{i,l}(-1) = 0\,, \quad F^{1/2}_{i,l-1}(-1) = B^{1/2}_{i-1,l-1}(-1) = \sqrt{\delta}\,,$$

for small $\delta > 0$. Set $\mathbf{\Theta}^b_{1,l}(-1) = \mathbf{I}_2$, $\epsilon^b_{L_s,l-1}(-1) = 0$ throughout chain 1 and $p^*_l(-1) = 0$ in the units of all ladder blocks.

Update recursions: At time n set the filter input as

$$\begin{aligned} &\epsilon^f_{i,0}(n) = \epsilon^b_{i,0}(n) = u^{(i)}(n)\,, && 1 \leq i \leq L_1\,,\\ &\epsilon_0(n) = u^{(i)}(n)\,, && L_1+1 \leq i \leq L\,,\\ &\epsilon_0(n) = d'(n)\,, \quad \gamma^{1/2}_{L_1,0}(n) = 1\,, && \text{reference}\,. \end{aligned}$$

In lattice block $s > 1$ incorporate new channels as described in Sec. III.

Lattice update: Compute Givens matrices and update lattice units as

$$\begin{bmatrix} \lambda^{1/2} F^{1/2}_{i,l-1}(n-1) & \epsilon^f_{i,l-1}(n) \\ \lambda^{1/2} p^{b\,*}_{i,l}(n-1) & \epsilon^b_{i-1,l-1}(n) \end{bmatrix} \mathbf{\Theta}^f_{i,l} = \begin{bmatrix} F^{1/2}_{i,l-1}(n) & 0 \\ p^{b\,*}_{i,l}(n) & \epsilon^b_{i,l}(n) \end{bmatrix}$$

$$\begin{bmatrix} \lambda^{1/2} B^{1/2}_{i-1,l-1}(n-1) & \epsilon^b_{i-1,l-1}(n) \\ \lambda^{1/2} p^{f\,*}_{i,l}(n-1) & \epsilon^f_{i,l-1}(n) \end{bmatrix} \mathbf{\Theta}^b_{i,l} = \begin{bmatrix} B^{1/2}_{i-1,l-1}(n) & 0 \\ p^{f\,*}_{i,l}(n) & \epsilon^f_{i,l}(n) \end{bmatrix}$$

In chain $i = 1$ use the precomputed (delayed) Givens matrix to update the second row of the backward prediction postarray above, then update the Givens matrix, prediction energy and conversion factor as

$$\begin{bmatrix} \lambda^{1/2} B^{1/2}_{L_s,l-1}(n-1) & \epsilon^b_{L_s,l-1}(n) \\ 0 & \gamma^{1/2}_{L_s,l-1}(n) \end{bmatrix} \mathbf{\Theta}^b_{1,l} = \begin{bmatrix} B^{1/2}_{L_s,l-1}(n) & 0 \\ \times & \gamma^{1/2}_{L_s,l}(n) \end{bmatrix}$$

Ladder update: Ladder arrays share their first row with chain 1. Use the updated Givens matrix $\mathbf{\Theta}^b_{1,l}$ to compute the second row

$$\begin{bmatrix} \lambda^{1/2} p^*_l(n-1) & \epsilon_{l-1}(n) \end{bmatrix} \mathbf{\Theta}^b_{1,l} = \begin{bmatrix} p^*_l(n) & \epsilon_l(n) \end{bmatrix}$$

Filter output: Obtain the *a priori* or *a posteriori* filter output from the final ladder block in the reference chain as $\xi_L(n) = \gamma^{-1/2}_{L,L}(n)\epsilon_L(n)$ or $e_L(n) = \gamma^{1/2}_{L,L}(n)\epsilon_L(n)$, respectively.

which is very useful for solving the often overlooked, but practically relevant, problem of selecting an appropriate equalizer order for a particular channel whose characteristics are not known beforehand. Decision-feedback equalizers (DFE) may be accommodated with this structure by inserting previously-decoded symbols into an additional input channel. Allowing for different channel orders turns out to be very useful because the required filter orders for acoustic inputs and previous symbols (processed by feedforward and feedback filters, respectively, in the usual transversal DFE architecture) can be quite disparate.

In addition to mitigating ISI, practical equalizers should include a number of auxiliary subsystems for synchronization (symbol and carrier) and slicing of the equalizer output when operating in decision-directed mode. This section discusses some issues where the technical solutions that are commonly used for transversal filters must be adapted to the lattice structure.

A. Carrier Recovery

A popular technique for carrier recovery in coherent underwater communications jointly optimizes the equalizer coefficients and estimated carrier phase offset to minimize the output MSE [5]. A phase rotator is placed at the output of the DFE feedforward filter, and iterative optimization of carrier phase is performed by computing the gradient of the MSE with respect to this parameter and using it as a phase error driving a first- or second-order PLL. The positioning of this phase rotator leads to expressions where the output of the feedforward filter appears explicitly, a requirement that is easy to meet in transversal DFEs but much less so in a lattice implementation where acoustic signals and previous decisions are mingled together.

A more suitable alternative is to turn the slicer into a rotating one, i.e., to allow the equalizer output to rotate freely, perform phase compensation before formulating symbol decisions (slicing), and then rotating the decisions back before feeding them to the DFE. Formally,

$$\tilde{d} = Q\{z \exp -j\theta\}\,, \qquad d' = \tilde{d} e^{j\theta}\,, \tag{8}$$

where θ is the estimated carrier phase, z denotes the equalizer output, $\tilde{d}$ is the symbol decision using slicer mapping $Q\{\cdot\}$, and d' is the rotated symbol fed back to the DFE. Assuming correct decisions, $\tilde{d} = d$, the angle θ is adjusted by stochastic gradient descent of the MSE cost function

$$J(\theta) = E\{\left|d - z \exp -j\theta\right|^2\}\,, \tag{9}$$

$$\frac{dJ}{d\theta} = -\mathrm{Im}\{E\{d^* z\} \exp -j\theta\}\,. \tag{10}$$

Similarly to [5], a practical second-order PLL update is then given by

$$\theta(n+1) = \theta(n) + K_P \Phi(n) + K_I \sum_{i=0}^{n} \Phi(i) \tag{11}$$

$$\Phi(n) = \mathrm{Im}\{d^*(n)\, z(n) \exp -j\theta(n)\}\,. \tag{12}$$

Other variants for the PLL phase detector (12) are discussed in [11]. Throughout this work the loop filter proportional and integral constants were empirically set to $K_P = 10^{-1}$, $K_I = 1.7 \times 10^{-3}$. Using (8), again assuming perfect decisions, a slightly more convenient expression is obtained for the phase detector in a lattice filter that computes the error e, rather than z,

$$\Phi = \mathrm{Im}\{z d'^*\} = \mathrm{Im}\{(d' - e) d'^*\} = -\mathrm{Im}\{e d'^*\}\,. \tag{13}$$

B. Slicing

In a transversal equalizer operating in decision-directed mode the *a priori* output is computed from the previous coefficient vector, and then the closest point of the signal constellation is chosen as the current decision to update the coefficients. In an array-based lattice equalizer this is not quite possible because the adaptation algorithm of Tab. I only computes the *a priori* output error $\xi(n)$ at the end of the iteration at time n. This would seem to create a kind of chicken-and-egg problem, as the current external reference $d'(n)$ must be externally supplied before computing the error.

As the reference chain does not influence the state of lattice blocks, a simple workaround would be to update the lattice

for the current set of multichannel inputs, then sequentially apply all points of the signal constellation to the reference input, commit to the one producing minimum *a priori* or *a posteriori* output error, and update the internal units in the reference chain accordingly. Note that this can be done because the reference chain is memoryless, i.e., once the feedback errors on which it operates are fixed the output becomes an instantaneous function of the reference input. This strategy can be further improved because the output error is actually a *linear* function of the reference. To see this remark that for a (complex) Givens rotation matrix of the form

$$\mathbf{\Theta} = \begin{bmatrix} c & -s \\ s^* & c \end{bmatrix} , \tag{14}$$

where $c \in \mathbb{R}$ and $s \in \mathbb{C}$ are the cosine and sine parameters [1], the expression for order-updating $\epsilon_l(n)$ in Tab. I reads

$$\epsilon_l(n) = c_l \epsilon_{l-1}(n) - \lambda^{1/2} s_l p_l^*(n-1) . \tag{15}$$

Applying it recursively to all blocks and stages in the reference chain produces the input/output mapping, for $\epsilon_0(n) = d'(n)$ (recall (8)),

$$\epsilon(n) = g d'(n) + h . \tag{16}$$

Determining the constants g and h above would only require applying two distinct values to the reference chain. However, the slope parameter in (16) satisfies $g = \prod_l c_l$, and can readily be shown to coincide with the final value $\gamma_{L,L}^{1/2}$ computed by the first lattice chain. Hence, all that is really needed to complete the characterization of this linear mapping is to apply 0 at the reference input and read h at the output. Then, for a (rotated) constellation $\mathcal{C}'$, the symbol decision is formulated as

$$\tilde{d}'(n) = \arg \min_{d' \in \mathcal{C}'} \left| g d' + h \right|^2 \tag{17}$$

or, in a more familiar form with nonrotated constellation,

$$\tilde{d}(n) = Q \left\{ -\frac{h}{g} \exp -j\theta \right\} . \tag{18}$$

C. Timing Recovery

Jointly-optimal MMSE-based timing recovery, as proposed in [5], presents challenges to lattice filters similar to those mentioned in Sec. IV-A; acoustic signals and previous decisions become tangled as they cross the lattice section, and computing gradients with respect to a subset of them is difficult. Furthermore, such schemes incorporate a highly complex equalizer transfer function inside the timing recovery loop, thus creating serious instability problems. A better approach in practice is to perform symbol synchronization as a preprocessing step using low-delay methods [11].

In this work the time scaling factor due to Doppler was directly estimated by measuring the duration[1] of packets and then resampling them prior to equalization as described in [12], [13]. This procedure also compensates a common Doppler-induced rotation of constellations in all acoustic signals, making it easier for the lattice DFE+PLL to subsequently track residual phase variations. Once the correct symbol rate was attained, fractionally-spaced sampling ensured near-perfect insensitivity to the actual choice of sampling phase, and no further synchronization refinements were needed.

[1]In practice this is often accomplished by detecting a known preamble and postamble in each packet by crosscorrelation, and measuring their time offset.

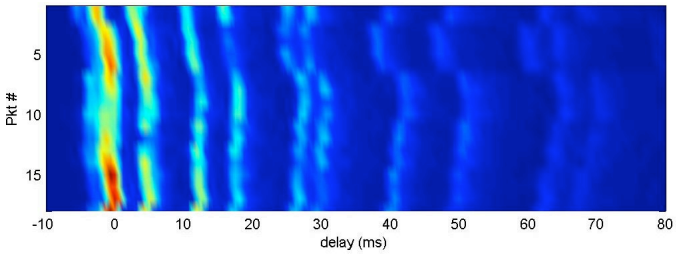

Fig. 2: [MREA'04] Averaged impulse response estimates

V. Experimental Results

A. MREA'04

The MREA'04 (Maritime Rapid Environmental Assessment) sea trial was conducted in the continental shelf off the west coast of Portugal in April 2004, in an area to the north of the Setúbal Canyon [14]. The receiver was an 8-element drifting array with hydrophones placed at depths 10, 15, 55, 60, 65, 70, 75, 80 m, and in the experiment reported here the acoustic source was suspended from the surface at a depth of about $60\,\mathrm{m}$. During a period of approximately 90 minutes modulated data were transmitted over a distance of about $2\mathrm{km}$, using a carrier frequency of $3600\,\mathrm{Hz}$, symbol rates of 200 or $400\,\mathrm{baud}$, and both 2-PSK and 4-PSK constellations. Throughout much of the experiment the transmitter was being towed along an approximately $110\,\mathrm{m}$-deep range-independent trajectory, at speeds of up to 2 m/s, thus inducing significant Doppler scaling in received waveforms. Environmental surveys at the test site revealed the presence of a $1.5\,\mathrm{m}$-thick silt and gravel sediment layer over a hard uniform sub-bottom.

Received signals were passband filtered, sampled at $20080\,\mathrm{Hz}$ and converted to baseband, then match-filtered (using a fourth-root raised-cosine pulse) and resampled at $L = 2$ times the symbol rate. Packet synchronization was performed by crosscorrelation with the beginning of the known transmitted sequence. Preprocessing for Doppler compensation was performed as described in [12]. The signals from the upper 2 hydrophones were discarded to match the conditions of Sec. V-C, resulting in a 12-channel external input to the DFE (6 hydrophones, each generating 2 symbol-rate-sampled streams).

This work focuses on the last 18 transmitted 2-PSK packets at $400\,\mathrm{baud}$, each having a duration of $20\,\mathrm{s}$, that span a total interval of about 25 minutes with variable inter-packet delays of $1\,\mathrm{s}$ or $277\,\mathrm{s}$. Fig. 2 depicts averaged estimates of the (Doppler-corrected) impulse responses for these packets, showing a fairly stable multipath profile with two strong arrivals at 0 and $4\,\mathrm{ms}$, and secondary arrivals up to about $50\,\mathrm{ms}$. Analysis of the data using MATLAB floating-point

algorithms showed that the best performance is obtained for a short DFE with 2 causal and 1 anticausal coefficients in each feedforward channel (abbreviated as $(2, 1)$ henceforth), 7 feedback coefficients, and forgetting factor $\lambda = 0.95$ empirically adjusted to minimize the residual error variance. The low value of λ and shortness of the feedback filter suggest that there is significant nonstationarity that prevents the coherent combination of multipath energy from late arrivals.

B. Fixed-point performance

This section presents equalization results using fixed-point implementations for RLS, QR-RLS (see, e.g., [1]), and the MQR algorithm of Tab. I. QR-RLS and MQR both use the Newton method to compute Givens matrices. Additional results are also provided for MQR using CORDIC to carry out Givens rotations, as discussed in Sec. III-A. Like RLS, the computational complexity of QR-RLS scales with the square of the total equalizer order. The particular implementation of RLS used here is the one designated by version II in [15, Tab. 13.2], which exhibits improved numerical behavior at the cost of moderate added redundancy.

The algorithms were coded in C targetting a 32-bit Intel processor (thus limiting to 32 bits as well the maximum numerical precision that can be specified, for reasons of code efficiency), with native compiler support (GCC) for complex data types, and using custom-made low-level arithmetic functions that operate in true fixed-point format with parametrizable integer and fraction lengths.

The integer/fractional part lengths to be used in each algorithm were determined by running the MATLAB prototype algorithms in floating point, examining the numerical range of internal variables, and choosing appropriate integer lengths to accomodate them. The global integer part lengths for RLS and QR-RLS, were set to $I_{\text{RLS}} = 11$ bits and $I_{\text{QR-RLS}} = 6$ bits, respectively. While integer part lengths can be individually specified for each block of the multichannel lattice filter, globally setting them to $I_{MQR} = 6$ proved to be appropriate. Fig. 3 shows the output (*a priori*) MSE attained by MQR over all the packets for 16, 24 and 32 bit resolutions, and also the lower bound obtained with a floating-point implementation. Also depicted are the output constellations for one of the packets, showing how the MSE degradation due to finite numerical precision impacts the scattering of points at the slicer input. Fig. 4 more explicitly shows the average increase in MSE for the full set of packets as a function of the numerical precision for all considered algorithms. QR-RLS and MQR with Newton-based Givens rotations exhibit very robust numerical behavior and comparable performance up to 16-bit resolution. CORDIC rotations are seen to be more sensitive to numerical roundoff, and this conclusion actually applies not only to MQR but to all other examined algorithms based on Givens rotations (not shown). Finally, RLS breaks down for less than 24 bit resolution, which is not surprising given the much larger range, and hence coarser quantization, of its internal variables. These results are in line with the abundant technical literature devoted to exposing and analyzing the numerically unstable behavior of RLS.

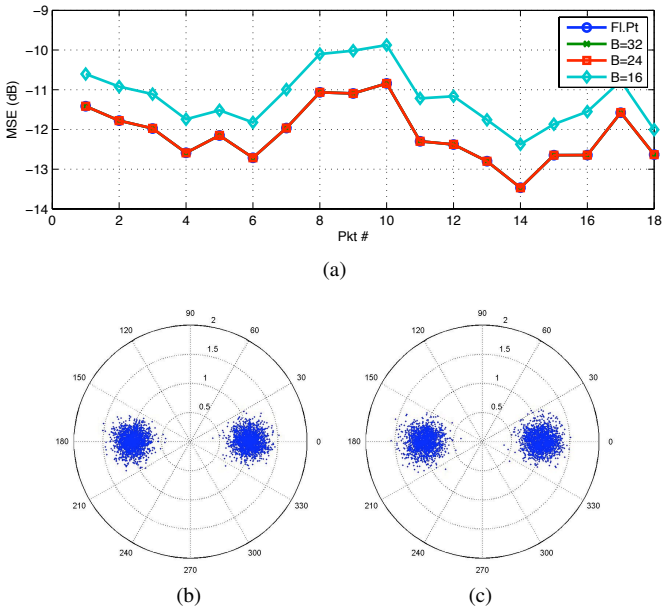

Fig. 3: [MREA'04] Performance of the MQR algorithm (a) Steady-state MSE (b) Output constellation for packet #14, floating-point (c) Same packet, fixed-point (Newton), 16 bits

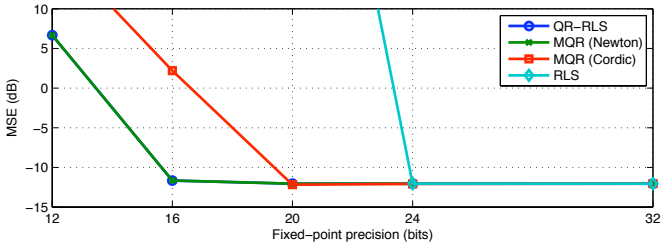

Fig. 4: [MREA'04] Evolution of MSE with fixed-point precision

C. MAKAI'05

The MAKAI'05 experiment took place off the island of Kauai, Hawaii, in September/October 2005, in an area that is part of the Pacific Missile Range Facility (PMRF) [16]. MAKAI'05 was specifically planned to support the High-Frequency initiative, whose goal is to gain a better understanding of acoustic propagation at frequencies on the order of tens of kHz. A large number of teams from various countries and institutions were involved, each focused on a specific set of objectives related to its equipment and scientific goals. The data examined in this paper pertains to an experiment designed by the University of Algarve, Portugal, in which 2-PSK data were transmitted at 2000 baud (3 kHz bandwidth) around a carrier frequency of 10 kHz. Both moored and towed sources were used, and source-receiver ranges were on the order of 1 to 4 km. The drifting receiver array was similar to the one

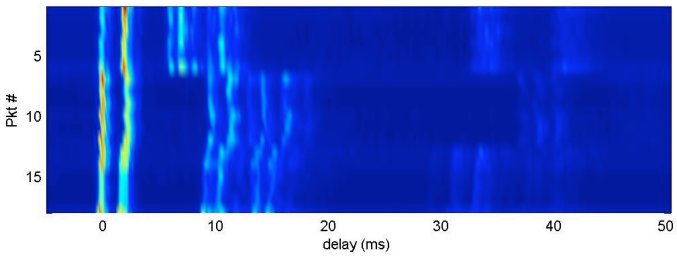

Fig. 5: [MAKAI'05] Averaged impulse response estimates

used in MREA'04, but only the lower 6 hydrophones were functional at the time.

The reported results focus on 3 bursts of 6 packets, each packet lasting for $15\,\mathrm{s}$ followed by a $2.9\,\mathrm{s}$ gap. The burst cycle was repeated every hour, for a total of 18 packets collected over a 2-hour period at the start of Drift 5 [16], when the receiver array was drifting eastward along a range-independent path of approximately $100\ \mathrm{m}$ depth. The transmitter was mounted on a vertical towfish and deployed at a depth of about $25\,\mathrm{m}$. During the period under analysis the towed source remained essentially stationary, at a range of little more than $1\ \mathrm{km}$ from the receiver.

Received signals were preprocessed as described in Sec. V-A, and Fig. 5 depicts the resulting averaged impulse response estimates. Two closely-spaced main arrivals of comparable magnitude can be seen, as well as significant contributions from several secondary paths with delays as long as $40\,\mathrm{ms}$. Preliminary (floating-point) equalization results showed that best results are obtained for $(5, 3)$ feedforward coefficients in each of the 12 input channels, 30 feedback coefficients, and $\lambda = 0.98$. This suggests that the time span of the feedback filter should extend to the group of arrivals starting at delay $10\ \mathrm{ms}$, which are likely to be sufficiently stable for coherent combining. As usual in a DFE, the time span of the feedforward filter is much shorter than the duration of ISI, its main goal being to capture most of the available input energy and cancel the precursor interference.

D. Fixed-point performance

As in Sec. V-B, fixed-point equalization results are presented in Figs. 6 and 7. The most striking difference with respect to the results for MREA'04 is that the least-squares problem solved by the adaptive equalizers is now considerably more ill-conditioned. This seems to stem from (*i*) the larger filter order, as is frequently the case when multichannel data and fractional sampling are used, and (*ii*) the improved temporal stability of the acoustic channel, as manifested in the larger value of λ. The latter results in weaker statistical fluctuations of the data covariance matrix, whose effect is akin to an additional noise source. In the case of plain RLS, which explicitly updates the inverse of the covariance matrix, it was found that 28 bits in integer parts were now needed to represent the range of internal variables observed in the floating-point algorithm. For the word lengths shown in Fig. 7 this left too few fractional bits for accurate quantization, resulting

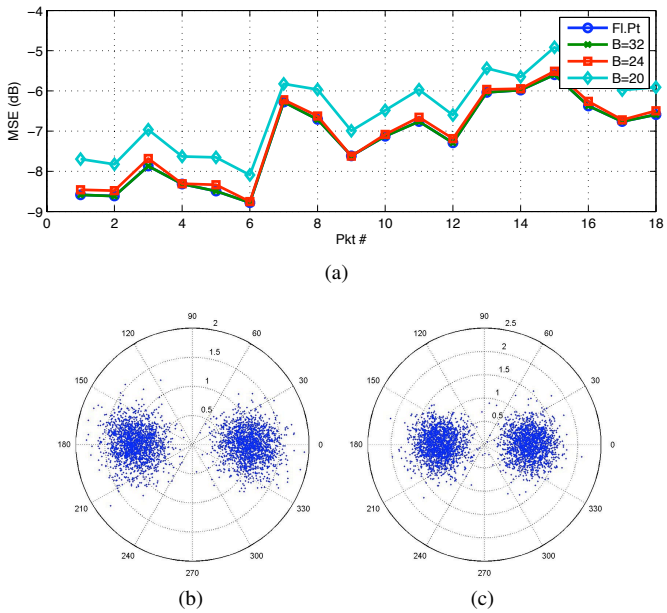

Fig. 6: [MAKAI'05] Performance of the MQR algorithm (a) Steady-state MSE (b) Output constellation for packet #1, floating-point (c) Same packet, fixed-point (Newton), 20 bits

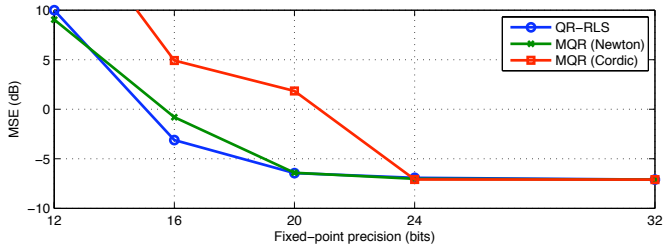

Fig. 7: [MAKAI'05] Evolution of MSE with fixed-point precision. RLS failed to converge for all word lengths shown

in numerical divergence. A floating-point implementation of RLS in MATLAB with pseudo-quantization (i.e., truncation to a specified word length after floating-point operations) successfully converged only for a minimum precision of 48 bits, with 32 bits allocated for integer parts. In contrast, QR-type algorithms were able to continue operating with integer part lengths of 6 bits, as in Sec. V-A.

In spite of the improved numerical stability relative to RLS, QR-type algorithms now require 4 additonal fractional bits for acceptable performance (note the difference in the legend of the upper curves of Figs. 3a and 6a). MQR still provides almost identical performance to QR-RLS using Newton iterations, and the CORDIC implementation again exhibits less robust behavior.

VI. Conclusion

The main goal of this work was to evaluate the performance of a new multichannel lattice RLS filtering algorithm (MQR) using data from underwater acoustic communication experi-

ments and fixed-point arithmetic. This type of filter may be well suited for channel equalization in underwater receivers due to the linear scaling of its computational complexity with filter order and robust numerical behavior. Practical aspects related to equalization, such as timing/carrier recovery and slicing of symbol estimates, were also addressed.

Data sets from the MREA'04 and MAKAI'05 experiments were examined, the former having mild ISI and noticeable time fluctuations, whereas the latter had stronger ISI but appeared to be more stable. Experimental results showed that the algorithm retains the same desirable numerical robustness characteristics of other square-root RLS variants, namely, QR-RLS. Minor degradations in output MSE were observed down to word lengths of 16 bit in MREA'04 and 20 bits in MAKAI'05. These values for fixed-point precision are well within the reach of current technology for embedded or hardware implementations of signal processing algorithms. By contrast, plain RLS was found to require higher precision for adequate operation, as much as 48 bits in MAKAI'05 data. The variant of MQR using Newton iterations for computing Givens rotations was found to tolerate shorter word lengths than the one based on CORDIC.

Given the availability of intermediate-order filtering residuals throughout the lattice filter, practical methods for automatically choosing a suitable equalizer order may be considered in future work. Developing alternative lattice-like multichannel decompositions that allow the order in a single input channel to be gradually increased may also be useful in practical DFEs for underwater receivers, where the (single) feedback filter is typically longer than the (multiple) feedforward filters.

ACKNOWLEDGMENT

This work was supported by Fundação para a Ciência e a Tecnologia (ISR/IST plurianual funding) through the POS–Conhecimento Program that includes FEDER funds.

REFERENCES

[1] S. Haykin, *Adaptive Filter Theory*, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[2] A. Sayed, *Fundamentals of Adaptive Filtering*. New York: Wiley-IEEE, 2003.

[3] J. Gomes, V. Barroso, "Array-based QR-RLS multichannel lattice filtering," *Submitted to IEEE Transactions on Signal Processing*, Mar. 2007.

[4] ——, "A CORDIC-based QR-RLS multichannel lattice filter," in *Proceedings of the 15th European Signal Processing Conference (EUSIPCO'07)*, Poznań, Poland, Sept. 2007.

[5] M. Stojanovic, J. Catipovic, J. Proakis, "Phase-coherent digital communications for underwater acoustic channels," *IEEE Journal of Oceanic Engineering*, vol. 19, no. 1, pp. 100–111, Jan. 1994.

[6] B. Yang, "A QR multichannel least squares lattice algorithm for adaptive nonlinear filtering," *International Journal of Electronics and Communications (AEÜ)*, vol. 49, no. 4, pp. 171–182, July 1995.

[7] B. Yang, J. Böhme, "Rotation-based RLS algorithms: Unified derivations, numerical properties, and parallel implementations," *IEEE Transactions on Signal Processing*, vol. 40, no. 5, pp. 1151–1167, May 1992.

[8] G. Glentis, N. Kalouptsidis, "A highly modular adaptive lattice algorithm for multichannel least squares filtering," *Signal Processing*, vol. 46, no. 1, pp. 47–55, 1995.

[9] F. Ling, J. Proakis, K. Zhao, "A systematic treatment of order-recursive least-squares algorithms," *Proceedings of SPIE Adaptive Signal Processing*, vol. 1565, pp. 296–306, 1991.

[10] M. Ercegovac, T. Lang, *Digital Arithmetic*. San Francisco, CA: Morgan Kaufmann, 2003.

[11] E. Lee, D. Messerschmitt, *Digital Communication*, 2nd ed. Kluwer Academic Publishers, 1994.

[12] J. Gomes, A. Silva, S. Jesus, "Adaptive spatial combining for passive time-reversed communications," *Submitted to the Journal of the Acoustical Society of America*, Mar. 2007.

[13] ——, "Joint passive time reversal and multichannel equalization for underwater communications," in *Proceedings of MTS/IEEE OCEANS'06*, Boston, MA, USA, Sept. 2006.

[14] S. Jesus, C. Soares, P. Felisberto, A. Silva, L. Farinha, C. Martins, "Acoustic maritime rapid environmental assessment during the mrea04 sea trial," CINTAL — Universidade do Algarve, Tech. Rep. 02/05, Mar. 2005. [Online]. Available: http://www.siplab.fct.ualg.pt/pubs/rep_0205.pdf

[15] S. Haykin, *Adaptive Filter Theory*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1991.

[16] S. Jesus, A. Silva, F. Zabel, "Acoustic oceanographic buoy data report — Makai Ex 2005," CINTAL — Universidade do Algarve, Tech. Rep. 04/05, Nov. 2005. [Online]. Available: http://www.siplab.fct.ualg.pt/pubs/rep_0405.pdf