EFFICIENT IMPLEMENTATION OF THE TIME-RECURSIVE CAPON AND APES SPECTRAL ESTIMATORS

*Stephen R. Alty**, *Andreas Jakobsson*† *and Erik G. Larsson***

*Dept. of Electronic Engineering, King's College London, London WC2R 2LS, UK
Phone: +44 20 7848 2108, email: *steve.alty@kcl.ac.uk*
†Dept. of Electrical Engineering, Karlstad University, SE-651 88 Karlstad, Sweden
Phone: +46 54 700 2330, email: *andreas.jakobsson@ieee.org*
**Dept. of Elec. & Comp. Engineering, George Washington University, Washington DC 20052, USA
Phone: +1 202 994 6083, email: *egl@gwu.edu*

ABSTRACT

A method for the computationally efficient sliding window time-updating of the Capon and APES spectral estimators based on the time-variant displacement structure of the data covariance matrix is presented. The proposed algorithm forms a natural extension of the computationally most efficient algorithm to date, and offers a significant computational gain as compared to the computational complexity associated with the batch re-evaluation of the spectral estimates for each time-update.

1. INTRODUCTION

Spectral estimation finds applications in a wide range of fields, and has received a vast amount of interest in the literature during the last century. Due to their inherent robustness to model assumptions, there has lately been a renewed interest in *non-parametric* spectral estimators. Among the non-parametric approaches, the *data-dependent* filterbank spectral estimators have many promising properties, allowing for very accurate, computationally efficient, high-resolution estimates (see, [1] and the references therein). Both the recent APES estimator [2] and the *amplitude spectrum Capon* estimator, (the estimator obtained when using the classical Capon filter [3, 4]), to estimate a sinusoidal component at the centre frequency of the bandpass filter, can be seen as *matched* filterbank methods [5]. Given the excellent performance of these estimators, several authors have worked on finding efficient implementations (see [1] for references); the most efficient implementation to date was been presented in [6]. This implementation is based on the evaluation of the inverse Cholesky factors of the covariance matrix estimate using its inherent displacement structure [7]. Given their low displacement rank, these Cholesky factors can be obtained efficiently using the generalized Schur recursion. In this work, we present a novel computationally efficient approach to time-updating the efficient estimator in [6] using a sliding window update of the measured data. The presented update is based on the *time-variant* displacement structure, allowing for the time-updating of the inverse Cholesky factors of the (forward-backward averaged) covariance matrix estimate using the numerically robust time-variant generalized Schur algorithm presented in [8]. The resulting time-updated spectral estimates offers a significant computational gain as compared with the previously required recalculation of the Cholesky factors of the covariance matrix.

2. THE MATCHED FILTERBANK SPECTRAL ESTIMATORS

The matched filterbank spectral estimators [2, 5] are constructed from a set of data-adaptive, frequency dependent, L-tap FIR filters, $\mathbf{h}_\omega$, such that

$$\mathbf{h}_\omega = \arg\min_{\mathbf{h}_\omega} \mathbf{h}_\omega^* \mathbf{Q}_\omega \mathbf{h}_\omega \quad \text{subject to} \quad \mathbf{h}_\omega^* \mathbf{a}_\omega = 1 \tag{1}$$

where $\mathbf{Q}_\omega$ is the $L \times L$ covariance matrix of the signal consisting of all frequencies except ω, $(\cdot)^*$ denotes the conjugate transpose, and $\mathbf{a}_\omega$ is a L-tap Fourier vector, that is,

$$\mathbf{a}_\omega = \begin{bmatrix} 1 & e^{i\omega} & \dots & e^{i\omega(L-1)} \end{bmatrix}^T \tag{2}$$

The classical Capon filter is obtained by minimizing (1) using the covariance matrix of the measured data as an estimate of $\mathbf{Q}_\omega$, where

$$\mathbf{Q}_\omega^{Capon} = \mathbf{R}_x \triangleq E\{\mathbf{x}_t \mathbf{x}_t^*\} \tag{3}$$

and

$$\mathbf{x}_t = \begin{bmatrix} x(t) & x(t+1) & \dots & x(t+L-1) \end{bmatrix}^T \tag{4}$$

Similarly, the APES filter is obtained by minimizing (1) using

$$\mathbf{Q}_\omega^{APES} = \mathbf{R}_x - \mathbf{X}_\omega \mathbf{X}_\omega^* \tag{5}$$

where $(\cdot)^*$ denotes the conjugate transpose, and

$$\mathbf{X}_\omega = \frac{1}{M} \sum_{t=1}^{M} \mathbf{x}_t e^{-i\omega t} \tag{6}$$

Here, $M = N - L + 1$. We remark that the choice of L is a compromise between resolution and statistical stability. The larger L, the better the resolution but the worse the statistical stability. Further, large values of L increase the dimension of $\mathbf{R}_x$ and thus the computational burden of evaluating the spectral estimate. The corresponding (amplitude) spectral estimate is obtained according to [2, 5] as

$$\hat{\phi}_x(\omega) = \mathbf{h}_\omega^* \mathbf{X}_\omega = \frac{\mathbf{a}_\omega^* \mathbf{Q}_\omega^{-1} \mathbf{X}_\omega}{\mathbf{a}_\omega^* \mathbf{Q}_\omega^{-1} \mathbf{a}_\omega} \tag{7}$$

We note that using the matrix inversion formula for the APES estimate, one may write (7) for both the Capon and APES estimates using a number of matrix-vector multiplications and

Fourier transforms of the inverse Choleksy factor of $\mathbf{R}_x$ (see also [1, 5]). This fact is exploited in the efficient implementation of (7) presented in [6], which is based on the inherent displacement structure of $\mathbf{R}_x$ to efficiently evaluate the inverse Cholesky factors of $\mathbf{R}_x$ using the generalized Schur algorithm (see, e.g., [7]); together with efficient matrix-vector products and the Fast Fourier transform (FFT) this forms the efficient implementation. As $\mathbf{R}_x$ is typically unknown, it is normally replaced by the forward-backward averaged *outer-product* estimate (see [9] for a more detailed discussion on the benefits of this estimator as compared to the forward-only estimator)

$$\hat{\mathbf{R}}_x^{fb} = \frac{1}{2}\left(\hat{\mathbf{R}}_x + \mathbf{J}\hat{\mathbf{R}}_x^T\mathbf{J}\right), \tag{8}$$

where $\mathbf{J}$ is the $L \times L$ exchange (or reversal) matrix formed as

$$\mathbf{J} = \begin{bmatrix} \mathbf{0} & & 1 \\ & \iddots & \\ 1 & & \mathbf{0} \end{bmatrix} \tag{9}$$

and

$$\hat{\mathbf{R}}_x = \frac{1}{M}\sum_{t=1}^{M} \mathbf{x}_t\mathbf{x}_t^* \tag{10}$$

In this work, we consider the problem of time-updating $\hat{\phi}_x(\omega)$ as additional data samples become available, by exploiting the time-variant displacement structure of $\hat{\mathbf{R}}_x^{fb}$ to efficiently form a time-update of the *inverse* Cholesky factors. As a step in the calculation of the update, it is necessary to update the Cholesky (i.e., non-inverse) factor itself.

3. TIME-UPDATING THE CHOLESKY FACTORS

Numerous signal processing problems form matrices exhibiting a significant degree of structure. This structure can be exploited to reduce the computational burden as well as the memory requirements for operations on such matrices. Herein, we focus on the displacement structure of the sample covariance matrix to find an efficient time-updating algorithm. A time-variant Toeplitz-like $n \times n$ matrix $\mathbf{R}(t)$ is said to have a time-variant displacement structure [7, 8] if the matrix difference $\nabla\mathbf{R}(t)$ defined by

$$\nabla\mathbf{R}(t) = \mathbf{R}(t) - \mathbf{F}(t)\mathbf{R}(t-\Delta)\mathbf{F}^*(t) \tag{11}$$

has low *rank*, say $r(t)$, where $r(t) \ll n$, for some lower triangular matrix $\mathbf{F}(t)$. The time-variant displacement rank, $r(t)$, provides a measure of the extent of the structure present, with lower rank indicating a stronger degree of structure. Thus, if $r(t)$ approaches n, there is little point in pursuing the displacement technique. Note that the sliding window time-updating of the estimated forward-backward covariance matrix estimate can be expressed, without loss of generality, as

$$\hat{\mathbf{R}}_x^{fb}(t) = \mathbf{F}(t)\hat{\mathbf{R}}_x^{fb}(t-1)\mathbf{F}^*(t) + \mathbf{G}(t)\mathbf{J}(t)\mathbf{G}^*(t) \tag{12}$$

allowing for a time-variant displacement structure with $\Delta = 1$ and $\mathbf{F}(t) = \mathbf{I}$. Combining (11) and (12) yields

$$\nabla\hat{\mathbf{R}}_x^{fb}(t) = \mathbf{G}(t)\mathbf{J}(t)\mathbf{G}^*(t) \tag{13}$$

where $\mathbf{G}(t)$ is an $n \times r(t)$ so-called *generator* matrix and $\mathbf{J}(t)$ is an $r(t) \times r(t)$ full rank *signature* matrix with either ± 1's along its diagonal. Here[1],

$$\mathbf{G}(t) = \begin{bmatrix} x^*_{(N+i)} & x_{(N-p+1+i)} & x^*_{(p-1+i)} & x_{(i)} \\ x^*_{(N-1+i)} & x_{(N-p+2+i)} & x^*_{(p-2+i)} & x_{(1+i)} \\ x^*_{(N-2+i)} & x_{(N-p+3+i)} & x^*_{(p-3+i)} & x_{(2+i)} \\ \vdots & \vdots & \vdots & \vdots \\ x^*_{(N-p+1+i)} & x_{(N+i)} & x^*_{(i)} & x_{(p-1+i)} \end{bmatrix}$$

where N is the sample frame length, p is the filter order and i is the integer time-update index (starting at $i = 1$) representing the shift of one sample in the frame. Further,

$$\mathbf{J}(t) = \begin{bmatrix} +1 & 0 & 0 & 0 \\ 0 & +1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \tag{14}$$

From (14), it can be seen that, $r(t) = 4$, for the forward-backward covariance matrix estimate. This value of $r(t)$ can often be significantly less than typical values of n, which depending on the application can easily be very large (i.e. > 1000). Also, note that the positive-definite nature of $\hat{\mathbf{R}}_x^{fb}(t)$ guarantees the existence of a unique (lower triangular) Cholesky factor, $\mathbf{L}(t)$, such that

$$\hat{\mathbf{R}}_x^{fb}(t) = \mathbf{L}(t)\mathbf{L}^*(t), \tag{15}$$

which by expanding (12), can be expressed as [8]

$$\begin{aligned} \begin{bmatrix} \mathbf{L}(t) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{L}^*(t) \\ \mathbf{0} \end{bmatrix} &= \begin{bmatrix} \mathbf{F}(t)\mathbf{L}(t-1) & \mathbf{G}(t) \end{bmatrix} \\ &\times \begin{bmatrix} \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{J}(t) \end{bmatrix} \begin{bmatrix} \mathbf{L}^*(t-1)\mathbf{F}^*(t) \\ \mathbf{G}^*(t) \end{bmatrix}. \end{aligned} \tag{16}$$

Hence, it follows that there exists an $[\mathbf{I}_n \oplus \mathbf{J}(t)]$-unitary *rotation* matrix, $\mathbf{\Gamma}(t)$, such that

$$\begin{bmatrix} \mathbf{L}(t) & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{F}(t)\mathbf{L}(t-1) & \mathbf{G}(t) \end{bmatrix}\mathbf{\Gamma}(t) \tag{17}$$

The matrix, $\mathbf{\Gamma}(t)$, has the effect of rotating the generator matrix onto the expression $\mathbf{F}(t)\mathbf{L}(t-1)$ to produce the updated Cholesky factor $\mathbf{L}(t)$ and a block zero entry in the left-hand side of (17). The rotational transform, $\mathbf{\Gamma}(t)$, is typically implemented as a sequence of elementary transforms, such that $\mathbf{\Gamma}(t) = \mathbf{\Gamma}_0(t)\mathbf{\Gamma}_1(t), \ldots, \mathbf{\Gamma}_n(t)$, where $\mathbf{\Gamma}_k(t)$ annihilates the k^{th} row of the generator matrix, for example[2],

$$\begin{aligned} &\begin{bmatrix} l & & & & g & g \\ l & l & & & g & g \\ l & l & l & & g & g \\ l & l & l & l & g & g \end{bmatrix} \xrightarrow{\mathbf{\Gamma}_0(t)} \\ &\begin{bmatrix} l' & & & & 0 & 0 \\ l' & l & & & g' & g' \\ l' & l & l & & g' & g' \\ l' & l & l & l & g' & g' \end{bmatrix} \xrightarrow{\mathbf{\Gamma}_1(t)} \cdots \\ &\cdots \xrightarrow{\mathbf{\Gamma}_n(t)} \begin{bmatrix} l' & & & & 0 & 0 \\ l' & l' & & & 0 & 0 \\ l' & l' & l' & & 0 & 0 \\ l' & l' & l' & l' & 0 & 0 \end{bmatrix} \end{aligned} \tag{18}$$

[1] The time indices are depicted as subscripts for space considerations.

[2] In this example, (18) shows a rank-2 generator matrix, with x' indicating time-updated elements of an given matrix.

The sequence of rotations in (18) updates one column of the Cholesky factor at a time, leaving the lower columns unchanged. Further, note that the remaining rows of the generator matrix are also updated, this enables the correct updating of the next column of the Cholesky factor in turn. This procedure continues until all the n columns of $\mathbf{L}(t-1)$ have been updated to $\mathbf{L}(t)$ and the entire generator matrix, $\mathbf{G}(t)$ has been completely nullified. In this way, the updated columns of the Cholesky factor are evolved in an efficient recursive manner. Such a recursion is also beneficial for efficient use of memory allocation and numerical contraction during matrix-vector products.

The rotation matrix $\mathbf{\Gamma}(t)$ can be formed in numerous different ways. Here, a combination of the *Householder* (or Hyperbolic) and *Givens* (or Circular) rotations are used. Both of these transforms have the general form

$$\begin{bmatrix} a & b \end{bmatrix} \mathbf{\Theta} = \begin{bmatrix} \alpha & 0 \end{bmatrix} \tag{19}$$

where

$$\alpha_H = \sqrt{|a|^2 - |b|^2} \tag{20}$$

$$\alpha_G = \sqrt{|a|^2 + |b|^2} \tag{21}$$

The corresponding rotation matrices for a Householder and a Givens rotation, respectively, are given as

$$\mathbf{\Theta}_H = \frac{1}{\sqrt{|a|^2 - |b|^2}} \begin{bmatrix} a & -b \\ -b^* & a \end{bmatrix} \tag{22}$$

$$\mathbf{\Theta}_G = \frac{1}{\sqrt{|a|^2 + |b|^2}} \begin{bmatrix} a & b \\ b^* & -a \end{bmatrix} \tag{23}$$

The Givens rotation is used for "*updating*" the factor with new samples and the Hyperbolic rotation has the effect of "*downdating*" the factor by removing those samples which are no longer present in the time-updated sample frame. In this way, an appropriate combination of rotations can be determined to correctly time-update each Cholesky factor column vector in turn. It should be noted that, in practice, each column of the Cholesky factor is concatenated with the generator matrix to make an $n \times \{r(t)+1\}$ matrix. Then, as each vector is updated, this process is repeated for $n = n-1$ as each row of $\mathbf{G}(t)$ is annihilated until all the column vectors of the new Cholesky factor are produced. Thus, an appropriate rotation matrix, $\mathbf{\Gamma}(t)$, is an $\{r(t)+1\} \times \{r(t)+1\}$ matrix of the form

$$\mathbf{\Gamma}_k(t) = \begin{bmatrix} \frac{L_0}{\delta} & \frac{G_1}{\alpha} & \frac{L_0.G_2}{\alpha.\beta} & -\frac{L_0.G_3}{\beta.\gamma} & -\frac{L_0.G_4}{\gamma.\delta} \\ \frac{G_1^*}{\delta} & -\frac{L_0}{\alpha} & \frac{G_1^*.G_2}{\alpha.\beta} & -\frac{G_1^*.G_3}{\beta.\gamma} & -\frac{G_1^*.G_4}{\gamma.\delta} \\ \frac{G_2^*}{\delta} & 0 & -\frac{\alpha}{\beta} & -\frac{G_2^*.G_3}{\beta.\gamma} & -\frac{G_2^*.G_4}{\gamma.\delta} \\ -\frac{G_3^*}{\delta} & 0 & 0 & \frac{\beta}{\gamma} & \frac{G_3^*.G_4}{\gamma.\delta} \\ -\frac{G_4^*}{\delta} & 0 & 0 & 0 & \frac{\gamma}{\delta} \end{bmatrix} \tag{24}$$

where

$$\alpha = \sqrt{|L_0|^2 + |G_1|^2}, \tag{25}$$

$$\beta = \sqrt{|\alpha|^2 + |G_2|^2} \tag{26}$$

$$\gamma = \sqrt{|\beta|^2 - |G_3|^2}, \tag{27}$$

$$\delta = \sqrt{|\gamma|^2 - |G_4|^2} \tag{28}$$

Whilst the time-updating of the Cholesky factor is interesting in itself, for instance for solving sets of linear equations by simple back-substitution or Gaussian elimination, it can not be used to efficiently find the time-update of the filterbank spectral estimate. However, it is possible to extend the above procedure to also yield the inverse Cholesky factor by augmenting (18) as follows [8]

$$\begin{bmatrix} l & & & & g & g \\ l & l & & & g & g \\ l & l & l & & g & g \\ l & l & l & l & g & g \\ l^{-*} & l^{-*} & l^{-*} & l^{-*} & 0 & 0 \\ 0 & l^{-*} & l^{-*} & l^{-*} & 0 & 0 \\ 0 & 0 & l^{-*} & l^{-*} & 0 & 0 \\ 0 & 0 & 0 & l^{-*} & 0 & 0 \end{bmatrix} \xrightarrow{\mathbf{\Gamma}_0(t)}$$

$$\begin{bmatrix} l' & & & & 0 & 0 \\ l' & l & & & g' & g' \\ l' & l & l & & g' & g' \\ l' & l & l & l & g' & g' \\ l'^{-*} & l^{-*} & l^{-*} & l^{-*} & w & w \\ 0 & l^{-*} & l^{-*} & l^{-*} & 0 & 0 \\ 0 & 0 & l^{-*} & l^{-*} & 0 & 0 \\ 0 & 0 & 0 & l^{-*} & 0 & 0 \end{bmatrix} \xrightarrow{\mathbf{\Gamma}_1(t)} \cdots$$

$$\cdots \xrightarrow{\mathbf{\Gamma}_n(t)} \begin{bmatrix} l' & & & & 0 & 0 \\ l' & l' & & & 0 & 0 \\ l' & l' & l' & & 0 & 0 \\ l' & l' & l' & l' & 0 & 0 \\ l'^{-*} & l'^{-*} & l'^{-*} & l'^{-*} & w''' & w''' \\ 0 & l'^{-*} & l'^{-*} & l'^{-*} & w'' & w'' \\ 0 & 0 & l'^{-*} & l'^{-*} & w' & w' \\ 0 & 0 & 0 & l'^{-*} & w & w \end{bmatrix} \tag{29}$$

Here, the upper triangular (or transpose conjugate) inverse Cholesky factor, $\mathbf{L}^{-*}(t-1)$, has been appended below the matrix in (18). Further, an $n \times r(t)$ matrix of zeros is also appended to produce the above $2n \times \{n + r(t)\}$ matrix. By applying the same combination of Householder and Givens rotations as in 24, the time-update of the inverse Cholesky factor can be achieved efficiently, yielding one column vector per iteration. The rotations onto the null matrix produces a kind of inverse generator matrix which is required to correctly update each inverse Cholesky factor column vector in turn.

4. NUMERICAL EXAMPLE

As an illustration of the superior quality of the filterbank spectral estimators, the amplitude spectral Capon estimate is shown in Figure 1 for a time-varying signal consisting of 3 complex-valued sinusoids. Here, the estimates are computed for 256 frequency points using $N = 64$ samples for each time step, using a $L = 16$-tap adaptive filter. As a comparison, Figure 2 shows the (unwindowed) Spectrogram spectral estimate for the same data set, clearly illustrating the superior resolution of the Capon estimator. Further, Figure 3 illustrates the computational gain of the proposed time-updating of the Capon and the APES spectral estimators for varying filter lengths as compared to the brute-force re-evaluation of the spectral estimates for each time-update using the efficient implementation in [6].

The results shown in Figure 3, were obtained from a Matlab® implementation comparing the time taken to com-

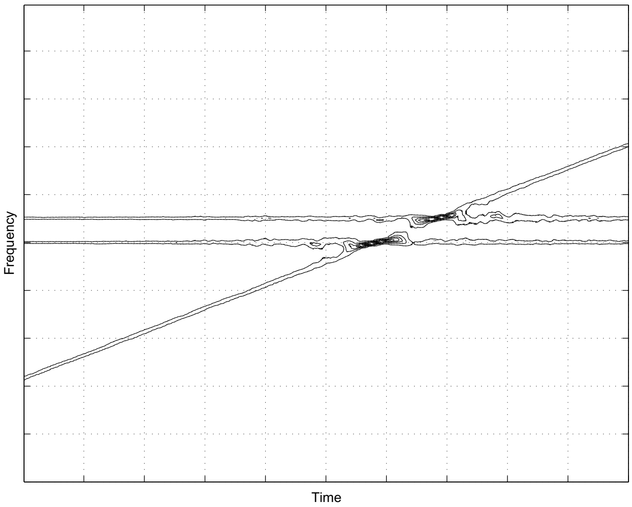

Figure 1: Capon estimate of time-varying signal.

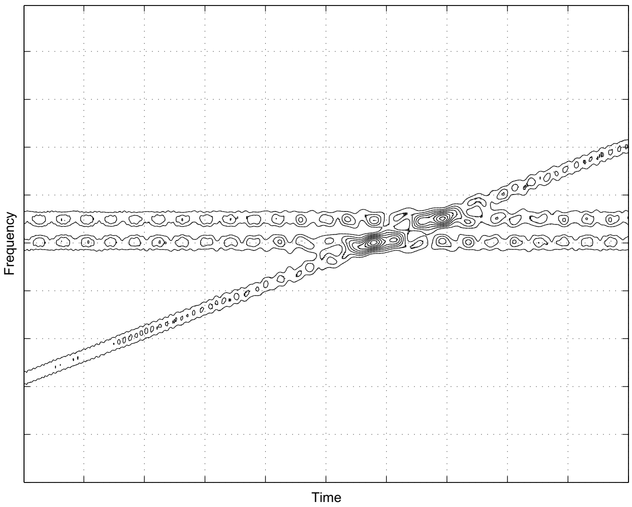

Figure 2: Spectrogram estimate of time-varying signal.

pute the spectral estimates of a given complex data sequence for varying filter lengths. Measurements were taken after each algorithm had been looped 100 times to minimise any error margins associated with spurious processor loading from the operating system (though this effect is still somewhat evident). The graph shows that significant computational gains of more than 2 times can be achieved with filter lengths > 800 for the Amplitude Spectrum Capon (ASC) estimator and this increases exponentially for greater L.

5. CONCLUSIONS

In the presented paper, we have proposed a time-updating implementation of the Capon and the APES spectral estimators based on the updating of the inverse Cholesky factors of the forward-backward averaged sample covariance matrix. This updating can be found via the numerically robust time-variant generalized Schur algorithm, providing a natural extension of the currently most efficient batch implementation of the estimators. Numerical simulations indicate a significant computational gain over the batch estimation methods

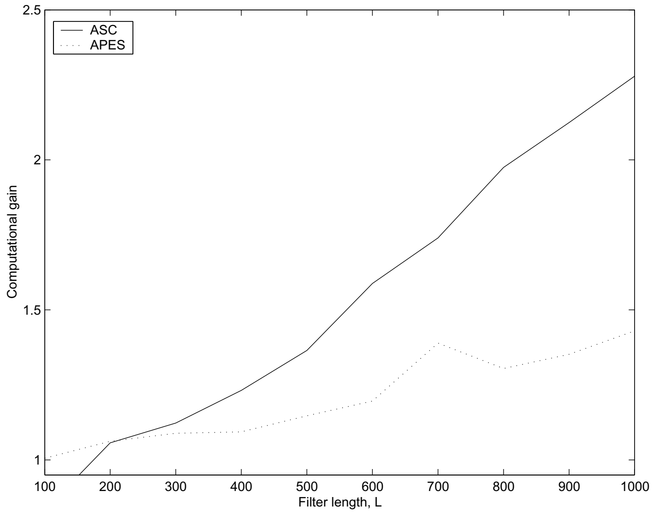

Figure 3: Computational gain of the proposed time-updated ASC and APES spectral estimators as compared to the brute-force re-evaluation using [6].

for larger filter lengths. Furthermore, initial studies of the error propagation show the proposed method to be preferable to a sliding window update of the covariance matrix estimate.

REFERENCES

[1] E. G. Larsson, J. Li, and P. Stoica, *"High-Resolution Nonparametric Spectral Analysis: Theory and Application"*. In *High-resolution and robust signal processing*, Y. Hua, A. B. Gershman and Q. Cheng, Eds., New York, N.Y., USA: Marcel-Dekker, 2003.

[2] J. Li and P. Stoica, "Adaptive Filtering Approach to Spectral Estimation and SAR Imaging," *IEEE Trans. Signal Processing*, vol. 44, pp. 1469–1484, June 1996.

[3] J. Capon, "High Resolution Frequency Wave Number Spectrum Analysis", *Proceedings of the IEEE*, vol. 57, pp. 1408–1418, 1969.

[4] J. Capon, *"Maximum-Likelihood Spectral Estimation"*. Chapter 5 in *Nonlinear Methods of Spectral Analysis*, 2nd ed., S. Haykin,ed., New York, USA: Springer-Verlag, 1983.

[5] P. Stoica, A. Jakobsson, and J. Li, "Matched-Filterbank Interpretation of Some Spectral Estimators," *Signal Processing*, vol. 66, pp. 45–59, April 1998.

[6] E. G. Larsson and P. Stoica, "Fast implementation of two-dimensional APES and Capon spectral estimators," *Multidimensional Systems and Signal Processing*, vol. 13, pp. 35–54, Jan. 2002.

[7] T. Kailath and A. H. Sayed, *Fast Reliable Algorithms for Matrices with Structure*. Philadelphia, USA: SIAM, 1999.

[8] A. H. Sayed, H. Lev-Ari, and T. Kailath, "Time-variant Displacement Structure and Triangular Arrays," *IEEE Trans. Sig. Proc.*, vol. 42, pp. 1052–1062, May 1994.

[9] M. Jansson and P. Stoica, "Forward-Only and Forward-Backward Sample Covariances – A Comparative Study," *Signal Processing*, vol. 77, no. 3, pp. 235–245, 1999.