LOW-COMPLEXITY ALGORITHM FOR INVERSION OF SPECIAL MATRICES IN SDR SYSTEMS

David Guevorkian (Tampere University of Technology, Tampere, Finland; david.guevorkian@tut.fi); Kim Rounioja (Renesas Mobile, Oulu, Finland; kim.rounioja@renesasmobile.com); and Jarmo Takala (Tampere University of Technology, Tampere, Finland; jarmo.takala@tut.fi)

ABSTRACT

Inversion of circulant Hermitian matrices is one of the most computationally complicated algorithms used in communication, signal processing, and other systems. In particular, it may be used in tap solvers of advanced HSDPA systems. In this paper, a novel fast method for finding the inverse of a circulant Hermitian matrix is proposed. The proposed method is based on using Discrete Cosine (DCT-1) and Discrete Sine (DST-1) transforms of Type 1. It reduces the number of operations approximately by a factor of four compared to conventional Fast Fourier Transform (FFT) based method.

I. INTRODUCTION

Tight computational requirements of Software Defined Radio (SDR) systems require careful optimization of the algorithms used in these systems. One of important algorithms used in advanced digital radio systems is inversion of special matrices. For example, in linear equalizer tap solvers of High Speed Downlink Packet Access (HSDPA) receivers, as well as in Wideband Code Division Multiple Access (WCDMA) systems inversion of circulant Hermitian matrices, is one of the most computationally intensive algorithms.

Matrix inversion as a fundamental operation of linear algebra being used in various application fields has always been in the focus of researchers. In the general case, for inversion of arbitrary $(N \times N)$ matrix using the most known methods such as Gauss elimination method, $O(N^3)$ operations are needed. In 1969, Strassen [1] proposed an algorithm for matrix inversion that needs approximately $O(N^{2.8})$ operations. Since then a number of algorithms further reducing this complexity were proposed. However, the complexity of arbitrary matrix inversion still remains well above $O(N^2)$ operations.

Luckily, matrix inversion operations used in communication systems are often related to matrices of special structure. Computational complexity of inversion of such matrices can significantly be reduced by making use the knowledge of their special structure. For example, it is well known that symmetric Toeplitz matrices can be inverted with the complexity of $O(N^2)$ (see, e.g. [2]). Furthermore, fast real-valued Fourier transform or fast Hartley transform may be used for inversion of symmetric real-valued Toeplitz matrices resulting in $O(N\log N)$ complexity.

Here we consider the problem of reducing the complexity of tap solvers in HSDPA systems where the most computationally complex operation is the inversion of complex-valued circulant Hermitian matrices. The complexity $O(N\log N)$ of inversion of such matrices may be reached by using complex-valued Fast Fourier Transform (FFT) algorithm (see [3]-[5]). In [6], a method for inverting $(N \times N)$ Toeplitz matrices using fast Discrete Cosine and Sine transforms of order N is described wherein inversion of a Toeplitz matrix is first reduced to inversion of a symmetric and an anti-symmetric matrices.

In this paper, we propose a new algorithm that further reduces the complexity of tap solvers by using circulant Hermitian matrix inversion method proposed in [7]. The proposed method is based on using slightly modified Type 1 Fast Discrete Cosine Transform and Type 1 Fast Discrete Sine Transform of orders $N/2$. The number of operations in the proposed algorithm is approximately four times lower than that of required by conventional FFT based algorithm.

II. DEFINITIONS AND CONVENTIONAL METHODS

Our main target is to reduce the complexity of tap solvers for advanced HSDPA equalizer receivers where inversion of circulant Hermitian matrices is a very computation demanding part. Aspects of the usage and implementation of such equalizers have been described by several authors, for example [10]. The chip-level signal model for one transmitter and one receiver antenna system can be represented by $\mathbf{y} = H\mathbf{s} + \mathbf{n}$, where $\mathbf{y}$ is the received chip

sequence (the input to the equalizer), H models the transmission channel including the influence of the transmit filter, multi-path fading and the receiver filter, $\mathbf{s}$ is the transmitted chip-sequence, and $\mathbf{n}$ represents the noise as seen by the receiver. The structure of H is:

$$H=\begin{bmatrix} h_0 & \cdots & h_{m-2} & h_{m-1} & 0 & \cdots & 0 \\ 0 & h_0 & \cdots & h_{m-2} & h_{m-1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & h_0 & \cdots & h_{m-2} & h_{m-1} \end{bmatrix}$$

where h_n, $n=0,...,m-1$, represents the channel impulse response and m represents the channel delay spread. The output of the chip-level equalizer receiver is $\hat{\mathbf{s}}=\mathbf{x}^H\mathbf{y}$: a delayed estimate of the transmitted chip $\mathbf{s}$. The vector $\mathbf{x}$ represents the filter taps obtained from the equalizer tap solver. The LMMSE equalizer solution for $\mathbf{x}$ is (see [11]):

$$\mathbf{x}=\sigma_s^2 H^H\left(\sigma_s^2 HH^H+\sigma_n^2 I\right)^{-1}\delta_D=\sigma_s^2 H^H A^{-1}\delta_D$$

where σ_s^2 and σ_n^2 denote variance of the chip and noise sequence, respectively. Thus, the essential task of the tap solver is to find the *D*-th, *D=0....,L-1,* column $\mathbf{w}_D$ of the inverse of an $L\times L$ Hermitian Toeplitz matrix *A*:

$$A=\begin{bmatrix} r_0 & \cdots & r_{m-1} & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\ r_1^* & r_0 & \cdots & r_{m-1} & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & & \ddots & \ddots & & \ddots & \ddots & \vdots \\ r_{m-2}^* & \cdots & r_1^* & r_0 & \cdots & r_{m-1} & 0 & \cdots & 0 & 0 \\ r_{m-1}^* & r_{m-2}^* & \cdots & r_1^* & r_0 & \cdots & r_{m-1} & 0 & \ldots & 0 \\ 0 & r_{m-1}^* & r_{m-2}^* & \ldots & r_1^* & r_0 & \ldots & r_{m-1} & \ldots & 0 \\ \vdots & \cdots & \ddots & \ddots & \ldots & \ddots & \ddots & \ldots & \ddots & \vdots \\ 0 & \ldots & 0 & r_{m-1}^* & r_{m-2}^* & \ldots & r_1^* & r_0 & \ldots & r_{m-1} \\ 0 & 0 & \ldots & 0 & r_{m-1}^* & r_{m-2}^* & \ldots & r_1^* & r_0 & \ldots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & & & \\ 0 & \ldots & 0 & 0 & \cdots & 0 & r_{m-1}^* & r_{m-2}^* & \ldots & r_0 \end{bmatrix}$$

where r_0 is a real number (since A is Hermitian). In general, this is not a circulant matrix. However, extensive simulations were carried out (see [3]) showing that the performance of tap solvers is only negligibly worsened if the matrix A is modified so that it becomes a circulant Hermitian $L\times L$ matrix of the form:

$$C=\begin{bmatrix} r_0 & \cdots & r_{m-1} & 0 & \cdots & 0 & r_{m-1}^* & \cdots & r_1^* & r_1^* \\ r_1^* & r_0 & \cdots & r_{m-1} & 0 & \cdots & 0 & r_{m-1}^* & \cdots & r_2^* \\ \vdots & \ddots & \ddots & & \ddots & \ddots & & \ddots & \ddots & \vdots \\ r_{m-2}^* & \cdots & r_1^* & r_0 & \cdots & r_{m-1} & 0 & \cdots & 0 & r_{m-1}^* \\ r_{m-1}^* & r_{m-2}^* & \cdots & r_1^* & r_0 & \cdots & r_{m-1} & 0 & \ldots & 0 \\ 0 & r_{m-1}^* & r_{m-2}^* & \ldots & r_1^* & r_0 & \ldots & r_{m-1} & \ldots & 0 \\ \vdots & \cdots & \ddots & \ddots & \ldots & \ddots & \ddots & \ldots & \ddots & \vdots \\ 0 & \ldots & 0 & r_{m-1}^* & r_{m-2}^* & \ldots & r_1^* & r_0 & \ldots & r_{m-1} \\ r_{m-1} & 0 & \ldots & 0 & r_{m-1}^* & r_{m-2}^* & \ldots & r_1^* & r_0 & \ldots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & & & \\ r_1 & \ldots & r_{m-1} & 0 & \cdots & 0 & r_{m-1}^* & r_{m-2}^* & \ldots & r_0 \end{bmatrix}, \quad (1)$$

With this modification, the problem of the tap solver is reduced to finding the vector

$$\mathbf{w}_D=C^{-1}\delta_D \quad (2)$$

Where δ_D is the Kronecker Delta vector consisting of all zeros except one unity at *D*-th position (multiplication of the matrix C^{-1} to δ_D results in the *D*-th column $\mathbf{w}_D$ of C^{-1}).

Inverting circulant and, in particular, Toeplitz matrices is an old well-studied problem. In our case, however, there are additional features of the matrix being Hermitian and, therefore, having a real valued main diagonal, which helps in further reduction of computational complexity.

In conventional implementations of HSDPA downlink receivers (see [3] – [5]), inversion of the circulant matrix C is implemented with the help of Discrete Fourier Transform (DFT) based on the fact that every circulant matrix is diagonalized by 2-D DFT and, therefore can be decomposed as follows:

$$C=FGF^H \quad (3)$$

where F and F^H are the DFT matrix and its conjugate transposed matrix (which coincides with the inverse DFT matrix), respectively, and $G=\mathrm{diag}(\mathbf{g})$ is a diagonal matrix such that the vector $\mathbf{g}$ of its main diagonal elements is the DFT of the first column $C(*,0)$ of the matrix *C*, that is

$$\mathbf{g}=F\cdot C(*,0)\,. \quad (4)$$

Substituting (3) into (2) yields:

$$\mathbf{w}_D=FG^{-1}F^H\delta_D \quad (5)$$

Also note that $F^H\delta_D$ is the D-th, $D=0....,L-1$, column of the (known) inverse DFT matrix, and $G^{-1}F^H\delta_D$ is a vector-column obtained as point-wise product of $F^H\delta_D$ with the vector $1./\mathbf{g}$ being point-wise inverse of the vector $\mathbf{g}$ from (4).

Therefore, the desired vector $\mathbf{w}_D$ can be obtained according to the following algorithm, which makes use of the well-known fast Fourier transform (FFT) algorithm for efficient computation of DFT.

1. Find vector $\mathbf{g}=F\cdot C(*,0)$ by implementing FFT over the first column of the matrix C.
2. Element-by-element (point-wise) invert the vector $\mathbf{g}$ obtained at the 1st step.
3. Point-wise multiply the D th, $D=0,...,L-1$, column $F^H\delta_D$ of the inverse DFT matrix by the inverted vector obtained at 2nd step.
4. Compute the desired vector $\mathbf{w}_D$ by implementing FFT over the vector obtained at the 3rd step.

This algorithm significantly reduces the number of operations required to invert a circulant matrix from $O(L^3)$ operations in conventional Gauss elimination method to $O(L\log L)$ operations.

III. THE PROPOSED ALGORITHM

It is very well known, that the inverse C^{-1} of an $L\times L$ circulant Hermitian matrix C (of the form (1)) is also circulant and Hermitian. Therefore, to find arbitrary D th column $\mathbf{w}_D=\left[w_D^0,w_D^1,...,w_D^{L-1}\right]^T$, $D=0,...,L-1$, of C^{-1} or, more in general, to invert the matrix C, it is sufficient to find only the subvector $\bar{\mathbf{w}}_0=\left[w_0^0,w_0^1,...,w_0^{L/2}\right]^T$ of the first $L/2+1$ components of only the 1st column $\mathbf{w}_0$ of C^{-1}. The other $L/2-1$ components of the 1st column may then be obtained as conjugates of these components based on the relation:

$$w_0^i=\left(w_0^{L-i}\right)^*,\ i=L/2+1,...,L-1, \tag{6}$$

which is true since the matrix C^{-1} is circulant and Hermitian. Then the D th column $\mathbf{w}_D$ of C^{-1}, for an arbitrary $D=0,...,L-1$, may be obtained by circular shift for D positions down from $\mathbf{w}_0$.

The proposed algorithm is based on the lemma below, for formulating which let us introduce a few notations and remind the definitions of Type 1 DCT and Type 1 DST transforms. First, we adopt the Re() and Im() notations for the real and the imaginary parts of a (scalar or vector) variable, respectively. Second, for a vector $\mathbf{x}$ we use the notation $\mathbf{x}(n:m)$ to denote the subvector of $\mathbf{x}$ consisting of the components indexed n through m.

Now, let us denote $C_1^{L/2+1}$ the $\left(\left(L/2+1\right)\cdot\left(L/2+1\right)\right)$ matrix with entries:

$$c_{n,m}=b_m\cos(\pi nm/(L/2)),$$
$$b_m=\begin{cases}1/2, & \text{for } m=0,L/2\\ 1, & \text{otherwise}\end{cases},\ n,m=0,...,L/2. \tag{7}$$

This is actually slightly different from matrix $\mathrm{DCT1}(L/2+1)$ of the well-known Type 1 DCT transform of order $L/2+1$, which consists of the entries:

$$c'_{n,m}=b_mb_n\cos(\pi nm/(L/2)),\ b_0=b_{L/2}=1/\sqrt{2},$$
$$n,m=0,...,L/2,$$

(see [8]). It is well known that the Type 1 DCT of order $N+1$ may be computed with a fast algorithm, which requires

$$N_{mul}(DCT1)=\left(N/2\right)\log_2 N-3N/4 \text{ and}$$
$$N_{add}(DCT1)=\left(3N/2\right)\log_2 N-N/2 \tag{8}$$

multiplications and additions, respectively [8]. The transform $\mathbf{y}=C_1^{L/2+1}\mathbf{x}$ with the matrix $C_1^{L/2+1}$ differs from ($L/2+1$)-point Type 1 DCT only in scaling two inputs and two outputs since

$$\mathbf{y}=C_1^{L/2+1}\cdot\mathbf{x}=B\cdot DCT1(L/2+1)\cdot B\cdot\mathbf{x}$$

where $B=\mathrm{diag}\left(1/\sqrt{2},1,...,1,1/\sqrt{2}\right)$. Therefore, there exists also a fast algorithm for implementing the transform with the matrix $C_1^{L/2+1}$, which requires at most four extra multiplications compared to fast ($L/2+1$)-point Type 1 DCT (an actually, even better algorithm may be developed but this is out of the scope in this work). Thus, the complexity of the transform with the matrix $C_1^{L/2+1}$ is not more than (substitute $N=L/2$ in (8)):

$$N_{mul}(C_1^{L/2+1}) = \left(L/4\right)\log_2 L - 5L/8 + 4 \text{ and}$$
$$N_{add}(C_1^{L/2+1}) = \left(3L/4\right)\log_2 L - L \quad (9)$$

multiplications and additions, respectively. Furthermore, let us denote by $S_1^{L/2-1}$ the $\left(L/2-1\right)\times\left(L/2-1\right)$ matrix of the Type 1 DST (see [9]) with the entries

$$s_{n,m} = \sin(\pi nm/(L/2)),\ n,m = 1,...,L/2-1 \quad (10)$$

Note that also for this transform there exists a fast algorithm with the complexity of

$$N_{\text{mul}}(S_1^{L/2-1}) = (L/4)\log_2 L - 3L/4 \text{ and}$$
$$N_{\text{add}}(S_1^{L/2-1}) = (3L/4)\log_2 L - 7L/4 - \log_2 L \quad (11)$$

multiplications and additions, respectively.

The proposed algorithm is based on the following lemma that was proven in [7].

Lemma. For an arbitrary $L\times L$ circulant Hermitian matrix C, the real and the imaginary parts of the subvector $\bar{\mathbf{w}}_0 = \mathbf{w}_0(0:L/2)$ of the first column $\mathbf{w}_0$ of C^{-1} can be expressed as follows:

$$\text{Re}(\bar{\mathbf{w}}_0) = C_1^{L/2+1}Q\mathbf{p},\ \ \text{Im}(\bar{\mathbf{w}}_0) = -\left[0, S_1^{L/2-1}Q\mathbf{s}, 0\right] \quad (12)$$

where

$$\mathbf{p} = \left[p_0, p_1, ..., p_{L/2}\right]^T = C_1^{L/2+1}\left(\text{Re}\left(\mathbf{c}_0(0:L/2)\right)\right),$$
$$\mathbf{s} = \left[s_1, s_2, ..., s_{L/2-1}\right]^T = S_1^{L/2-1}(\text{Im}(\mathbf{c}_0(1:L/2-1))), \quad (13)$$

$\mathbf{c}_0$ is the first column of the matrix C, and $Q = diag(\mathbf{q})$ is a diagonal matrix where $\mathbf{q} = \left[q_0, q_1, ..., q_{L/2}\right]$ with

$$q_0 = 1/p_0,\ q_i = 1/(p_i^2 - s_i^2),\ i = 1,...,L/2-1,$$
$$q_{L/2} = 1/p_{L/2}. \quad (14)$$

Based on this Lemma, the following algorithm may be applied to invert a circulant Hermitian matrix.

Algorithm.

Input: An $((L/2+1))$-point subvector $\mathbf{c}_0(0:L/2)$ of the first column $\mathbf{c}_0$ of a circulant Hermitian matrix *C*.

Note that, in the case of HSDPA receiver this is the same as the $(L/2+1)$-point subvector of first components of the first column of the original Toeplitz matrix *A*, and therefore there is no need for constructing the circulant matrix *C* as in the conventional implementations.

Output: The subvector $\bar{\mathbf{w}}_0 = \mathbf{w}_0(0:L/2)$ of the first column $\mathbf{w}_0$ of C^{-1}.

If needed, the whole first column and all the other columns of C^{-1} may be formed by making use of equation (6) and circular shifts.

Computation:

Step 1. Find vectors $\mathbf{p}$ and $\mathbf{s}$ according to (13). For this, implement fast $(L/2+1)$-point fast transform with the matrix $C_1^{L/2+1}$ over the vector $\text{Re}(\mathbf{c}_0(0:L/2))$ and $(L/2-1)$-point fast Type 1 DST over the vector $\text{Im}(\mathbf{c}_0(1:L/2-1))$.

Step 2. Find the vector $\text{Re}(\mathbf{c}_0(0:L/2))$ according to (14).

Step 3. Point-wise multiply the vectors $\mathbf{p}$ and $\mathbf{s}$ obtained at the 1st step by the components of the vector obtained at the 2nd step.

Step 4. Find the real and the imaginary parts of the vector $\bar{\mathbf{w}}_0 = \mathbf{w}_0(0:L/2)$ according to (12), that is, implement $(L/2+1)$-point fast transform with the matrix $C_1^{L/2+1}$ and $(L/2-1)$-point fast Type 1 DST.

IV. COMPLEXITY ANALYSIS AND DISCUSSION

In this section we analyze the complexity of the Algorithm. The proposed algorithm takes into account that the matrix C is not only circulant but also Hermitian, thus its diagonal elements are real and furthermore it reduces the number of required operations by approximately a factor of four. Instead of implementing *L*-point complex-valued FFT, (*L/2*)-point real-valued fast DCT-1 and fast DST-1 transforms are implemented in the proposed algorithm. This gives opportunity to reduce the implementation time (due to less number of operations) as well as the required memory and data movements (due to shorter arrays), and possibly the energy consumption.

The complexity of the proposed Algorithm in comparison with the conventional FFT-based algorithm is summarized in Table 1.

At each of the Steps 1 and 4 of the Algorithm, one $(L/2+1)$-point fast transform with the matrix $C_1^{L/2+1}$ and one $(L/2-1)$-point fast Type 1 DST transforms are

implemented. According to (9) and (11), total of

$$N_{mul}(\text{step1}) = \left(L/2\right)\log_2 L - 11L/8 + 4$$

real multiplications and

$$N_{add}(steps1) = \left(3L/2\right)\log_2 L - 11L/4 - \log_2 L + 3$$

real additions will be required for each of these two steps.

At step 2 of the Algorithm, where equations (14) are implemented,

$$N_{mul}(\text{step2}) = L - 2$$

real multiplications,

$$N_{add}(\text{step2}) = L/2 - 1$$

real additions, and

$$N_{div}(\text{step2}) = L/2 + 1$$

real divisions are needed.

And, in addition

$$N_{mul}(\text{step3}) = L$$

real multiplications are needed to complete Step 3.

The last row of Table 1 presents the total operation count. As can be seen from Table 1, the number of real operations in the proposed algorithm is approximately the same as the number of complex operations in the conventional FFT based algorithm. In general, one complex multiplication "costs" approximately four real multiplications and two real additions, while one complex addition "costs" two real additions. (Although in another approach one complex multiplication can be implemented with three real multiplications and more additions, this does not significantly change the conclusions.) The operation counts with respect to these costs are presented in the last three columns of Table 1. As can be seen, the proposed method is approximately four times less costly than the current FFT-based method since it uses more than four times less multiplications, twice less divisions and approximately three times less additions.

Also, the proposed method deals with shorter input and output arrays and separately with real and imaginary parts of the arrays. In a careful implementation, this will give opportunity to reduce the required memory size, data bandwidth, and data transfers. Due to less amount of computations needed, the proposed method should also give opportunity to reduce energy consumption.

Figure 1 presents the plots of the total operations counts of the proposed method and of the FFT-based method for typical values of the matrix size L in communication systems ($L = 64$, 128, 256, 512, 1024, 2048). One can notice a large absolute difference in the numbers of operations. For the larger values of L (which will likely be used in emerging communication systems) the absolute difference is even larger.

V. CONCLUSION

New fast DCT-1 and fast DST-1 based algorithm for inversion of circulant Hermitian matrices was proposed. This algorithm is approximately four times less costly compared to the conventional FFT based method. The proposed algorithm may be applied in advanced HSDPA and WCDMA downlink receivers and other communication systems.

REFERENCES

[1] V. Strassen, "Gaussian Elimination is Not Optimal." Numerische Mathematik 13, pp. 354-356, 1969.

[2] L. Rodman, T. Shalom, "On inversion of symmetric Toeplitz matrices", *SIAM J. Matrix Analysis and Application*, 13 (1992), 530-549.

[3] J. Zhang, T. Bhatt, G. Mandyam, "Efficient linear equalization for high data rate downlink CDMA signalling," 2003.Conference Record of the Thirty-Seventh Asilomar Conference on Signals, Systems and Computers, 2003. Volume 1, 9-12 Nov. 2003, pp. 141 – 145.

[4] US patent application No. 20050025267, "Reduced complexity sliding window based equalizer," A. Reznik, R. Yang, B. Li, A. Zeira, 2005.

[5] US patent application No. 20050180559, "Apparatus and method of echo cancellation," F. Pisoni, R. Hug, M. Bonaventura, 2005.

[6] US patent No. 6597745B1, "Reduced complexity multicarrier precoder," E.M. Dowling, 2003.

[7] D. Guevorkian, K. Rounioja and J. Takala, "Circulant Hermitian Matrix Inversion Method Based on Discrete Cosine and Sine Transforms," in *Proceedings of 2012 IEEE Workshop on Signal Processing Systems (SIPS-2012),* Québec City, Québec, Canada Oct., 2012.

[8] P.C. Yip and K. Rao, "DIF algorithms for DCT and DST," Proceedings of IEEE International Conference ICASSP'85, pp.776-779, 1985.

[9] Z. Wang, "Fast discrete sine transform algorithms," Signal Processing, vol. 19, pp. 91-102, 1990.

[10] Mailaender, L.; "Low-complexity implementation of CDMA downlink equalization", 3G Mobile Communication Technologies, 2001. Second International Conference on (Conf. Publ. No. 477) 26-28 March 2001 Page(s):396 – 400

[11] Krauss, T.P.; Zoltowski, M.D.; Leus, G.; "Simple MMSE equalizers for CDMA downlink to restore chip sequence: comparison to zero-forcing and RAKE", Proceedings.of 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2000. ICASSP '00.

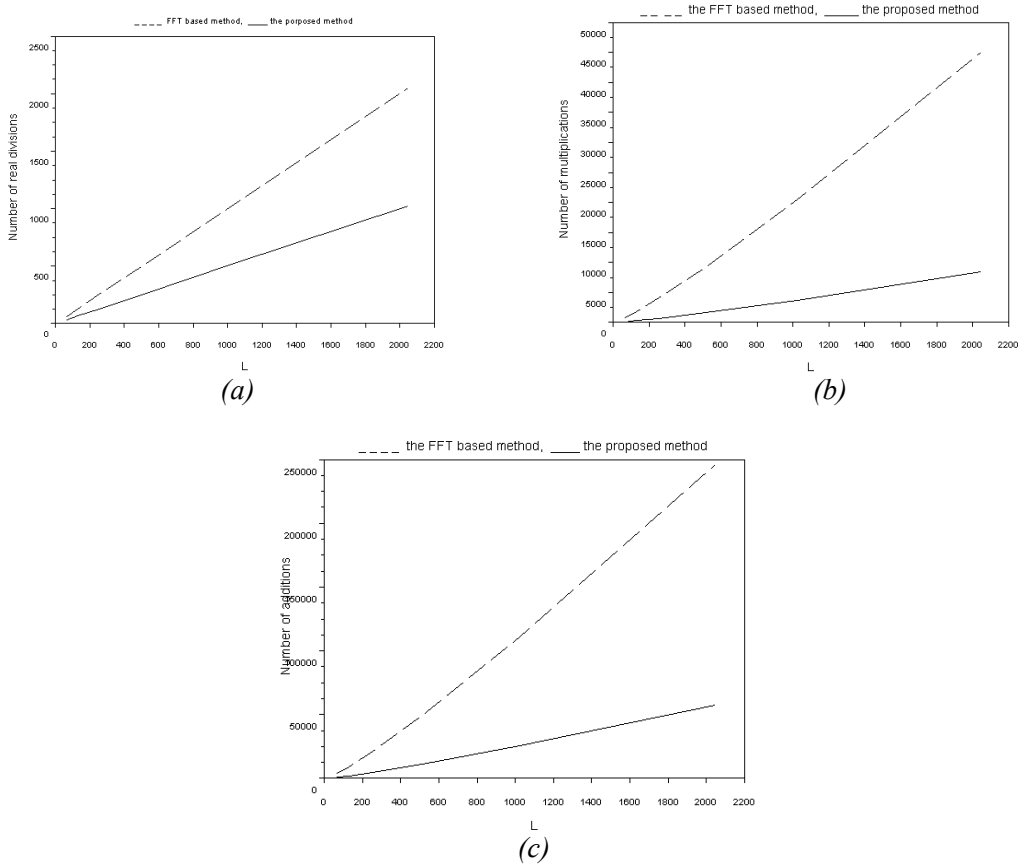

Figure 1. Comparison of the complexities of the conventional FFT based method and the proposed method: a) number of real divisions; b) number of real multiplications; c) number of real additions.

TABLE I. Number of operations implemented in the proposed and conventional algorithms

Step	Proposed algorithm			FFT- based algorithm					
				Count in complex ops			Count in real ops		
	Div	Mul.	Add	Div	Mul	Add	Div	Mul	Add
1	0	$\frac{L}{2}\log_2 L - 11\frac{L}{8} + 4$	$\frac{3L}{2}\log_2 L - 11\frac{L}{4} - \log_2 L + 3$	0	$\frac{L}{2}\log_2 L$	$L\log_2 L$	0	$2L\log_2 L$	$3L(\log_2 L - 1)$
2	$L/2+1$	$L-2$	$L/2-1$	L	0	0	L	0	0
3	0	L	0	0	L	0	0	4L	2L
4	0	$\frac{L}{2}\log_2 L - 11\frac{L}{8} + 4$	$\frac{3L}{2}\log_2 L - 11\frac{L}{4} - \log_2 L + 3$	0	$\frac{L}{2}\log_2 L$	$L\log_2 L$	0	$2L\log_2 L$	$3L(\log_2 L - 1)$
Total	$L/2+1$	$L\log_2 L - \frac{3L}{4} + 6$	$3L\log_2 L - 5L - 2\log_2 L + 5$	L	$L\log_2 L$	$2L\log_2 L$	L	$4L\log_2 L$	$8L(\log_2 L - 1)$