A Fast Minimal-Symbol Subspace Approach to Blind Identification and Equalization

by B. Sampath, Y. Li and K.J.R. Liu

T.R. 95-98

A Fast Minimal-Symbol Subspace Approach to Blind Identification and Equalization*

Balaji Sampath, Ye Li and K. J. Ray Liu

Electrical Engineering Department and Institute for Systems Research
University of Maryland, College Park, MD 20742
Fax:1-301-405-6707
Email: bsampath(liye, kjrliu)@src.umd.edu

ABSTRACT

A subspace based blind channel identification algorithm using only the fact that the received signal can be oversampled is proposed. No direct use is made of the statistics of the input sequence or even of the fact that the symbols are from a finite set and therefore this algorithm can be used to identify even channels in which arbitrary symbols are sent. A modification of this algorithm which uses the extra information in the more common case when the symbols are from a finite set is also presented. This LS-Subspace algorithm operates directly on the data domain and therefore avoids the problems associated with other algorithms which use the statistical information contained in the received signal. In the noiseless case, it is possible for the proposed Basic Subspace algorithm to identify the channel exactly using the least number of symbols that can possibly be used. Thus, if the length of the impulse response of a channel is JT, T being the symbol interval, then it is possible to use this algorithm to identify the channel using an observation interval of just $(J+3)T$. In the noisy case, simulations have shown that almost exact identification can be obtained by using a few more symbols than the theoretical minimum. This is orders of magnitude better than the other blind algorithms. Moreover, this algorithm is computationally very efficient and has no convergence problems.

*The work was supported in part by the NSF grants MIP9309506 and MIP9457397.

1 Introduction

Most digital communication systems are subject to intersymbol interference (ISI). In many cases this is so severe that the correct reception of the transmitted sequence is hindered unless specific equalization procedures are adopted. Classically, these procedures are based on the knowledge of the channel which is obtained by sending a known training sequence. But, when the channel is varying, even slowly, the training sequence has to be sent periodically so that the channel estimates can be updated and this reduces the effective channel rate.

In contrast to the classical channel identification methods, blind channel identification methods are very attractive since they do not require training sequences. Existing blind identification methods use the statistics of the transmitted sequence instead of the explicit knowledge of the sequence itself. Since communication channels are very likely to be nonminimum phase, most of the existing blind channel identification algorithms have used higher order statistics [10]-[13]. Elegant solutions using this approach have been given but all of them need a very large number of symbols (typically, much more than a few hundred symbols).

Therefore, an algorithm by Tong, Xu, and Kailath [1] which allows the blind identification of channels using only second-order statistics is considered to be a breakthrough. Their algorithm relies on the cyclo-stationarity of communication signals and makes explicit use of the second-order cyclo-stationary statistics of the channel output. Recently a number of algorithms [3]-[9] have been proposed which make use of this idea.

But none of these algorithms have used the signal structure inherent in the oversampled channel output. In our approach we make full use of this structure and therefore find that it is not necessary to explicitly use any statistics. Our proposed Basic Subspace algorithm does not need any information about the transmitted symbols and therefore this method works in a very general set of situations.

Although the Basic Subspace algorithm does not use any property of the symbol set and gives exact results in the noiseless case, in the noisy case a much better algorithm can be obtained by making use of the known properties of the transmitted sequence. We have therefore modified the Basic Subspace algorithm by incorporating least-square techniques. By doing this we not only use the explicit knowledge of the transmitted symbols but also implicitly make use of the statistics of all orders and not just the second or third order statistics.

The received baseband signal $x(\cdot)$ can be written as

$$r(t) = \sum_{k=-\infty}^{\infty} s_k h(t - kT) \tag{1}$$

$$x(t) = r(t) + n(t) \tag{2}$$

where s_k is an information symbol in a signal constellation $\mathcal{S}$ ($\mathcal{S}$ may be an infinite set), $h(\cdot)$ is the discrete-time channel impulse response, T is the symbol interval and $n(\cdot)$ is the additive noise. In the sequel we will assume that the impulse response $h(\cdot)$ has finite support, i.e. $h(t) = 0$ for $t \geq JT$, $J \in \mathcal{N}$.

The channel identification problem requires us to estimate the channel impulse response $h(t)$ (or at least samples of $h(t)$). Classical channel identification procedures using a training sequence have the knowledge of both $x(t)$ as well as the transmitted symbols $\{s_k\}$. A blind channel identification algorithm on the other hand has to estimate the channel response $h(\cdot)$ given only the received signal $x(\cdot)$.

In this paper we derive and study a new blind channel identification algorithm which needs very few symbols[1] to estimate the channel. The organization of the paper is as follows - in Section 2, we derive the Basic Subspace algorithm and this forms the backbone of our approach. In the next section we derive the conditions for the identifiability of the channels using this method. In Section 4, we modify the Basic Subspace algorithm by incorporating least-square techniques and obtain a robust algorithm which works very well in practical situations. Finally we discuss a simulation example and the conclusions that can be drawn from it.

2 The Basic Subspace Algorithm

In this section we develop the Basic Subspace algorithm. In order to simplify the presentation we will ignore the noise for the moment. We will first discuss the case when the impulse response is of length $2T$, i.e. $J = 2$ and then we will present the more general case.

2.1 The Basic Subspace Algorithm for the $J = 2$ Case

We sample the signal at twice the baud rate, i.e. at $t = nT + \delta_1$ and $t = nT + \delta_2$, $0 \leq \delta_1, \delta_2 \leq T$. Using the fact that the impulse response has length $2T$, it can be easily seen that the following

[1]Under certain conditions, just $J + 3$ symbols are enough to obtain the exact impulse response in the noiseless case.

$2M$ equations are true:

$$y(2j-2) = s_j h_0 + s_{j-1} h_2 \quad , \quad y(2j-1) = s_j h_1 + s_{j-1} h_3 \quad , \quad 1 \le j \le M$$

where $y(2n+i-3) = x(nT+\delta_i)$, $h_{2n+i-1} = h(nT+\delta_i)$, for $i = 1, 2$.

An important fact to note here is that while there are $2M$ equations there are $M+5$ unknown variables (i.e., h_i and s_i). Therefore when $M \le 5$ we certainly cannot solve for the unknown variables, but if $M \ge 5$ we may be able to do so. Therefore the minimum length of the observation interval required to find $\{h_i\}$ is no less than $5T$. We will now show that under certain conditions we can in fact find the exact impulse response with an observation interval of just $5T$. Define

$$\mathbf{x}_0 = \begin{bmatrix} y(0) & y(2) & y(4) & \cdots & y(2M-2) \end{bmatrix}^T \quad , \quad \mathbf{x}_1 = \begin{bmatrix} y(1) & y(3) & y(5) & \cdots & y(2M-1) \end{bmatrix}^T$$

$$\mathbf{s}_0 = \begin{bmatrix} s_0 & s_1 & s_2 & \cdots & s_{M-1} \end{bmatrix}^T \quad , \quad \mathbf{s}_1 = \begin{bmatrix} s_1 & s_2 & s_3 & \cdots & s_M \end{bmatrix}^T$$

$$\mathbf{H} = \begin{bmatrix} h_2 & h_3 \\ h_0 & h_1 \end{bmatrix}$$

then it is clear that

$$\begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 \end{bmatrix} = \begin{bmatrix} \mathbf{s}_0 & \mathbf{s}_1 \end{bmatrix} \mathbf{H}. \tag{3}$$

Assuming that $\mathbf{H}$ is invertible, i.e.

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_0 & \lambda_2 \\ \lambda_1 & \lambda_3 \end{bmatrix} = \mathbf{H}^{-1} \tag{4}$$

exists, we see by post-multiplying by $\mathbf{\Lambda}$ in (3), that the vectors $\mathbf{s_0}$ and $\mathbf{s_1}$ lie in the span of the vectors $\mathbf{x_0}$ and $\mathbf{x_1}$ and we have

$$\mathbf{s}_0 = \lambda_0 \mathbf{x}_0 + \lambda_1 \mathbf{x}_1, \tag{5}$$

$$\mathbf{s}_1 = \lambda_2 \mathbf{x}_0 + \lambda_3 \mathbf{x}_1. \tag{6}$$

But $\mathbf{s}_0$ and $\mathbf{s}_1$ are not just any two vectors in the span of $\mathbf{x}_0$ and $\mathbf{x}_1$, they have a very special structure : The bottom $M-1$ elements of $\mathbf{s}_0$ are the same as the top $M-1$ elements of $\mathbf{s}_1$. We will now exploit this relationship. Partition the vectors $\mathbf{x}_0$ and $\mathbf{x}_1$ as below

$$\begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 \end{bmatrix} = \begin{bmatrix} y(0) & y(1) \\ \xi_0 & \xi_1 \end{bmatrix} = \begin{bmatrix} \eta_0 & \eta_1 \\ y(2M-2) & y(2M-1) \end{bmatrix} \tag{7}$$

where η_i and ξ_i are the top and bottom $M-1$ elements of $\mathbf{x}_i$. The constraint that the bottom $M-1$ elements of $\mathbf{s}_0$ are the same as the top $M-1$ elements of $\mathbf{s}_1$, translates into the following relation

$$\lambda_0\xi_0 + \lambda_1\xi_1 = \lambda_2\eta_0 + \lambda_3\eta_1. \tag{8}$$

This in turn implies that $\lambda = \begin{bmatrix} \lambda_0 & \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix}^T$ is in the null space of $\mathbf{\Phi}$ where

$$\mathbf{\Phi} = \begin{bmatrix} \xi_0 & \xi_1 & -\eta_0 & -\eta_1 \end{bmatrix}.$$

If we make the further assumption that $\mathbf{\Phi}$ has only a single dimensional null space[2], we can then find λ uniquely (up to a multiplication factor) and therefore $\mathbf{H} = \mathbf{\Lambda}^{-1}$ is also determined up to a multiplication factor. Note that this method needs an observation interval of only $5T$ to identify the channel, if the corresponding $\mathbf{\Phi}$ matrix has a one-dimensional null space.

2.2 The Basic Subspace Algorithm for the arbitrary J Case

We will now present the main steps of the algorithm for the identification of an arbitrary FIR channel. We assume that the impulse response has length JT and that the received signal is not corrupted by noise. In the noiseless case, this method can be used to identify the channel exactly[3] with an observation length of just $(J+3)T$. We use the same notations as for the $J=2$ case. Here the received signal is sampled at J times the baud rate. It is easy to see that we obtain the following equations for $0 \le i \le J-1$:

$$\begin{aligned} y(i) &= s_{J-1}h_i + s_{J-2}h_{J+i} + \cdots + s_0 h_{(J-1)J+i} \\ y(J+i) &= s_J h_i + s_{J-1}h_{J+i} + \cdots + s_1 h_{(J-1)J+i} \\ \vdots & \;\vdots\; \vdots \\ y(JM-J+i) &= s_{M+J-2}h_i + s_{M+J-3}h_{J+i} + \cdots + s_{M-1}h_{(J-1)J+i} \end{aligned}$$

where $y(Jn-J+j-1) = x(nT+\delta_j)$, $h_{Jn+j-1} = h(nT+\delta_j)$, $1 \le j \le J$.

Following the same procedure as in the $J=2$ case, we now define the vectors $\mathbf{x}_i$ and $\mathbf{s}_i$ for $0 \le i \le J-1$ as below

$$\mathbf{x}_i = \begin{bmatrix} y(i) & y(J+i) & y(2J+i) & \cdots & y((M-1)J+i) \end{bmatrix}^T$$

[2]This fact will be proved later

[3]Provided the corresponding $\mathbf{\Phi}$ matrix in (11) has a one-dimensional null space.

$$\mathbf{s}_i = \begin{bmatrix} s_i & s_{i+1} & s_{i+2} & \cdots & s_{M+i-1} \end{bmatrix}^T .$$

As before we obtain the relation

$$\mathbf{X} = \mathbf{SH} \tag{9}$$

where

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{J-2} & \mathbf{x}_{J-1} \end{bmatrix} \;,\; \mathbf{S} = \begin{bmatrix} \mathbf{s}_0 & \mathbf{s}_1 & \cdots & \mathbf{s}_{J-2} & \mathbf{s}_{J-1} \end{bmatrix}$$

and the k^{th} column of $\mathbf{H}$ is $\mathbf{h}_k$, $1 \le k \le J$, where

$$\mathbf{h}_k = \begin{bmatrix} h_{J(J-1)+k-1} & h_{J(J-2)+k-1} & \cdots & h_{J+k-1} & h_{k-1} \end{bmatrix}^T .$$

Inverting this relation we see that $\mathbf{s}_i \in span\{\mathbf{x}_0, \mathbf{x}_0, \cdots, \mathbf{x}_{J-1}\}$. Therefore we have for $0 \le i \le J-1$,

$$\mathbf{s}_i = \lambda_0^{(i)}\mathbf{x}_0 + \lambda_1^{(i)}\mathbf{x}_1 + \cdots + \lambda_{J-1}^{(i)}\mathbf{x}_{J-1} \tag{10}$$

where the $\{\lambda_j^{(i)}\}$, taken appropriately, form the matrix $\mathbf{\Lambda} = \mathbf{H}^{-1}$.

Let ξ_i and η_i be the bottom $M-1$ and top $M-1$ elements of $\mathbf{x_i}$, respectively. Then the constraint imposed by the structure of $\mathbf{s}_i$ (i.e. bottom $M-1$ elements of $\mathbf{s}_i$ is the same as the top $M-1$ elements of $\mathbf{s}_{i+1}$) translates into the following relation.

$$\mathbf{\Phi}\lambda = 0 \tag{11}$$

where

$$\lambda = [\lambda_0^{(0)}, \cdots, \lambda_{J-1}^{(0)}, \lambda_0^{(1)}, \cdots, \lambda_{J-1}^{(1)}, \cdots, \lambda_0^{(J-1)}, \cdots, \lambda_{J-1}^{(J-1)}]^T$$

$$\mathbf{\Phi} = \begin{bmatrix} \xi & \eta & 0 & 0 & \cdots & 0 & 0 \\ 0 & \xi & \eta & 0 & \cdots & 0 & 0 \\ 0 & 0 & \xi & \eta & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \xi & \eta \end{bmatrix}$$

with the number of block-columns being J, 0 representing a matrix of zeros and

$$\xi = \begin{bmatrix} \xi_0 & \xi_1 & \cdots & \xi_{J-1} \end{bmatrix} \;,\; \eta = -\begin{bmatrix} \eta_0 & \eta_1 & \cdots & \eta_{J-1} \end{bmatrix} . \tag{12}$$

Therefore we know that $\lambda \in Null(\mathbf{\Phi})$, and again if we know that $\mathbf{\Phi}$ has a one-dimensional null space, then we can find λ and therefore $\mathbf{H}$ and $\{s_k\}$.

3 Identifiability

The two assumptions under which the Basic Subspace algorithm will work are:

- **H** is invertible (or in the noisy case **H** is well-conditioned).

- $\mathbf{\Phi}$ has a one-dimensional null space.

The first assumption is violated in many systems of practical importance and **H** is often ill-conditioned. A careful look at the derivation of the Basic Subspace algorithm shows that the only crucial fact assumed about the **H** matrix is its full rankness, ie. we only need: rank(**H**)=J. If sampling the received signal at J-times the baud rate does not make the H matrix full rank, then it may be possible to construct a full rank impulse response matrix by sampling the signal at a higher rate. If that happens to be the case, the Basic Subspace algorithm can be correspondingly modified to take into account the fact that the **H** matrix is not square. This can be done by looking at pseudo-inverses instead of inverses. If the **H** matrix becomes full rank by this construction, ie. by sampling at a higher rate, then it has J linearly independent columns. Therefore, we can instead just sample the signal at the instants corresponding to these J columns and implement the Basic Subspace algorithm with a square invertible **H** matrix. This means that choosing the sampling instants appropriately and using the Basic Subspace algorithm, is equivalent to sampling the signal at a higher rate and using a modification of the algorithm using pseudo-inverses.

Hence, the only real problem arises when the **H** matrix remains singular for all choices of sampling instants. In section 4 we discuss the reasons which make the **H** matrix ill-conditioned. We then modify the Basic Subspace algorithm by a method analogous to the 'lower-rank' and 'lower-order approximation' methods used in situations when the numerical rank or numerical order is not the same as the actual rank or actual order. Simulation results using an an ill-conditioned **H** matrix show that the estimates obtained by using the modified algorithm are very close to the actual result.

If we assume that the **H** matrix is indeed invertible, it can be shown that the second assumption is almost never violated. We now prove the following theorem.

Theorem 3.1 *For every non-trivial*[4] *communication system, if* $\mathbf{H}$ *is invertible then the probability that* $\mathbf{\Phi}$ *has a one-dimensional null space tends to 1 as the observation interval increases. The rate of convergence is at least exponential if the symbols are independent.*

Theorem 3.1 ensures that as the observation interval increases, the probability of the violation of the second assumption tends to zero very fast and therefore even with a very small observation interval we can use the Basic Subspace algorithm to estimate the channel. Similar conditions based on persistently exciting sequences for the identifiability of a FIR channel have been discussed in [15].

We will now prove Theorem 3.1. In proving this theorem, we will also discover a very elegant approach to the Basic Subspace algorithm.

Let $\mathbf{s}_{\xi_i}$ and $\mathbf{s}_{\eta_i}$ be the bottom $M-1$ and top $M-1$ elements of the vector $\mathbf{s}_i$. Then we know that

$$\mathbf{s}_{\xi_0} = \mathbf{s}_{\eta_1} \ , \ \mathbf{s}_{\xi_1} = \mathbf{s}_{\eta_2} \ , \ \cdots, \ \mathbf{s}_{\xi_{J-3}} = \mathbf{s}_{\eta_{J-2}} \ , \ \mathbf{s}_{\xi_{J-2}} = \mathbf{s}_{\eta_{J-1}}.$$

It is therefore clear that of the $2J$ vectors $\mathbf{s}_{\xi_i}$ and $\mathbf{s}_{\eta_i}$, we only have $J+1$ different vectors $\mathbf{s}_{\eta_0}$,$\mathbf{s}_{\xi_0}$,$\mathbf{s}_{\xi_1}$,$\cdots$,$\mathbf{s}_{\xi_{J-1}}$ and for convenience we will call them $\mathcal{B}_0$, $\mathcal{B}_1$, $\cdots$, $\mathcal{B}_J$ respectively.

Lemma 3.1 *If* $\mathbf{H}$ *is invertible and the* $J+1$ *vectors* $\mathcal{B}_0$, $\mathcal{B}_1$, $\cdots$, $\mathcal{B}_J$, *are linearly independent, then the matrix* $\mathbf{\Phi}$ *has a one-dimensional null space.*

Proof: From (9) and (12) we get

$$\xi = \begin{bmatrix} \mathcal{B}_1 & \mathcal{B}_2 & \cdots & \mathcal{B}_J \end{bmatrix} \mathbf{H} \quad , \quad \eta = -\begin{bmatrix} \mathcal{B}_0 & \mathcal{B}_1 & \cdots & \mathcal{B}_{J-1} \end{bmatrix} \mathbf{H}.$$

Using this relation in $\mathbf{\Phi}$, we get

$$\mathbf{\Phi} = \mathbf{\Psi}\tilde{\mathbf{H}}, \tag{13}$$

where

$$\tilde{\mathbf{H}} = \begin{bmatrix} \mathbf{H} & 0 & 0 & \cdots & 0 \\ 0 & \mathbf{H} & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{H} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{H} \end{bmatrix}$$

[4]A trivial communication system is one in which all the future transmitted symbols are decided by a finite number of transmitted symbols and this is clearly not a very useful communication system!

and

$$\Psi = \begin{bmatrix} \mathcal{B}_1\mathcal{B}_2\cdots\mathcal{B}_J & \mathcal{B}_0\mathcal{B}_1\cdots\mathcal{B}_{J-1} & 0 & \cdots & 0 & 0 \\ 0 & \mathcal{B}_1\mathcal{B}_2\cdots\mathcal{B}_J & \mathcal{B}_0\mathcal{B}_1\cdots\mathcal{B}_{J-1} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mathcal{B}_1\mathcal{B}_2\cdots\mathcal{B}_J & \mathcal{B}_0\mathcal{B}_1\cdots\mathcal{B}_{J-1} \end{bmatrix}.$$

Since $\mathbf{H}$ is invertible, $\tilde{\mathbf{H}}$ is also invertible. Therefore $\boldsymbol{\Phi}$ will have a one-dimensional null space if $\boldsymbol{\Psi}$ has a one-dimensional null space. Lemma 3.2 confirms that $\boldsymbol{\Psi}$ does have a one-dimensional null space under the hypothesis of Lemma 3.1 and therefore we have proved Lemma 3.1. □

Lemma 3.2 *If the $J+1$ vectors $\mathcal{B}_0$, $\mathcal{B}_1$, $\cdots$, $\mathcal{B}_J$ are linearly independent, then the matrix $\boldsymbol{\Psi}$ has a one-dimensional null space and*

$$\mathcal{N}(\boldsymbol{\Psi}) = \{c \begin{bmatrix} e_1 & e_2 & e_3 & \cdots & e_J \end{bmatrix}^T : c \in \Re\}$$

where $e_i = \begin{bmatrix} 0 & 0 & \cdots & 1 & 0 & \cdots & 0 \end{bmatrix}$ (1 is in the i^{th} position).

Proof: Let the vector β be a vector in the null space of $\boldsymbol{\Psi}$ where

$$\beta = [\beta_{1,1}, \beta_{2,1}, \cdots, \beta_{J,1}, \beta_{1,2}, \beta_{2,2}, \cdots, \beta_{J,2}, \cdots, \beta_{1,J}, \beta_{2,J}, \cdots, \beta_{J,J}]^T.$$

Then

$$\boldsymbol{\Psi}\beta = 0.$$

From this equation, and, using the fact that since the vectors $\mathcal{B}_0$, $\mathcal{B}_1$, $\cdots$, $\mathcal{B}_J$ are linearly independent, a linear combination of them cannot be zero unless all the corresponding coefficients are zero, we get the following equations :

$$\begin{aligned} \beta_{1,i} &= 0 \quad for \;\; all \;\; i = 2, 3, \cdots, J-1, J \\ \beta_{1,i} - \beta_{2,i+1} &= 0 \quad for \;\; all \;\; i = 1, 2, 3, \cdots, J-1 \\ \beta_{2,i} - \beta_{3,i+1} &= 0 \quad for \;\; all \;\; i = 1, 2, 3, \cdots, J-1 \\ &\vdots \vdots \qquad\qquad\quad \vdots \vdots \\ \beta_{J-1,i} - \beta_{J,i+1} &= 0 \quad for \;\; all \;\; i = 1, 2, 3, \cdots, J-1 \\ \beta_{J,i} &= 0 \quad for \;\; all \;\; i = 1, 2, 3, \cdots, J-1. \end{aligned}$$

From these equations it can be concluded that $\beta_{i,j}$ for $i \neq j$ are all zero and $\beta_{1,1} = \beta_{2,2} = \cdots = \beta_{J,J}$. This means that the vector β is fixed up to a multiplication factor and since β is an arbitrary vector in the null space of $\boldsymbol{\Psi}$, we infer that $\boldsymbol{\Psi}$ has a one-dimensional null space generated by the vector $\begin{bmatrix} e_1 & e_2 & e_3 & \cdots & e_J \end{bmatrix}^T$. □

Therefore we come to the conclusion that if the **H** matrix is invertible, $\boldsymbol{\Phi}$ does not have a one-dimensional null space only if the $J+1$ vectors $\mathbf{s}_{\eta_0}, \mathbf{s}_{\xi_0}, \mathbf{s}_{\xi_1}, \cdots, \mathbf{s}_{\xi_{J-1}}$ are linearly dependent. This condition forces the following matrix to have a non-trivial null space:

$$\mathbf{Q} = [\mathbf{s}_{\eta_0}\ \mathbf{s}_{\xi_0}\ \mathbf{s}_{\xi_1}\ \cdots\ \mathbf{s}_{\xi_{J-1}}].$$

This in turn trivially implies the following necessary condition.

Lemma 3.3 *If* **H** *is invertible, then* $\boldsymbol{\Phi}$ *does not have a one-dimensional null space only if*

$$det \begin{pmatrix} s_i & s_{i+1} & \cdots & s_{i+J} \\ s_j & s_{j+1} & \cdots & s_{j+J} \\ \vdots & \vdots & \vdots & \vdots \\ s_k & s_{k+1} & \cdots & s_{k+J} \end{pmatrix} = 0 \quad \forall\, i, j, \cdots, k \;:\; i \neq j \neq \cdots \neq k.$$

Let the first $2J$ symbols $\{s_0, s_1, \cdots, s_{2J-1}\}$ take on a set of arbitrary values. Now we will look at the only sequences for which it may be possible for the necessary condition to hold.

When we use the necessary condition for $(i, j, \cdots, k) = (0, 1, \cdots, J)$ we fix the value of s_{2J}. When we then apply the necessary condition for $(i, j, \cdots, k) = (1, 2, \cdots, J+1)$ we have no choice but to fix the value of s_{2J+1}. Similarly applying the necessary condition successively we will fix the values of all the symbols $\{s_{2J}, \cdots, s_M\}$. Therefore there is only one sequence of length M with its first $2J$ elements specified that can **possibly** satisfy the necessary condition. It may also happen that for certain choices of the $2J$ elements no sequence may satisfy the necessary condition in its entirety. (Note moreover the fact that this is only a necessary condition and not a necessary and sufficient condition.)

In any sensible communication system the probability that a particular sequence of length M is transmitted given that the first $2J$ elements of the sequence take on specified values tends to zero as M increases. In particular if the symbols are from a finite set of size q then the number of possible values the first $2J$ elements can take is q^{2J} and the total number of sequences of length M are q^M. If the transmitted symbols are independent then the probability that the transmitted sequence satisfies the necessary condition is less than or equal to q^{2J-M}. Therefore we come to the conclusion that if $\mathbf{H}$ is invertible then

$$Prob(\mathbf{\Phi} \textit{ has a one}-\textit{dimensional null space}) \geq 1 - \frac{1}{q^{M-2J}}.$$

This means that as the observation interval M increases, the probability that the Basic Subspace algorithm fails tends to zero exponentially and if the symbols are not from a finite set, then clearly the probability of failure is zero if $M > 2J$ and we have therefore proved Theorem 3.1. □

The proof of Theorem 3.1 also opens up a new way of deriving the Basic Subspace algorithm. From (13) we know that vectors in the null space of $\mathbf{\Phi}$, when pre-multiplied by $\tilde{\mathbf{H}}$, produce vectors in the null space of $\mathbf{\Psi}$. Conversely, vectors in the null space of $\mathbf{\Psi}$, when premultiplied by $\tilde{\mathbf{H}}^{-1}$, produce vectors in the null space of $\mathbf{\Phi}$. This combined with the fact that the only vector (disregarding multiplication factors) in the null space of $\mathbf{\Psi}$ is $[e_1, e_2, e_3, \cdots, e_J]^T$ implies that if λ is a vector in the null space of $\mathbf{\Phi}$, then $\mathbf{H} = \mathbf{\Lambda}^{-1}$, where the matrix $\mathbf{\Lambda}$ is constructed by appropriately arranging the elements of the vector λ and then scaling it suitably. That this is indeed the case can be easily verified by noting that the vector $\begin{bmatrix} e_1 & e_2 & e_3 & \cdots & e_J \end{bmatrix}^T$ picks out the columns of the matrix $\mathbf{H}^{-1}$ from the matrix $\tilde{\mathbf{H}}^{-1}$ and arranges them one below the other to form the vector λ.

It must be noted that it is possible to use this algorithm to identify a general FIR channel of length JT exactly (in the noiseless situation) with an observation length of just $(J+3)T$. In the noisy situation, the same algorithm can be applied with the small modification that in solving for λ, instead of finding the null space of $\mathbf{\Phi}$, we find the singular vector corresponding to the smallest singular value of $\mathbf{\Phi}$.

4 The Least-Squares Subspace Algorithm

In this section we will attack the important problem of the $\mathbf{H}$ matrix being ill-conditioned. Before that we first look at the Exhaustive LS-Search algorithm, a **conceptual algorithm** for solving the blind identification problem. In the classical identification procedures, which use a training sequence, the problem is often solved in this way : Assuming that the received signal is sampled at the baud rate, we have the following matrix equation:

$$\mathbf{Sh} + \mathbf{n} = \mathbf{y} \tag{14}$$

where

$$\mathbf{S} = \begin{bmatrix} s_1 & s_2 & \cdots & s_{J-1} & s_J \\ s_2 & s_3 & \cdots & s_J & s_{J+1} \\ s_3 & s_4 & \cdots & s_{J+1} & s_{J+2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{M-J+1} & s_{M-J+2} & \cdots & s_{M-1} & s_M \end{bmatrix}$$

$$\mathbf{h} = [h_{J-1}, h_{J-2}, \cdots, h_1, h_0]^T \quad , \quad h_i = h(iT)$$

$$\mathbf{n} = [n_J, n_{J+1}, n_{J+2}, \cdots, n_M]^T \quad , \quad n_i = n(iT)$$

$$\mathbf{y} = [y_J, y_{J+1}, y_{J+2}, \cdots, y_M]^T \quad , \quad y_i = x(iT).$$

Since we know the **S** matrix, we can find the least-squares solution for the vector **h** which minimizes the norm-square error $||\mathbf{Sh} - \mathbf{y}||^2$. In the blind identification problem, we do not know the matrix **S**, because we do not know the transmitted sequence. But if the transmitted symbols are from a finite set (of say size 2, i.e. $s_i = \pm 1$) then we do know that the symbols $\{s_i\}_{i=1}^{M}$ can take one of 2^M possibilities. For each of these possibilities we will have a **S** matrix. For each of these possible **S** matrices we can find the least-squares solution for **h** and also the corresponding error $||\mathbf{Sh}-\mathbf{y}||^2$. We can then choose as our 'best' **S**, the one which minimizes the error and therefore our estimate of **h** will be the least-squares **h** corresponding to the 'best' **S**. It seems very likely that since in most cases the 'best' **S** that we obtain will be close to the actual transmitted **S**, our estimates of the channel impulse response **h** will be almost the same as that obtained by sending a training sequence and so will provide us with an excellent estimate of the channel impulse response. We performed a few simulation runs to check this fact and they confirmed this.

But the main problem with this exhaustive-search approach is that we will have to perform the least squares estimate 2^M times and this makes the practical implementation of this algorithm impossible. But if we could somehow get a reasonable estimate of the actual transmitted sequence, then we would need to search for the 'best' **S** only in the vicinity of the estimated **S**. This will considerably reduce the computational cost and if the estimate is very good (as it is when we use the Basic Subspace algorithm) then the computational cost may even be much lesser than the cost for other standard blind identification algorithms which make use of the statistics of the received signal.

4.1 The LS-Subspace Algorithm

Since the Basic Subspace algorithm provides a way of directly finding the transmitted sequence, we will use it to estimate $\mathbf{S}$. But the problem is that $\mathbf{H}$ is ill-conditioned and so we will not be able to use the Basic Subspace algorithm. But there is a way of circumventing this problem as we now illustrate.

Why is it that in many channels the $\mathbf{H}$ matrix is ill-conditioned? The Fig.1 shows a typical channel response. The tail of the response is very long and has a very small magnitude. The k^{th} column of $\mathbf{H}$ is $\mathbf{h}_k$, $1 \leq k \leq J$, where

$$\mathbf{h}_k = \left[\begin{array}{cccccc} h_{J(J-1)+k-1} & h_{J(J-2)+k-1} & \cdots & h_{J+k-1} & h_{k-1} \end{array} \right]^T$$

where $h_i = h(\frac{iT}{J})$. Near the tail ends the magnitude of the response is very small and so all the h_i which are obtained by sampling the impulse response near its tails are very close to zero and hence to each other. As can be seen these h_i's occupy the ends of the vector $\mathbf{h}_i$. Two vectors become very "similar" to each other if the number of almost-equal-elements becomes much more than the number of different elements and therefore a matrix composed of these vectors becomes ill-conditioned. It is therefore because of these end-elements that the $\mathbf{H}$ matrix becomes ill-conditioned. If we can get rid of the end-elements then we could use the Basic Subspace algorithm (for a smaller $\mathbf{H}$ matrix).

We now note that the tail of the impulse response contributes very little to the actual received signal and so we can neglect its contribution for our purpose (though it will be used in finding the transmitted sequence once the full impulse response is estimated). If we neglect the tail, the total length of the impulse becomes much less than it was before. Therefore the effective length of the impulse response is reduced to $J'T$ from the original JT. For HF channel and mobile radio channels J' is usually very small - about 2 or 3. We therefore need to sample the received signal at J' times the baud rate and perform the Basic Subspace algorithm under the assumption that the length of the response is only $J'T$. This will give us an estimate of the transmitted sequence $\widehat{s}_k$ and an estimate of the shortened impulse response $\widehat{\mathbf{h}}_s$. We can then use the estimate of the transmitted sequence $\widehat{s}_k$ to obtain a very good estimate of the full (unshortened) impulse response by solving the matrix least squares estimation problem of minimizing $||\mathbf{SH} - \mathbf{Y}||$ over all matrices $\mathbf{H}$. We could either solve of $\mathbf{H}$ by using the estimated

S matrix directly or by adopting one of following search-approaches to improve the estimate of the S matrix and using the 'best' S matrix thus obtained.

4.2 Approaches for Performance Improvement

Once we have a first estimate of the transmitted symbols using the shortened channel, we can then use a number of methods to improve our estimate of s_k and hence obtain a very good estimate of the complete impulse response **h**. The following are some of the proposed methods.

4.2.1 LS-Subspace Algorithms with Error Correction

Using the shortened impulse response $\mathbf{h}_s$ we can find the estimates $\widehat{\mathbf{s}}_i$ of the vectors $\mathbf{s}_i$. These vectors are then thrown into the symbol set $\mathcal{S}$ by hard-limiting. But since the vectors $\mathbf{s}_i$ have overlapping elements, we have the problem of choosing the right value in case the overlapping elements of $\widehat{\mathbf{s}}_i$ do not coincide.

To illustrate this point let us look at the following example in which $\mathcal{S} = \{\pm 1\}$ and we have the two 'transmitted vectors'

$$\mathbf{s}_0 = [1,1,-1,-1,1,1,-1,1] \quad , \quad \mathbf{s}_1 = [1,-1,-1,1,1,-1,1,-1]$$

and also the two 'received' estimates

$$\widehat{\mathbf{s}}_0 = [0.9,0.8,-0.5,-0.4,0.5,0.2,-0.1,0.4] \ , \ \widehat{\mathbf{s}}_1 = [0.3,-0.4,-0.7,0.8,-0.1,-0.3,0.1,-0.8]$$

which hard-limit to

$$\tilde{\mathbf{s}}_0 = [1,1,-1,-1,1,1,-1,1] \quad , \quad \tilde{\mathbf{s}}_1 = [1,-1,-1,1,-1,-1,1,-1].$$

As can be seen $\tilde{\mathbf{s}}_1$ has an error in the 5th bit, but the corresponding bit in $\tilde{\mathbf{s}}_0$ (i.e. the 6th bit) is correct. So there is clearly a contradiction and this signals the fact that there is an error in one of the estimates. If we use the arithmetic average of the two estimates $\tilde{\mathbf{s}}_0$ and $\tilde{\mathbf{s}}_1$ we will get zeros in places where the two estimates do not match. If there are d places in which we get zeros, then the correct estimate (at least in those d positions) is clearly one of the 2^d possible sequences (with ± 1 in the zero-positions). We can therefore apply the LS-search algorithm to this smaller search area and since we would expect d to be quite small, the LS-search algorithm is quite practical. For instance in our simulations d was in most cases 1 or 2, and in rare cases went up to a maximum of 6 and therefore in all cases the LS-search added only a small computational cost. Therefore the first step in all the versions in the sequel is compensating for these inherently self-detected errors. These represent something analogous to receiving an erasure - we know that there is an error in that position and so it does not take too much effort to correct it. Let the estimate after correcting for the erasures be the sequence $\{\hat{s}_k\}$. Now we can incorporate different levels of sophistication into the algorithm depending on the purpose for which it is needed. The following are some of the levels of sophistry that we propose:

Level 1: We need not make any correction at all. Just use the arithmetic average of the J estimates for $\{s_k\}$ to construct the matrix $\mathbf{S}$ and find the LS estimate for $\mathbf{h}$. We do not search for the 'best' LS estimate. We just use the $\mathbf{S}$ matrix that we get. The advantage is that this has a small computational cost. This will be useful in applications in which computational cost is a much bigger criteria as compared to the accuracy. But at any rate even here the accuracy of the estimate is much better than the conventional algorithms.

Level 2: We can just correct the erasures by the procedure already discussed. This will increase the computational cost a little but increases the accuracy of the estimate. We will henceforth refer to this algorithm as the **Erasure-Correcting LS-Subspace Algorithm** or the **ERC-LS-Subspace Algorithm**.

Level 3: We can introduce one more level of correction by correcting for single errors. For this we search for the LS-minimum over the space of all matrices $\mathbf{S}$ for which the corresponding sequence $\{s_k\}$ differs from the estimate $\{\hat{s}_k\}$ in at most one position. By this procedure we can find the best LS estimate if it lies within a distance of a single bit from the erasure-corrected estimate.

Level 4: We can generalize Level 3 by correcting for n-errors. For this we will have to expand the search area to cover all the matrices $\mathbf{S}$ for which the corresponding sequence $\{s_k\}$ differs from the estimate $\{\hat{s}_k\}$ in at most n positions.

The problem with the last method is that the computational cost increases almost exponentially as n increases, but the corresponding increase in accuracy is not so significant since in most cases there are very few errors. Even with one-error or two-error correcting procedure the estimate becomes very close to the exhaustive search LS algorithm, which is asymptotically the 'best' possible estimate we can have. We can avoid the increasing computational cost with increasing n by adopting the following method which is by far the best method we have come across both in maintaining a low computational cost as well as in achieving an estimate almost as good as the exhaustive search LS algorithm. By 'first estimate' we shall mean the sequence obtained by correcting for the erasures i.e. the sequence obtained at the end of Level 2.

The EC-LS-Subspace Algorithm

1. First let the search area be all the sequences which differ from the 'first sequence' in one position.
2. Find the 'best' sequence in this area.
3. If this sequence is the same as the 'first sequence' then stop and decide that this is the 'best' estimate.
4. If this sequence is not the same as the 'first sequence' then replace the 'first sequence' by this sequence and start all over again from step 1.

Why will this method work? The reason is that in most cases if the actual 'best' sequence differs from the estimated sequence in n positions then the 'best' sequence can be reached from the estimated sequence in steps and the corresponding sequences in the intermediate steps will be 'best' sequences in their respective local areas. It is very clear that this method reduces the computational complexity since at each stage we only test the sequences which differ in one position (if the number of symbols is m then it means m LS-computations) and we will have in most cases, n stages if there are n positions which have errors. This means we will totally have mn LS-computations in contrast to $\sum_{i=0}^{n} \begin{pmatrix} m \\ i \end{pmatrix}$ LS-computations if we use a search area with an ability to correct n errors. We will henceforth call this algorithm the **Error-Correcting-LS-Subspace Algorithm** or the **EC-LS-Subspace Algorithm**.

An important point to note here is that we have used the intuitively clear fact that in most cases if the starting point is quite close to the global minimum, then the global minimum of a set can be reached by finding the successive local minimums in smaller subsets (where the successive subsets are chosen properly depending on the previous local minimum). This point is very well illustrated by the Fig. 2 where each search area is a circle centered at the local minimum of the previous search area. The stopping condition of the algorithm is reached when the local minimum lies in the interior of its subset - this condition is analogous to the condition in elementary calculus, that, when a function attains its minimum or maximum, the derivative of the function is zero. The great reduction in the computational cost results because of a two-fold effect -

- In the n-error correcting algorithm the original area is of size $\sum_{i=0}^{n} \begin{pmatrix} m \\ i \end{pmatrix}$ as compared to a maximum search area for the EC-LS-Subspace algorithm (i.e. mn).
- The n-error correcting algorithm is designed for the worst case (i.e. n errors) and even if there are fewer errors the whole area will have to be searched whereas the EC-LS-Subspace algorithm is designed to be iterative and so when there are fewer errors (as is most often the case) it searches only a correspondingly smaller area.

But since this is an iterative method it may be argued that it will have the associated problems of convergence and will not be able to find the global minimum if one of the local minimums lies in the interior of its subset. It may either need too many iterations or may not converge if the global minimum can not be reached by moving through successive local minimums. But our simulation studies have shown that in almost all the cases (in fact in all the cases in our simulations) this method converges without any problems. We can take care of even the rare cases when there are convergence problems, by specifying an upper limit to the number of iterations, and, if the algorithm has not ended before the upper limit is reached, we can either discard the result obtained and look for a fresh set of samples, or, simply use the original estimate without corrections for finding $\mathbf{h}$. But these are merely technical issues, since, as our simulation results in the next section show, this algorithm is very good on all counts - low computational cost, high accuracy of estimate, and small observation interval.

4.2.2 The LS Iteration Algorithm

As we saw in the previous paragraphs, once we have a sufficiently good estimate of the transmitted symbols, we can employ various error correcting mechanisms to improve the performance and bring it very close to the limiting case of the exhaustive search LS algorithm without increasing the computational cost too much. In the following paragraphs we will discuss another method which attempts to improve the performance of the LS-Subspace algorithm iteratively.

We first note that we have to solve for the **S** and the **H** matrices and they are related by

$$\mathbf{SH} + \mathbf{N} = \mathbf{Y}$$

where **N** and **Y** represent the noise and received matrices. Our aim is therefore to minimize the error, $\|\mathbf{SH} - \mathbf{Y}\|$, over all the possible matrices **S** and **H**. It is a joint minimization problem over a two-dimensional matrix space. It might therefore be possible to solve it iteratively, by finding the minimum over a one-dimensional matrix space at each step.

We have an estimate of the **S** matrix obtained from the Basic Subspace algorithm. The idea is that we can solve for the least squares **H** by using the estimated **S** and then obtain a 'better' estimate of the **S** matrix from the equation $\mathbf{S} = \mathbf{Y}\mathbf{H}^{\dagger}$[5]. We can then repeat this procedure iteratively untill a good estimate for **H** is obtained.

We made a simulation study of this algorithm and the results show that this method is not as good as the EC-LS-Subspace algorithm. The reason is that this algorithm is to be used only when the **H** matrix is ill-conditioned and we will therefore be using an ill-conditioned matrix explicitly to obtain an estimate of the **S** matrix. If it was only an intermediate result that was ill-conditioned then maybe it could happen that after a few iterations the result will improve. But since here the final answer itself is ill-conditioned, we can not expect iterations to improve the result.

5 A Simulation Example

For our simulation we used the same channel which was used in the simulation example of [1] and the impulse response is shown in Fig. 1 and in Table 1. The source symbols were

[5] $\mathbf{A}^{\dagger}$ is the pseudo-inverse of the matrix **A**

drawn from a BPSK signal constellation with a uniform distribution. We implemented both - our algorithms and the algorithm by Tong et al [1] and made a comparative study.

A simulation of 100 independent trials was conducted for each algorithm under the same simulation scenario. Fig.3a, Fig.3b, Fig.3c, and Fig.3d show the 100 estimates for the EC-LS-Subspace algorithm, ERC-LS-Subspace algorithm, the LS-Iteration algorithm, and the algorithm by Tong et al. [1] each estimate using only 30 symbols. Fig.4a and Fig.4b show the 100 estimates for the EC-LS-Subspace algorithm and Tong's algorithm when each estimate is based on 100 symbols. These graphs show clearly that even if the number of symbols used for each estimate increases, the EC-LS-Subspace algorithm performs much better than Tong's algorithm. Fig.5a, Fig.5b, Fig.5c and Fig.5d show the 100 estimates for the EC-LS-Subspace algorithm for SNR = 20,15,10 and 5 dB, respectively. Simulations showed that even the few aberrations that occur in Fig.5b and Fig.5d vanish if the number of symbols used in the estimation is increased slightly. These aberrations occur because the EC-LS-Subspace algorithm is not able to converge to the correct sequence as the starting sequence was very different from the correct sequence. The starting sequence is very different from the correct sequence because the $\mathbf{\Phi}$ matrix does not have a 1-dimensional null space. We proved in Section 3, that the probability that the $\mathbf{\Phi}$ matrix does not have a 1-dimensional null space decreases exponentially with an increase in the number of symbols. Therefore if the number of symbols increases even a little, the probability that the starting sequence is very different from the correct one decreases rapidly and hence the aberrations vanish.

To obtain a performance measure of the channel estimation, the normalized root-mean-square error (NRMSE) of the estimator is defined by

$$NRMSE = \frac{1}{\|\mathbf{h}\|}\sqrt{\frac{1}{M}\sum_{i=1}^{M}\|\widehat{\mathbf{h}}_{(i)} - \mathbf{h}\|^2}$$

where M is the number for independent trials (100 in our case), and $\widehat{\mathbf{h}}_{(i)}$ is the estimate of the channel from the i^{th} trial. Fig.6a shows the NRMSE's of the different algorithms versus SNR in a series of 100 independent runs using 35 symbols for each estimate. Fig.6b shows the NRMSE's of the EC-LS-Subspace algorithm versus SNR in a series of 100 independent runs using different numbers of symbols for each set in the series. These plots show that unlike Tong's algorithm, the subspace algorithms do not have a cut-off SNR below which the performance decreases rapidly.

Moreover it can be seen that the subspace algorithms converge to a much smaller NRMSE than Tong's algorithm. Initially, with an increase in the number of symbols, the performance of the EC-LS-Subspace algorithm improves but the rate of this increase decreases and beyond 35 symbols the improvement in performance is not very significant since its performance is already very close to that of the conventional least squares algorithm.

We also tested the bit error rate (BER) against the SNR. Fig.7 shows the typical probability of error vs SNR curves that would be obtained for randomly chosen sequences, if either Tong's algorithm or the EC-LS-Subspace algorithm is used to first estimate the channel,[6] and then this channel is used to estimate the rest of the transmitted symbols. From the figure it can been seen that while Tong's algorithm does not perform well at low SNR's, the EC-LS-Subspace algorithm maintains a very good bit error rate even for very small SNR's. As can been seen at higher SNR's also, the performance of the EC-LS-Subspace algorithm is significantly better than that of Tong's algorithm.

Fig.8 describe the computational complexity of the different LS based Subspace algorithms as well as Tong's algorithm under different running conditions. Fig.8a shows the plot of the number of floating point operations (FLOPS) used in the MATLAB program as a function of the number of symbols used for the estimate for each of the algorithms. Fig.8b shows the same plot for the EC-LS-Subspace algorithm for different SNR's. These plots give us a general idea of the complexity of each algorithm. The algorithms were not optimized for the number of operations and so the estimates shown are well-above the actual complexity. A glance at the plots shows that at small observation intervals (which will be the running condition in practice) the complexity of the ERC-LS-Subspace algorithm is an order of magnitude lower than Tong's algorithm, and EC-LS-Subspace algorithm's complexity is also significantly lower than that of Tong's algorithm. But as the observation interval increases the complexities of all the subspace algorithms increase rapidly. This is due to the fact that we have to compute the SVD of a matrix whose size increases with the number of symbols used in the estimation. But, for the subspace algorithms we only need the singular vector corresponding to the smallest singular value and therefore if other optimum algorithms for computing the smallest singular vector were

[6]Using only the first few symbols for channel identification. For example, in our simulations, we used the first 25,30 and 100 symbols for identifying the channel.

to be used instead of the SVD, then the complexities of the subspace algorithms can be reduced significantly.

Another fact that can be observed is that as the SNR decreases, the complexity of the EC-LS-Subspace algorithm increases whereas the corresponding increase in Tong's algorithm is not that predominant. This is because of the fact that the EC-LS-Subspace algorithm directly deals with noise and tries to eliminate it by correcting the estimated sequence. This is what makes the EC-LS-Subspace algorithm so robust to noise and it is because of the increase in the number of corrections that have to be made as the SNR decreases that the complexity of this algorithm increases. On the other hand Tong's algorithm does not deal with the noise directly and therefore its complexity does not depend very much on SNR (except through the SVD computation) but this again means that it is not robust to noise and therefore breaks down at lower SNR.

The basic subspace algorithm without any correction will demonstrate similar variations in complexity with SNR. Moreover, it is unfair to compare the complexity of the two algorithms just based on the number of symbols used. It is more sensible to compare the complexity of the two algorithms when their performance is the same. Even with 1000 symbols the performance of Tong's algorithm is worse than that of the EC-LS-Subspace algorithm with just 35 symbols. Therefore if we use this criterion for comparing the complexities then the complexity of the EC-LS-Subspace algorithm is an order of magnitude better than Tong's algorithm.

These simulations show that the algorithms that we have proposed in this paper perform much better than Tong's algorithm with respect to fewer number of symbols needed for estimating the impulse response, robustness to noise - both in terms of the NRMSE as well as the BER, complexity and accuracy of estimate. The algorithm proposed in [7] by Moulines et.al. is better than Tong's algorithm in some of these respects, but the algorithms proposed in this paper easily outperform the algorithm in [7].

6 Conclusions

Blind identification and equalization is very useful in many communication systems. Traditionally blind identification algorithms need a large observation interval to obtain a reasonable estimate of the channel. This is because they have used higher order statistics for making their estimates. Even a cyclo-stationary second order statistics based approach by Tong, Xu and Kailath [1] needs a very large observation interval as compared to identification algorithms which use training sequences. This has therefore prevented an effective use of blind equalization algorithms in communication systems.

In this paper, we have proposed a new method for blind identification using subspace and least-square techniques. By exploiting the inherent structure in the received signal and using the ability to correct errors we are able to obtain a fast identification procedure which uses very few symbols, slightly more than what is used in identification procedures with training sequences. The proposed method leads to a very accurate estimate (much better than any of the previous algorithms) of the impulse response with a much smaller sample size (comparable to estimation procedures using a training sequence) than any other blind identification algorithm proposed till now. Moreover, the computational complexity of this algorithm is also lower than other algorithms. This is especially true for smaller observation intervals and equivalent performance levels. Furthermore the algorithm performs very well even under small SNR's and so it can be used in practical applications which use rapidly varying channels with very low SNR's. The main features of our algorithm are summarized below:

1. This algorithm needs very few symbols. In the noiseless case, it is possible to use this algorithm to obtain the exact impulse response[7] using an observation interval of just $(J + 3)T$, if JT is the length of the impulse response. In the noisy case, very good estimates are obtained with very small observation intervals. In our simulation example for instance, we need 30 symbols as compared to a 100 symbols in Tong's algorithm [1] and in fact our estimate of the impulse response with 30 symbols is much better than Tong's estimate with 100 symbols. This implies that this algorithm can be used in a wide range of applications which have rapidly changing channels.

2. The computational cost of the Basic Subspace algorithm is quite small and even with the extra additions for the least squares approach, the complexity is significantly lower than other blind equalization techniques. This is especially true for observation intervals of practical interest.

3. The algorithm provides a wide range of choices to the user and the user can choose between

[7]Provided the $\mathbf{\Phi}$ matrix in (11) has a one-dimensional null space

different levels of computational costs, accuracy levels and length of the estimation period depending on his or her need.

4. The performance of this algorithm is very close to the asymptotically 'best' algorithm, i.e. the least squares algorithm.

5. Since the received signal is oversampled, it has a better immunity to noise and interference [1], [14].

6. The algorithm works directly on the data domain and so the problems associated with other algorithms which are based on explicitly computing the statistics are not encountered.

7. The Basic Subspace algorithm (although not the LS-Subspace algorithm) can be used to estimate the channel even when nothing is known about the transmitted sequence. This will be useful in applications when the signal is received from an unknown transmitter.

References

[1] L. Tong, G. Xu and T. Kailath, "Blind identification and equalization based on second-order statistics: A time domain approach," *IEEE Trans. Inform. Theory,* vol. 40, pp. 340-350, Mar. 1994.

[2] D. Yellin and B. Porat, "Blind identification of FIR systems excited by discrete-alphabet inputs," *IEEE Trans. Signal Processing,* vol. 41, pp. 1331-1339, Mar. 1993.

[3] D. Hatzinakos, "Nonminimum phase channel deconvolution using the complex cepstrum of the cyclic autocorrelation," *IEEE Trans. Signal Processing,* vol. 42, pp. 3026-3042, Nov. 1994.

[4] Y. Li and Z. Ding, "Blind channel identification based on second order cyclostationary statistics," *Proc. Int. Conf. Acoust. Speech, Signal Processing.,* pp. IV:81-84, April 1993.

[5] H. Liu, G. Xu and L. Tong, " A deterministic approach to blind equalization," *Proc. 27th Asilomar Conf. Signals, Syst. Comput.,* 1993.

[6] D. T. M. Slock, " Blind fractionally-spaced equalization, perfect-reconstruction filter banks and multichannel linear prediction," *Proc. IEEE ICASSP,* 1994, pp. IV:573-576.

[7] E. Moulines, P. Duhamel, J. Cardoso and S. Mayrargue, " Subspace methods for the blind identification of multichannel FIR filters," *IEEE Trans. Signal Processing,* vol. 43, pp. 516-525, Feb. 1994.

[8] S. Talwar, M. Viberg and A. Paulraj, " Blind estimation of multiple co-channel digital signals using antenna array," *IEEE Signal Processing Lett.,* vol. 1, pp. 29-31, Feb. 1994.

[9] A. J. van der Veen, S. Talwar and A. Paulraj, " Blind estimation of multiple digital signals transmitted over FIR channels," *IEEE Signal Processing Lett.,* vol. 2, pp. 99-102, May 1995.

[10] G. Giannakis, Y. Inouye and J. Mendel, "Cumulant-based identification of multichannel moving average models," *IEEE Trans. Commun.*, vol. 34, pp. 783-787, 1989.

[11] G. Giannakis and J. Mendel, "Identification of non-minimum phase systems using higher-order statistics," *IEEE Trans. Acoust. Speech, Signal Processing.*, vol. 37, pp. 360-377, 1989.

[12] J. Mendel, "Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications,"*Proc. IEEE,* vol. 79, pp. 278-305, Mar. 1991.

[13] C. Nikias, "Blind deconvolution using higher-order statistics,"*Proc. 2nd Int. Conf. Higher-Order Stat.* Elsevier, 1992, pp. 49-56.

[14] W. A. Gardner and W. A. Brown, "Frequency-shift filtering theory for adaptive co-channel interference removal,"*Proc. 23rd Asilomar Conf. Signals,Syst., and Comput.*, Pacific Grove, CA, Oct.1989, pp. 562-567.

[15] F. Gustafsson, "Blind equalization by direct examination of the input sequences," *IEEE Trans. Commun.*, vol. 43, pp. 2213-2222, July 1995.

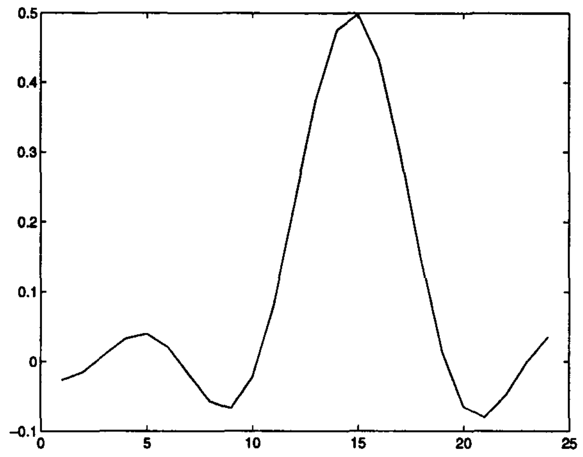

Figure 1: A three-ray multipath channel impulse response

n	1	2	3	4	5	6
h(n)	-0.02788	-0.01556	0.009773	0.0343	0.04142	0.0216
n	7	8	9	10	11	12
h(n)	-0.01959	-0.06035	-0.07025	-0.0241	0.08427	0.2351
n	13	14	15	16	17	18
h(n)	0.3874	0.4931	0.5167	0.4494	0.3132	0.152
n	19	20	21	22	23	24
h(n)	0.01383	-0.06754	-0.08374	-0.05137	-0.001258	0.03679

Table 1: Channel Impulse Response

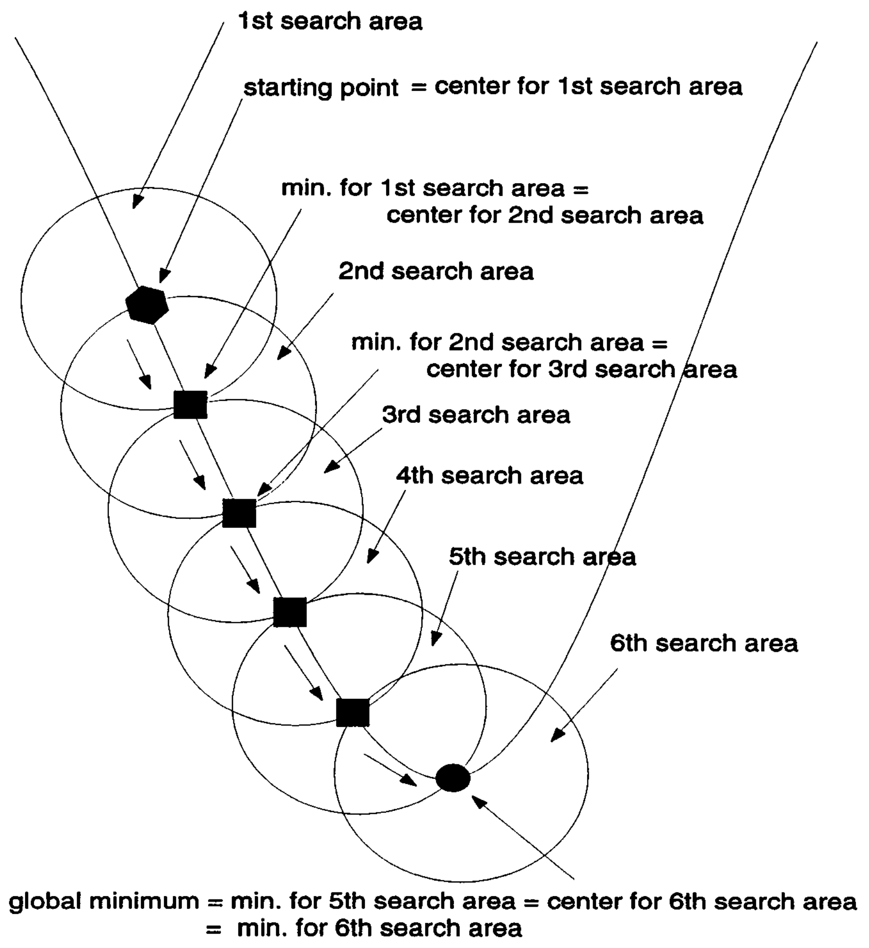
1st search area
starting point = center for 1st search area
min. for 1st search area = center for 2nd search area
2nd search area
min. for 2nd search area = center for 3rd search area
3rd search area
4th search area
5th search area
6th search area
global minimum = min. for 5th search area = center for 6th search area = min. for 6th search area

Figure 2: Graphical demonstration of the fact that the global (interior) minimum can often be reached through successive local (boundary) minimums

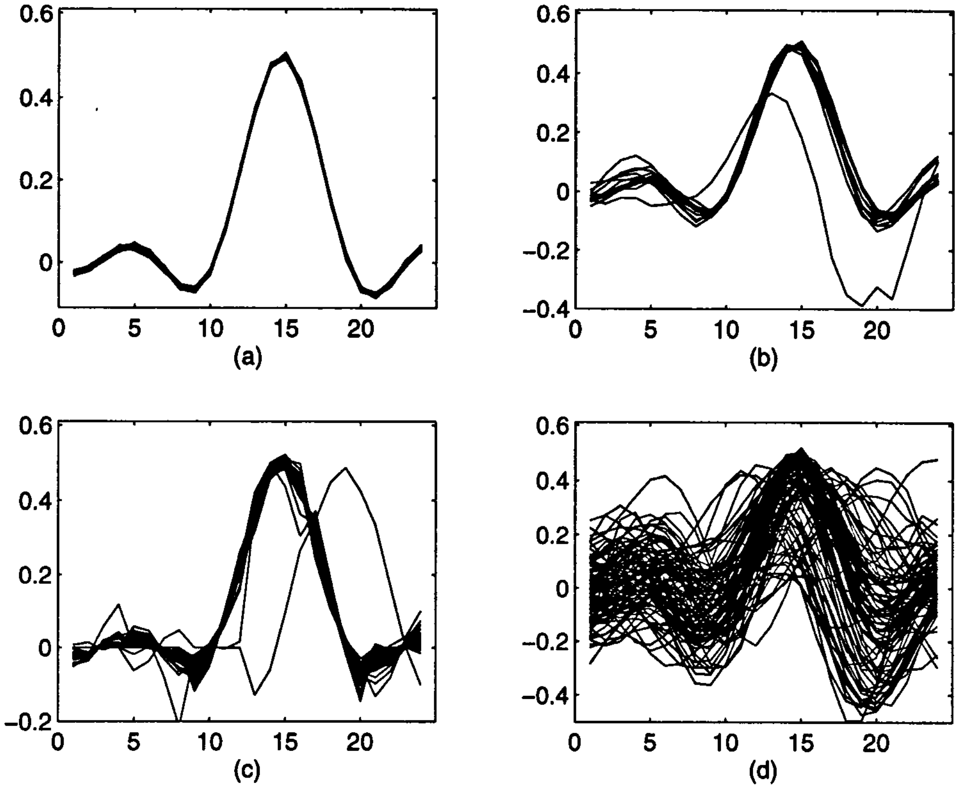

Figure 3: 100 estimates of the channel using 30 symbols for each estimate for (a) EC-LS-Subspace Algorithm (b) ERC-LS-Subspace Algorithm (c) LS-Iteration Algorithm and (d) Tong's Algorithm. In all cases SNR=30dB.

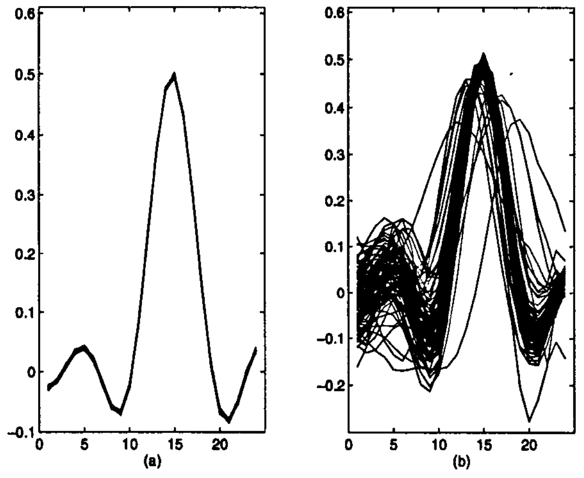

Figure 4: 100 estimates of the channel using 100 symbols for each estimate for (a) EC-LS-Subspace Algorithm (b) Tong's Algorithm. In both cases SNR=30dB.

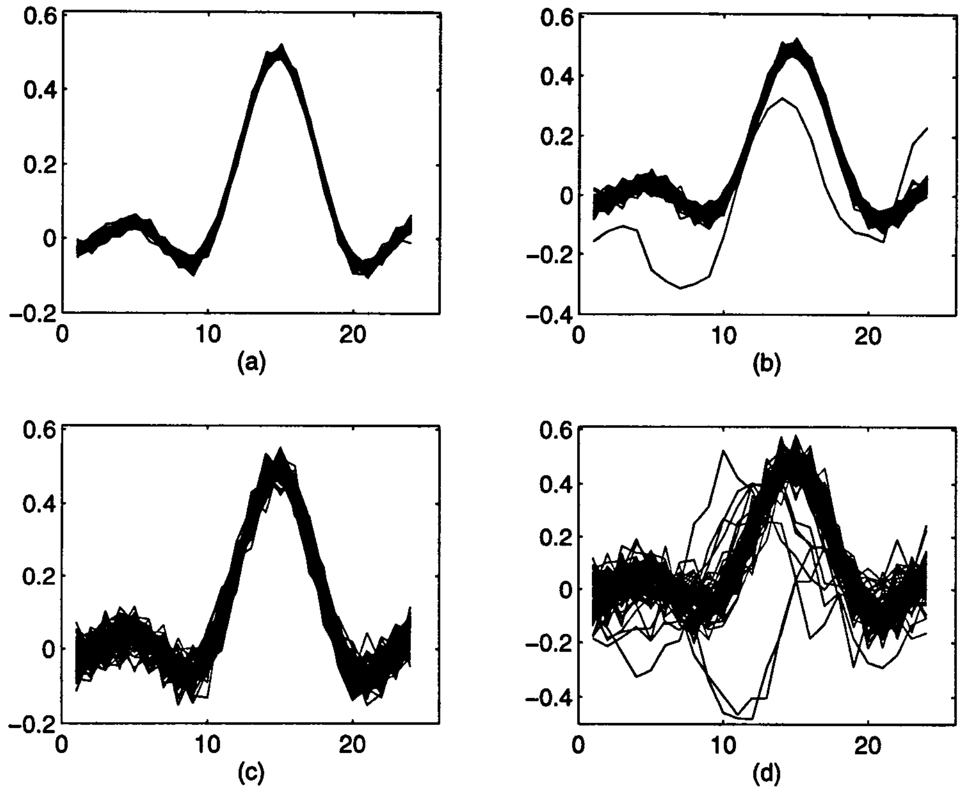

Figure 5: 100 estimates of the channel for (a) SNR=20 dB (b) SNR=15 dB (c) SNR=10 dB and (d) SNR=5 dB for the EC-LS-Subspace Algorithm. In all cases the number of symbols used for each estimate is 30.

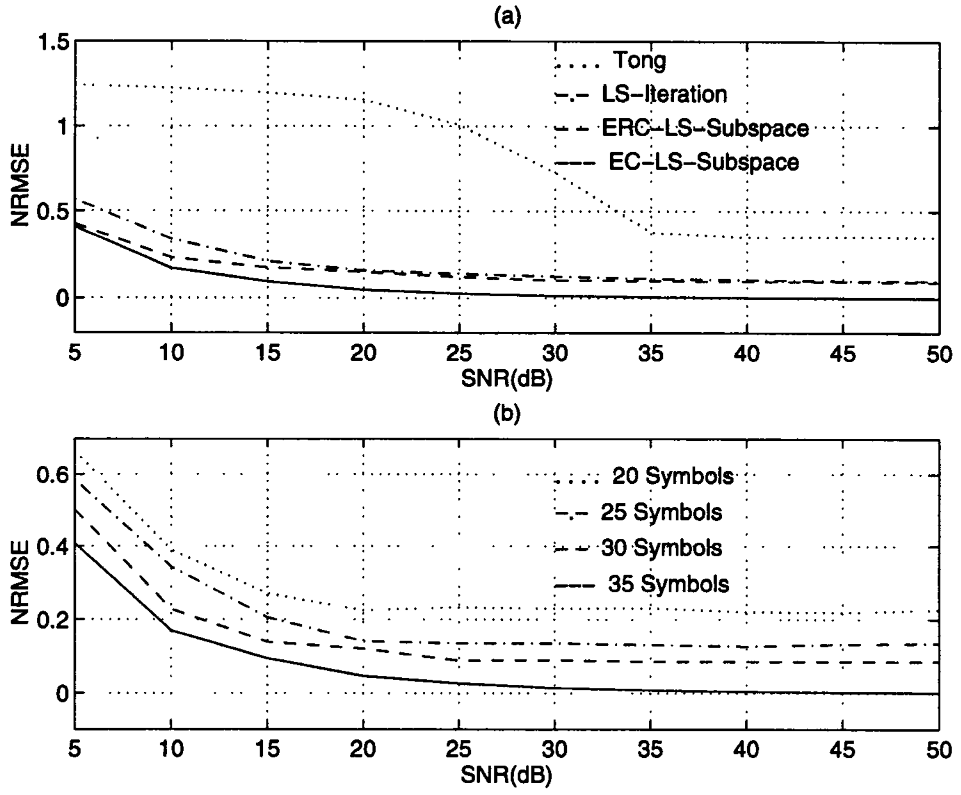

Figure 6: NRMSE versus SNR. 100 independent runs were used for the estimates and (a) Each estimate of the channel used 35 symbols (b) All estimates are for the EC-LS-Subspace Algorithm

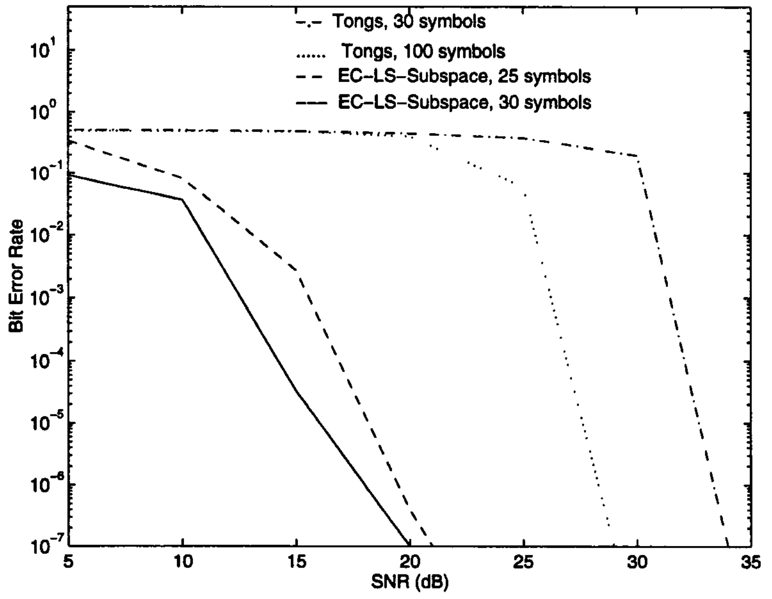

Figure 7: Bit error rate versus SNR for Tong's and EC-LS-Subspace algorithm. The index in the figure shows the number of symbols used in estimating the channel.

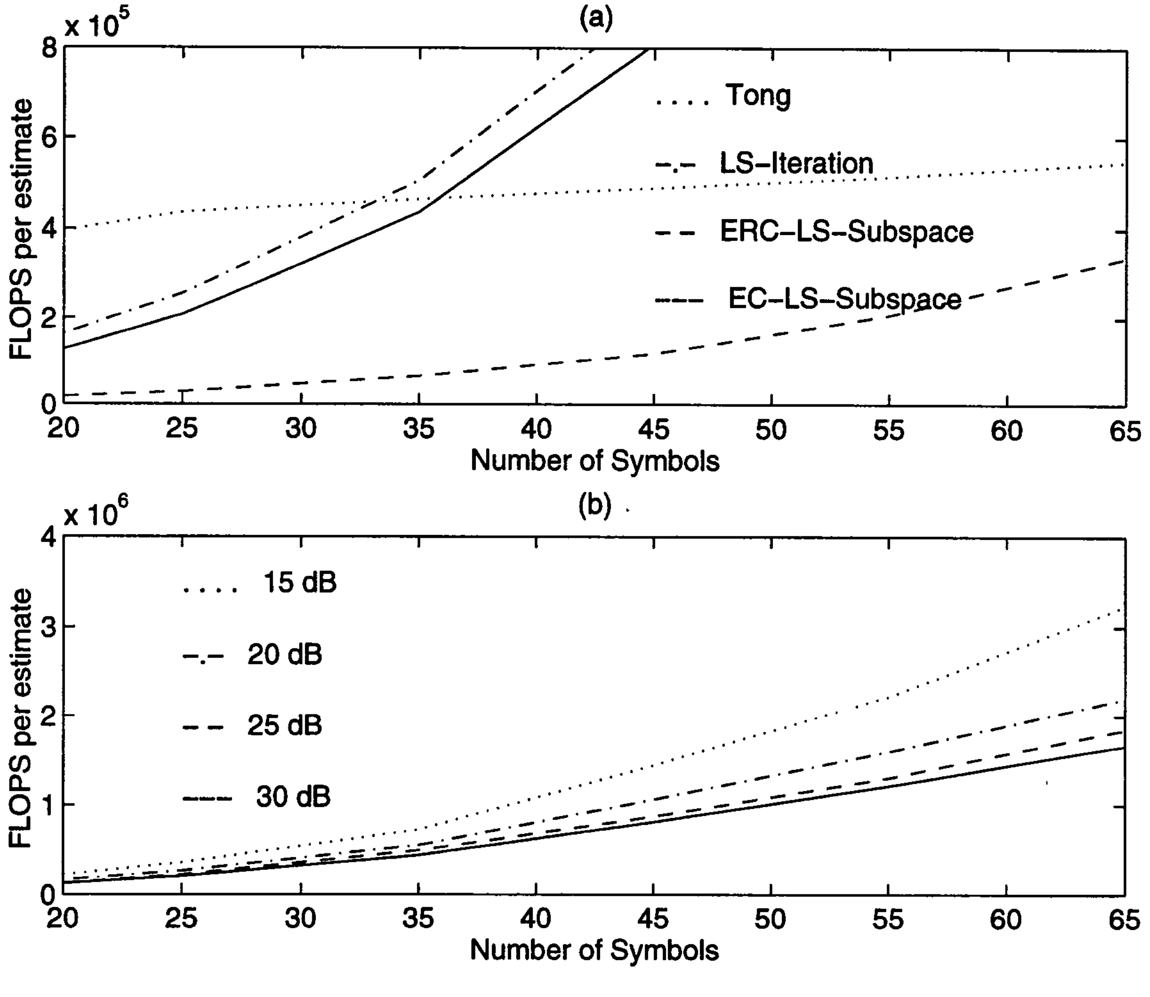

Figure 8: Number of FLOPS per estimate versus number of symbols used for each estimate. (a) Different Algorithms, SNR=30 dB (b) Different SNR's for the EC-LS-Subspace Algorithm. Note that even when the EC-LS-Subspace algorithm uses only 30 symbols, the estimate is much better than the estimate obtained by using Tong's algorithm with 100 symbols.