FAST ALGORITHMS FOR ITERATIVE ADAPTIVE APPROACH
SPECTRAL ESTIMATION TECHNIQUES∗

George-Othon Glentis
University of Peloponnese
Dept. of Science and Technology of Telecom.
Tripolis, 22100 Greece
email: gglentis@uop.gr

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

ABSTRACT

This paper presents computationally efficient implementations for Iterative Adaptive Approach (IAA) spectral estimation techniques for uniformly sampled data sets. By exploiting the methods inherent low displacement rank, together with the development of suitable Gohberg-Semencul representations, and the use of data dependent trigonometric polynomials, the proposed implementations are shown to offer a reduction of the necessary computational complexity with at least one order of magnitude. Numerical simulations together with theoretical complexity measures illustrate the achieved performance gain.

Index Terms— Spectral analysis, estimation, fast algorithms.

1. INTRODUCTION

Spectral estimation finds applications in a wide range of fields, and has received substantial and well-justified interest in the literature over the last century. Due to their inherent robustness to model assumptions, there has recently been a notable interest in various semi-parametric spectral estimation techniques. One such promising technique that is currently widely studied is the so-called iterative adaptive approach (IAA), originally presented in [1] to provide a sparse signal representation for passive sensing, channel estimation, and single-antenna radar applications. As shown in [1], the technique is able to outperform conventional data-adaptive techniques, and, as a result, the technique has since been expanded to handle angle-Doppler imaging for MIMO radar [2], missing data recovery [3], non-uniformly sampled real-valued data [4], coherence and polychromatic estimation [5, 6], as well as applications in spectroscopy [7], and blood velocity estimation using ultrasound [8]. As can be seen in these papers, the IAA-based estimation techniques are able to provide accurate estimates even when only a few data snapshots are available. However, as is also mentioned in several of these works, the improved performance comes at a cost of a considerable computational complexity, suggesting the need for computationally efficient implementations and for approximations allowing for computational savings without sacrificing too much of the performance quality. One such approximation was introduced in [5], where a segmented IAA (SIAA) algorithm that forms averaged spectral estimates of lower dimensions was proposed, allowing both for computational savings and for a trade-off between variance and bias in the resulting estimates. In this work, we instead focus on deriving computationally efficient implementations, maximally exploiting the structure of the algorithms.

2. PROBLEM FORMULATION

Let \{y_n\}_{n=0}^{N-1} denote a sequence of observations for which one wish to compute a spectral estimate. We assume that the observation times \{n\}_{n=0}^{N-1} are given, and that \(y_n \in \mathbb{C}\). Form the data and frequency vectors

\[
y_N = \begin{bmatrix} y_0 & \cdots & y_{N-1} \end{bmatrix}^T
\]

\[
f_N(\omega_k) = \begin{bmatrix} 1 & e^{j\omega_k} & \cdots & e^{j\omega_k(N-1)} \end{bmatrix}^T
\]

where \((\cdot)^T\) denotes the transpose, and where \(\omega_k = 2\pi \frac{k}{N}, k = 0, 1, \ldots, K - 1\). Let \(\Phi(\omega_k) = |\alpha(\omega_k)|^2\) denote the power of the signal at frequency \(\omega_k\), where \(\alpha(\omega_k)\) is the complex-valued spectral amplitude at frequency \(\omega_k\). Let \(R_N\) denote an estimate of the sample covariance matrix. Then, the IAA spectral estimate is formed by iteratively estimating \(\alpha(\omega_k)\) and \(R_N\), until practical convergence, as (see [1, 3] for details)

\[
\alpha(\omega_k) = \frac{f_N^H(\omega_k)R_N^{-1}y_N}{\bar{f}_N^H(\omega_k)f_N(\omega_k)}
\]

\[
\Phi(\omega_k) = \frac{|\alpha(\omega_k)|^2}{\sum_{k=0}^{K-1} \Phi(\omega_k)f_N(\omega_k)f_N^H(\omega_k)}
\]

where \((\cdot)^H\) denotes the conjugate transpose, with \(R_N\) initialized to the identity matrix \(I_N\). Alternatively, one may wish to allow for the segmentation proposed in [5], forming the SIAA algorithm. As detailed in [5], this version of IAA allows for filter lengths shorter than \(N\), typically offering estimates with lower variance. As is well-known in case of e.g. Capon-based spectral estimators, the use of longer filter lengths leads to spectral estimates with high variance, often resulting in strong spurious peaks (see, e.g., [9]). On the other hand, a shorter filter length increases the estimator bias. To allow for a trade-off between variance and bias, it was proposed in [5] that the used filter length was reduced by dividing the data into \(L\) segments of length \(N_s\) each. In the case of non-overlapping segments, \(N_s = N/L\). Thus, consider the data vectors \(y_{N_s}^i, i = 1, 2, \ldots, L\), with \(N_s < N\), that correspond to possibly overlapping segmentation of the original data vector. Define the frequency vector that corresponds to the \(i\)-th segment

\[
\bar{f}_{N_s}^i(\omega_k) = [e^{j\omega_k n_i} e^{j\omega_k(n_i+1)} \ldots e^{j\omega_k(n_i+N_s-1)}]^T
\]
over a uniformly spaced grid of frequencies, $\omega_k = 2\pi k/N$. Here, $n_i$ denotes the time index that corresponds to the first sample of the $i$-th segment. Then, the Segmented IAA, hereafter termed the SIAA-I, is formed by iterating

$$\alpha_s^i(\omega_k) = \frac{\hat{F}_{Nk}^H(\omega_k) R_{Nk}^{-1} y_{Nk}}{\hat{F}_{Nk}^H(\omega_k) R_{Nk}^{-1} \hat{F}_{Nk}(\omega_k)}$$

until convergence, for all $i = 1, 2, \ldots, L$ (see [5] for further details).

Finally, the averaged spectral estimate is formed by averaging the spectra of the segmented data as

$$\Phi_s(\omega_k) = \frac{1}{L} \sum_{i=1}^{L} |\alpha_s^i(\omega_k)|^2.$$  

In an attempt to reduce the computational cost further, an alternative formulation of the Segmented IAA may be formed, instead iterating

$$\alpha_s^i(\omega_k) = \frac{\hat{f}_{Nk}^H(\omega_k) R_{Nk}^{-1} y_{Nk}}{\hat{f}_{Nk}^H(\omega_k) R_{Nk}^{-1} \hat{f}_{Nk}(\omega_k)}$$

until convergence, for all $i = 1, 2, \ldots, L$. The resulting algorithm is termed SIAA-II. The advantage of using this re-formulation is that the complexity is now reduced, since a single covariance matrix has
to be calculated and be inverted, instead of $L$ required by the SIAA-I. This also offers an improved robustness in the estimation scheme, since $R_{Nk}$ is now an average of the covariance matrices that correspond to each individual segment. Indeed,

$$R_{Nk} = \frac{1}{L} \sum_{i=1}^{L} R_{Nk}$$

where $R_{Nk}$ is given by (11). The brute force implementation of the IAA method as described by (3)-(5) results in a computational cost of $2KN^2 + KN + N^3$ operations per IAA iteration. Let $m$ be the number of IAA iterations necessary to allow for convergence; typically, no more than 10 or 15 iterations are needed [1]. Then, the total cost of the IAA method is $m(2KN^2 + KN + N^3)$. The complexity of the brute force implementation of the SIAA-I and SIAA-II methods are $mL(2KN^2 + KN + N^2)$ and $m(L(2KN^2 + N^2 + 2KN_L))$, respectively. In the case of non-overlapping segments, $N_k = N/L$, which indicates that, even for the brute force implementation, the complexity of the SIAA-I and the SIAA-II is reduced by a factor of $L$ and $L^2$, respectively, compared to the complexity of the IAA method. However, these figure can be drastically improved by taking into account the special structure of $R_N$ and the operations required to update $\alpha(\omega_k)$ and $R_N$ each iteration.

### 3. FAST IAA IMPLEMENTATIONS

To form the efficient algorithms, define a matrix containing the frequency vectors, $f_N(\omega_k)$, $k = 0, 1, \ldots, K - 1$, as

$$F_{NK} = \left[ f_N(\omega_0) \ldots f_N(\omega_{K-1}) \right],$$

and express the sample covariance matrix estimate in (5) as

$$R_N = F_{NK} \text{diag} \left( |\alpha(\omega_0)|^2 \ldots |\alpha(\omega_{K-1})|^2 \right) F_{NK}^H,$$

where $\text{diag}\{x\}$ denotes a diagonal matrix formed with the vector $x$ along the diagonal. Clearly, $R_N$ is a Toeplitz matrix. Consider the partition

$$W_{NN}^H = \left[ F_{NK} \times \right],$$
where $W_{KK}$ is the Discrete Fourier Transform (DFT) matrix of size $K \times K$, and the symbol $\times$ denotes unspecified terms of no relevance. Using (19), one notes that $R_N$ is the upper left part of the circulant matrix

$$C_K = W_{KK}^H \text{diag} \{ |\alpha(\omega_0)|^2, \ldots, |\alpha(\omega_{K-1})|^2 \} W_{KK}, \quad (20)$$

i.e.,

$$C_K = \begin{bmatrix} R_N & \times \\ \times & \times \end{bmatrix}. \quad (21)$$

Since $C_K$ is a circulant matrix, with its first column denoted $c_K$, it can be computed using the Inverse DFT (IDFT) as

$$c_K = W_{KK}^H a_K, \quad (22)$$

where $a_K = [ |\alpha(\omega_0)|^2 \ldots |\alpha(\omega_{K-1})|^2 ]^T$. As a consequence of (21), the first column of $R_N$, denoted by $r_N$, is obtained by the partition

$$c_K = \begin{bmatrix} r_N \\ \times \end{bmatrix}. \quad (23)$$

Once $r_N$ has been computed, the solution of the linear system of equations that appears in the numerator of (3), i.e.,

$$d_N = R_N^{-1} \gamma_N \quad (24)$$

can be solved using the celebrated Levinson-Durbin algorithm, at a cost of approximately $2N^2$ operations. This figure can be halved, using the Gohberg-Semencul (GS) factorization of the matrix $R_N^{-1}$ for the computation of the matrix vector product that is involved in (24). Consider the partition

$$R_N = \begin{bmatrix} r_0 & r_{N-1}^H \\ r_{N-1} & R_{N-1} \end{bmatrix}. \quad (25)$$

The linear predictor of order $N - 1$ is defined as

$$R_{N-1} a_{N-1} = -r_{N-1}, \quad (26)$$

and the associated prediction error is given by

$$\alpha_N^f = r_0 + r_{N-1}^H a_{N-1}. \quad (27)$$

The linear system of equations (26) can be solved using the Levinson-Durbin algorithm at a cost of $(N - 1)^2$ operations. Let

$$t_N = \begin{bmatrix} 1 \\ a_{N-1} \end{bmatrix} / \sqrt{\alpha_N^f}. \quad (28)$$

Then, the GS factorization of $R_N^{-1}$ has the form (see, e.g. [9])

$$R_N^{-1} = L(t_N) L^H(t_N) - \mathcal{L}(Z_N J_N t_N^*) \mathcal{L}^H(Z_N J_N t_N) \quad (29)$$

where $L(\cdot)$ is a lower triangular Toeplitz matrix, $Z_N$ is the unit lower shift matrix, and $J_N$ is the exchange matrix. Thus, the vector $d_N$ can be computed by means of fast Toeplitz vector multiplication methods at a cost proportional to $\phi(N)$, where $\phi(N)$ denotes the cost of performing a Fast Fourier Transform (FFT) (or an Inverse FFT, IFFT) of size equal to $N$. The contribution of this part to the overall cost of the algorithm is marginal, and we will thus not go into a detailed analysis of the specific implementation of the fast Toeplitz vector multiplication methods. The obvious way to perform this task is by circular embedding of a Toeplitz matrix, which results in a cost of $7\phi(2N)$. It thus remains to show how the numerator and the denominator of (3) can be efficiently evaluated over the grid of frequencies, $\omega_k$, $k = 0, 1, \ldots, K - 1$. Using (17), (19) and (24), the numerator of (3) can be expressed as

$$P_{KK}^H d_N = W_{KK} \begin{bmatrix} d_N \\ 0_{K-N} \end{bmatrix}. \quad (30)$$

which, obviously, can be implemented using the FFT. The denominator of (3) can similarly be handled using a trigonometric polynomial, defined as (see also [10])

$$\varphi(\omega_k) = t_N^H(\omega_k) R_N^{-1} r_N(\omega_k) = \sum_{m=-\infty}^{N-1} \varphi(x) e^{2\pi i \omega_k m}. \quad (31)$$

The coefficients of the trigonometric polynomial defined by (31) can efficiently be computed using the GS representation of $R_N^{-1}$ [11], at a cost of $5\phi(2N)$. Finally, $\varphi(\omega_k)$, $k = 0, 1, \ldots, K - 1$ is computed using an FFT of size equal to $K$. Thus, the computational
4. SIMULATIONS

The validity of the proposed algorithms is demonstrated by typical numerical examples. First, the simulation scenario of [12] is adopted, where a mixture of sinusoidal signals corrupted by additive zero-mean complex Gaussian noise is utilized as a test signal. The sinusoidal signal is composed by four dominant sinusoids and nine other sinusoids of lower energy, located at frequencies 0.025, 0.0875, 0.25, 0.295, 0.33, 0.35, 0.37, 0.39, 0.41, 0.43, 0.45, 0.47 and 0.49. The SNR with respect to the first sinusoid is set equal to 20dB. The simulation scenario of [3] is then considered. In this case, a mixture of sinusoidal signals corrupted by additive zero-mean complex Gaussian noise is utilized as a test signal. The sinusoidal signal is composed by four sinusoids located at frequencies 0.025, 0.05, 0.27 and 0.28. The complex amplitude of the fourth signal is set equal to 0.5 and all the rest are set equal to one. The noise variance is set equal to $\sigma^2 = 0.01$. In all cases, $N = 256$ data are considered, using $K = 1024$. For IAA, the result have been obtained using $m = 10$ iterations, and, for the SIAA, four non-overlapping segments of length equal to $N_s = N/4 = 64$ have been used. As for the APES, the filter bank length is set to $M = N/4 = 64$. Figures 1-4 illustrates the excellent performance and resolution capabilities of the IAA-based algorithms for the two data sets, as well as the mentioned trade-off for the SIAA-algorithms (as discussed in Section 2). It is worth noting in these figures that the IAA algorithm actually allows for an improved performance as compared to the APES algorithm, confirming the results in [1]. Finally, the computational complexity of the proposed fast IAA algorithms is shown in Figure 5, clearly illustrating the achieved complexity gain from using the proposed implementations. As expected, the FSIAA algorithms allow for a noticeable complexity reduction as compared to the FIAA algorithm, in particular when using the proposed FSIAA-II technique.

5. REFERENCES