# A Stable and Efficient Adaptive Notch Filter for Direct Frequency Estimation

Gang Li, *Member, IEEE*

***Abstract*—In this paper, we investigate the problem of direct frequency estimation. A new adaptive algorithm is developed for constrained pole-zero notch filters. The basic characteristic of this algorithm is that the notch filter is cascaded and, hence, purely parametrized by the notching frequencies. This makes the algorithm much simpler and, hence, more efficient and more robust such that the poles and zeros are automatically located on the respective constrained circle, no matter if the algorithm is implemented with infinite or finite precision. The algorithm is analyzed and compared with some existing algorithms. Several numerical examples are given as well as the corresponding simulations.**

## I. Introduction

In MANY applications, one often confronts the problem of estimating frequencies in a desired signal of multiple sinusoids buried in additive noise. There are two classes of processing. The first class, which is called off-line processing, includes the methods such as spectral estimation techniques, which are based on the discrete Fourier transform (DFT) (see, e.g., [1] and [2]) and some multiple signal classification (MUSIC) based algorithms [3], [4]. The frequencies are found by locating the maximal or minimal values of the computed spectrum. These methods usually have a high cost of computation. The second class, which is called on-line processing, is based on adaptive notch filtering techniques. One of the popularly used models is the constrained notch filter (see, e.g., [5]–[7]), which has some superior properties such as better stability, fast convergence, and computation efficiency to unconstrained adaptive filters. Classically, the adaptive notch filter is parametrized with the polynomial coefficients of its transfer function. These coefficients are a function of the notching frequencies for which the notch filter has or nearly has a zero gain. The frequencies are then computed from the estimated transfer function coefficients. These techniques, which are referred as *indirect* frequency estimation methods, require stability monitoring, that is, the model stability has to be checked after each adaptation, which leads to a lot of additional computation. In [8], the lattice structure was used to overcome the stability monitoring problem for the general adaptive IIR filtering problem.

Based on the work reported in [5] and [6], an adaptive notch filter algorithm, which is partially parametrized with the notching frequencies, was proposed by Chen *et al.*[1] in [7], and the frequencies can then be estimated *directly*. They showed that their frequency parametrized adaptive algorithm has several nice properties such as ideal notching performance (due to the zeros are automatically ensured on the constrained circle), faster convergence speed than the classically used transfer function coefficient parametrizations, and no need for model stability monitoring. We observe, however, that this algorithm also has a crucial weakness, that is, the updated frequencies have to be converted into transfer function coefficients at each iteration, and the algorithm is then implemented with these coefficients. This may lead to a series of problems. In fact, besides the requirement of additional computations, *more seriously*, most of its nice properties may be lost when this algorithm is implemented with *finite precision*, which is always the case for real-time applications. As to be shown in the next section, the automatically ensured stability and zeros on the constrained circle may no longer hold when this algorithm is implemented with a digital signal processor (DSP).

In this paper, motivated by the work of Chen *et al.*, we develop a new adaptive algorithm for constrained notch filters with the interest of direct frequency estimation. The basic idea is to avoid implementing the transfer function coefficients and use a *pure* frequency parametrization for the constrained notch filter. This can be done by cascading the filter of order $2n$ into $n$ stages of second-order subfilters. Compared with the Chen *et al.* work reported in [7], our proposed adaptive algorithm has the following advantages:

- It is much simpler and, hence, more efficient,
- It is much more robust, that is, all the nice properties of CYL's algorithm are preserved no matter whether it is implemented with infinite or finite precision.

An outline of this paper is as follows. Section II is devoted to formulating the problem. We introduce CYL's algorithm and demonstrate its faults, which motivates us to develop a new adaptive algorithm for the constrained notch filters. Our main results are found in Section III and IV. The derivation of a new adaptive algorithm is given in Section III. The new algorithm is based on a simple idea of cascading a constrained notch filter of order $2n$ into $n$ stages of second-order filters such that the adaptive algorithm is purely parametrized by the $n$ notching frequencies. The performance of this algorithm is analyzed, discussed, and compared with some existing algorithms in Section IV. The comparison is done especially with CYL's algorithm. Some new sights are also given on the better convergence behavior of the frequency parametrized

Manuscript received February 4, 1996; revised January 15, 1997. The associate editor coordinating the review of this paper and approving it for publication was Dr. Victor E. DeBrunner.

The author is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore (e-mail: egli@ntu.edu.sg).

Publisher Item Identifier S 1053-587X(97)05783-8.

[1] For convenience, this algorithm will be referred as CYL's algorithm in the sequel.

algorithms in this section. To confirm the theoretical analyzes and show the actual performance of the proposed algorithm, numerical examples are presented in Section V, among which there is an application case for speech processing. Concluding remarks are given in Section VI.

## II. Preliminaries and CYL's Algorithm

Let $y(t)$ be a measurable signal, which consists of $n$ sinusoids $s(t)$ with an additive broadband noise $e(t)$

$$y(t)=\sum_{k=1}^{n}A_k\cos(\theta_k^0 t-\phi_k)+e(t)\triangleq s(t)+e(t) \quad (1)$$

where $s(t)$ and $e(t)$ are assumed to be independent, and $\{A_k\neq 0,\theta_k^0,\phi_k\}$ are the amplitude, (angular) frequency, and phase parameter of the sinusoids, respectively.

To estimate the frequencies $\{\theta_k^0\}$ with the *only* available signal $y(t)$, Rao and Kung [5] proposed to use the constrained adaptive notch filter

$$H(z^{-1})=\frac{A(z^{-1})}{A(\alpha z^{-1})} \quad (2)$$

where $0<\alpha<1$, and

$$\begin{aligned}A(z^{-1})&=1+a_1z^{-1}+\cdots+a_nz^{-n}+\cdots+a_{2n}z^{-2n}\\&=\prod_{k=1}^{n}(1-2z^{-1}\cos\theta_k+z^{-2})\triangleq\prod_{k=1}^{n}A_0(\theta_k,z^{-1})\end{aligned} \quad (3)$$

where

$$A_0(\theta_k,z^{-1})=1-2z^{-1}\cos\theta_k+z^{-2}. \quad (4)$$

*Comment 1:* It is easy to see that $H(z^{-1})$ has its zeros on the unit circle $\{e^{\pm j\theta_k}\}$, which means a zero transmission gain at the frequencies $\{\theta_k\}$. The notch filtering performance depends on the choice of the constant $\alpha$, which controls the null bandwidth. In fact, for each pole/zero pair, the corresponding bandwidth of $A_0(\theta_k,z^{-1})/A_0(\theta_k,\alpha z^{-1})$ is approximately given by

$$BW\approx\pi(1-\alpha). \quad (5)$$

Clearly, when $\alpha$ is close to 1, say, $\alpha=0.9950$, the corresponding transfer function $H(z^{-1})$ behaves like an ideal notch filter.

In [6], noting the fact that the notch filter can be parametrized by the coefficient parameter vector $\hat{a}$

$$\hat{a}\triangleq(a_1a_2\cdots a_k\cdots a_n)^T$$

Nehorai derived an adaptive algorithm to update these $n$ parameters. Based on the fact that $H(z^{-1})$ given by (2) and (3) can also be parametrized by the notching frequency vector $\hat{\theta}\triangleq(\theta_1\ \theta_2\cdots\theta_k\cdots\theta_n)^T$ and that in this parametrization, $H(z^{-1})$ has automatically its $2n$ zeros on the unit circle and $2n$ poles, on the circle of radius $\alpha$, Chen *et al.* [7], adapting Nehorai's work, proposed an adaptive algorithm for estimating the frequency parameters directly. This algorithm, that is CYL's algorithm, is summarized as follows:

### A. CYL's Algorithm:

Initialization: $\hat{\theta}(0),\hat{a}(0),P(0),\alpha_0,\alpha_\infty,\alpha(1),\lambda_0,\lambda_\infty,\lambda(1),y(t)=y_F(t)=e_F(t)=\overline{e}(t)=0,t<1$.

Then, we have the main loop for $t=1,2,\cdots$.

- Step 1: Prediction error $\hat{e}(t)$ computation:

$$\hat{e}(t)=y(t)+y(t-2n)-\alpha^{2n}(t)\overline{e}(t-2n)-\Gamma^T(t)\hat{a}(t-1)$$

where $\Gamma(t)\triangleq(\gamma_1(t)\ \gamma_2(t)\ \cdots\ \gamma_k(t)\ \cdots\ \gamma_n(t))^T$ with

$$\gamma_k(t)=\begin{cases}-y(t-k)-y(t-2n+k)+\alpha^k(t)\overline{e}(t-k)\\ \quad+\alpha^{2n-k}(t)\overline{e}(t-2n+k) & k=1,\cdots,n-1\\ -y(t-n)+\alpha^n(t)\overline{e}(t-n) & k=n.\end{cases} \quad (6)$$

- Step 2: Jacobian matrix $J_m(t-1)=\{\partial a_j/\partial\theta_i\}|_{\hat{\theta}=\hat{\theta}(t-1)}$ computation:

$$\frac{\partial a_j}{\partial\theta_i}=\begin{cases}2\dfrac{\partial a_{j-1}}{\partial\theta_i}\cos\theta_i-\dfrac{\partial a_{j-2}}{\partial\theta_i}\\ \quad+2a_{j-1}\sin\theta_i & \forall i,j=2,\cdots,n\\ \dfrac{\partial a_0}{\partial\theta_i}=0,\dfrac{\partial a_1}{\partial\theta_i}=2\sin\theta_i & \forall i.\end{cases} \quad (7)$$

- Step 3: Derivative vector $\Psi_\theta(t-1)$: $\Psi_\theta(t-1)=J_m(t-1)\Psi_a(t-1)$, where $\Psi_a(t-1)=\Gamma(t-1)$ but with $y(\cdot)$ and $\overline{e}(\cdot)$ in (6) replaced by $y_F(\cdot)$ and $e_F(\cdot)$, respectively

$$y_F(t)\triangleq\frac{1}{A_0(\alpha z^{-1})}y(t),\quad e_F(t)\triangleq\frac{1}{A_0(\alpha z^{-1})}\overline{e}(t)$$

in which $A_0(\alpha z^{-1})$ corresponds to $\hat{a}(t-1)$.

- Step 4: Parameter updating:

$$\begin{aligned}K(t)&=\frac{P(t-1)\Psi_\theta(t-1)}{\lambda(t)+\Psi_\theta(t-1)P(t-1)\Psi_\theta(t-1)}\\P(t)&=\lambda^{-1}(t)[P(t-1)-K(t)\Psi_\theta(t-1)P(t-1)]\\\hat{\theta}(t)&=\hat{\theta}(t-1)+K(t)\hat{e}(t).\end{aligned} \quad (8)$$

- Step 5: Conversion of $\hat{a}(t)$ from $\hat{\theta}(t)$: $\hat{\theta}(t)\rightarrow\hat{a}(t)$.
- Step 6: *A posteriori* prediction error $\overline{e}(t)$ with $\hat{a}(t)$

$$\overline{e}(t)=y(t)+y(t-2n)-\alpha^{2n}(t)\overline{e}(t-2n)-\Gamma^T(t)\hat{a}(t).$$

- Step 7: Constant updating:

$$\begin{aligned}\alpha(t+1)&=\alpha_\infty-[\alpha_\infty-\alpha(t)]\alpha_0\\\lambda(t+1)&=\lambda_\infty-[\lambda_\infty-\lambda(t)]\lambda_0.\end{aligned} \quad (9)$$

This is a Gaussian-Newton type recursive prediction error (RPE) based adaptive algorithm. Discussions on the general RPE can be found in [9] and [10], and the performance of this algorithm on adaptive notch filters was analyzed in [6] and [11]. We will go back this issue later on.

The CYL's algorithm has several nice properties (see [7]):

- The zeros of the notch filter are automatically on the unit circle, which guarantees a zero gain of the notch filter at its notching frequencies—this is one of the requirements of an ideal notch filter.
- The stability of the notch filter is ensured at every iteration without monitoring at all.
- A very fast convergence can be achieved since there exist several equivalent global minima for the cost function in $\hat{\theta}$ (while there is only one global minimum for the cost function in $\hat{a}$).

One observes, however, that every iteration of the CYL's algorithm requires the conversion of $\hat{a}(t)$ from the updated frequency vector $\hat{\theta}(t)$ (see Step 5) since the prediction error computation and some other steps need $\hat{a}$, and the computation of the Jacobian matrix $J_m$ and the derivative $\Psi_a$ with respect to $\hat{a}$ in order to get the derivative $\Psi_\theta$ with respect to $\hat{\theta}$ (see Steps 2 and 3).

These may well prevent this algorithm from real-time applications, where the algorithm has to be implemented with a DSP that has a limited memory space and computation capacity. In fact, these steps, on one hand, require a lot of computation, and *more seriously*, on the other hand, the requirement of implementing the transfer function coefficient $\hat{a}$ may cause instability if the adaptive algorithm is implemented with finite wordlength (FWL), which is always the case for real-time applications! To show this, let us look at the following example: Suppose $\hat{\theta} = (0.0500\,0.1000\,0.5000)^T$, and $\alpha = 0.9750$. This represents a stable notch filter with its zeros and poles on the circles of radius 1 and 0.9750. The corresponding $\hat{a}$ is $(-5.7427\ 13.9738\ -18.4622)^T$. Now, if the adaptive algorithm is implemented with a DSP of 16 bits, the actually implemented transfer function coefficient vector is $\hat{a}_{fwl} = (-5.7422\ 13.9736 - 18.4619)^T$, the corresponding zeros of the actually implemented notch filter are $1.1897e^{\pm j0.1786}$, $e^{\pm j0.5092}$, and $0.8406e^{\pm j0.1786}$, and the poles are $1.1599e^{\pm j0.1786}, 0.9750e^{\pm j0.5092}$, and $0.8196e^{\pm j0.1786}$. This clearly shows that *the actually implemented notch filter is not only unstable, but in addition, its notching performance degrades due to the fact that the actual zeros are no longer on the unit circle.* In one word, the nice properties of the CYL's algorithm may no longer hold when this algorithm is implemented with finite precision!

These observations motive us to develop a new adaptive algorithm for notch filters, which are more efficient and robust. That is what we are going to do in the next section.

## III. A New Adaptive Algorithm for Notch Filters

In this section, we develop a new adaptive algorithm for direct frequency estimation. The basic idea is to parametrize the filter into a cascade form such that the transfer function coefficients $\hat{a}$ are no longer required.

Let us consider the following transfer function of a notch filter $H(z^{-1})$

$$H(z^{-1}) = \prod_{k=1}^{n} H_0(\theta_k, z^{-1}) \tag{10}$$

where

$$\begin{aligned} H_0(\theta_k, z^{-1}) &= \frac{1 - 2\beta z^{-1}\cos\theta_k + \beta^2 z^{-2}}{1 - 2\alpha z^{-1}\cos\theta_k + \alpha^2 z^{-2}} \\ &\triangleq \frac{A_0(\theta_k, \beta z^{-1})}{A_0(\theta_k, \alpha z^{-1})} \end{aligned} \tag{11}$$

with $A_0(\cdot,\cdot)$ as defined by (4) and $\alpha < \beta < 1$. $\beta$ is a constant close to 1, say, $\beta = 0.9999$. This notch filter has its poles at $\{\alpha e^{\pm j\theta_k}\}$ and zeros at $\{\beta e^{\pm j\theta_k}\}$, which is slightly different from the one given by (2)–(4), but they have almost the same performance since $\beta$ is very close to 1 (see [6]).

When the only available signal $y(t)$ passes through the filter, the corresponding output is given by $\hat{e}(t) \triangleq H(z^{-1})y(t)$. Denoting

$$x_i(t) \triangleq \prod_{k=1}^{i} H_0(\theta_k, z^{-1})y(t) \tag{12}$$

we have the iterative equations

$$x_i(t) = H_0(\theta_i, z^{-1})x_{i-1}(t) \tag{13}$$

which leads to the *state-space equations*

$$\begin{aligned} x_i(t) &= (2(\alpha-\beta)\cos\theta_i\ \beta^2 - \alpha^2)Z_i(t) + x_{i-1}(t) \\ Z_i(t+1) &= \begin{pmatrix} 2\alpha\cos\theta_i & -\alpha^2 \\ 1 & 0 \end{pmatrix} Z_i(t) + \begin{pmatrix} 1 \\ 0 \end{pmatrix} x_{i-1}(t) \end{aligned} \tag{14}$$

for $i = 1, 2, \cdots, n$ with $x_0(t) = y(t)$, where $Z_i(t)$ is the state vector. Clearly, $\hat{e}(t) = x_n(t)$. Therefore, the prediction error $\hat{e}(t)$ can be computed iteratively.

*Comment 2:* One can see that unlike CYL's algorithm, for the prediction error computation using (14), the frequency vector $\hat{\theta}$ is used directly, and the transfer function coefficient vector $\hat{a}$ is not required. Therefore, the conversion from $\hat{\theta}$ to $\hat{a}$ is no longer needed. More interestingly, (14) *is always stable, and the corresponding zeros are automatically located on the circle of radius $\beta$ during any iteration,* no matter how the relevant parameters are implemented (with or without infinite precision). This is a very desirable property for real-time applications.

We now define a cost function $V(\hat{\theta}, t)$ as

$$V(\hat{\theta}, t) \triangleq \frac{1}{2t}\sum_{j=1}^{t} \hat{e}^2(j). \tag{15}$$

By minimizing this cost function with respect to $\hat{\theta}$, one can obtain the estimate of the frequencies $\{\theta_k^0\}$. The algorithm we are to develop is based on the same RPE, that is, (8), for frequency parameter updating.

To compute $\theta(t)$ with $P(t-1)$ and $\theta(t-1)$, one needs $\hat{e}(t)$ and $\Psi_\theta(t-1)$. $\hat{e}(t)$ can be computed with (14) since $\{Z_i(t), x_i(t)\}$ are available at time $t$. Now, let us compute $\Psi_\theta(t-1) \triangleq -[\partial\hat{e}(t)/\partial\hat{\theta}]$. First of all, we note that

$$\hat{e}(t) = \prod_{k=1}^{n} H_0(\theta_k, z^{-1})y(t).$$

It then follows that

$$\begin{aligned} \frac{\partial\hat{e}(t)}{\partial\theta_i} &= \frac{\partial H_0(\theta_i, z^{-1})}{\partial\theta_i}\prod_{k\neq i} H_0(\theta_k, z^{-1})y(t) \\ &= \frac{\partial H_0(\theta_i, z^{-1})}{\partial\theta_i} H_0^{-1}(\theta_i, z^{-1})\hat{e}(t). \end{aligned} \tag{16}$$

Noting the equation at the bottom of the next page and $\partial A_0(\theta_i, \gamma z^{-1})/\partial\theta_i = 2\gamma z^{-1}\sin\theta_i, \gamma = \alpha, \beta$, one has

$$\begin{aligned} \frac{\partial\hat{e}(t)}{\partial\theta_i} &= 2\left[\frac{\beta z^{-1}}{A_0(\theta_i, \beta z^{-1})}\hat{e}(t) - \frac{\alpha z^{-1}}{A_0(\theta_i, \alpha z^{-1})}\hat{e}(t)\right]\sin\theta_i \\ &\triangleq 2[e_{Fi}(\beta, t) - e_{Fi}(\alpha, t)]\sin\theta_i \end{aligned} \tag{17}$$

where $e_{Fi}(\gamma,t) = [\gamma z^{-1}/A_0(\theta_i,\gamma z^{-1})]\hat{e}(t)$, which can be computed with state-space equations

$$e_{Fi}(\gamma,t) = (\gamma\ 0)W_i(\gamma,t)$$
$$W_i(\gamma,t+1) = \begin{pmatrix} 2\alpha\cos\theta_i & -\alpha^2 \\ 1 & 0 \end{pmatrix} W_i(\gamma,t) + \begin{pmatrix} 1 \\ 0 \end{pmatrix}\hat{e}(t) \quad (18)$$

for $\gamma = \alpha, \beta$, where $W_i(\gamma,t)$ is the corresponding state vector.

Now, we summarize this new algorithm as follows.

### A. Proposed Algorithm

Initialization: $\hat{\theta}(0), P(0), \alpha_0, \alpha_\infty, \alpha(1), \lambda_0, \lambda_\infty, \lambda(1), \beta$, $Z_i(1) = W_i(\gamma,1) = 0$ for $\gamma = \alpha, \beta, \forall i$.

Main loop: for $t = 1, 2, \cdots$

- Step 1: Prediction error $\hat{e}(t)$ computation: for $i = 1 : n$

$$x_i(t) = 2(\alpha(t)-\beta)\cos\theta_i(t-1)\ \beta^2-\alpha^2(t))Z_i(t)+x_{i-1}(t) \quad (19)$$

  with $x_0(t) = y(t)$ and $\hat{e}(t) = x_n(t)$.
- Step 2: Computing the derivative $\Psi_\theta(t-1) = (\psi_1(t-1)\cdots\psi_i(t-1)\cdots\psi_n(t-1))^T$: for $i = 1 : n$

$$e_{Fi}(\beta,t) = (\beta\ 0)W_i(\beta,t), e_{Fi}(\alpha,t) = (\alpha(t)\ 0)W_i(\alpha,t)$$
$$\psi_i(t-1) = -2[e_{Fi}(\beta,t) - e_{Fi}(\alpha,t)]\sin\theta_i(t-1). \quad (20)$$

- Step 3: Parameter updating $\hat{\theta}(t)$ from $\hat{\theta}(t-1)$: the same as those given by (8).
- Step 4: *A posteriori* prediction error $\overline{e}(t)$ computation and state vector updating: for $i = 1 : n$

$$\overline{x}_i(t) = (2(\alpha(t)-\beta)\cos\theta_i(t)\ \beta^2-\alpha^2(t))Z_i(t)+\overline{x}_{i-1}(t)$$
$$Z_i(t+1) = \begin{pmatrix} 2\alpha(t)\cos\theta_i(t) & -\alpha^2(t) \\ 1 & 0 \end{pmatrix} Z_i(t) + \begin{pmatrix} 1 \\ 0 \end{pmatrix}\overline{x}_{i-1}(t) \quad (21)$$

  with $\overline{x}_0(t) = x_0(t) = y(t)$ and $\overline{e}(t) = \overline{x}_n(t)$. For $i = 1 : n$

$$W_i(\gamma,t+1) = \begin{pmatrix} 2\alpha(t)\cos\theta_i(t) & -\alpha^2(t) \\ 1 & 0 \end{pmatrix} W_i(\gamma,t) + \begin{pmatrix} 1 \\ 0 \end{pmatrix}\overline{e}(t) \quad (22)$$

  for $\gamma = \alpha, \beta$.
- Step 5: Constants $\alpha(t)$ and $\beta(t)$ updating: the same as those given by (9).

*Comment 3:*

- The choice of $\alpha$ is a very important issue. Looking at (5), one can see that the null bandwidth of each stage depends on the value of $\alpha$. When $\alpha$ is close to 1, one can achieve an almost ideal notch filter. However, if no *a priori* information is available on the true frequencies, the notches may not fall over the desired frequencies if the bandwidth of each stage is too narrow (that is, when $\alpha$ is too close to 1) such that the algorithm cannot sense the existence of the sine wave signals. Mainly based on this observation, Nehorai [6] proposed using a time-varying $\alpha(t)$ that starts from a small value and increases its value gradually. A simple way to achieve this is to use the exponentially iterative equation given by (9), where $\alpha_0$ is a parameter determining the rate of change in $\alpha(t)$. Normally, $\alpha_0$ is chosen close to 1, and a typical value is $\alpha_0 = 0.9900$. $\alpha_\infty$ is the steady value of $\alpha(t)$ that determines the bandwidth of the converged notch filter. By setting $\alpha(1)$, which is the initial condition, one can achieve a time-varying bandwidth, which helps the algorithm search and then asymptotically converge to the true frequencies. Normally, $\alpha(1) = 0.8000, \alpha_\infty = 0.9950$. For a detailed discussion, we refer to [6].
- The use of $\lambda(t)$ determined by (9) is very classical. The basic idea is to estimate the parameters using the more recent data (that is why it is usually called the forgetting factor). This recursion enables the algorithm to process nonstationary signals. More detailed discussions can be found in, e.g., [6], [10], and [12].
- In our algorithm, the *a posteriori* prediction errors (those variables with *bar*) are used in variable updating. These procedures have been shown to have the potential of improving the convergence speed of the adaptive algorithm (see, [6], [11], and [13]). This is mainly due to the fact that the *a posteriori* prediction errors are usually expected to be a better estimate of the desired values than the *a priori* errors.
- One of the differences between our proposed algorithm from CYL's is that the state-space equation model is used in our algorithm instead of the classical input/output representation. The state-space representation ensures that we use the latest estimates (predictions) to compute the next ones. In fact, looking at (19), the prediction error $\hat{e}(t)$ depends on only the information at $t$ (that is, $\{Z_i(t), x_i(t)\}$), whereas in CYL's, the prediction error depends on $\overline{e}(t-i)$ for $i = 1, 2, \cdots, n$ [see also (6)]. It has been found that this representation can improve the convergence behavior of the algorithm.

In the next section, we will analyze the performance of this proposed algorithm and reveal some nice properties it possesses. The comparison will be done with some existing algorithms.

## IV. Performance Analysis and Comparison

In this section, we analyze and discuss the performance of the new adaptive algorithm and compare it with some existing algorithms in terms of convergence speed, stability, robustness, and computation efficiency.

---

$$\frac{\partial H_0(\theta_i, z^{-1})}{\partial\theta_i} = \frac{\dfrac{\partial A_0(\theta_i,\beta z^{-1})}{\partial\theta_i}A_0(\theta_i,\alpha z^{-1}) - A_0(\theta_i,\beta z^{-1})\dfrac{\partial A_0(\theta_i,\alpha z^{-1})}{\partial\theta_i}}{A_0^2(\theta_i,\alpha z^{-1})}$$

### A. Convergence Analysis

Clearly, our algorithm is of recursive prediction error (RPE). The convergence analysis has set up some specific results for the asymptotic properties of the RPE-type algorithms [9], [10]. Under some assumptions, it was shown that the algorithm converges to a local minimum of the cost function or to the boundary of the model set. These assumptions are generally weak and hold for the constrained notch filters [6]. Therefore, the convergence analysis are applicable in our model. In [6] and [11], it was shown that for sine waves in additive noise, the only possible convergent points of the algorithm are the true parameters. Therefore, our algorithm asymptotically converges to the true frequencies.

The convergence speed is a very important issue. In [6], Nehorai showed that the algorithm based on the minimally parametrized notch filter by $\hat{a}$ converges much faster than on nonminimally parametrized notch filter. Clearly, the notch filter parametrized by $\hat{\theta}$ is also of minimal parametrization. Chen *et al* demonstrated that CYL's algorithm is even faster than Nehorai's algorithm in [6]. This is, as pointed out in [7], due to the fact that the cost function in $\hat{\theta}$ has *several* equivalent global minima such that the frequency vector goes to the nearest one quickly from a given initial condition (while this function in $\hat{a}$ has a unique minimum). Here, we give some new sights on the fast convergence.

Decoupling between parameters is a strongly desired property in adaptive parameter estimation algorithm and a measure of coupling is the off-diagonal elements of the covariance matrix $R_x \triangleq E[\Psi_x(t)\Psi_x^T(t)]$, where $-\Psi_x(t)$ is the corresponding derivative vector of the prediction error with respect to the parametrization $x$ [14], [15]. To achieve a fast convergence, it is desired to have a parametrization for which $R_x$ is diagonal, and its condition number is one (see, e.g., [14]–[16]). In Section V, it will be shown that $R_\theta$ is much better than $R_a$ in terms of condition number and zero off-diagonal elements. Therefore, there exists a weaker coupling between the frequency parameters than the transfer function coefficients. This is believed to be the most important reason for the better convergence behavior of the frequency parametrized adaptive algorithms.

In our algorithm, the same frequency parametrization is used. Therefore, all the nice properties of CYL's algorithm, such as very fast convergence, are preserved. Simulations show that our proposed algorithm has even a better convergence behavior than CYL's. This is believed to be due to the use of the state-space representation in our algorithm (see also Comment 3).

### B. Stability Analysis

We always want to keep the model stable during the adaptation. This is a crucial issue in developing adaptive algorithms because any instability in the model may stop the algorithm from working. Therefore, in the classical adaptive algorithms, model stability monitoring is absolutely necessary [9], [10]. This means that one has to check the model stability after each iteration, which generally costs time and computation. In CYL's and in our proposed algorithm, the model stability is automatically ensured. Therefore, no stability monitoring is needed at all during the adaptation. This is really a very nice property. As pointed out before in Section II, this property holds for CYL's algorithm *if and only if* the algorithm is implemented in infinite precision. This means for a real-time application, where a digital signal processing device of finite precision has to be used, the model stability monitoring may still be required, while for our proposed algorithm, the stability is guaranteed all the time, no matter if it is implemented with infinite or finite precision!

### C. Cramer-Rao Bound

The Cramer-Rao bound (CRB) is a lower bound of the covariance matrix of the parameter estimation error

$$\mathrm{cov}(\hat{\theta}) \triangleq E[(\hat{\theta}-\hat{\theta}^0)(\hat{\theta}-\hat{\theta}^0)^T] \geq J_\theta^{-1} \tag{23}$$

where $J_\theta$ is called the Fisher information matrix.

We note that $y(t)$ given by (1) is an ARMA process. For ARMA processes, it was shown in [10] that the corresponding Fisher matrix is given by

$$J_\theta = \frac{N}{\sigma_e^2} E[\Psi_\theta(t)\Psi_\theta^T(t)] \tag{24}$$

with $\sigma_e^2 \triangleq E[e^2(t)]$, and the corresponding RPE algorithm is expected to approach to the CRB for a sufficiently data length $N$.

Denoting $R = \{r_{kl}\} \triangleq E[\Psi_\theta(t)\Psi_\theta^T(t)]/\sigma_e^2$, one has $r_{ij} = E[(\partial\hat{e}(t)/\partial\theta_i)(\partial\hat{e}(t)/\partial\theta_j)]/\sigma_e^2$. With (16), it can be shown [17] that

$$\begin{aligned} r_{kl} &= \frac{1}{j2\pi}\oint_C \frac{\partial H_0(\theta_k, z^{-1})}{\partial\theta_k} H_0^{-1}(\theta_k, z^{-1}) \\ &\quad \cdot \frac{\partial H_0(\theta_l, z)}{\partial\theta_l} H_0^{-1}(\theta_l, z) z^{-1}\, dz \\ &= \frac{1}{j2\pi}\oint_C T(\theta_k, z^{-1}) T(\theta_l, z) z^{-1} (4\sin\theta_k \sin\theta_l)\, dz \end{aligned} \tag{25}$$

where

$$\begin{aligned} T(\theta_i, z^{-1}) &\triangleq \frac{\partial H_0(\theta_i, z^{-1})}{\partial\theta_i} H_0^{-1}(\theta_i, z^{-1}) \\ &= \frac{(\beta-\alpha)(1-\alpha\beta z^{-1})z^{-1}}{A_0(\theta_i, \beta z^{-1}) A_0(\theta_i, \alpha z^{-1})} \end{aligned} \tag{26}$$

and $C$ denotes the unite circle in a counterclockwise direction for the integration.

The integration (25) can be evaluated using residual theory. Some efficient methods can be found in [17] and [18]. An alternative expression for $J_\theta$ was derived in [7] in terms of $J_a$, which is the corresponding Fisher information matrix, to the transfer function coefficients $\hat{a}$. Due to the involvement of $\hat{a}$, their expression is more complicated than ours.

### D. Computation Efficiency

As pointed out before, the classical adaptive notch filters require model stability monitoring. Therefore, one has to compute the poles of the notch filter at each iteration. This is usually rather computationally costly. For CYL's and our proposed algorithm, no stability monitoring is needed (assuming the implementation is done in infinite precision). Therefore, on one hand, the frequency parametrized algorithms are more

TABLE I
SIMULATION RESULTS IN EXAMPLE I

| $\theta_1 = 0.5000$ | | $\theta_2 = 1.0000$ | | $\theta_3 = 2.0000$ | |
|---|---|---|---|---|---|
| CYL's | GL's | CYL's | GL's | CYL's | GL's |
| 0.5001 | 0.5005 | 1.0002 | 1.0004 | 2.0020 | 2.0022 |
| 0.5005 | 0.4998 | 0.9991 | 1.0002 | 2.0002 | 1.9985 |
| 0.4983 | 0.4992 | 2.4134 | 1.0008 | 2.0001 | 1.9996 |
| 0.5004 | 0.4997 | 0.9992 | 0.9996 | 1.9994 | 1.9987 |
| 0.5012 | 0.5019 | 1.0002 | 1.0000 | 2.0010 | 2.0009 |
| 0.5010 | 0.5020 | 1.0002 | 0.9993 | 1.9994 | 1.9997 |
| 0.4996 | 0.4996 | 1.0008 | 1.0008 | 2.0000 | 2.0001 |
| 0.4987 | 0.4987 | 0.9987 | 0.9986 | 2.0000 | 1.9990 |
| 0.4994 | 0.5000 | 1.0002 | 1.0013 | 2.0013 | 2.0023 |
| 0.5004 | 0.5020 | 1.0010 | 1.0015 | 2.0004 | 2.0013 |
| 0.5002 | 0.4993 | 1.0003 | 0.9998 | 1.9994 | 1.9971 |
| 0.5007 | 0.5005 | 0.9998 | 0.9999 | 2.0004 | 1.9993 |
| 0.5001 | 0.5000 | 1.0009 | 1.0011 | 2.0000 | 2.0009 |
| 0.4991 | 0.4992 | 0.9997 | 0.9994 | 1.9998 | 1.9997 |
| 0.4997 | 0.4996 | 0.9999 | 0.9989 | 2.0004 | 2.0009 |
| 0.4998 | 0.5008 | 0.9998 | 0.9994 | 1.9998 | 2.0000 |
| 0.5003 | 0.5016 | 1.0003 | 0.9997 | 2.0002 | 1.9998 |
| 0.4992 | 0.4988 | 0.9980 | 0.9982 | 2.0006 | 2.0011 |
| 0.5005 | 0.4998 | 1.0010 | 1.0012 | 2.0000 | 1.9997 |
| 0.4991 | 0.4996 | 0.4991 | 0.9992 | 2.0002 | 1.9991 |

efficient. In CYL's algorithm, one has to convert $\hat{\theta}$ into $\hat{a}$ (see Step 5) to compute the Jacobian matrix $J_m$ and, hence, to compute $\Psi_\theta$ (see Steps 2 and 3) at each iteration. This is really a computational burden, which does not exist in our proposed algorithm. Therefore, on the other hand, one can easily conclude that our proposed algorithm is much simpler and more efficient than CYL's.

To confirm these analyzes and demonstrate the performance of our proposed algorithm, in the next section, we will present three examples as well as the corresponding simulation results.

## V. NUMERICAL EXAMPLES AND SIMULATIONS

In this section, we present three numerical examples and the corresponding simulations to examine the performance of our algorithm. Some comments are also given. The first two are basically for examining the performance of our proposed algorithm, and the third example is a studied case for speech signal processing.

*Example I:*

This example was used in [7]. The signal $y(t)$ is given by

$$y(t) = 2\sin(0.5t) + 2\sin(1t) + 2\sin(2t) + e(t)$$

where $e(t)$ is a white noise with zero-mean and unit variance. Clearly, the signal-to-noise ratio (SNR) is 3 dB for each sinusoid. We generate 30 different frames $y(t)$, each of them contains 500 samples. We tested three algorithms: Nehorai's, CYL's, and our proposed one, which is denoted GL's.

In order to have a fair comparison, we set the same initial conditions for the three algorithms. For Nehorai's algorithm, $\hat{a}(0) = (0\,0\,0)^T$, which is equivalent to $\hat{\theta}(0) = (2.6180\ 1.5708\ 0.5236)^T$ for CYL's and GL's. $P(0) = \gamma I$ with $\gamma$ taking one of the three values 0.1, 1, and 10.

In the 30 trials, both Nehorai's and CYL's algorithm converge to their true parameter vector 23 times after the 500 iterations, whereas GL's algorithm converges 27 times. The results of 20 trials are presented in Table I for CYL's and GL's.

It is interesting to note that for the third trial on Table I, both Nehorai's and CYL's algorithms cannot converge to their corresponding true value after 500 iterations, no matter what $\gamma$ takes. For Nehorai's, the algorithm becomes unstable when $\gamma \geq 0.25$. A typical convergence behavior of the three algorithms is shown in Fig. 1, with $\gamma = 0.10$. For comparison, we have converted $\hat{a}(t)$ for Nehorai's algorithm into $\hat{\theta}(t)$ at each iteration, as shown in Fig. 1(a).

As pointed out previously, the convergence behavior of an adaptive algorithm is strongly related to the covariance matrix $R_x = \frac{1}{500}\Sigma_{t=1}^{500}\ \Psi_x(t)\Psi_x^T(t)$, where $-\Psi_x(t)$ is the derivative of prediction error with respect to the parametrization $x$. For Example I, computation shows that with Nehorai's algorithm

$$R_a = \begin{pmatrix} 0.8047 & 0.6274 & 0.2242 \\ 0.6274 & 1.8473 & 1.1671 \\ 0.2242 & 1.1671 & 0.8116 \end{pmatrix} \times 10^4$$

and the corresponding condition number is 189.0925.

Foe CYL's and GL's algorithm

$$R_\theta = \begin{pmatrix} 1.3294 & 0.0016 & 0.0007 \\ 0.0016 & 1.5193 & -0.0005 \\ 0.0007 & -0.0005 & 1.2613 \end{pmatrix} \times 10^4$$

and the corresponding condition number is 1.2046.

Clearly, the two frequency parametrized algorithms have a much better covariance matrix than the one parametrized with the transfer function coefficients. Therefore, a better convergence behavior can be expected.

*Example II:*

Now, let us consider another example

$$y(t) = 2\sin(0.25t) + 2\sin(0.5t) + 2\sin(0.75t) + e(t)$$

which is exactly the same as the previous one except the frequencies. For Nehorai's algorithm

$$R_a = \begin{pmatrix} 0.5553 & 0.8964 & 0.5098 \\ 0.8964 & 1.4655 & 0.8364 \\ 0.5098 & 0.8364 & 0.4780 \end{pmatrix} \times 10^4$$

and the condition number of this matrix is $3.9869 \times 10^4$.

For CYL's and GL's algorithm

$$R_\theta = \begin{pmatrix} 1.3280 & -0.0012 & 0.0002 \\ -0.0012 & 0.0147 & 0.0000 \\ 0.0002 & 0.0000 & 0.0051 \end{pmatrix} \times 10^4$$

and the condition number for this matrix is 260.7926.

The same observations can be made. We note that for this example, the condition number for each algorithm is much bigger than that for the first example. This is due to the fact that the frequencies in the second example are much closer to each other. In both examples, we note that the off-diagonal elements of $R_\theta$ are much smaller. This means that the frequencies are almost independent one to another, and hence, a faster convergence speed can be expected. To show this,

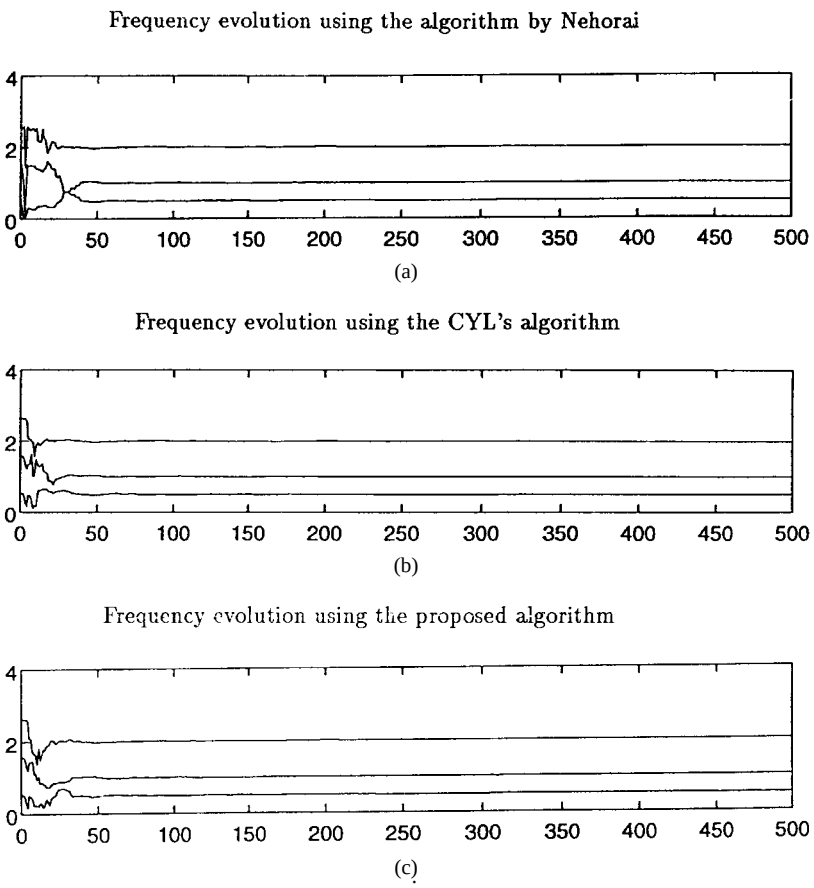


Fig. 1. Frequency evolution comparison of three algorithms for Example I.

we run the three algorithms for the second example with the initial condition $\theta(0) = (0.1500\ 0.3500\ 0.5236)^T$ for CYL's and GL's and the equivalent $\hat{a}(0)$ for Nehorai's. For this example, it is found that Nehorai's algorithm cannot converge to the true values with the 500 data. From Fig. 2(a), one can see that it goes to the true frequencies slowly. Fig. 2(b) and (c) show the evolution of convergence for CYL's and GL's algorithms, respectively, from which one can see that both of them converge quickly to the true frequencies.

*Example III—A Studied Case for Speech Signal Processing*

We now look at an application in speech analysis and synthesis. It has been noted that voiced speech signal can be modeled with (1) for a short interval of 20–30 ms. For some cases, the speech signal shows highly periodicity. The period, which is called *pitch*, is a very important parameter in speech encoder design (see, e.g., [19]). In Fig. 3, the solid line presents a typical voiced speech signal. We then model it using

$$y(t) = \sum_{k=1}^{n} A_k \cos(k\theta_0 t - \phi_k) + e(t) \tag{27}$$

where $\theta_0$ is the fundamental (angular) frequency.

In this case, we have one frequency $\theta_0$ to determine instead of $n$. Our proposed algorithm can be adapted for this situation easily. In fact, with the constraint

$$\begin{aligned}\hat{\theta}(t) &= (\theta_1(t) \quad \theta_2(t) \quad \cdots \quad \theta_k(t) \quad \cdots \quad \theta_n(t))^T \\ &= (1 \quad 2 \quad \cdots \quad k \quad \cdots \quad n)^T \hat{\theta}_0(t)\end{aligned} \tag{28}$$

one has

$$\begin{aligned}\Psi_{\theta_0}(t) \triangleq -\frac{d\hat{e}(t)}{d\theta_0} &= -\sum_{k=1}^{n} \frac{d\hat{e}(t)}{d\theta_k}\frac{d\theta_k}{d\theta_0} \\ &= (1 \quad 2 \quad \cdots \quad k \quad \cdots \quad n)^T \hat{\Psi}_\theta\end{aligned} \tag{29}$$

where $\Psi_\theta$, as defined before, is the derivative of the prediction error with respect to the frequency vector $\hat{\theta}$.

Therefore, one can get the corresponding algorithm to the fundamental frequency estimation just by replacing Step 3 of our proposed algorithm with

$$\begin{aligned}K(t) &= \frac{P(t-1)\Psi_{\theta_0}(t-1)}{\lambda(t) + \Psi_{\theta_0}(t-1)P(t-1)\Psi_{\theta_0}(t-1)} \\ P(t) &= \lambda^{-1}(t)[P(t-1) - K(t)\Psi_{\theta_0}(t-1)P(t-1)] \\ \hat{\theta}_0(t) &= \hat{\theta}_0(t-1) + K(t)\hat{e}(t) \\ \hat{\theta}(t) &= (1 \quad 2 \quad \cdots \quad k \quad \cdots \quad n)^T \hat{\theta}_0(t)\end{aligned} \tag{30}$$

where $K(t)$ and $P(t)$ are *scalar* instead of *matrix* of order $n \times n$.

Taking $n = 12$ in (27), we run the proposed adaptive algorithm for fundamental frequency estimation with $\hat{\theta}_0(0) = 0.05$ and $P(0) = 20$ and obtain $\hat{\theta}_0(200) = 0.2334 \triangleq \hat{\theta}_0$ with which the corresponding synthesized speech signal $\hat{y}(t)$ can be obtained with

$$\hat{y}(t) = \sum_{k=1}^{12} A_k \cos(k\hat{\theta}_0 t - \phi_k) \tag{31}$$

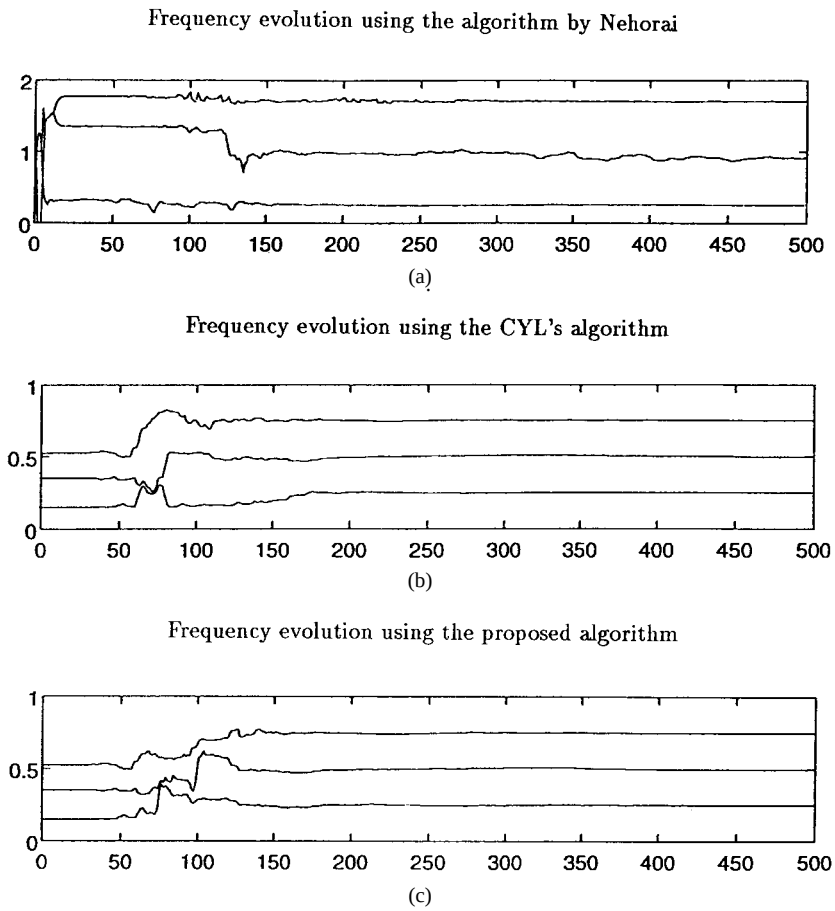


Fig. 2. Frequency evolution comparison of three algorithms for Example II.

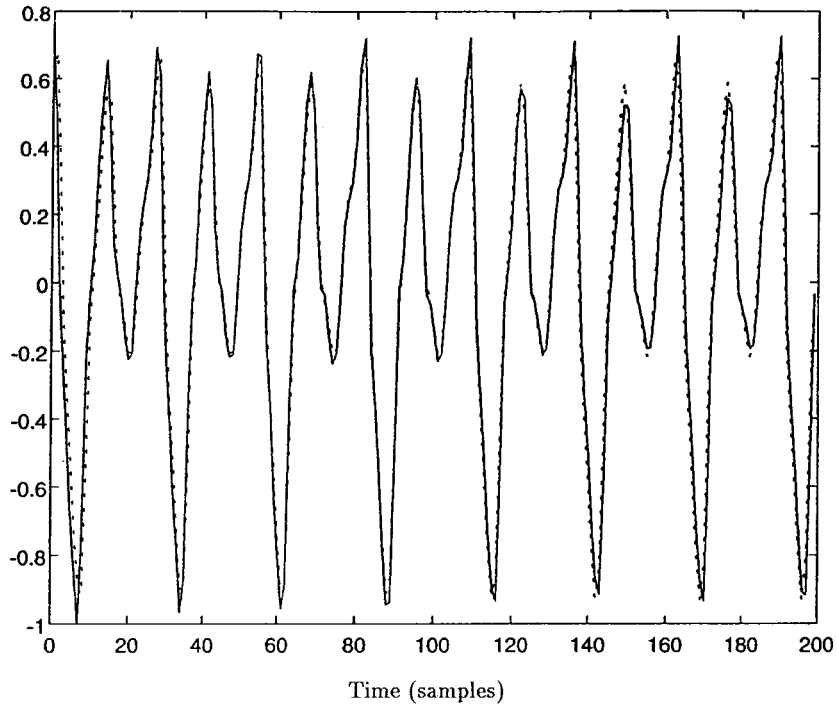


Fig. 3. Speech signal (solid line) and its synthesized version (dotted line).

TABLE II
ESTIMATED PARAMETERS IN (31) FOR EXAMPLE III

| *Harmonics* | $A_k$ | $\phi_k$ |
|---|---|---|
| 1 | 0.2703 | -1.4136 |
| 2 | 0.5384 | 0.0878 |
| 3 | 0.0745 | 1.8662 |
| 4 | 0.0279 | 1.2808 |
| 5 | 0.0257 | 0.3380 |
| 6 | 0.0984 | 1.4808 |
| 7 | 0.0120 | 2.9031 |
| 8 | 0.0077 | 1.5536 |
| 9 | 0.0161 | 1.8886 |
| 10 | 0.0122 | -2.4339 |
| 11 | 0.0008 | -1.1600 |
| 12 | 0.0002 | 0.3624 |

where $\{A_k, \phi_k\}$ can be determined by minimizing $\sigma^2 \triangleq \Sigma_{t=1}^{200}\ [y(t) - \hat{y}(t)]^2$ with respect to these variables. In fact, denoting $A_k^c \triangleq A_k \cos\phi_k$ and $A_k^s \triangleq A_k \sin\phi_k$ for all $k$, one can see that $\sigma^2$ is a quadratic function in $\{A_k^c, A_k^s\}$ with $\hat{\theta}_0$ and $y(t)$ given. Therefore, $\sigma^2$ can be minimized with respect to $\{A_k^c, A_k^s\}$ easily. With the obtained optimal $\{A_k^c, A_k^s\}$, one can then convert them into $\{A_k, \phi_k\}$ steadily. In Table II, one can find the estimated parameters $\{A_k, \phi_k\}$.

The synthesized speech signal $\hat{y}(t)$ is presented in Fig. 3 with the dotted line. As seen, $\hat{y}(t)$ follows $y(t)$ well. This confirms the validity of the model and the fundamental frequency or the pitch obtained.

## VI. CONCLUSIONS

We have studied the problem of direct frequency estimation using an adaptive constrained notch filter. After pointing out some flaws in CYL's algorithm, we have derived a new adaptive algorithm, which is purely parametrized by the notching frequencies of the filter. This characteristic makes the new algorithm not only much simpler and, hence, more computationally efficient but also more robust, compared with the partially frequency parametrized CYL's. Some new sights have also been given to explain the faster convergence speed achieved with the frequency parametrized algorithms. The actual performance of the proposed adaptive algorithm has been demonstrated with three numerical examples, which confirmed our theoretical results. The proposed algorithm has also been modified for direct fundamental frequency estimation, which is used in the third example for pitch estimation of the speech signal.

It has been seen that the performance of an adaptive notch filter is related to the Fisher information matrix. This matrix is determined by the parametrization (or structure) of the filter. Examples I and II show that the frequency parametrized CYL's and GL's adaptive filters yield a better performance in terms of convergence than the polynomial coefficient parametrized Nehorai's adaptive filter. It is well known that for a filter, there are a number of equivalent parametrizations. It is interesting to investigate theoretically the relationship between the parametrizations and a given performance (say, e.g., the convergence speed) of an adaptive notch filter.

## REFERENCES


[1] S. M. Kay, *Modern Spectral Estimation: Theory and Application*. Englewood Cliffs, NJ: Prentice-Hall, 1988.
[2] P. T. Gough, "A fast spectral estimation algorithm based on the FFT," *IEEE Trans. Signal Processing*, vol. 42, pp. 1317–1322, June 1994.
[3] S. M. Kay and A. K. Shaw, "Frequency estimation by principal component AR spectral estimation method without eigendecomposition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 95–101, Jan. 1988.
[4] J. T. Karhunen and J. Joutsensalo, "Sinusoidal frequency estimation by signal subspace approximation," *Signal Process.*, vol. 40, no. 12, pp. 2961–2972, 1992.
[5] D. V. Rao and S. Y. Kung, "Adaptive notch filtering for the retrieval of sinusoids in noise," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 791–802, Aug. 1984.
[6] A. Nehorai, "A minimal parameter adaptive notch filter with constrained poles and zeros," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 983–996, 1985.
[7] B. S. Chen, T. Y. Yang, and B. H. Lin, "Adaptive notch filter by direct frequency estimation," *Signal Process.*, vol. 27, pp. 161–176, 1992.
[8] P. A. Regalia, "Stable and efficient lattice algorithms for adaptive IIR filtering," *IEEE Trans. Signal Processing*, vol. 40, pp. 375–388, 1992.
[9] L. Ljung, "Analysis of a recursive prediction error identification algorithm," *Automatica*, vol. 17, pp. 89–100, Jan. 1981.
[10] L. Ljung and T. Soderstrom, *Theory and Practice of Recursive Identification*. Cambridge, MA: MIT Press, 1983.
[11] P. Stoica and A. Nehorai, "Performance analysis of an adaptive notch filter with constrained poles and zeros," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 911–919, June 1988.
[12] G. C. Goodwin and K. J. Sin, *Adaptive Filtering, Prediction and Control*. Englewood Cliffs, NJ: Prentice-Hall, 1984.
[13] T. S. Ng, "Some aspects of an adaptive digital notch filter with constrained poles and zeros," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 158–161, 1987.
[14] G. C. Goodwin, "Some observations on robust stochastic estimation," in *Proc. 8th IFAC/IFORS Symp. Identification Syst. Parameter Estimation*, Beijing, China, 1988, pp. 22–32.
[15] G. Li and M. Gevers, "Data filtering, representation, and the numerical accuracy of parameter estimator," in *Proc. 31st IEEE Conf. Decision Contr.*, 1992, vol. 4, pp. 3692–3697.
[16] M. Gevers and G. Li, *Parametrizations in Control, Estimation and Filtering Problems: Accuracy Aspects*. London, U.K.: Springer Verlag, 1993.
[17] K. J. Astrom, *Introduction to Stochastic Control Theory*. New York: Academic, 1970.
[18] T. Kailath, A. Viera, and M. Morf, "Inverses of Toeplitz operator, innovations, and orthogonal polynomials," *SIAM Rev.*, vol. 20, pp. 106–110, Jan. 1978.
[19] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*. Englewood Cliffs, NJ: Prentice-Hall, 1978.



**Gang Li** (M'92) received the B.Eng. degree in electrical engineering from Beijing Institute of Technology, Beijing, China, in 1982, and the M.Eng and the Ph.D. degrees, both from Louvain University, Louvain, Belgium, in 1988 and 1990, respectively.

He was with the Control Group at Louvain University as a Post-doctoral Researcher until April 1992. Since May 1992, he has been with Nanyang Technological University, Singapore, as a lecturer. His research interests include digital filter and controller design, numerical problems in estimation and control applications, adaptive signal processing, and speech signal analysis and synthesis. He is a co-author (with M. Gevers) of the book *Parametrizations in Control, Estimation, and Adaptive Filtering Problems: Accuracy Aspects* (London, U.K.: Springer-Verlag, Communications and Controls Engineering Series, 1993).