COMPLEX SINUSOID FREQUENCY ESTIMATION AT VERY LOW SNR

T.Ganesan
Motorola India Electronics (Pvt) Ltd.,
Bangalore INDIA 560 001
email: gana@winner.miel.mot.com

Abstract

A frequency estimator for complex sinusoids in white noise is proposed for very low *SNR* scenario. This algorithm possesses the computational simplicity of Discrete Fourier Transform (DFT) and resolution advantages of signal eigenvector methods. Asymptotic expressions are derived to explain the behaviour of the estimator for high and low *SNR*. Simulation results shows that the estimator provides reasonably good estimates even at lower *SNR* as compared to the existing techniques.

1. Introduction

Frequency estimation of sinusoids in white noise has been a classical problem in spectral estimation. Starting from the simple Discrete Fourier Transform (DFT) many methods have been proposed based on parametric and/or non-parametric models [1,2]. For short data records all the parametric methods like AR, MA and ARMA provide better resolution than DFT methods. But at low *SNR* they fail to work. Recently high resolution methods have been proposed for frequency estimation [3,4]. Even though these methods have infinite resolution capability for short data records, at low *SNR* they fail to work.

Fitting model to the derived data rather than to the raw data is shown to improve the performance of the estimators [4,5]. Tufts and Kumaresan used SVD to get the low rank approximation of the data matrix and thereby improved the frequency estimates. Kay's Iterative Filtering Algorithm (IFA) uses iterative filtering on the prediction error. All these techniques reduces the *Threshold-SNR*, but at the cost of increased computation. In this paper, a new algorithm is proposed for estimating the frequencies of complex sinusoids in white noise at lower *SNR* as compared to the existing methods. This algorithm is based on the data modelling of signal eigenvector and uses only the Fast Fourier Transform (FFT). Thus it possesses the computational advantages of DFT and resolution advantages of signal eigenvector methods at low *SNR*.

2. Proposed Algorithm

The aim here is to estimate the frequencies of the multiple complex sinusoids in white noise using L data points of the signal. The assumed signal model is

$$x(t) = \sum_{i=1}^{D} a_i e^{j2\pi\omega_i t + \phi_i} + n(t) \tag{1}$$

where $n(t)$ is complex zero mean, white Gaussian noise of variance $2\sigma_n^2$, a_i is the amplitude of i-th sinusoid and ω_i is the digital frequency of i-th sinusoid. It is also assumed that the number of sinusoids D is known and initial random phases ϕ_i's are independent and uniformly distributed over $[0, 2\pi]$. Dividing the data into M segments of N data points each, the estimated covariance matrix can be written as

$$R'_x = \frac{1}{N} \sum_{i=1}^{M} x_i x_i^H \tag{2}$$

where $x_i = Ab_i + n_i$, A is a Vandermonde matrix, n_i is a vector of noise samples and b_i is a vector as shown below.

$$b_i = \left[a_1 e^{j\phi_1 + 2\pi\omega_1 (i-1)N} \ \dots \ a_D e^{j\phi_D + 2\pi\omega_D (i-1)N} \right]^T \tag{3}$$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ e^{j2\pi\omega_1} & e^{j2\pi\omega_2} & e^{j2\pi\omega_D} \\ \dots & \dots & \dots \\ e^{j2\pi(N-1)\omega_1} & e^{j2\pi(N-1)\omega_2} & e^{j2\pi(N-1)\omega_D} \end{bmatrix} \tag{4}$$

$$n_i = \left[n(N[i-1]) \ \dots \ n(N[i-1]+N-1) \right]^T \tag{5}$$

Here the superscript 'T' represents the Transpose operator. There are several ways in which covariance matrix can be computed from the given number of samples. Eqn. (2) is one of the many ways and is a Maximum Likelihood (ML) estimate. The proposed algorithm is based on the following theorem[6].

Theorem: *For large values of N, given a signal eigenvector of R'_x, there exists only one column of A which is closest to that* *in l_2-norm sense when all the sinusoids are not equipowered.*

Corollary: For large values of *N*, each signal eigenvector becomes approximately equivalent to one of the columns of *A* matrix, which can be used to estimate the frequency of the sinusoids.

Algorithm: Based on the above theorem the proposed algorithm can be described as follows.

Step 1 Compute the estimated covariance matrix R'_x form the *L* data points using eqn.(2). Discussion about the choice of *N* or *M* is deferred till the next section.

Step 2 Compute *D* number of signal eigenvectors of the estimated covariance matrix.

Step 3 Apply Fast Fourier Transform(FFT) to each signal eigenvector. The frequency corresponding to the peak value in the magnitude of FFT gives the estimate of the sinusoid frequency.

Remarks: Some of the salient features of the algorithm are listed below.

- Algorithm is very easy to implement since it uses only the FFT.
- Inaccurate knowledge of *D* is not going to affect the frequency estimates since every sinusoid is isolated before frequency estimation is carried out. If *D* is under/over estimated then fewer/more number of sinusoid frequencies will be estimated than the true number.
- If the eigenvectors are arranged in descending order according to the eigenvalues, the first signal eigenvector identifies the sinusoid with the highest power, the second eigenvector identifies the next higher power sinusoid and so on. And the eigenvalues are proportional to the sinusoid powers.
- Since signal eigenvector is used, there is significant reduction in the *Threshold-SNR* as observed by Totarang[7]. Moreover due to decoupling of the sinusoids (according to the theorem), significant improvement in the Sinusoid to Interference Ratio(*SIR*) is obtained which helps to improve the estimate at very low *SNR*. Here *SIR* is defined as the ratio of the power of the desired sinusoid and power of the rest of the signal in the signal eigenvector.

3. Asymptotic Analysis

For simplicity let us consider two sinusoids case. It can be shown that the asymptotic signal eigenvectors are related to the columns of *A* matrix as shown below.

$$e_1 = c\left[h_1 - \frac{a_2^2 \text{sinc}}{N(a_2^2 - a_1^2)} h_2\right] \qquad (6)$$

$$e_2 = c\left[h_2 - \frac{a_1^2 \text{sinc}^*}{N(a_2^2 - a_1^2)} h_1\right] \qquad (7)$$

where, h_1 and h_2 are the first and second column vectors of *A* matrix, '*' represents the complex conjugation operator and

$$\text{sinc} = \frac{\sin((\omega_1 - \omega_2)\pi N)}{\sin((\omega_1 - \omega_2)\pi)} e^{j(\omega_1 - \omega_2)\pi(N-1)} \qquad (8)$$

Interpretation:

Case 1. Power of the sinusoids are not equal: From the above equations, it can be inferred that even in asymptotic case (*M* is infinite or *SNR* is infinite), e_1 is not equal to h_1, if *N* is not sufficiently large. Hence if *SNR* is increased infinitely for some *N*, the closeness between the signal eigenvectors and the column vectors of *A* does not improve beyond certain *SNR*. This leads to a bias in the estimates. Hence variance does not monotonicallty decrease with *SNR*. But for low *SNR* scenario, small increase in the *SNR* of the sinusoid increases the eigenvalue separation between the two eigenvalues and hence improves the *SNR* of the signal eigenvectors. Therefore to get accurate frequency estimates *N* should be as large as possible for the given *L=M N*. But number of segments *M* should be at least equal to *D*. So one possible solution would be using overlapping segments to estimate the covariance matrix. If *N* is larger and *M* is at least *D*, then Sinusoid to Interference Ratio(*SIR*) improves with *N*.

Case 2. Power of the sinusoids are equal: Let $A = QE_s$, where $E_s = \left[e_1\ e_2\right]$. It is shown in [6] that when the sinusoid powers are equal, the coefficient matrix *Q* relating the signal eigenvectors and column vectors of *A* matrix will become the eigenvectors of a nearly identity matrix. But eigenvectors of a nearly identity matrix are not nearly euclidean basis vectors. Hence it causes a limitation in the closeness of the signal eigenvectors to the column vectors of *A* matrix, irrespective of the data length and sinusoid *SNR*. So when the sinusoid powers are equal there is a limitation in the variance of the estimate that can be achieved in this algorithm even with large *N*. This limit is a function of the difference between the power of the sinusoids. Higher the difference between the sinusoid powers, better will be the performance.

4. Simulation Results and Discussion

To demonstrate the capability of the method at low *SNR*, the test example chosen by Tufts and Kumaresan[4] is taken. The same example is used by Kay in [2] to compare the sinusoid frequency estimators. The frequencies and phases are chosen to be $\omega_1 = 0.50$, $\phi_1 = 0^o$, $\omega_2 = 0.52$, $\phi_2 = 45^o$. The amplitudes are $a_1 = a_2 = 1$ and the noise variance σ_n^2 is chosen to get the desired *SNR* where

$$SNR = 10\log\left(\frac{1}{\sigma_n^2}\right) \quad (9)$$

The data record length is $L = 24$ complex points (Note that Tufts and Kumaresan used 25 points to estimate the frequencies) To compute the *MSE*, the 24 point data is divided into 2 segments. The covariance matrix was computed according to the eqn.(2). Then 2048 point FFT is applied to the first 2 signal eigenvectors (eigenvectors corresponding to large eigenvalues) of the estimated covariance matrix to estimate the frequencies. The frequency corresponding to the peak value in the FFT gives the estimate of the frequency. The *MSE* was calculated over 100 experiments with different random noise at every experiment. Fig. 1 shows the Mean Square Error (*MSE*) of the estimate of the first frequency along with that of IFA and Maximum Likelihood Estimate (MLE) for $\omega = 0.52$. Results for IFA and MLE were extracted from page 437 of [2]. It can be clearly seen that *MSE* of the estimate of the sinusoid frequency 0.52 reaches the CR bound at 0 *dB SNR* which is much below the *Threshold-SNR* of existing methods. But it is seen by simulations that by using the overlapping segments with segment length = 20, *Threshold -SNR* upto -12 *dB* can be achieved. Due to inherent bias in the estimate, performance does not improve with *SNR* as predicted in the analysis. Another peculiar behaviour of the algorithm is that the estimation of the other frequency is not good at very low *SNR* and it improves with *SNR* upto certain level.

Table 1 gives the performance of the proposed algorithm for two different lengths. It is seen that *Threshold-SNR* goes down with length *N*. For example, when $N = 24$, *Threshold-SNR* is -2 *dB* and for $N = 32$, it is -5 *dB* approximately. It is also seen that, estimation does not improve with *SNR* as predicted in the analysis. But increasing *N* helps in achieving lower and lower *Threshold-SNR*. Tables 2 and 3 give the performance of the algorithm for same frequencies and phases, but with different amplitudes for the sinusoids along with CR bounds for $N = 24$ and $N = 32$. Tables 2 and 3 show that the algorithm performs better when the amplitudes are different as seen in the analysis. For this case *SNR* is calculated after taking the power of the sinusoids into consideration.

5. Conclusions

A new algorithm is proposed to estimate complex sinusoid frequencies at low *SNR* when compared to the existing techniques. This algorithm is computationally attractive since it uses only the FFT. The behaviour of the algorithm is similar to that of a simple Fourier Transform(FT) except for the fact that two frequencies separated by a distance less than the Fourier distance can be resolved. In two sinusoid case, one sinusoid is estimated with greater accuracy and other one with less accuracy at moderate *SNRs*. At low and very low SNRs atleast one frequency is estimated with greater accuracy. It is observed that the variance of the estimates of the two estimates are not equal even when the sinusoids are equipowered. Theoretical explanation for such behaviour of the algorithm is to be arrived at.

SNR	$N = 24$		$N = 32$	
	$MSE(\omega_1)$	$MSE(\omega_2)$	$MSE(\omega_1)$	$MSE(\omega_2)$
-10	11.31	15.74	11.54	15.69
-5	11.0	26.39	10.79	33.26
-2	12.03	44.6	11.35	47.25
0	10.83	45.75	12.58	50.87
5	12.64	48.32	19.71	53.47
10	17.25	49.51	30.78	55.67
20	26.81	49.90	31.61	58.47
50	27.73	50.72	31.64	60.38

Table 1: Performance for two different lengths (All units in dB)

References:

[1] S. M. Kay and S. L. Marple, "Spectrum Analysis - A Modern Perspective", *Proc. of IEEE*, vol. 69, pp.1380-1419, Nov. 1981.

[2] S. M. Kay, "*Modern Spectral Estimation - Theory and Applications* ", Prentice Hall, Englewood Cliffs, New Jersey, 1988.

[3] R. O. Schmidt, "Multiple Emitter Location Signal Parameter Estimation", *Proc. of RADC Spectral Estimation Workshop*, pp. 243-256, 1974.

[4] D. W. Tufts and R. Kumaresan, "Estimation of Frequencies of Multiple Sinusoids: Making Linear Prediction to Perform like Maximum Likelihood", *Proc. of IEEE*, vol. 70, pp.975-989, Sep. 1982.

[5] S. M. Kay, "Accurate Frequency Estimation at Low SNR", *IEEE Trans. on ASSP*, vol. 32, pp.540-547, Jun. 1984.

[6] T. Ganesan, "*A New algorithm For Source Localization in Shallow Water*", Masters Thesis, Dept. of Electrical Communication Engg., Indian Institute of Science, Bangalore, India, Dec. 1993.

[7] P. Totarong et al, "Robust High Resolution DOA estimation via Signal Eigenvector Domain", *IEEE Jl. on Oceanic Engg.*, vol.18, no.4, Oct. 1993.

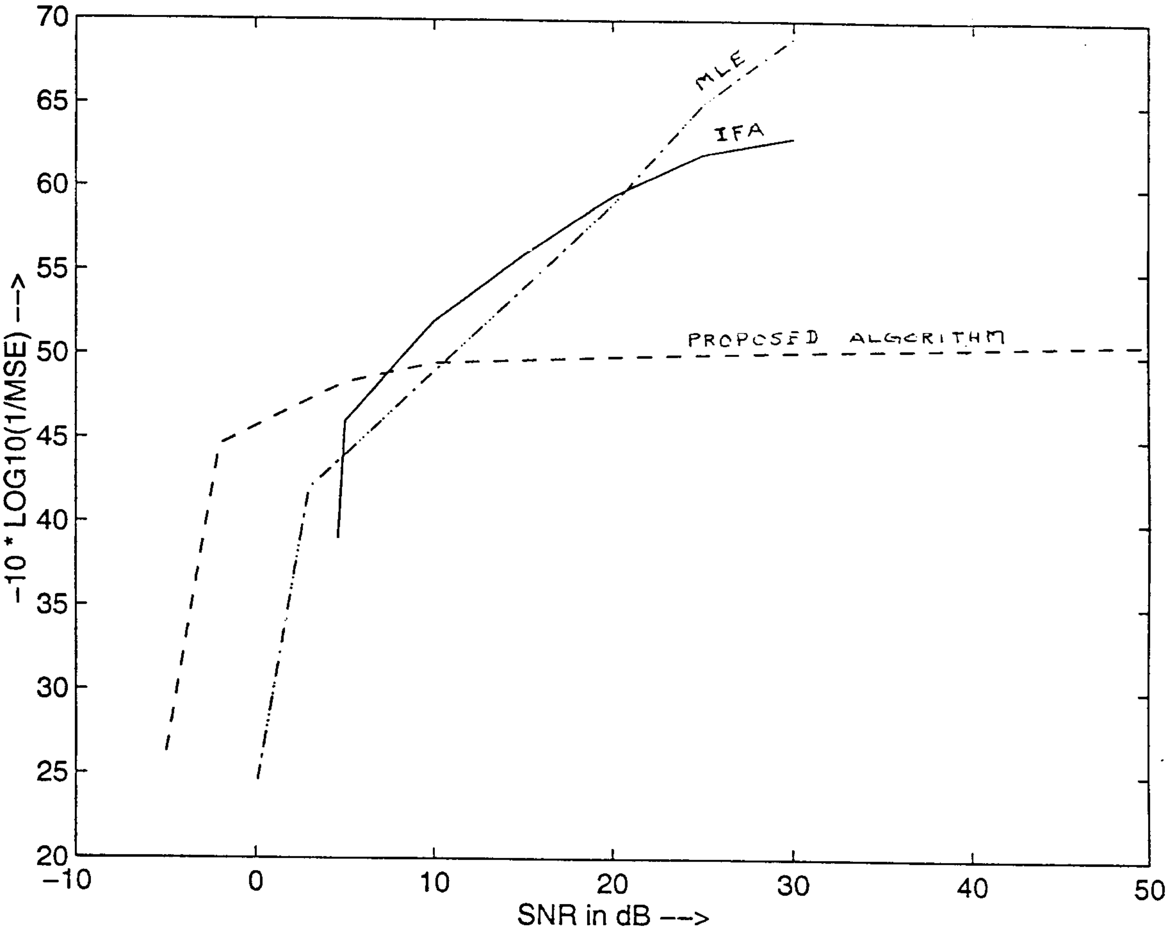

Fig1: Comparison of Proposed Method with IFA and MLE

SNR	CR Bound $N = 24$	$N = 24$		$N = 32$	
		$MSE(\omega_1)$	$MSE(\omega_2)$	$MSE(\omega_1)$	$MSE(\omega_2)$
-10	29.6	11.08	18.81	10.50	24.84
-5	34.6	10.31	45.35	10.60	44.42
-2	37.6	10.12	46.49	11.18	46.86
0	39.6	10.73	49.63	10.27	49.25
5	44.6	10.52	54.50	11.20	55.52
10	49.6	13.85	58.82	12.86	50.04
20	59.6	25.11	62.51	27.42	61.42
50	89.6	27.99	62.05	27.99	62.14

Table 2: Performance of the method for Different amplitudes (All units in dB) a1 = 1, a2 = 2

SNR	CR Bound $N = 32$	$N = 24$		$N = 32$	
		$MSE(\omega_1)$	$MSE(\omega_2)$	$MSE(\omega_1)$	$MSE(\omega_2)$
-10	31.7	15.31	10.24	12.95	12.82
-5	36.7	26.91	10.92	20.61	11.70
-2	38.7	38.35	10.59	36.98	12.20
0	40.7	38.90	10.42	37.40	14.12
5	45.7	38.59	11.96	37.41	22.29
10	50.7	38.86	16.56	38.47	26.95
20	60.7	38.93	24.47	38.53	26.47
50	90.7	38.99	23.82	38.62	26.05

Table 3: Performance of the method for Different amplitudes (All units in dB) a1 = 2, a2 = 1