Superfast Approximative Implementation of the IAA Spectral Estimate

G. O. Glentis*, *Member, IEEE*, and A. Jakobsson†, *Senior Member, IEEE*

***Abstract*—In this correspondence, we develop superfast approximative one-dimensional algorithms for the computationally efficient implementation of the recent Iterative Adaptive Approach (IAA) spectral estimate. The proposed methods are based on rewriting the IAA algorithm using suitable Gohberg-Semencul representations, solving the resulting linear systems of equations using the preconditioned conjugate gradient method, where a novel preconditioning is applied using an incomplete factorization of the Toeplitz matrix. Numerical simulations illustrate the efficiency of both the proposed preconditioning as well as the overall algorithm, offering a computational reduction of up to two orders of magnitude as compared to our recently proposed efficient and exact IAA implementation.**

***Index Terms*—Toeplitz inversion, preconditioned conjugate gradient, spectral estimation, Iterative Adaptive Approach (IAA), fast algorithms.**

I. Introduction

COMPUTATIONALLY efficient high-resolution spectral estimation algorithms are of great importance in numerous applications. Typically, to achieve improved resolution, higher than the periodogram, one has to resort to using parametric or data-adaptive non-parametric estimation techniques. Due to their inherent robustness to model assumptions, the data-adaptive approaches are often of particular interest, and the topic has attracted increasing interest during the last decade. One such promising technique that is currently widely studied is the so-called iterative adaptive approach (IAA), recently proposed in [1] for passive sensing, channel estimation, and single-antenna radar applications, and there shown to outperform the well-known Capon and APES spectral estimation techniques [2] for data with a sparse spectrum. As a result, the technique has attracted significant interest in a variety of topics [3]–[5] As noted in these papers, the IAA-based estimation techniques allow for accurate high-resolution estimates even when only a few data snapshots are available. However, this improved performance comes at the cost of a notably high computational complexity, suggesting the need for computationally efficient implementations of such estimates. In [6], we presented several *exact* implementations for various IAA-based estimators, exploiting efficient formulations of data dependent trigonometric polynomials as well as suitable Gohberg-Semencul (GS) representations of the estimated inverse covariance matrix. This approach allows for substantial computational savings, but given the requirement of an exact implementation, one is still limited to the original formulation of the methods. In [7], we extended on this work allowing also for non-stationary signals, formulating both exact and approximative time-recursive sliding-window implementations. As shown there, the approximative solutions allow for significant additional computational savings, while still resulting in estimates close to the exact solution. In this work, we return to the problem of stationary signals, instead formulating computationally efficient *approximate* solutions of the IAA spectral estimate. Extending the GS representations formulated in [6], we propose to solve the resulting linear set of equations using a novel Quasi-Newton (QN) preconditioned conjugate gradient (PCG) algorithm, termed the QN-PCG algorithm. The proposed preconditioner is motivated by the QN algorithm formulated in [8], and approximate the resulting Toeplitz covariance matrix as being formed from a low-order autoregressive (AR) process. It should be stressed that the AR-model is merely used to roughly capture the main features of the spectrum in order to form the preconditioning, but does not necessitate that the signal is an AR-process. Using a GS factorization of the inverse of this approximative covariance matrix as a preconditioning to the CG algorithm will gather the eigenvalues of the matrix in a narrow range, substantially improving the conditioning of the system. The preconditioner is closely related to the technique just proposed in [9], although, in that work, the preconditioner is derived using a rather different approach, resulting eventually in the same form as if it had been derived using an AR-process of order equal to half the data length, whereas in our work it is allowed to be selected as a user parameter, typically being significantly smaller than half the data length. The here presented implementation is also different, and more efficient, than the one in [9]. The resulting QN-PCG-IAA algorithm will, using a sufficient number of iterations, yield an exact implementation of the IAA estimate. However, the resulting algorithm will then be computationally more demanding as compared to our earlier fast IAA (FIAA) implementation proposed in [6]. Here, we are primarily interested in using the proposed reformulation to allow for approximative solutions using only a few PCG iterations, and as a result allowing for substantial computational reductions as compared to the FIAA algorithm, without more than a marginal loss of accuracy in the resulting estimates. In the following section, we briefly overview the IAA algorithm, the FIAA implementation, as well as introduce some notations and novel results used in the latter sections. In the interest of brevity, the presentation of the background details is somewhat dense, and the reader is referred to [1], [4], [6] for further details on the basics of the IAA method and the FIAA implementation. Then, in Section III, we proceed to discuss the proposed QN-PCG-IAA algorithm, as well as an approximative algorithm, termed QN-IAA, that use the estimated preconditioning matrix in place of the estimated covariance matrix of the signal. In Section IV, we illustrate the achieved performance and complexity of the discussed algorithms. Finally, Section V contains our conclusions.

II. An Overview of IAA and FIAA

Let $\{y_n\}_{n=0}^N \in \mathbb{C}$ denote a uniformly sampled sequence of observations for which one wish to compute a spectral estimate. Form the data and frequency vectors $\mathbf{y}_{N+1} = \begin{bmatrix} y_0 & \dots & y_N \end{bmatrix}^T$, $\mathbf{f}_{N+1}(\omega_k) = \begin{bmatrix} 1 & e^{j\omega_k} & \dots & e^{j\omega_k N} \end{bmatrix}^T$, where $(\cdot)^T$ denotes the transpose, and where $\omega_k = 2\pi\frac{k}{K}$, $k = 0, 1, \dots, K-1$, typically[1] with $K > N+1$. Denote the power of the signal $\Phi_s(\omega_k) = |\alpha(\omega_k)|^2$, where $\alpha(\omega_k)$ is the complex-valued spectral amplitude at frequency

Parts of this work has been published at EUSIPCO 2011. This work was supported in part by the Swedish Research Council and Carl Trygger's foundation.

*G. O. Glentis is with the Department of Science and Technology of Telecommunications, University of Peloponnese, Tripolis, 22100 Greece, email: gglentis@uop.gr.

†A. Jakobsson is with the Dept. of Mathematical Statistics, Lund University, P.O. Box 118, SE-221 00 Lund, Sweden, email: aj@maths.lth.se.

[1]For $K = N$, IAA yields the DFT spectrum for all iterations.

ω_k, and let $\mathbf{R}_{N+1}$ denote an estimate of the sample covariance matrix. Then, for all frequencies of interest, the IAA spectral estimate is formed by iteratively estimating $\alpha(\omega_k)$ and $\mathbf{R}_{N+1}$, until practical convergence, as (see [1], [4] for details)

$$\alpha(\omega_k) = \frac{\mathbf{f}_{N+1}^H(\omega_k)\mathbf{R}_{N+1}^{-1}\mathbf{y}_{N+1}}{\mathbf{f}_{N+1}^H(\omega_k)\mathbf{R}_{N+1}^{-1}\mathbf{f}_{N+1}(\omega_k)}, \quad (1)$$

$$\mathbf{R}_{N+1} = \sum_{k=0}^{K-1} |\alpha(\omega_k)|^2 \mathbf{f}_{N+1}(\omega_k)\mathbf{f}_{N+1}^H(\omega_k) \quad (2)$$

where $(\cdot)^H$ denotes the conjugate transpose, with $\mathbf{R}_{N+1}$ initialized to the identity matrix $\mathbf{I}_{N+1}$. Throughout this paper, unless otherwise stated, boldface capital letters, single or double indexed, denote square or rectangular matrices of appropriate size, i.e., $\mathbf{X}_N$ and $\mathbf{X}_{N,M}$ denote $N\times N$ and $N\times M$ matrices, respectively. Typically, no more than $m = 10-15$ iterations are needed to allow for convergence [1]. The brute force implementation of m iterations of the IAA method, as described by (1) and (2), results in a computational cost of $C^{IAA} = m\left[2K(N+1)^2 + K(N+1) + (N+1)^3\right]$ operations. However, this figure can be drastically improved by taking into account the special structure of $\mathbf{R}_{N+1}$ and the operations required to update $\alpha(\omega_k)$ and $\mathbf{R}_{N+1}$ each iteration. To do so, one may note that $\mathbf{R}_{N+1}$ is the upper left part of the circulant matrix [6] $\mathbf{C}_K = \mathbf{W}_K^H \text{diag}\left\{|\alpha(\omega_0)|^2, \ldots, |\alpha(\omega_{K-1})|^2\right\}\mathbf{W}_K$,, where $\mathbf{W}_K$ is the Discrete Fourier Transform (DFT) matrix of size $K \times K$, i.e.,

$$\mathbf{C}_K = \begin{bmatrix} \mathbf{R}_{N+1} & \times \\ \times & \times \end{bmatrix} \quad (3)$$

with the symbol $\times$ denoting unspecified terms of no relevance. Since $\mathbf{C}_K$ is a circulant matrix, with its first column denoted $\mathbf{c}_K$, it can be computed using the Inverse DFT (IDFT) as $\mathbf{c}_K = \mathbf{W}_K^H\boldsymbol{\alpha}_K$, where $\boldsymbol{\alpha}_K = \begin{bmatrix}|\alpha(\omega_0)|^2 & \ldots & |\alpha(\omega_{K-1})|^2\end{bmatrix}^T$. As a consequence of the embedding of $\mathbf{R}_{N+1}$ in $\mathbf{C}_K$, the first column of $\mathbf{R}_{N+1}$, denoted by $\mathbf{r}_{N+1}$, can be extracted as the $N+1$ initial elements of $\mathbf{c}_K$.

$$\mathbf{R}_{N+1}\mathbf{d}_{N+1} \triangleq \mathbf{y}_{N+1} \quad (4)$$

can be solved using the celebrated Levinson-Durbin (LD) algorithm. To do so, partition $\mathbf{R}_{N+1}$ as

$$\mathbf{R}_{N+1} = \begin{bmatrix} r_0 & \mathbf{r}_N^{fH} \\ \mathbf{r}_N^f & \mathbf{R}_N \end{bmatrix} = \begin{bmatrix} \mathbf{R}_N & \mathbf{J}_N\mathbf{r}_N^{f*} \\ \mathbf{r}_N^{fT}\mathbf{J}_N & r_0 \end{bmatrix}, \quad (5)$$

with r_0 and $\mathbf{r}_N^f$ defined accordingly, where $(\cdot)^*$ denotes the conjugate, and $\mathbf{J}_N$ is the exchange matrix, which, by using the matrix inversion lemma (see, e.g., [2]), yields

$$\mathbf{R}_{N+1}^{-1} = \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{R}_N^{-1} \end{bmatrix} + \bar{\mathbf{a}}_{N+1}\bar{\mathbf{a}}_{N+1}^H \quad (6)$$

$$= \begin{bmatrix} \mathbf{R}_N^{-1} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} + \mathbf{J}_{N+1}\bar{\mathbf{a}}_{N+1}^*\bar{\mathbf{a}}_{N+1}^T\mathbf{J}_{N+1} \quad (7)$$

where

$$\bar{\mathbf{a}}_{N+1} = \begin{bmatrix} 1 \\ \mathbf{a}_N \end{bmatrix} / \sqrt{\alpha_N^f} \quad (8)$$

and

$$\mathbf{a}_N = -\mathbf{R}_N^{-1}\mathbf{r}_N^f, \quad \alpha_N^f = r_0 + \mathbf{r}_N^{fH}\mathbf{a}_N \quad (9)$$

The resulting LD algorithm (see page 100 in [2]), allows for a solution of (9) at a cost of approximately N^2 operations. Extension of the LD algorithm to handle (4) as well results in an additional cost of N^2 operations. The overall complexity can be halved by instead using the GS factorization of $\mathbf{R}_{N+1}^{-1}$ for the computation of the matrix vector product that is involved in solving (4). To see this, define the down-shifting operator

$$\mathbf{Z}_{N+1}(\nu) = \begin{bmatrix} \mathbf{0}^T & \nu \\ \mathbf{I}_N & \mathbf{0} \end{bmatrix} \quad (10)$$

and $\mathbf{C}(\boldsymbol{\xi}_{N+1}, \nu)$ as the *ν-circulant* matrix having $\boldsymbol{\xi}_{N+1}$ along its first column, given by

$$\mathbf{C}(\boldsymbol{\xi}_{N+1}, \nu) = \left[\boldsymbol{\xi}_{N+1}\ \mathbf{Z}_{N+1}(\nu)\ldots(\mathbf{Z}_{N+1}(\nu))^N\boldsymbol{\xi}_{N+1}\boldsymbol{\xi}_{N+1}\right]. \quad (11)$$

Clearly, $\mathbf{C}(\boldsymbol{\xi}_{N+1}, 1)$ and $\mathbf{C}(\boldsymbol{\xi}_{N+1}, -1)$ define a *circulant* matrix and *skew-circulant* matrix, respectively. Moreover, $\mathbf{C}(\boldsymbol{\xi}, 0)$ coincides with the Toeplitz lower matrix $\mathbf{L}(\boldsymbol{\xi}_{N+1})$, formed with $\boldsymbol{\xi}_{N+1}$ along its first column. Let μ and ν be two constants. Then, it can be shown that the two following lemmas hold:

Lemma 1: The inverse covariance matrix $\mathbf{R}_{N+1}^{-1}$ can be computed by the following GS-type factorization, provided that $\mu\nu \neq 1$,

$$\mathbf{R}_{N+1}^{-1} = \frac{1}{1-\nu\mu}\sum_{i=1}^{2}\sigma_i\mathbf{C}(\mathbf{t}_{N+1}^i, \nu)\mathbf{C}^H(\mathbf{s}_{N+1}^i, \mu), \quad (12)$$

where $\sigma_1 = 1$ and $\sigma_2 = -1$, and

$$\mathbf{t}_{N+1}^1 \triangleq \bar{\mathbf{a}}_{N+1} \qquad \mathbf{s}_{N+1}^1 = \mathbf{t}_{N+1}^1$$
$$\mathbf{t}_{N+1}^2 \triangleq \mathbf{Z}_{N+1}(\nu)\mathbf{J}_{N+1}\bar{\mathbf{a}}_{N+1}^* \qquad \mathbf{s}_{N+1}^2 \triangleq \mathbf{Z}_{N+1}(\mu)\mathbf{J}_{N+1}\bar{\mathbf{a}}_{N+1}^*$$

Lemma 2: The lower order matrix $\mathbf{R}_N^{-1}$ allows for a GS factorization of the form

$$\begin{bmatrix} \mathbf{R}_N^{-1} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} = \frac{1}{1-\nu\mu}\sum_{i=1}^{2}\sigma_i\mathbf{C}(\bar{\mathbf{t}}_{N+1}^i, \nu)\mathbf{C}^H(\bar{\mathbf{s}}_{N+1}^i, \mu), \quad (13)$$

provided that $\nu\mu \neq 1$, where

$$\bar{\mathbf{t}}_{N+1}^1 \triangleq \bar{\mathbf{a}}_{N+1} \qquad \bar{\mathbf{s}}_{N+1}^1 = \bar{\mathbf{t}}_{N+1}^1$$
$$\bar{\mathbf{t}}_{N+1}^2 \triangleq \mathbf{J}_{N+1}\bar{\mathbf{a}}_{N+1}^* \qquad \bar{\mathbf{s}}_{N+1}^2 \triangleq \mathbf{J}_{N+1}\bar{\mathbf{a}}_{N+1}^*$$

Proof: See the Appendix for proofs of both lemmas. ■

Using $\nu = \mu = 0$, one obtains the standard lower and upper Toeplitz GS decomposition. If instead using $\nu = 0$ and $\mu = 1$, one obtains a lower triangular and circulant GS decomposition variant [10]. Finally,

TABLE I
THE MODIFIED PRECONDITIONED CONJUGATE GRADIENTS METHOD

Initialization

$$\begin{aligned} \mathbf{a}_N &= \mathbf{a}_N^{\text{INI}} \\ \mathbf{e}_N^f &= -\mathbf{r}_N - \mathbf{R}_N\mathbf{a}_N \\ \rho_0 &= \mathbf{e}_N^{fH}\mathbf{e}_N^f \end{aligned}$$

Iterate until practical convergence, for $\kappa = 1, 2, \ldots$

$$\begin{aligned} \mathbf{z}_N &= \mathbf{P}_N^{-1}\mathbf{e}_N^f \\ g_{k-1} &= \mathbf{z}_N^H\mathbf{e}_N^f \\ \gamma &= 0, \quad (\text{if } \kappa = 1) \\ \mathbf{v}_N &= \mathbf{0}, \quad (\text{if } \kappa = 1) \\ \gamma &= g_{\kappa-1}/g_{\kappa-2}, \quad (\text{if } \kappa \neq 1) \\ \mathbf{v}_N &= \mathbf{z}_N + \gamma\mathbf{v}_N, \quad (\text{if } \kappa \neq 1) \\ \mathbf{w}_N &= \mathbf{R}_N\mathbf{v}_N \\ \delta &= g_{\kappa-1}/(\mathbf{v}_N^H\mathbf{w}_N) \\ \mathbf{a}_N &= \mathbf{a}_N + \delta\mathbf{v}_N \\ \mathbf{e}_N^f &= \mathbf{e}_N^f - \delta\mathbf{w}_N \\ \rho_\kappa &= \mathbf{e}_N^{fH}\mathbf{e}_N^f \end{aligned}$$

if $\nu = 1$ and $\mu = -1$, a circulant and skew-circulant GS decomposition is obtained [11]. Using (12), one may reduce the cost of solving (4) to about $\phi(N+1)$ operations, where $\phi(N)$ denotes the cost of performing an FFT of length N, providing that $\bar{\mathbf{a}}_{N+1}$, given by (8), is available. Here, the circulant and skew-circulant GS decomposition is adopted, i.e. (4) is computed via (12) using $\nu = 1$ and $\mu = -1$. This results in a lower computational complexity as compared to the standard approach, bounded by $10\phi(N+1)$ operations, where $4\phi(N+1)$ accounts for the preprocessing and $6\phi(N+1)$ results from the actual matrix vector multiplication [11], a fact that can be useful in the case when the products of the same matrix with several vectors are required. Thus, the numerator of (1), $\mathbf{f}_{N+1}^H(\omega_k)\mathbf{d}_{N+1}$, can be computed by means of fast Toeplitz vector multiplication methods at a cost proportional to $\phi(N+1)$. The denominator of (1) can similarly be handled using a trigonometric polynomial, defined as (see also [12]) $\varphi(\omega_k) = \mathbf{f}_N^H(\omega_k)\mathbf{R}_{N+1}^{-1}\mathbf{f}_N(\omega_k) = \sum_{i=-N}^{N}\varphi_i e^{j\frac{2\pi}{K}k}$. The coefficients φ_i of the trigonometric polynomial can efficiently be computed using the GS representation of $\mathbf{R}_{N+1}^{-1}$ at a cost of $5\phi(2(N+1))$ [13]. Finally, $\varphi(\omega_k)$, for $k = 0, 1 \ldots, K-1$, is computed using an FFT of size equal to K. Thus, the computational complexity of the fast IAA (FIAA) implementation is $C^{FIAA} = m\left[(N+1)^2 + 12\phi(2(N+1)) + 3\phi(K)\right]$. With these results, we are now ready to proceed to formulating the proposed PCG algorithm.

III. The Proposed PCG-IAA implementation

The CG method can be used to form an iterative solution of a set of equations such as (4). The advantage of using the CG method stems from the fact that the matrix vector products involved can be accomplished using efficient techniques based on the FFT due to the Toeplitz nature of $\mathbf{R}_{N+1}$. In theory, after $N+1$ iterations, the CG algorithm provides the exact solution of (4). In practice, however, this may not be so, due to the round-off errors caused by the finite precision arithmetic implementation. Moreover, the convergence rate of the algorithm may seriously be affected when $\mathbf{R}_{N+1}$ is an ill-conditioned matrix. In this case, an increased number of iterations is required to obtain an accurate solution. One way to handle this effect is to apply a preconditioning to speed up the convergence rate of the algorithm, i.e., instead of solving (4), one solves the preconditioned system (see, e.g., [14]) $\mathbf{P}_{N+1}^{-1}\mathbf{R}_{N+1}\mathbf{d}_{N+1} = \mathbf{P}_{N+1}^{-1}\mathbf{y}_{N+1}$, where $\mathbf{P}_{N+1}$ is the so-called the *preconditioning matrix*, which should be easy to construct and to invert, and have the property that the condition number $\kappa(\mathbf{P}_{N+1}^{-1}\mathbf{R}_{N+1})$ should be close to one, or, at least, it should be much lower than the condition number of the original matrix $\mathbf{R}_{N+1}$. As we are here not only interested in estimating the solution of (4), but also in computing the displacement representation of $\mathbf{R}_{N+1}^{-1}$, we present a slightly different version of the standard PCG algorithm, first computing the generators of $\mathbf{R}_{N+1}^{-1}$, then applying the PCG on (9), and, finally, estimating $\mathbf{d}_{N+1}$ using (12). The modified PCG algorithm is tabulated in Table I, where $\mathbf{P}_N^{-1}\mathbf{R}_N\mathbf{a}_N = -\mathbf{P}_N^{-1}\mathbf{r}_N^f$ is solved first, followed by a GS-based computation of $\mathbf{d}_{N+1}$. The computational complexity of PCG methods depends on the type of preconditioning matrix $\mathbf{P}_N$ utilized. Using a circulant preconditioning matrix is perhaps the most popular approach adopted, since the inverse of a circulant matrix is easily computed and a variety of circulant matrices have been proposed and have been studied in terms of convergence efficiency. Regrettably, we have using numerical simulations found that most of the available circulant preconditioning matrices do not work well, or at all, when used in combination with IAA due to the inherent assumptions these make on the spectrum of the Toeplitz matrix, which are not fulfilled in the here examined case of data with sparse spectra (see also the numerical examples given in Section IV). As a result, we will instead propose a novel QN based preconditioner, constructed from an incomplete factorization of $\mathbf{R}_{N+1}^{-1}$. The QN adaptive algorithm, originally proposed in [8] provides an efficient and low complexity implementation scheme of approximate recursive least squares algorithms, by imposing a low order AR approximation on the input signal of the adaptive algorithm. A similar method has also be presented in [15] in the context of AEC. Motivated by [8], we are here applying an QN methodology when forming the iterative solution of Toeplitz systems using the PCG method. Given a Toeplitz matrix $\mathbf{R}_N$, the new preconditioned matrix is constructed from an incomplete factorization of the inverse $\mathbf{R}_N^{-1}$, as if it had been derived using an AR-process of order $M \ll N$.

TABLE II
Ratio the complexity of the PCG method over the complexity of the LD algorithm, $C_R = C^{QN-PCG}/C^{LD}$, when the Number of the PCG iterations $k = 10$.

Using 10 iterations						
N	C-PCG	QN-PCG				
		32	64	128	256	512
64	2.34	3.81	-	-	-	-
128	1.34	1.86	2.32	-	-	-
256	0.76	0.98	1.09	1.44	-	-
512	0.42	0.53	0.56	0.64	0.92	-
1024	0.23	0.29	0.3	0.32	0.39	0.62
2048	0.13	0.16	0.16	0.16	0.18	0.24
4096	0.07	0.09	0.09	0.09	0.09	0.11

Step 1. Assuming that $\mathbf{R}_\ell$ is positive definite, at least for $\ell = M$, $\mathbf{R}_M^{-1}$ is factored using (12) as

$$\mathbf{R}_M^{-1} = \frac{1}{2}\sum_{i=1}^{2}\sigma_i\mathbf{C}(\mathbf{t}_M^i)\mathbf{S}^H(\mathbf{s}_M^i), \tag{14}$$

where, using (8)-(9), $\mathbf{a}_{M-1} = -\mathbf{R}_{M-1}^{-1}\mathbf{r}_{M-1}^f$, and

$$\bar{\mathbf{a}}_M = \begin{bmatrix}1\\ \mathbf{a}_{M-1}\end{bmatrix}/\sqrt{\alpha_{M-1}^f}, \quad \alpha_{M-1}^f = r_0 + \mathbf{r}_{M-1}^{fH}\mathbf{a}_{M-1} \tag{15}$$

with $\mathbf{a}_{M-1}$ and α_{M-1}^f computed either using the LD algorithm, provided that all principal minors up to size $M-1$ are positive definite, or by using another standard linear systems solution method, such as Cholesky's algorithm. This step requires $\mathcal{O}(M^2)$ operations, when the LD algorithm is utilized.

Step 2. Given $\mathbf{a}_M$, the novel QN preconditioner is constructed as the Toeplitz matrix $\mathbf{Q}_N$, whose first column is formed from the autocorrelation sequence q_ℓ, $\ell = 0, 1, \ldots N-1$, defined as

$$q_\ell \triangleq \begin{cases} r_\ell, & \ell = 0, 1, \ldots, M-1 \\ -\sum_{i=1}^{M}\mathbf{a}_{M-1}^{[i]}q_{\ell-i+1}, & \ell = M, \ldots, N-1 \end{cases} \tag{16}$$

The special structure of $\mathbf{Q}_N$ allows for the computation of the inverse matrix $\mathbf{Q}_N^{-1}$ using (6) as

$$\mathbf{Q}_N^{-1} = \begin{bmatrix}\mathbf{0} & \mathbf{0}^T\\ \mathbf{0} & \mathbf{R}_M^{-1}\end{bmatrix} + \mathbf{A}_{N,N-M-1}\mathbf{A}_{N,N-M-1}^H, \tag{17}$$

where, defining $\tilde{a}_N = [\bar{a}_M^T\ \mathbf{0}_{N-M}^T]^T$,

$$\mathbf{A}_{N,N-M-1} = \left[\tilde{a}_N\ Z_{N+1}(0)\tilde{a}_N\ \ldots (Z_{N+1}(0))^{N-M}\,\tilde{a}_N\right] \tag{18}$$

since, by construction, the reflection coefficient $k_\ell = 0$, for $\ell = M+1, M+2, \ldots, N$ (see also (3.5.6) and (3.5.7) in [2]. In practice, $\mathbf{Q}_N$ is never computed, since its inverse is given directly by (17).

Step 3. We proceed to use $\mathbf{P}_N = \mathbf{Q}_N$ as the preconditioning matrix in the PCG scheme described in Table I. In this case, $\mathbf{P}_N^{-1} = \mathbf{Q}_N^{-1}$ can efficiently be implemented using (17). Indeed, due

to the special structure of $\mathbf{Q}_N^{-1}$, the required matrix vector products can be computed using FFT based schemes, at a cost of no more that $\mathcal{O}(\phi(N)) + \mathcal{O}(\phi(M))$ operations. The computational complexity of the proposed PCG scheme for the computation of the generator of $\mathbf{R}_{N+1}^{-1}$ is thus $C^{QN-PCG} = k\Big(8\phi(N) + 9N + 6\phi(M) + 6M\Big) + M^2 + 9\phi(N) + 3N$, where the first term corresponds to the repetitive computations imposed by the method and the second term to the initialization of the algorithm. The computational gain achieved by the proposed QN-PCG method over the standard LD algorithm is illustrated in Table III, giving the ratio of the complexity of the PCG method over that of the LD algorithm, for matrix sizes up to $N = 4096$, and various QN sizes and when the number of iterations is set $k = 10$. As is clear from the table, the QN-PCG offers a substantial computational saving as compared to the LD algorithm, especially for longer vectors. We proceed to use the above QN-PCG algorithm to estimate the displacement representation of $\mathbf{R}_{N+1}^{-1}$, i.e., the estimation of $\bar{\mathbf{a}}_{N+1}$, as defined in (8), by means of computing $\mathbf{a}_N$ in (9) using the PCG algorithm tabulated in Table I along with the QN preconditioner in (17). We term the resulting scheme the *Quasi-Netwon PCG IAA* (QN-PCG-IAA) algorithm. The initialization of the QN-PCG algorithm of Table I can be done either by setting $\mathbf{a}_N^{\text{INI}} = \mathbf{0}_N$, or more efficiently, by using the estimate of $\mathbf{a}_N$ obtained in the previous IAA iteration. This stems from the fact that, upon convergence, $\mathbf{R}_{N+1}$ will not change too much in between successive IAA iterations. The successive IAA iteration initialization scheme converges faster, or equivalently, a lower number of PCG iterations are required to achieve a given accuracy. Finally, better performance can be achieved by introducing a Newton-based refinement scheme following the QN-PCG solution, with [16]

$$\mathbf{e}_N^f = -\mathbf{r}_N^f - \mathbf{R}_N \mathbf{a}_N \tag{19}$$

$$\mathbf{a}_N = \mathbf{a}_N + \mathbf{R}_N^{-1} \mathbf{e}_N^f \tag{20}$$

iterated until practical convergence. Usually, only very few steps are required for the convergence of the algorithm, provided that a good guess for $\mathbf{a}_N$ is used for initialization, and an accurate estimate of $\mathbf{R}_N^{-1}$ is available. Here, the output of the QN-PCG is used for initialization and $\mathbf{R}_N^{-1}$ is re-estimated at each step from its displacement representation, using (13), at $O(\phi(N))$ cost. As an alternative, one may form an approximative IAA algorithm by instead of computing $\mathbf{R}_{N+1}^{-1}$ use the proposed matrix $\mathbf{Q}_{N+1}^{-1}$, as defined in (17), in place of $\mathbf{R}_{N+1}^{-1}$ directly in the IAA algorithm. In this way, an approximate IAA algorithm is formed by iteratively estimating $\alpha(\omega_k)$ and $\mathbf{Q}_{N+1}$,

$$\alpha(\omega_k) = \frac{\mathbf{f}_{N+1}^H(\omega_k)\mathbf{Q}_{N+1}^{-1}\mathbf{y}_{N+1}}{\mathbf{f}_{N+1}^H(\omega_k)\mathbf{Q}_{N+1}^{-1}\mathbf{f}_{N+1}(\omega_k)}, \tag{21}$$

$$\mathbf{R}_M = \sum_{k=0}^{K-1} |\alpha(\omega_k)|^2 \mathbf{f}_M(\omega_k)\mathbf{f}_M^H(\omega_k) \tag{22}$$

$$\mathbf{Q}_{N+1}^{-1} = \begin{bmatrix} \mathbf{0} & \mathbf{0}^T \\ \mathbf{0} & \mathbf{R}_M^{-1} \end{bmatrix} + \mathbf{A}_{N+1,N-M}\mathbf{A}_{N+1,N-M}^H \tag{23}$$

until practical convergence, for $k = 0, 1, \ldots, K-1$. Using $M \ll N$, a significant computation reduction can then be achieved, at the expense of a possible degradation in the quality of the spectrum estimate. Since $M \ll N$, the LD algorithm is employed for the computation of the generators of $\mathbf{R}_M^{-1}$. Moreover, the denominator of (21) can be computed efficiently, since $\varphi(\omega_k) \triangleq \mathbf{f}_{N+1}^H(\omega_k)\mathbf{Q}_{N+1}^{-1}\mathbf{f}_{N+1}(\omega_k)$ equals to $\varphi(\omega_k) = \hat{\varphi}(\omega_k) + (N-M)|\mathbf{f}_M^H\bar{\mathbf{a}}_M|^2$ where $\hat{\varphi}(\omega_k) \triangleq \mathbf{f}_M^H(\omega_k)\mathbf{R}_M^{-1}\mathbf{f}_M(\omega_k)$. We denote the resulting approximative algorithm the QN-IAA algorithm. It is worth noting that, using (15), one obtain the interpretation of $\varphi(\omega_k)$ as the harmonic mean of all lower order AR spectra, i.e., for the original IAA algorithm

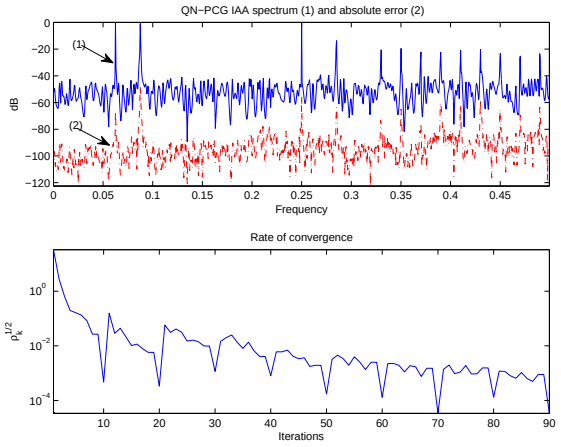

Fig. 1. a) Spectral estimates obtained using IAA for the data set in [17], with $N + 1 = 512$, $m = 10$, and $K = 2048$ and the absolute error between the IAA and the QN-PCG-IAA algorithm using only $k = 8$ QN-PCG iterations followed by 2 Newton iterations. b) Rate of convergence

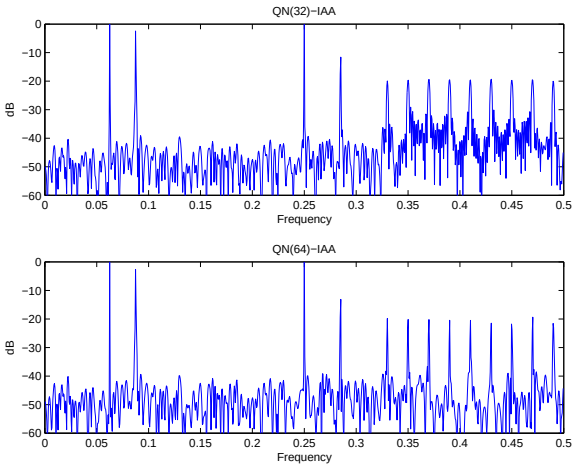

Fig. 2. Performance of the QN-IAA algorithm, with $M = 32$ and 64, for $N = 511$, $m = 10$, and $K = 2048$.

$\varphi(\omega_k) = \sum_{n=0}^{N} \left(\alpha_i^{AR}(\omega_k)\right)^{-1}$, whereas for the QN-IAA algorithm, one obtain

$$\hat{\varphi}\omega_k) = \sum_{n=0}^{M-1} \frac{1}{\alpha_i^{AR}(\omega_k)} + \frac{(N-M)}{\alpha_M^{AR}(\omega_k)} \tag{24}$$

where $\alpha_i^{AR}(\omega_k)$ denotes the spectrum estimated by an AR model of order i.

IV. Performance evaluation

In this section, we examine the performance of the proposed QN-PCG-IAA algorithm using numerical simulations. For simplicity, the data set described in [17] is used, with the number of available data samples set equal to $N + 1 = 512$. We use $m = 10$ IAA iterations, and, when relevant, the predictor size associated with the QN preconditioner set equal to $M = 62$. The size of the frequency grid is set to $K = 2048$. Figure 1 (a) illustrates the IAA spectrum estimate of the examined data set. However, to be competitive with the FIAA algorithm, the proposed QN-PCG-IAA method should be able to provide fairly accurate results using only a few PCG iterations.

The performance of the proposed scheme when $M = 64$, the number of QN-PCG iterations is $k = 8$, followed by $k_1 = 2$ Newton refinement iterations, is illustrated in Figure 1 (a), where the norm error, over all frequencies, in this case equals $1.2 \cdot 10^{-5}$. As can be seen from the figure, the algorithm offers an almost as good estimate in this case, although at a computational complexity of only 55% of the one required by the FIAA algorithm. Here, the jagged nature of the rate of convergence is due to the evolution over the IAA iterations, with the error temporarily increasing as a new covariance matrix is formed in the beginning of a new iteration. Working on larger data sets while keeping the QN-PCG parameters the same as before, a significant complexity reduction is achieved, reducing the computational load to a small fraction of that required by the FIAA algorithm, namely 30%, 17%, and 9% at the expense of a higher error norm, being $8 \cdot 10^{-5}$, $7.3 \cdot 10^{-3}$ and $9 \cdot 10^{-3}$, when N=1023, 2047, and 4095, respectively. Clearly, the computational reduction versus accuracy trade-off is prominent, especially as N becomes larger. Increasing the QN parameter, M, or the number of PCG iterations, k, more accurate estimates are obtained. Using, for example, $M = 512$ or $M = 1024$, the error norm is reduced to $1.5 \cdot 10^{-4}$ and $3 \cdot 10^{-6}$, while the complexity remains at 12% and 17% of that of the FIAA method, respectively, for the case when $N = 4095$. Should one increases the number of the PCG iterations to $k = 18$ and $k_1 = 2$, even better results are obtained, namely $2.1 \cdot 10^{-5}$ and $4.8 \cdot 10^{-9}$ at a cost of 17% and 26% of that of the FIAA, respectively. However, it is worth noting that, from a practical point of view, the lower accuracy does not affect the quality of the estimated spectra noticeably, since in all cases the possible amplitude misadjustment is far below that of the noise margin at the particular frequencies, and does not affect the frequency discrimination at all. We proceed to examine the performance of the proposed QN-IAA algorithm for various values of the parameter M, noting that, when $M = 0$, the QN-IAA algorithm degenerates to the DFT. As is shown in Figure 2, QN-IAA performs pretty well, even for small values of M as compared to the data size $N + 1$, which is a remarkable achievement taking into account the crude approximation imposed on the data covariance matrix. The computational complexity of the proposed methods is illustrated in Figure 3. The curves designated by QN-PCG(k)-IAA correspond to the cost of the proposed QN-PCG-IAA algorithm, using $k-2$ PCG and 2 Newton refinement iterations, for $M = 32$, $M = 64$, and $M = 128$. Clearly, the algorithm allows for up to an order of magnitude improvement as compared to the complexity of the FIAA algorithm, where QN-PCG(10)-IAA becomes more efficient than the FIAA for $N > 256$ and QN-PCG(20)-IAA for $N > 512$, and where the influence of the size of M on the overall complexity is noticeable only for relatively small values of N. From the figure, the proposed approximate QN-IAA algorithm is also seen to offers a significant further cost reduction (up to two orders of magnitude) for $M = 16$, $M = 32$, $M = 64$, and $M = 128$. It is worth noting that the proposed preconditioning method can also be used for the solution of Hermitian positive definite Toeplitz equations. As shown in Figure 4, illustrating the eigenvalue spread for the covariance matrix with and without preconditioning, i.e., $\hat{\mathbf{R}}_N^{-1}\mathbf{R}_N$ and $\mathbf{R}_N$, respectively, for $N + 1 = 512$, the proposed QN-PCG-IAA algorithm performs quite good, even when a very few number of iterations are used for the QN-PCG Toeplitz solver. Let the elements of the first column of a Toeplitz matrix, $\mathbf{T}$, be the Fourier coefficients of the generating function $f(\theta)$, i.e., (see also [14]) $t_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(\theta)e^{-jn\theta}d\theta$, $n = 0, \pm 1, \pm 2, \ldots$. The effectiveness of the proposed preconditioner can then be examined in the context of the iterative Toeplitz solvers, using the generating functions $f_1(\theta) = \theta^4 + 1$, $f_2(\theta) = |\theta|^3 + 1$, $f_3(\theta) = \theta^4$, and $f_4(\theta) = \theta^4(\pi^2 - \theta^2)$ to generate Toeplitz matrices of size $N = 1024$, here designated by

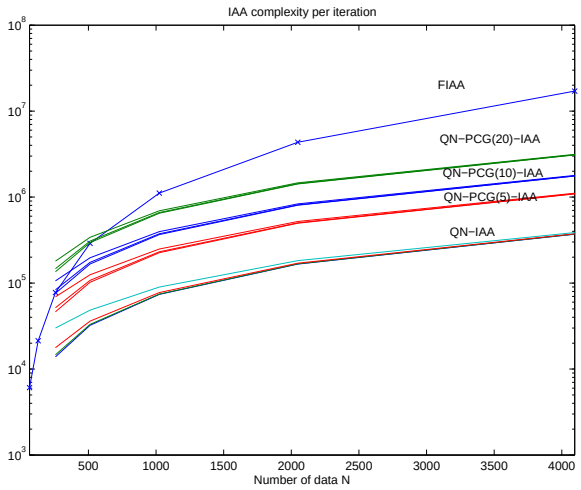

Fig. 3. The computational complexity of the discussed algorithms for different predictor sizes, M.

$\mathbf{T}_1$, $\mathbf{T}_2$, $\mathbf{T}_3$, and $\mathbf{T}_4$, respectively. Moreover, a fifth matrix, $\mathbf{T}_5$, is considered, as resulting from the $m = 10$ iteration of the standard FIAA algorithm, $\mathbf{R}_N$. The right hand side vector of the Toeplitz linear system to be solved is set equal to ones, i.e., $-\mathbf{r}_N^f \triangleq \mathbf{b}_N = [1 \ldots 1]^T$. The PCG algorithms are all initialized by setting the initial guess of the solution sought equal to zero, while iterations are performed until a desired threshold of the solution error norm, $\sqrt{\rho}$, is reached, usually expressed as $tol \cdot \sqrt{\mathbf{b}_N^T\mathbf{b}_N}$, where $tol = 10^{-7}$. The number of iterations required by each PCG algorithm to reach the desired error threshold are tabulated in Table III, where † indicates that the CG iterations did not converge. Apart from the standard CG algorithm (without preconditioning), here denoted I, the PCG using various circulant preconditioners are examined, namely T. Chan's preconditioner, denoted C_F, and the preconditioners resulting from the use of the r-th order generalized Jackson kernels, denoted $K_{\kappa,2r}$, $r = 2, 3, 4$, which are the preconditioners considered to be the more efficient in the case of ill-conditioned Toeplitz systems [14]. The proposed preconditioning method, denoted $QN(M)$, is considered for several values of M; it should be stressed that in the final case, when $M = N/2$, the method is equivalent to the preconditioning approach just presented in [9]. As is clear from the table, the proposed preconditioner offers substantial improvements as compared to the typically used preconditioners, especially for large values of M. However, it is worth noting that the proposed version, allowing for the possibility of using $M < N/2$, enables the use of a lower complexity preconditioner [2], while still offering almost the same performance as the high order preconditioner proposed in [9].

V. Conclusions

In this work, we have presented computationally efficient approximative formulations of our earlier *exact* implementations of the recent IAA spectral estimator. The proposed one- and two-dimensional *approximate* algorithms have been shown to offer a

[2]It is worth noting that the implementation in [9] is based on an upper-triangular fast Toeplitz computation, as compared to the here presented circulant skew-circulant fast Toeplitz and inverse Toeplitz computations, with the former requiring at least double, or more, computations as compared to the latter. Even though our implementation also offer some other further simplifications, these here introduced simplifications could also be included in [9], and therefore, as such, does not make the results differ. The notable improvement instead lies in the possibility of using lower order models.

TABLE III
NUMBER OF PCG ITERATIONS FOR VARIOUS TOEPLITZ MATRICES AND PRECONDITIONING MATRICES, FOR $N = 1024$ AND $tol = 10^{-7}$.

	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$	$\mathbf{T}_5$
I	71	430	†	†	728
C_F	5	10	587	350	544
$K_{\kappa,4}$	5	6	24	25	893
$K_{\kappa,6}$	5	6	23	24	929
$K_{\kappa,8}$	5	6	22	25	1004
$QN(8)$	3	17	902	817	3162
$QN(16)$	3	5	280	290	1737
$QN(32)$	2	3	94	95	552
$QN(64)$	2	2	38	40	164
$QN(128)$	1	2	21	23	111
$QN(256)$	1	1	13	15	58
$QN(512)$	1	1	9	10	28

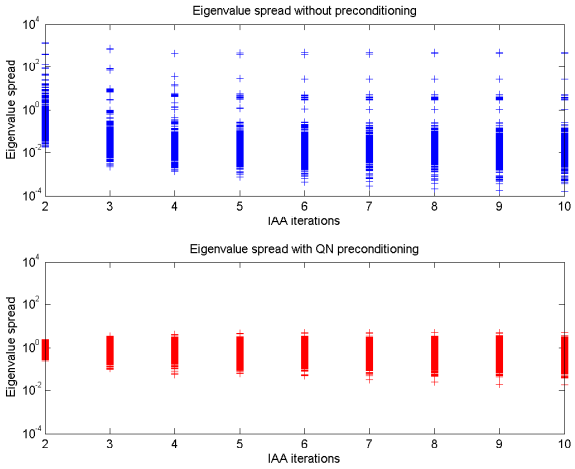

Fig. 4. Evolution of the eigenvalue spread of the covariance matrix, without and with preconditioning, using $N + 1 = 512$, $M = 64$, and $k = 10$.

substantial complexity reduction as compared to their exact counterparts, although without more than a marginal loss of performance. The proposed algorithms exploit a novel preconditioning of an incomplete factorization of a Toeplitz matrix, generalizing a just proposed preconditioning algorithm formed using a similar principle. The estimates are then formed by rewriting the IAA estimate using suitable Gohberg-Semencul representations, and solving the resulting linear set of equations using a preconditioned conjugate gradient technique. Complexity calculations and numerical examples illustrate the accuracy and efficiency of the proposed algorithm.

APPENDIX

A. Proof of Lemma 1

Using (6), (7), along with that $\mathbf{Z}_{N+1}^{N+1}(\nu) = \nu \mathbf{I}_{N+1}$, one can express the displacement of $\mathbf{R}_{N+1}^{-1}$ with respect to the lower shifting block matrices $\mathbf{Z}_{N+1}(\nu)$ and $\mathbf{Z}_{N+1}(\mu)^T$ as

$$\nabla \mathbf{R}_{N+1}^{-1} \triangleq \mathbf{R}_{N+1}^{-1} - \mathbf{Z}_{N+1}(\nu)\mathbf{R}_{N+1}^{-1}\mathbf{Z}_{N+1}^T(\mu)$$

$$= \bar{\mathbf{a}}_{N+1}\bar{\mathbf{a}}_{N+1}^H - \mathbf{Z}_{N+1}(\nu)\mathbf{J}_{N+1}\bar{\mathbf{a}}_{N+1}^*\bar{\mathbf{a}}_{N+1}^T\mathbf{J}_{N+1}\mathbf{Z}_{N+1}^T(\mu),$$

which, via repeated multiplications by the down-shifting operator (10), and by the subsequent summation of the resulting equations, leads to the desired expressions [18].

B. Proof of Lemma 2

Using (6) and (7), one obtain

$$\begin{bmatrix} \mathbf{R}_N^{-1} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} - \mathbf{Z}_{N+1}(\nu) \begin{bmatrix} \mathbf{R}_N^{-1} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} \mathbf{Z}_{N+1}^T(\mu) =$$
$$\bar{\mathbf{a}}_{N+1}\bar{\mathbf{a}}_{N+1}^H - \mathbf{J}_{N+1}\bar{\mathbf{a}}_{N+1}^*\bar{\mathbf{a}}_{N+1}^T\mathbf{J}_{N+1}$$

which, via repeated multiplications by the down-shifting operator (10), and by the subsequent summation of the resulting equations, yields (13).

REFERENCES

[1] T. Yardibi, J. Li, P. Stoica, M. Xue, and A. B. Baggeroer, "Source Localization and Sensing: A Nonparametric Iterative Approach Based on Weighted Least Squares," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 46, no. 1, pp. 425–443, January 2010.

[2] P. Stoica and R. Moses, *Spectral Analysis of Signals*, Prentice Hall, Upper Saddle River, N.J., 2005.

[3] W. Roberts, P. Stoica, J. Li, T. Yardibi, and F. A. Sadjadi, "Iterative Adaptive Approaches to MIMO Radar Imaging," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 1, pp. 5–20, Feb 2010.

[4] P. Stoica, J. Li, and J. Ling, "Missing Data Recovery Via a Nonparametric Iterative Adaptive Approach," *IEEE Signal Processing Letters*, vol. 16, no. 4, pp. 241–244, April 2009.

[5] N. R. Butt and A. Jakobsson, "Coherence Spectrum Estimation From Nonuniformly Sampled Sequences," *IEEE Signal Processing Letters*, vol. 17, no. 4, pp. 339–342, April 2010.

[6] G.-O. Glentis and A. Jakobsson, "Efficient Implementation of Iterative Adaptive Approach Spectral Estimation Techniques," *IEEE Trans. Signal Processing*, vol. 59, no. 9, pp. 4154 – 4167, Sept. 2011.

[7] G.-O. Glentis and A. Jakobsson, "Time-Recursive IAA Spectral Estimation," *IEEE Signal Processing Letters*, vol. 18, no. 2, pp. 111–114, Feb. 2011.

[8] G. V. Moustakides and S. Theodoridis, "Fast Newton Transversal Filters - A New Class of Adaptive Estimation Algorithms," *IEEE Trans. Signal Processing*, vol. 39, no. 10, pp. 2184–2193, Oct. 1991.

[9] Y.-W. Wen, W.-K. Ching, and M. Ng, "Approximate Inverse-Free Preconditioners for Toeplitz Matrices," *Applied Mathematics and Computation*, vol. 217, no. 16, pp. 6856–6867, April 2011.

[10] G. Ammar and P. Gader, "A Variant of the Gohberg–Semencul Formula Involving Circulant Matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 12, no. 3, pp. 534–540, July 1991.

[11] I. Gohberg and V. Olshevksy, "Complexity of multiplication with vectors for structured matrices," *Lin. Algebra Appl.*, vol. 202, pp. 163–192, 1994.

[12] H. Li, P. Stoica, and J. Li, "Capon Estimation of Covariance Sequences," *Circuits, Systems, and Signal Processing*, vol. 17, no. 1, pp. 29–49, 1998.

[13] G.-O. Glentis, "A Fast Algorithm for APES and Capon Spectral Estimation," *IEEE Trans. on Signal Processing*, vol. 56, no. 9, pp. 4207–4220, Sept. 2008.

[14] R. Chan and X. Jin, *An introduction to iterative Toeplitz solvers*, SIAM, 2007.

[15] B. Farhang-Boroujeny, "Fast LMS/Newton algorithms based on autoregressive modeling and their application to acoustic echo cancellation," *IEEE Trans. Signal Processing*, vol. 45, no. 8, pp. 1987–2000, Aug. 1997.

[16] V. Pan, *Structured matrices and polynomias: Unified superfast algorithms*, Birkhauser Boston, 2001.

[17] P. Stoica, A. Jakobsson, and J. Li, "Matched-Filterbank Interpretation of Some Spectral Estimators," *Signal Processing*, vol. 66, no. 1, pp. 45–59, April 1998.

[18] T. Kailath and A. H. Sayed, "Displacement Structure: Theory and Applications," *SIAM Review*, vol. 37, no. 3, pp. 297–386, Sept. 1995.