APPLICATIONS OF THE TOTAL LEAST SQUARES TECHNIQUE TO FREQUENCY ESTIMATION

by

Chung-Jung Chiang
September 1992

Thesis Advisor: Monique P. Fargues

Approved for public release; distribution is unlimited
APPLICATIONS OF THE TOTAL LEAST SQUARES TECHNIQUE TO FREQUENCY ESTIMATION

Chung-Jung Chiang

Master's Thesis

FROM__ TO __

September 1992

98

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

Estimating the frequencies of signals in a noisy environment has numerous applications in digital signal processing. In December 1980, Golub and Van Loan proposed a spectral estimator called the Total Least Squares (TLS) technique which is a refinement of the Least Squares (LS) technique. In this thesis, we first describe the origin of the TLS technique and its applications to frequency estimation. Furthermore, we present a numerical implementation for resolving two damped/undamped closely-spaced sinusoidal signals in white noise.

Next, we introduce TLS extensions such as the Constrained Total Least Squares (CTLS) technique and the Linear Constraint Total Least Squares (LCTLS) technique. The CTLS addresses the case where the noise components are related and the LCTLS addresses the case where one desires to resolve between two narrowband signals close in frequency, one of which is known. Finally, we present a numerical implementation of the Recursive Total Least Squares (RTLS) technique and apply it to the case of a signal with a fixed frequency together with a signal with a time-varying frequency.
Applications of the Total Least Squares Technique to Frequency Estimation

by

Chung-Jung Chiang
Lieutenant Commander, Taiwan Republic of China Navy
B.S., Chung-Cheng Institute of Technology, 1982

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1992

Author: Chung-Jung Chiang

Approved by: Monique P. Fargues, Thesis Advisor
Roberto Cristi, Second Reader

Michael A. Morgan, Chairman,
Department of Electrical and Computer Engineering
ABSTRACT

Estimating the frequencies of signals in a noisy environment has numerous applications in digital signal processing. In December 1980, Golub and Van Loan proposed a spectral estimator called the Total Least Squares (TLS) technique which is a refinement of the Least Squares (LS) technique. In this thesis, we first describe the origin of the TLS technique and its applications to frequency estimation. Furthermore, we present a numerical implementation for resolving two damped / undamped closely-spaced sinusoidal signals in white noise.

Next, we introduce TLS extensions such as the Constrained Total Least Squares (CTLS) technique and the Linear Constraint Total Least Squares (LCTLS) technique. The CTLS addresses the case where the noise components are related and the LCTLS addresses the case where one desires to resolve between two narrowband signals close in frequency, one of which is known. Finally, we present a numerical implementation of the Recursive Total Least Squares (RTLS) technique and apply it to the case of a signal with a fixed frequency together with a signal with a time-varying frequency.
# TABLE OF CONTENTS

I. **INTRODUCTION** ........................................................................................................ 1

II. **THE TOTAL LEAST SQUARES TECHNIQUE** ......................................................... 4
   A. **INTRODUCTION** .................................................................................................. 4
   B. **LEAST SQUARES (LS) ALGORITHM** .................................................................. 4
      1. Introduction ........................................................................................................ 4
      2. Algorithm Derivation ......................................................................................... 5
   C. **TOTAL LEAST SQUARES (TLS) ALGORITHM** ................................................... 5
      1. Introduction ........................................................................................................ 5
      2. Algorithm Derivation ......................................................................................... 8
   D. **CONSTRAINED TOTAL LEAST SQUARES (CTLS) ALGORITHM** ....................... 14
      1. Introduction ........................................................................................................ 14
      2. Algorithm Derivation ......................................................................................... 14
   E. **LINEARLY CONSTRAINT TOTAL LEAST SQUARES (LCTLS) ALGORITHM** ............ 20
      1. Introduction ........................................................................................................ 20
      2. Algorithm Derivation ......................................................................................... 20

III. **TOTAL LEAST SQUARES TECHNIQUE IMPLEMENTATION** .................................... 26
   A. **INTRODUCTION** .................................................................................................. 26
   B. **ALGORITHM IMPLEMENTATION** ....................................................................... 27
   C. **EXPERIMENTAL OBSERVATIONS** ................................................................. 28
      1. Performance of the Algorithm .......................................................................... 28
      2. Effects of Signal Phase Changes .................................................................... 29
      3. Effects of Filter Order Changes .................................................................... 29
      4. Effects of Errors in the Subspace Partition Estimate ..................................... 30
      5. Application to Damped Signals ...................................................................... 31
IV. RECURSIVE TLS TECHNIQUE
IMPLEMENTATION ........................................................................... 32

A. THEORETICAL BACKGROUND .......................................................... 32
1. Introduction .................................................................................. 32
2. Algorithm Derivation .................................................................. 32
3. Simulations .................................................................................. 37
   a. Case (1) ................................................................................... 37
   b. Case (2) ................................................................................... 37
   c. Case (3) ................................................................................... 37
   d. Case (4) ................................................................................... 37

B. EXPERIMENTAL OBSERVATIONS ................................................... 38
1. Estimated Signal Subspace Obtained with Averaging the
   Eigenvalues ................................................................................ 38
2. Estimated Signal Subspace Obtained without Averaging
   Eigenvalues ................................................................................ 39
3. Estimated Noise Subspace Obtained with Averaging the
   Eigenvalues ................................................................................ 39

V. CONCLUSIONS ............................................................................... 41

APPENDIX A : MATLAB SOURCE CODE ........................................ 66
1. IMPLEMENTATION OF THE TLS ALGORITHM ......................... 66
2. IMPLEMENTATION OF THE RTLS ALGORITHM USING
   THE SIGNAL SUBSPACE AND PERFORMING THE
   EIGENDECOMPOSITION OF MATRIX ' S ' ............................... 71
3. IMPLEMENTATION OF THE RTLS ALGORITHM USING
   THE SIGNAL SUBSPACE AND PERFORMING THE
   EIGENDECOMPOSITION OF MATRIX ' R ' ............................... 75
4. IMPLEMENTATION OF THE RTLS ALGORITHM USING
   THE NOISE SUBSPACE AND PERFORMING THE
   EIGENDECOMPOSITION OF MATRIX ' S ' ............................... 79

REFERENCES .................................................................................. 83

INITIAL DISTRIBUTION LIST ...................................................... 85
LIST OF TABLES

TABLE 1. EFFECTIVE STANDARD DEVIATIONS OBTAINED BY USING THE TLS ALGORITHM ........................................ 43
LIST OF FIGURES

Figure 3.1.1 TLS solution for SNR=20dB, P=8, N=50, f1=0.23, f2=0.27, undamped sinusoids.......................................................... 44
Figure 3.1.2 TLS solution for SNR=12dB, P=8, N=50, f1=0.23, f2=0.27, undamped sinusoids.......................................................... 44
Figure 3.1.3 TLS solution for SNR=5dB, P=8, N=50, f1=0.23, f2=0.27, undamped sinusoids.......................................................... 45
Figure 3.1.4 TLS solution for SNR=3dB, P=8, N=50, f1=0.23, f2=0.27, undamped sinusoids.......................................................... 45
Figure 3.1.5 TLS solution for SNR=1dB, P=8, N=50, f1=0.23, f2=0.27, undamped sinusoids.......................................................... 46
Figure 3.1.6 TLS solution for SNR=0dB, P=8, N=50, f1=0.23, f2=0.27, undamped sinusoids.......................................................... 46
Figure 3.2.1 TLS solution for SNR=3dB, P=8, N=50, f1=0.25, f2=0.27, undamped sinusoids.......................................................... 47
Figure 3.2.2 TLS solution for SNR=12dB, P=8, N=50, f1=0.25, f2=0.27, undamped sinusoids.......................................................... 47
Figure 3.2.3 TLS solution for SNR=3dB, P=8, N=50, f1=0.25, f2=0.27, \( \phi_1=\pi/4, \phi_2=-\pi/4\), damped sinusoids................................. 48
Figure 3.2.4 TLS solution for SNR=12dB, P=8, N=50, f1=0.25, f2=0.27, \( \phi_1=\pi/4, \phi_2=-\pi/4\), damped sinusoids................................. 48
Figure 3.3.1 TLS solution for SNR=12dB, P=2, N=50, f1=0.25, f2=0.27, undamped sinusoids.......................................................... 49
Figure 3.3.2 TLS solution for SNR=12dB, P=4, N=50, f1=0.25, f2=0.27, undamped sinusoids.......................................................... 49
Figure 3.3.3 TLS solution for SNR=12dB, P=8, N=50, f1=0.25, f2=0.27, undamped sinusoids.......................................................... 50
Figure 3.3.4 TLS solution for SNR=12dB, P=18, N=50, f₁=0.25, f₂=0.27, undamped sinusoids
Figure 3.3.5 TLS solution for SNR=12dB, P=28, N=50, f₁=0.25, f₂=0.27, undamped sinusoids
Figure 3.3.6 TLS solution for SNR=12dB, P=38, N=50, f₁=0.25, f₂=0.27, undamped sinusoids
Figure 3.4.1 TLS solution for SNR=12dB, P=8, N=50, f₁=0.25, f₂=0.27, undamped sinusoids, signal and noise subspaces mixed by 3 columns
Figure 3.4.2 TLS solution for SNR=12dB, P=8, N=50, f₁=0.25, f₂=0.27, undamped sinusoids, signal and noise subspaces mixed by 6 columns
Figure 3.5.1 TLS solution for SNR=5dB, P=8, N=50, f₁=0.25, f₂=0.27, damped sinusoids
Figure 3.5.2 TLS solution for SNR=20dB, P=8, N=50, f₁=0.25, f₂=0.27, damped sinusoids
Figure 3.5.3 TLS solution for SNR=5dB, P=18, N=50, f₁=0.25, f₂=0.27, damped sinusoids
Figure 3.5.4 TLS solution for SNR=20dB, P=18, N=50, f₁=0.25, f₂=0.27, damped sinusoids
Figure 4.1.1 RTLS solution for SNR=12dB, P=8, N=100, α=0.25, weight=3%, f₁ is time-varying, f₂=0.27, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'
Figure 4.1.2 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, α=0.25, weight=3%, f₁ is time-varying, f₂=0.27, solution derived by using the signal subspace information
Figure 4.1.3 RTLS solution for SNR=12dB, P=8, N=100, α=0.95, weight=3%, f₁ is time-varying, f₂=0.27, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'
Figure 4.1.4 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, α=0.95, weight=3%,
f1 is time-varying, f2=0.27, solution derived
by using the signal subspace information ........................................ 56

Figure 4.2.1 RTLS solution for SNR=12dB, P=8, N=100, α=0.95,
weight=10%, f1 is time-varying, f2=0.27, using the
signal subspace information & performing the
eigendecomposition of spherical matrix 'S' ........................................ 57

Figure 4.2.2 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, α=0.95, weight=10%,
f1 is time-varying, f2=0.27, solution derived
by using the signal subspace information ........................................ 57

Figure 4.3.1 RTLS solution for SNR=12dB, P=5, N=100, α=0.95,
weight=10%, f1 is time-varying, f2=0.27, using the
signal subspace information & performing the
eigendecomposition of spherical matrix 'S' ........................................ 58

Figure 4.3.2 RTLS solution for SNR=12dB, P=3, N=100, α=0.95,
weight=10%, f1 is time-varying, f2=0.27, using the
signal subspace information & performing the
eigendecomposition of spherical matrix 'S' ........................................ 58

Figure 4.4.1 RTLS solution for SNR=0dB, P=8, N=100, α=0.95,
weight=10%, f1 is time-varying, f2=0.27, using the
signal subspace information & performing the
eigendecomposition of spherical matrix 'S' ........................................ 59

Figure 4.4.2 RTLS solution for SNR=-5dB, P=8, N=100, α=0.95,
weight=10%, f1 is time-varying, f2=0.27, using the
signal subspace information & performing the
eigendecomposition of spherical matrix 'S' ........................................ 59

Figure 4.5.1 RTLS solution for SNR=12dB, P=8, N=100, α=0.95,
weight=10%, f1 is time-varying, f2=0.27, using the
signal subspace information & performing the
eigendecomposition of correlation matrix 'R' ..................................... 60

Figure 4.5.2 non-averaged eigenvalues of correlation matrix 'R'
SNR=12dB, P=8, N=100, α=0.95, weight=10%,
f1 is time-varying, f2=0.27, solution derived
by using the signal subspace information ........................................ 60
Figure 4.6.1  RTLS solution for SNR=0dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, using the signal subspace information & performing the eigendecomposition of correlation matrix 'R'................................. 61

Figure 4.6.2  non-averaged eigenvalues of correlation matrix 'R'
SNR=0dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, solution derived by using the signal subspace information.................................................. 61

Figure 4.6.3  RTLS solution for SNR=-5dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, using the signal subspace information & performing the eigendecomposition of correlation matrix 'R'................................. 62

Figure 4.6.4  non-averaged eigenvalues of correlation matrix 'R'
SNR=-5dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, solution derived by using the signal subspace information.................................................. 62

Figure 4.7.1  RTLS solution for SNR=12dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, using the noise subspace information & performing the eigendecomposition of spherical matrix 'S'........................................ 63

Figure 4.7.2  averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, solution derived by using the noise subspace information.................................................. 63

Figure 4.8.1  RTLS solution for SNR=12dB, P=5, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, using the noise subspace information & performing the eigendecomposition of spherical matrix 'S'........................................ 64

Figure 4.8.2  RTLS solution for SNR=12dB, P=3, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, using the noise subspace information & performing the eigendecomposition of spherical matrix 'S'........................................ 64

Figure 4.9.1  RTLS solution for SNR=15dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying, f2=0.27, using the noise subspace information & performing the eigendecomposition of spherical matrix 'S'........................................ 65
Figure 4.9.2  RTLS solution for SNR=30dB, P=8, N=100, α=0.95, weight=10%, f₁ is time-varying, f₂=0.27, using the noise subspace information & performing the eigendecomposition of spherical matrix 'S'........................................ 65
ACKNOWLEDGMENTS

First, I would like to thank my second reader, Professor Roberto Cristi, who helped me to understand the field of adaptive filtering. I would also like to thank Colin R. Cooper for his much needed help with word processing.

Finally, I sincerely express my gratitude to my thesis advisor, Professor Monique P. Fargues. Without her enthusiastic guidance, constant encouragement and long-term patience, this thesis would not have been completed. Thanks, Professor Fargues.
I. INTRODUCTION

Estimating the frequencies of signals in a noisy environment has numerous applications in digital signal processing. Various techniques have been proposed to solve that problem such as the Pisarenko algorithm [1, p. 623] and the MUSIC algorithm [1, p. 626]. These two algorithms adopt the eigen-decomposition technique to obtain the solution from the partitioned signal and noise subspaces. In this thesis, we present a subspace technique for frequency estimation called the Total Least Squares algorithm. Two extensions, the Constrained Total Least Squares algorithm and the Recursive Total Least Squares algorithm, are also introduced.

In Chapter II, we first introduce the ordinary Least Squares (LS) algorithm [1, p. 518]. The LS algorithm is used to solve the overdetermined set of linear prediction equations $Ax=b$, where $A$ and $b$ are $mxn$ and $mx1$ matrices depending on data, and $x$ is an array of parameters. In other words, the LS algorithm computes the solution $x_{LS}$ so that the Euclidean norm $\|Ax-b\|$ is minimum. Note that the LS algorithm considers the errors or perturbations in the observation vector $b$ only. However, in practice, we also need to consider the errors or perturbations occurring in the data matrix $A$. Thus, we consider the Total Least Squares (TLS) algorithm [2] which is an extension of the classical LS algorithm and has been applied to frequency estimation in signal processing to locate signals in the presence of additive noise. This algorithm can be viewed as a generalized LS algorithm and it has better performance than the ordinary LS algorithm. However, it is computationally more expensive.
We obtain the TLS solution $x_{\text{TLS}}$ by simultaneously minimizing the perturbations $\Delta A$ and $\Delta b$ which exist in the data matrix $A$ and the observation vector $b$, respectively.

When there exists some linear dependence algebraic relationship among the correlated noise perturbation components in $A$ and $b$, the TLS algorithm has to be modified to take this dependence into account. The modification leads to the Constrained Total Least Squares (CTLS) algorithm [6]. To be more specific, the CTLS algorithm results from an unconstrained minimization problem over a small set of variables. It uses the fact that the elements in perturbations $\Delta A$ and $\Delta b$ are algebraically correlated. By employing a new matrix $F$ with a smaller dimension and a white noise random vector $w$, it minimizes the Frobenius norm of $\Delta C$ to derive the solution $x_{\text{CTLS}}$. Next, we introduce the LCTLS algorithm [10]. This algorithm can be considered as an extension of the TLS algorithm and a special case of the CTLS algorithm. It is used to resolve closely-spaced frequencies with a given known frequency component. An example and a summary of the main steps needed to be implemented the LCTLS are presented.

Chapter III presents a numerical implementation of the TLS algorithm for resolving two closely-spaced damped / undamped sinusoids. We investigate the effects of choosing different signal phase angles, signal-to-noise ratios (SNR) and prediction filter orders. In addition, we explore the estimation behavior of the algorithm due to incorrect singular subspace partitions.

In Chapter IV, we introduce the Recursive Total Least Squares (RTLS) algorithm [11]. The algorithm is used in this thesis to estimate the frequencies of two complex sinusoidal signals in white noise. We adopt a spherical
subspace updating [12] technique and use an eigendecomposition method to obtain the solution $x_{\text{RTLS}}$. Four parameters are varied: the signal-to-noise (SNR) ratio, the prediction filter order, the forgetting factor and the range of the frequencies involved. We also study the algorithm performances obtained by using the signal or the noise subspace to estimate the frequencies. Last, Chapter V presents conclusions.
II. THE TOTAL LEAST SQUARES TECHNIQUE AND ITS EXTENSIONS

A. INTRODUCTION

Resolving two closely spaced sinusoidal signals in a noisy environment is a problem of importance in digital signal processing. This problem becomes rather difficult when the number of data samples is small or the signal-to-noise ratio (SNR) is low. Here, we adopt the Total Least Squares (TLS) approach to solve a Linear Prediction Equation (LPE) to improve the resolution. Note that the performance of the algorithm deteriorates when SNR is low, say below 3dB.

The TLS technique is the generalization of the ordinary Least Squares (LS) technique [1] and it is often used to solve an overdetermined set of linear equations obtained from noisy observations. When there are more equations than unknowns, the system $Ax=b$ is said to be overdetermined. The TLS method takes into account the errors that occur in the data matrix $A$ as well as in the observation vector $b$ and try to attenuate these noise effects on both sides simultaneously. Further details are presented later in this section.

B. LEAST SQUARES (LS) ALGORITHM

1. Introduction

First, recall the ordinary Least Squares (LS) technique. For the LS algorithm, the data matrix $A$ is assumed to be free of error and all errors are confined to the observation vector $b$. However, this assumption is not always realistic since many errors (perturbations / interference) like sampling errors,
modeling errors and instrument errors, etc. may result in inaccuracies or uncertainties in $A$.

2. Algorithm Derivation

We obtain the LS solution $x$ by minimizing the Euclidean norm $\|Ax - b\|$. Thus, it is equivalent to the following expression:

$$\min_{x, \Delta b} \|\Delta b\|$$

subject to

$$Ax = b + \Delta b$$

which leads to

$$x_{LS} = (A^HA)^{-1}A^Hb = A^+b$$

where $A^+$ is the pseudoinverse of $A$, $A^HA$ is the correlation matrix. The superscript notation $'H'$ denotes the complex conjugate transposition. Note that the LS algorithm works well when considering the noise only in the observation vector $b$.

C. TOTAL LEAST SQUARES (TLS) ALGORITHM

1. Introduction

First, we represent the forward linear prediction equation $Ax = b$ in a matrix form:
\[
\begin{bmatrix}
\xi_0 & \xi_1 & \cdots & \xi_{P-1} \\
\xi_1 & \xi_2 & \cdots & \xi_P \\
\vdots & \vdots & \ddots & \vdots \\
\xi_{N-P-1} & \xi_{N-P} & \cdots & \xi_{N-2} \\
\end{bmatrix}
= 
\begin{bmatrix}
x_P \\
x_{P-1} \\
\vdots \\
x_1 \\
\end{bmatrix}
= 
\begin{bmatrix}
\xi_P \\
\xi_{P+1} \\
\vdots \\
\xi_{N-1} \\
\end{bmatrix}
\]

(2.4)

\(\xi_i\) is the observed noisy signal, \(P\) is the prediction filter order and is usually larger than the number \(M\) of exponentials but smaller than the number \(N\) of data samples.

If the observed data is noisy, from a statistical viewpoint the LS solution will no longer be optimal because it can be biased and suffer covariance from the accumulation of noise errors in \(A^H A\). To alleviate this problem, we adopt the Total Least Squares (TLS) technique [2]. It has been shown to be equivalent to the minimum norm method [3] and it is used to diminish the bias accrued in \(A^H A\) or to remove the noise components by using a perturbation analysis on \(A\) and \(b\) of the smallest 2-norm that makes the system equations consistent. The TLS solution can be obtained by minimizing

\[\| \Delta C \| = \| [ \Delta A | \Delta b ] \| \]

(2.5)

subject to

\[(A + \Delta A) x = b + \Delta b\]

(2.6)

which leads to

\[x_{TLS} = \left( A^H A - \sigma^2 I \right)^{-1} A^H b\]

(2.7)

where \(\Delta A\) and \(\Delta b\) are two independent noise perturbations and \(\sigma^2\) is the minimum eigenvalue of \([ A b ]^H [ A b ]\).
Note that we use a zero mean white *identically independent distributed (i.i.d.) Gaussian noise* in our analysis of the TLS algorithm. When the noise is *not white* but *colored* and still uncorrelated with the signals, a *whitening filter* ought to be applied. This kind of filter is capable of transforming a stationary discrete-time non-white process at the input into a white uncorrelated data sequence at the output. The *whitening procedure* takes the non-white noise vector $w$ and computes $R = E[ww^*] = ZZ^*$, using a *Cholesky decomposition*. This leads to the definition of a new white noise vector $\delta$ where
\[
\delta = Z^{-1}w.
\] (2.8)

The *whitening procedure* can be performed by using a prediction-error filter which offers a linear prediction equation same as $Ax=b$. Basically, the prediction is dependent on the presence of correlation between adjacent samples of the input non-white stochastic process. Then, we can successively decrease the correlations by increasing the prediction-error filter order high enough until it ultimately gets to a point at which the output sequence is composed of (white) uncorrelated samples [4, p. 216].

The *TLS solution* $x_{TLS}$ is obtained by using information obtained from the *singular value decomposition (SVD)* of the matrix $C = [A \mid b]$ with a dimension $(N-P)\times(P+1)$, i.e.,
\[
C = U\Sigma V^H
\] (2.9)

where $U$ and $V$ are unitary square matrices with a dimension $(N-P)\times(N-P)$ and $(P+1)\times(P+1)$, respectively; $\Sigma$ is a rectangular diagonal matrix with a dimension $(N-P)\times(P+1)$ consisting of the singular values in decreasing order with the largest
The matrices $U$, $V$ and $\Sigma$ can be partitioned as

$$U = \begin{bmatrix} u_1 & u_2 & \cdots & u_{N,P} \end{bmatrix}$$

(2.10)

$$V = \begin{bmatrix} v_1 & v_2 & \cdots & v_{P+1} \end{bmatrix}$$

(2.11)

and

$$\Sigma = \text{diag} (\sigma_1, \cdots, \sigma_P, \sigma_{P+1})$$

(2.12)

The columns of the matrices $U$ and $V$ are the unitary eigenvectors of $AA^H$ and $A^HA$ respectively.

2. Algorithm Derivation

The TLS algorithm described above can be summarized into the following equations [5]:

$$\left( \begin{bmatrix} Ab \end{bmatrix} + \begin{bmatrix} \Delta Ab \end{bmatrix} \right) \begin{bmatrix} x \\ -1 \end{bmatrix} = 0$$

(2.13)

or

$$\left( C + \Delta C \right) z = 0$$

(2.14)

where

$$C = \begin{bmatrix} Ab \end{bmatrix}, \quad \Delta C = \begin{bmatrix} \Delta Ab \end{bmatrix}, \quad z = \begin{bmatrix} x \\ -1 \end{bmatrix}$$

(2.15)

From the homogeneous equation (2.13), we see that the TLS problem finds the minimum norm of the perturbation matrix $\Delta C$ such that $(C + \Delta C)$ is
rank deficient. In other words, the TLS solution can be formulated as seeking a solution vector $x$ [6] to the following problem

$$\min_{\Delta C, x} \|\Delta C\|_F^2,$$  \hspace{1cm} (2.16)

subject to

$$(C + \Delta C) \begin{pmatrix} x \\ -1 \end{pmatrix} = 0$$  \hspace{1cm} (2.17)

where

$$\|\Delta C\|_F^2 = \text{tr}(\Delta C^* \Delta C),$$  \hspace{1cm} (2.18)

$\| \cdot \|_F$ denotes the Frobenius norm and " tr " is the trace of a matrix.

Next, the matrix $C$ is decomposed in the following form

$$C = [U_1 \quad U_2] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^H \\ V_2^H \end{bmatrix}$$  \hspace{1cm} (2.19)

where $U_1, \Sigma_1$ and $V_1$ correspond to the signal subspace; $U_2, \Sigma_2$ and $V_2$ correspond to the noise subspace. We separate the signal subspace from the noise subspace by using the number of sinusoids. Two different cases need to be considered. The first one considers the case where the singular values are all distinct. The second one considers the case where some of the singular values are multiple. For the first case, we usually assume that the last element $\sigma_{p+1}$ in the diagonal matrix $\Sigma$ is the smallest singular value and it corresponds to the last vector $v_{p+1}$ in the unitary matrix $V$. Then we get
$$x_{\text{TLS}} = \frac{1}{(v_{p+1})_{p+1}} \begin{bmatrix} v_{(p+1),1} \\ \vdots \\ v_{(p+1),r} \end{bmatrix}$$  \hspace{1cm} (2.20)

where \((v_{p+1})_{p+1}\) is the last element of the last singular vector \(v_{p+1}\) and is assumed to be non-zero for normalization. If \((v_{p+1})_{p+1}\) is equal to zero, a TLS solution does not exist. For the second case, we assume that the \(P-M+1\) smallest singular values are equal, i.e.,

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_M > \sigma_{M+1} = \cdots = \sigma_{P+1},$$  \hspace{1cm} (2.21)

and set \(V_2\) to be a column partition of \(V\) where

$$V_2 = (v_{M+1}, \ldots, v_{p+1})$$  \hspace{1cm} (2.22)

or

$$V_2 = \begin{pmatrix} c^T \\ V_2' \end{pmatrix}$$  \hspace{1cm} (2.23)

where \(c^T\) represents the first row of the matrix \(V_2\).

Next, we introduce the *Householder* (reflection) matrix \(Q\) which is used to solve the TLS problem. It has the property to zero out specific entries in a matrix or vector. Usually, it is defined as [4, p. 428]:

$$Q = I_p - 2 \frac{vv^*}{v^*v}$$  \hspace{1cm} (2.24)
where

$I$ is an $m \times m$ identity matrix,

$v$ is an $m \times 1$ complex vector which corresponds to the complex vector $v_i$ in $V_2$ defined in (2.22) and has the Euclidean norm

$$
\|v\| = (v^H v)^{1/2}.
$$

(2.25)

The vector $v$ can be defined below for analysis:

$$
v = c^* - \alpha^* e_p \quad \Rightarrow \quad v^H = c^T - \alpha^T e_p^H
$$

(2.26)

$$
\alpha^* = \pm \frac{c_p^*}{|c_p|} |c| \quad \Rightarrow \quad \alpha = \pm \frac{c_p}{|c_p|} |c|
$$

(2.27)

$$
e_p^T = [0, \cdots, 0, 1]_{1 \times p} \quad \Rightarrow \quad v_p^* = c_p - \alpha.
$$

(2.28)

For any given vector, we can construct a Householder matrix so that all the energy is compressed into a selected partitioned column vector [7]. This leads to the following TLS solution:

$$
V_2 Q = \begin{bmatrix} 0 & \alpha \\ W & y \end{bmatrix}
$$

(2.29)

$$
c^T Q = [0, \cdots, 0, \alpha].
$$

(2.30)

Now we express $V_2 Q$ to extract the last column, which is used in the solution:
\[ V_2Q = V_2 [ q_1 | q_2 | q_3 | \ldots | q_p ] \] (2.31)

and let

\[ V_2q_p = \begin{bmatrix} \alpha \\ y \end{bmatrix}. \] (2.32)

Note that

\[ v^H v = (c^T - \alpha^T e_p^H)(c^* - \alpha^* e_p) \]
\[ = \|c\|^2 - \alpha^* c_p - \alpha c_p^* + \|\alpha\|^2 \]
\[ = \|c\|^2 - 2\alpha c_p + \|\alpha\|^2 \]
\[ = \|c\|^2 - 2\frac{|c_p|^2}{|c_p|} \|c\| + 2\frac{|c_p|^2}{|c_p|^2} \|c\|^2 \]
\[ = 2\left(\|\alpha\|^2 - \frac{c_p c_p^*}{|c_p|^2} \|c\|\right) \]
\[ = 2\left(\alpha \alpha^* - \alpha^* c_p\right). \] (2.33)

The last column of the matrix \( Q \) is given by

\[ q_p = e_p - \frac{2v^*_p v}{v^H v} \]
\[ = e_p - \frac{2(c_p - \alpha)}{2\alpha^* (\alpha - c_p)} (c^* - \alpha^* e_p) \]
\[ = e_p - \frac{c_p c^* - c_p \alpha e_p}{\alpha^* (\alpha - c_p)} + \frac{\alpha \alpha^* e_p}{\alpha^* (\alpha - c_p)} \]
\[ = \frac{c^* (\alpha - c_p)}{\alpha^* (\alpha - c_p)} = \frac{c^*}{\alpha}. \] (2.34)
Thus,

\[ y = V_2 q_p = \frac{V_2 c^*}{\alpha^*} \]  

(2.35)

and

\[ x_{\text{TLS}} = -\frac{y}{\alpha} = -\frac{V_2 c^*}{\alpha \alpha^*} = -\frac{V_2 c^*}{c^H c} \]  

(2.36)

Equation (2.36) corresponds to the noise subspace case since \( V_2 \) and \( c \) are obtained from \( V_2 \) which is described in (2.19). Similarly, we can derive the solution vector from the signal subspace. First, we use the identity

\[
\begin{bmatrix}
g^H & c^H \\
V_1' & V_2'
\end{bmatrix}
\begin{bmatrix}
g \\
V_1' \\
V_2'
\end{bmatrix}
= I
\]  

(2.37)

which leads to

\begin{align*}
g^H g + c^H c &= 1 \quad \Rightarrow \quad c^H c = 1 - g^H g \\
V_1' g + V_2' c &= 0 \quad \Rightarrow \quad V_2' c = -V_1' g
\end{align*}

(2.38)

Next, we substitute (2.38) into (2.36) and obtain the alternative TLS solution corresponding to the signal subspace case:

\[ x_{\text{TLS}} = \frac{V_1' g^*}{1 - g^H g} \]  

(2.39)

where \( V_1' \) and \( g \) are obtained from \( V_1 \) which is also described in (2.19).
D. CONSTRAINED TOTAL LEAST SQUARES (CTLS) ALGORITHM

1. Introduction

When the noise components contained in the augmented matrix $C$ are algebraically correlated or linearly dependent (so called constraints), the TLS technique is no longer an optimal frequency spectral estimator. In this case, we have to reformulate the problem to reduce the dimensionality of the disturbances. Thus, the TLS needs to be reformulated to reduce the dimensionality of the noise entries. Thus, the Constrained Total Least Squares (CTLS) algorithm, proposed by Abatzoglou et al [6], can be viewed as the natural extension of the TLS algorithm. In addition, the CTLS can be shown to be equivalent to the Maximum Likelihood (ML) algorithm. The CTLS solution can be obtained from a constrained state-space-parameter ML estimator together with a Newton algorithm [8].

2. Algorithm Derivation

In order to remove the correlated noise perturbation components among $A$ and $b$, we derive the CTLS algorithm from an unconstrained minimization problem over a small set of variables. Let us redefine $\Delta A$ and $\Delta b$ both as linearly constrained.

\[
\Delta A = [ F_1 w | F_2 w | \cdots | F_p w ] \quad (2.40)
\]

\[
\Delta b = F_{p+1} w \quad (2.41)
\]

Then the augmented error matrix $\Delta C$ is formed by

\[
\Delta C = [ \Delta A | \Delta b ] = [ F_1 w | F_2 w | \cdots | F_p w | F_{p+1} w ] \quad (2.42)
\]
where $F$ is a matrix whose elements are related to those of independent variables $\Delta C$ and $w$ is a zero mean white noise random vector of minimal possible dimensionality. The reason for $w$ holding the minimum dimension is to increase the number of algebraic relationships the entries of $\Delta C$ can satisfy and to accomplish the minimization faster. So the $CTLS$ solution can be formulated as follows:

$$\min_{w,x} \|w\|^2$$

subject to

$$(C + \Delta C) \begin{pmatrix} x \\ -1 \end{pmatrix} = 0.$$  

Next, we consider the quadratic constraint equation in (2.46) which is expanded into:

$$C \begin{pmatrix} x \\ -1 \end{pmatrix} + \Delta C \begin{pmatrix} x \\ -1 \end{pmatrix} = 0.$$
where

\[
\Delta C \left( \begin{array}{c} x \\ -1 \end{array} \right) = \left[ F_1 w \mid F_2 w \mid \cdots \mid F_p w \mid F_{p+1} w \right] \left( \begin{array}{c} x \\ -1 \end{array} \right)
\]

\[
= \sum_{i=1}^{p} x_i F_i w - F_{p+1} w = \left( \sum_{i=1}^{p} x_i F_i - F_{p+1} \right) w.
\] (2.48)

Let

\[
H_x = \sum_{i=1}^{p} x_i F_i - F_{p+1},
\] (2.49)

then we have

\[
C \left( \begin{array}{c} x \\ -1 \end{array} \right) + H_x w = 0 \quad \Rightarrow \quad w = -H_x^+ C \left( \begin{array}{c} x \\ -1 \end{array} \right)
\] (2.50)

where \(H_x^+\) is the pseudoinverse of \(H_x\) defined as

\[
H_x^+ = H_x^* \left( H_x H_x^* \right)^{-1}.
\] (2.51)

For any \(w\), the solution vector \(x\) can satisfy (2.45) and (2.50); i.e., we have

\[
\min_w \|w\|^2 = \left( \begin{array}{c} x \\ -1 \end{array} \right)^* C^* H_x^+ H_x^* C \left( \begin{array}{c} x \\ -1 \end{array} \right).
\] (2.52)

Next, let us define the analytic function \(J(x)\)

\[
J(x) = \left( \begin{array}{c} x \\ -1 \end{array} \right)^* C^* \left( H_x^+ \right)^* H_x^* C \left( \begin{array}{c} x \\ -1 \end{array} \right)
\] (2.53)
\[ x = (x_1, \ldots, x_p). \]

\( J(x) \) can be simplified to the following expressions by substituting (2.51) into (2.53).

\[
J(x) = \begin{pmatrix} x \end{pmatrix}^* C^* H_x (H_x H_x^*)^{-1} H_x^* (H_x H_x^*)^{-1} C \begin{pmatrix} x \end{pmatrix} \\
= \begin{pmatrix} x \end{pmatrix}^* C^* [(H_x H_x^*)^{-1} H_x H_x^*] (H_x H_x^*)^{-1} C \begin{pmatrix} x \end{pmatrix} \\
= \begin{pmatrix} x \end{pmatrix}^* C^* (H_x H_x^*)^{-1} C \begin{pmatrix} x \end{pmatrix}.
\]

\[ (2.54) \]

Hence, we conclude that the CTLS solution is obtained by minimizing \( J(x) \). From (2.45) and (2.52), we try to find out the minimum of \( J(x) \) with respect to \( x \). It is necessary to compute the first-order complex partial derivative or complex gradient and set the derivative to be zero; namely,

\[
\frac{\partial J}{\partial x} = \begin{pmatrix} \frac{\partial J}{\partial x_1} & \ldots & \frac{\partial J}{\partial x_p} \end{pmatrix} = 0.
\]

Due to the complex data, Abatzoglou et al [6] defined a complex version of the Newton method which uses the second-order complex partial derivative of \( J(x) \) to update the values of \( x \). The Newton iteration (recursion) is briefly described by

\[
x(n+1) = x(n) + (\overline{A}B^{-1}A - \overline{B})^{-1}(\overline{a} - \overline{A}B^{-1}a)
\]

\[ (2.56) \]
where

\[
a = \frac{\partial J^T}{\partial x} = \left( \frac{\partial J}{\partial x_1} \ldots \frac{\partial J}{\partial x_p} \right)^T = \text{complex gradient of } J
\]

\[
A = \frac{1}{2} \left( \frac{\partial^2 J}{\partial x^2} + \frac{\partial^2 J^T}{\partial x^2} \right) = \text{unconjugated complex Hessian of } J
\]

\[
B = \frac{\partial^2 J}{\partial x \partial x} = \text{conjugated complex Hessian of } J.
\]

Hence, by referring to the derivation in [6], we obtain the following closed-form solutions for \(a\), \(A\) and \(B\):

\[
a = (h \ast \tilde{B})^T
\]

\[
A = -\tilde{F}^* H_x'(H_x H_x^*)^{-1} \tilde{B} - (\tilde{F}^* H_x'(H_x H_x^*)^{-1} \tilde{B})^T
\]

\[
B = [\tilde{B}^*(H_x H_x^*)^{-1} \tilde{B}]^T + \tilde{F}^* [H_x'(H_x H_x^*)^{-1} H_x - I] \tilde{F}
\]

(2.58)

where

\[
h = (H_x H_x^*)^{-1} C \begin{pmatrix} x \\ -1 \end{pmatrix}
\]

\[
\tilde{B} = CI_{P+1,P} - [F_1 h^* \cdots |F_p h^*]
\]

\[
\tilde{G} = [F_1^* h^* \cdots |F_p^* h^*].
\]

(2.59)

The Newton method is implemented by starting with an Iterative Quadratic Maximum Likelihood (IQML) algorithm [9]. This algorithm solves a quadratic constraint minimization problem and derives the solution vector from the coefficients of the linear prediction polynomial which correspond to the parameters of the signals in a fast convergent way. We summarize its main steps below:
(a). initialization: set $x(0) = x_0 = 1$ at $n=0$, $x$ is viewed as the vector composed of the prediction polynomial coefficients and obtained using Newton iteration

$$x = (x_0, x_1, \ldots, x_p)^T$$  \hspace{1cm} (2.60)

(b). compute:

$$Z_Y = \sum_{j=1}^{N} y_j^H (X^H X)^{-1} y_j$$  \hspace{1cm} (2.61)

where

$$Y = \begin{bmatrix} y_p & \cdots & y_1 & y_0 \\ y_{p+1} & \cdots & y_2 & y_1 \\ \vdots & \ddots & \vdots & \vdots \\ y_{m-1} & \cdots & y_{m-p-2} & y_{m-p-1} \end{bmatrix}$$  \hspace{1cm} (2.62)

$y_i$ being the observations, and

$$X = \begin{bmatrix} x_p & \cdots & x_1 & x_0 \\ \vdots & \ddots & \vdots & \vdots \\ x_p & \cdots & x_1 & x_0 \end{bmatrix}^H$$  \hspace{1cm} (2.63)

c). solve the Quadratic Minimization Problem:

$$\min_{x_{(n+1)}} \left( x_{(n+1)}^H Z_Y^{(n)} x_{(n+1)} \right)$$  \hspace{1cm} (2.64)

d). set $n=n+1$ for real-time iteration

e). check the convergence: $\|x(n-1) - x(n)\| < \varepsilon$

(f). identify the roots of the prediction vector polynomial $x$ which represent the parameters of the received signal.
E. LINEARLY CONSTRAINT TOTAL LEAST SQUARES (LCTLS) ALGORITHM

1. Introduction

In this section we introduce a special case of the CTLS algorithm called the Linearily Constrained Total Least Squares (LCTLS) technique. The LCTLS technique is used to locate and resolve closely spaced frequencies in the presence of a given known frequency component [10].

Recall that the TLS method solves the unconstrained problem:

\[(A + \Delta A) x = b + \Delta b.\] (2.65)

Using (2.13) to (2.18), we compute the minimum norm of the total perturbation matrix \(\Delta C\) which makes \((C + \Delta C)\) rank deficient and generates a vector \(z\) defined from a null space. Equation (2.65) has a solution provided

\[(b + \Delta b) \in \text{Range} (A + \Delta A).\] (2.66)

In this case, \(z_{n+1} \neq 0\) and we can solve for \(x\). If \(z_{n+1} = 0\), the TLS problem has no solution. In addition, the non-zero last element \(z_{n+1}\) is used to normalize the vector \(z\); i.e., we have

\[- \begin{pmatrix} z \\ z_{n+1} \end{pmatrix} = \begin{pmatrix} x \\ -1 \end{pmatrix} \Rightarrow z = -z_{n+1} \begin{pmatrix} x \\ -1 \end{pmatrix}.\] (2.67)

2. Algorithm Derivation

To solve the LCTLS problem, we first look at the constraint equation:

\[\Psi x = \Gamma\] (2.68)
where $\Psi$, $x$ and $\Gamma$ have dimensions $p_x(n+1)$, $(n+1)x1$ and $p_x1$, respectively.

Equation (2.69) can also be expressed as below:

$$\begin{bmatrix} \Psi & \Gamma \end{bmatrix} \begin{bmatrix} x \\ -1 \end{bmatrix} = [ \Psi \Gamma ] z = 0. \quad (2.69)$$

Let us introduce a physical example to define the matrix $[ \Psi \Gamma ]$. First, we consider a signal $\zeta(n)$ composed of two complex sinusoidal signals in white noise:

$$\zeta(n) = A_1e^{j\omega_1} + A_2e^{j\omega_2} + w(n), \quad n = 1, \ldots, N \quad (2.70)$$

where $\zeta(n)$ is the observed noisy data; $w(n)$ is the white Gaussian noise; $A_i$ is the amplitude; $\omega_i$ is the angular frequency and $N$ is the data length.

Assume that $\omega_1$ is known and $\omega_2$ is unknown. Create a signal source vector $\lambda(\omega)$ defined as:

$$\lambda^H(\omega) = [1, e^{j\omega}, e^{j2\omega}, e^{j3\omega}, \ldots, e^{jN\omega}]. \quad (2.71)$$

We make $\lambda(\omega)$ satisfy the constraint equation in (2.69), which leads to

$$\lambda^H(\omega) \begin{bmatrix} x \\ -1 \end{bmatrix} = 0. \quad (2.72)$$

The constraint forces $e^{j\omega}$ to be one of the roots of the prediction-error polynomial $e(\omega)$ defined as:

$$e(\omega) = \lambda^H(\omega) \begin{bmatrix} x \\ -1 \end{bmatrix} = 0. \quad (2.73)$$
Then, the angle of the next closest root on the unit circle is chosen as the estimate of the unknown angular frequency $\omega_2$.

Next, considering the TLS equation given in (2.65) and constraint given in (2.69) leads to

\[
\begin{pmatrix}
\Psi & \Gamma \\
A + \Delta A & b + \Delta b
\end{pmatrix}
\begin{pmatrix}
x \\
-1
\end{pmatrix} = 0.
\]  

(2.74)

Next, using a SVD partition method to obtain the solution. To ensure the constraint equation in (2.75) to be satisfied, we have to constrain the vector $z$ to fall in the null space of $[ \Psi \Gamma ]$ which can be further factored out using QR decomposition into

\[
[ \Psi \Gamma ] = [ \Omega_1 0 ] \Lambda^H = \Omega_1 A^H_1
\]  

(2.75)

\[
\Lambda^H = \begin{pmatrix}
\Lambda^H_1 \\
\Lambda^H_2
\end{pmatrix}
\]  

(2.76)

where $\Omega_1$ is an upper triangular matrix with a dimension $(p \times p)$ and $\Lambda^H$ is a unitary matrix with a dimension $(n+1) \times (n+1)$. By comparing the matrix dimensions from (2.69) and (2.75), we see that $z$ should be the product of the orthonormal basis $A_2$ with a dimension $(n+1) \times (n+1-p)$ for the null space of $[ \Psi \Gamma ]$ and an arbitrary non-zero vector $\eta$ with a dimension $(n+1-p) \times 1$, i.e.,

\[
z = A_2 \eta.
\]  

(2.77)
Next, define the matrix $\tilde{C}$ as

$$\tilde{C} = C \Lambda_2 \Lambda_2^H$$

(2.78)

with

$$C = [A \mid b].$$

(2.79)

The SVD of the matrix $\tilde{C}$ is performed and $\tilde{C}$ is partitioned into its signal and noise subspaces, which leads to:

$$\tilde{C} = USV^H = \sum_{i=1}^{n+p} \sigma_i u_i v_i$$

(2.80)

or

$$\tilde{C} = \begin{bmatrix} \tilde{U}_1 & \tilde{U}_2 & \tilde{U}_3 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 & 0 \\ 0 & \Sigma_2 & 0 \\ 0 & 0 & \Sigma_3 \end{bmatrix} \begin{bmatrix} \tilde{V}_1^H \\ \tilde{V}_2^H \\ \tilde{V}_3^H \end{bmatrix}$$

(2.81)

with

$\tilde{V}_1$ : $r$-dimensional unconstrained part of the signal subspace
$\tilde{V}_2$ : $s$-dimensional noise subspace
$\tilde{V}_3$ : $p$-dimensional constrained part

$\tilde{\Sigma}_1 = \text{diag}(\tilde{\sigma}_1, \ldots, \tilde{\sigma}_r)$

$\tilde{\Sigma}_2 = \tilde{\sigma}_{n+1-p} \mathbf{I}_s, \quad s = n+1-p-r$

$\tilde{\Sigma}_3 = 0.$
Using the fact that

$$\Lambda_1 \Lambda_1^H + \Lambda_2 \Lambda_2^H = I, \quad (2.82)$$

we get

$$C = CA_1A_1^H + CA_2A_2^H. \quad (2.83)$$

Dowling et al [10] have suggested to enforce sphericity in the concerned partitioned subspace to improve the frequency spectral estimation performance. The term "sphericity" means that the singular values in the signal and noise spaces are replaced with their mean values. For example, assume that there exist (s = n+1-p-r) equal singular values in the noise subspace $\tilde{V}_2$; i.e., the noise singular values are replaced with the variance of the random noise. By partitioning the truncated unitary matrix $\tilde{V}$ below and doing the same operations as in (2.37) and (2.38)

$$\tilde{V} = \begin{bmatrix} \tilde{V}_1 & \tilde{V}_2 & \tilde{V}_3 \end{bmatrix} = \begin{bmatrix} V_1' & V_2' & V_3' \\ g^T & c^T & f^T \end{bmatrix}, \quad (2.84)$$

we obtain the minimum norm solution corresponding to the noise subspace:

$$x_{LCLS} = -\frac{V_2'c}{c^Hc}. \quad (2.85)$$
Further using the following identity

\[
\begin{bmatrix}
g^H & c^H & f^H \\
V_1 & V_2 & V_3
\end{bmatrix}
\begin{bmatrix}
g & V_1^H \\
c & V_2^H \\
f & V_3^H
\end{bmatrix} = I,
\] (2.86)

which leads to the following two equations:

\[
\begin{align*}
g^Hg + c^Hc + f^Hf &= 1 \\
\Rightarrow & \\
g^Hc &= 1 - g^Hg - f^Hf \\
V_1'g + V_2'c + V_3'f &= 0 \\
\Rightarrow & \\
V_2'c &= -V_1'g - V_3'f
\end{align*}
\] (2.87)

we obtain the minimum norm solution corresponding to the signal subspace:

\[
x_{\text{LCTLS}} = \frac{V_1'g^* + V_3'f^*}{1 - g^Hg - f^Hf}.
\] (2.88)

Note that the minimum LCTLS solution should be based on the subspace \( \tilde{V}_2 \) or \( \begin{bmatrix} \tilde{V}_1 & \tilde{V}_3 \end{bmatrix} \), whichever one has the smaller dimension.
III. TOTAL LEAST SQUARES TECHNIQUE IMPLEMENTATION

A. INTRODUCTION

First, we study the performance of the TLS algorithm to estimate the frequency of sinusoidal signals in noise. We assume the received signal $x(n)$ consists of $M$ uniformly spaced signal data samples corrupted by a complex white Gaussian noise $w(n)$. The received signal is expressed by the following equation and is observed over a duration of $N$ seconds, assuming the sampling frequency $f_s=1$ Hz:

$$x(n) = \sum_{k=1}^{M} a_k e^{j2\pi f_k n} + w(n), \quad n = 0, 1, \ldots, N-1 \quad (3.1)$$

where

- $\{a_k\}$: a set of signal amplitudes
- $\{f_k\}$: a set of true signal frequencies
- $\{x(n)\}$: measured complex-valued data samples
- $\{w(n)\}$: random complex-valued noise samples and uncorrelated with signals

Real and imaginary part of $w(n)$ are assumed to have a variance $\sigma^2$.

We are interested in estimating the signal frequency parameters $f_k$; the parameters $a_k$ are chosen to have unit magnitude for the sake of simplicity. Note that we use the singular value decomposition (SVD) technique to solve the linear prediction equation $Ax=b$ which matrix form is given in (2.4). The TLS implementation is expected to decrease the noise effects occurring both in
the data matrix $A$ and in the observation vector $b$ simultaneously. First we choose the order $P$ of the prediction-error filter and solve a $P$-degree prediction-error filter polynomial. Next, we determine the signal parameters from the $M$ zeros which are the closest to the unit circle. The other $P-M$ zeros are called extraneous zeros; they are approximately uniformly distributed inside the unit circle. Note that the choice of optimal prediction-error filter order is important since if the filter order is chosen too low, the closely-spaced sinusoids can not be resolved. In addition, if the filter order is chosen too high, the noise zeros may be mistaken for the true signal zeros as the noise zeros become very close to the unit circle. Thus, we have to select the "optimal" prediction filter order carefully.

B. ALGORITHM IMPLEMENTATION

The parameters used in the simulation are $\phi_1 = \frac{\pi}{4}$, $\phi_2 = 0$; $f_1$ and $f_2$ represent the frequency locations; $w(n)$ are independent identically distributed (i.i.d.) white complex Gaussian noise samples with zero mean and variance $\sigma^2$ for both real and imaginary parts. The signal-to-noise ratio (SNR) of the received noisy signal is given by the peak amplitude of sinusoids to noise level and defined as $10 \times \log\left(\frac{1}{2\sigma^2}\right)$ dB. The sampling frequency is set to be 1Hz. The frequency separation is chosen smaller than the reciprocal of the observation time in order to resolve these two close-together frequencies and raise the resolution. Simulations are performed using an ensemble of 30 independent trials. For each trial, a data block of 50 ($N=50$) data samples is used to resolve two ($M=2$) sinusoids. The frequency estimates are computed by
choosing the two roots of the prediction-error filter transfer function or the two zeros of the prediction-error filter polynomial which are the closest to the unit circle in the complex z-plane. Different values of the filter order P in the range of 2~49 can be used to observe the variations of estimation.

C. EXPERIMENTAL OBSERVATIONS

1. Performance of the Algorithm

We investigate the performance of the algorithm with a statistical quantity called the effective standard deviation (ESD) defined below:

\[
ESD = \sqrt{\frac{1}{T} \sum_{i=1}^{T} \left| \hat{f}_i - f \right|^2}
\]  

(3.2)

where

\[T : \text{the total number of independent trials}\]
\[\hat{f}_i : \text{the estimated frequency at the } i^{th} \text{ realization}\]
\[f : \text{the true signal frequency}\]

The ESD gives a measure of the distance between the true and the estimated frequency locations. In this simulation, we first use \(f_1=0.23\text{Hz}\) while \(f_2 = 0.27\text{Hz}\). The data including \(\phi_1=\frac{\pi}{4}, \phi_2=0, P=8, N=50\) and 30 independent trials are used to compute the ESD. The selection of the two signal zeros is made by creating two sector regions designated from the origin to point \(z_1=0.3+j*0.7\) and to point \(z_2= -0.3+j*0.7\), respectively. Varying the SNR from 20dB to 0dB, we observe that the sector region for \(f_1=0.23\text{Hz}\) at SNR=0dB needs to be expanded by changing \(z_1=0.4+j*0.7\) and the sector region for \(f_2=0.27\text{Hz}\) at SNR=1dB needs to be expanded by changing \(z_2=-0.4+j*0.8\). Here we display six different SNR
cases which are 20dB, 12dB, 5dB, 3dB, 1dB and 0dB. We also observe that the signal and noise zeros resolution becomes worse as the SNR becomes smaller, as shown in Figures 3.1.1-3.1.6. Figure 3.1.6 shows the worst resolution as a couple of noise zeros are on the unit circle and could potentially be considered as the signal zeros. In addition, Table 1 shows that as the SNR becomes smaller, the estimated deviations for both signal frequencies are getting larger. The above simulation results compare with those obtained in [5].

2. Effects of Signal Phase Changes

Two sets of phases are chosen to investigate the effects of phase changes: (1) $\phi_1=\frac{\pi}{4}, \phi_2=0$, (2) $\phi_1=\frac{\pi}{4}, \phi_2=-\frac{\pi}{4}$. The filter order is $P=8$; the number of data samples points is $N=50$; 30 independent trials are used; the signal frequencies are chosen as $f_1=0.25Hz, f_2=0.27Hz$; the SNR values considered are 3dB and 12dB. Figures 3.2.1-3.2.4 show that phase changes do not affect the resolution.

3. Effects of Filter Order Changes

Next, we consider the effect of changes in the prediction-error filter order. Recall that it is important to choose the smallest value of $P$ which achieves a good frequency estimation performance. For this experiment, we choose $f_1=0.25Hz, f_2=0.27Hz$, SNR=12dB, $N=50$, and 10 independent trials are used. Figures 3.3.1 to 3.3.6 show the effects of changing the filter order. When $P=2$, we have very poor resolution, as shown in Figure 3.3.1. Performance slightly improves when $P=4$, as shown in Figure 3.3.2. Note that increasing the filter order $P$ improves the resolution of the estimated frequencies. For example, when choosing $P=8$ and $P=18$, we observe that two sinusoidal signals
are located exactly at their desired positions on the unit circle and the extraneous zeros are uniformly distributed inside the unit circle, as shown in Figures 3.3.3 & 3.3.4. However, if $P$ is chosen too large, $P=28$ or $P=38$ for example, we observe some of the $P-M$ extraneous zeros of the prediction polynomial are getting very close to the unit circle which make the distinction between signal and noise zeros difficult to make, as shown in Figures 3.3.5 & 3.3.6.

4. Effects of Errors in the Subspace Partition Estimate

It is possible to express the TLS solution in the sense of minimum norm. By applying the SVD technique to the linear prediction equation, we can partition the unitary matrix $V$ (for handling complex data) into two subspaces: the signal subspace and the noise subspace. If the partition is performed correctly, we can obtain a good zeros resolution by using a moderate filter order $P$ and high enough SNR, choosing $P=8$ and SNR=$12$dB for example, as shown in Figure 3.3.3. However, if the partition is not performed correctly; i.e., the numbers of column vector of the signal subspace and the noise subspace are not estimated properly, we observe that the extraneous zeros of the prediction polynomial are randomly distributed. The more the signal and the noise subspaces are incorrectly chosen, i.e., the worse the partition is, the worse the resolution to identify the true sinusoidal signal locations becomes, as shown in Figures 3.4.1 & 3.4.2. Figure 3.4.2 displays the worst degradation of the performance since these two subspaces are mixed up by 6 column vectors under the condition of SNR=$12$dB, $P=8$, $N=50$ and 10 independent trials.
5. Application to Damped Signals

In the following, we consider the case of a sum of two (M=2) damped exponential sinusoids described as follows:

\[ x(n) = e^{-\alpha_1 n + j(2\pi f_1 n + \phi_1)} + e^{-\alpha_2 n + j(2\pi f_2 n + \phi_2)} + w(n), \]

\[ n = 0, 1, \ldots, N-1 \]

where angular frequencies are \( f_1 = 0.25 \text{Hz}, f_2 = 0.27 \text{Hz} \); the signal phases are fixed at \( \phi_1 = \frac{\pi}{4}, \phi_2 = 0 \); the two damping factors are chosen \( \alpha_1 = 0.8, \alpha_2 = 0.9 \); the number of data sample is \( N = 50 \); the filter order is \( P = 8 \); 30 independent trials are used for the simulation, and \( w(n) \) is still defined as the complex white Gaussian noise. We observe that it is quite difficult to identify the exact locations of these two damped sinusoidal signals at low SNR, for example 5dB, or at high SNR, for example 20dB, as shown in Figures 3.5.1 & 3.5.2. Even if we increase the filter order up to \( P = 18 \), the resolution has not been improved still, as shown in Figures 3.5.3 & 3.5.4.
IV. RECURSIVE TLS TECHNIQUE
IMPLEMENTATION

A. THEORETICAL BACKGROUND

1. Introduction

As discussed before, the Total Least Squares (TLS) algorithm has been shown to be more accurate than the Least Squares (LS) algorithm when the errors or noise perturbations exist on both sides of a linear prediction equation $Ax=b$. We now introduce an adaptive algorithm called the Recursive Total Least Squares (RTLS) algorithm. In this algorithm, we perform the updates by tracking the smaller one of the signal subspace or the noise subspace partitioned from a unitary matrix containing complex data. It is shown that the RTLS algorithm works better than the Recursive Least Squares (RLS) algorithm [4, p. 480] and the Least Mean Squares (LMS) algorithm [4, p. 302] in terms of tracking property as well as steady-state tap-weight error norms [11]. Furthermore, the updates can also be done using a computationally efficient non-iterative subspace updating technique [12].

2. Algorithm Derivation

First, let us give a brief review of an adaptive process. An adaptive process involves the automatical adjustment of a set of tap weights of a prediction-error filter; i.e., it obtains an estimate of a desired response which results from the inner product of a set of tap input and its corresponding set of tap weights produced by the adaptive filter. Then, it compares this obtained estimate with the actual values of the desired response and generates an
estimation error. The estimation error is fed back to the adaptive process and stimulates the prediction filter in a recursive way. Therefore, this is a closed-loop operation and the final goal of the adaptive process is to make the output error reach its minimum value. There are two good examples: the LMS algorithm and the RLS algorithm.

In the TLS algorithm, we apply the *singular value decomposition (SVD)* technique to the augmented matrix spanned by the data matrix $A$ and the observation vector $b$. However, in the RTLS algorithm, we apply an *eigen-decomposition* technique to a specific augmented matrix $C(n)$ defined below to get the desired subspaces and determine the effective rank of $C(n)$ from its singular values. In addition, we adopt a parameter $\alpha$, called the *exponential weighting factor*, which is similar to $\lambda$ (*forgetting factor*) in the RLS algorithm.

$$C(n) = D_\alpha^{(n)} \begin{bmatrix} A & b \end{bmatrix}$$

(4.1)

where

$$D_\alpha^{(n)} = \sqrt{\text{diag}(\alpha^n, \ldots, \alpha^0)}.$$  

(4.2)

It is worthy to investigate the impact of the *forgetting weight factor* $\alpha$ on the tracking capability. The factor $\alpha$ is used to ensure that the data samples in the distant past are "forgotten" so as to afford the possibility of following the statistical variations of the observable data when the filter operates in a non-stationary environment. It is designated to be a positive constant close to, but smaller than one. Further speaking, the reciprocal of $1-\alpha$ can be considered as a measure of "memory". When $\alpha$ is equal to 1, which corresponds to infinite
memory or most unforgettable case, the algorithm keeps all data samples and the estimated frequencies have low variance. On the other hand, when \( \alpha \) is approaching to 0, which represents the most forgetful case, the algorithm keeps only a few data samples and the estimated frequencies have high variance.

As mentioned above, the RTLS algorithm applies the eigen-decomposition technique to the newly augmented matrix \( C(n) \). The estimated time-varying correlation matrix \( R(n) \) is expressed as

\[
R(n) = \alpha R_{(n-1)} + (1 - \alpha)y(n)y^H(n)
\]  \hspace{1cm} (4.3)

where \( R(n) \) is defined as

\[
R(n) = C^H(n)C(n).
\]  \hspace{1cm} (4.4)

In this tracking (updating) method, we use the rank-one eigenstructure update recursively. In order to diminish the computational load, we use the sphericity property proposed by DeGroat [12]. Spherical updating forces both of the eigenvalues in the signal subspace and the noise subspace to be replaced with their mean values. Thus, we form two different spherical subspaces without changing the dimension of the diagonal matrix \( D \) of \( R(n) \). The spherical subspace updating might be viewed as the nucleus of the RTLS adaptive filter algorithm. We implement the eigendecomposition of the one-step past-time correlation matrix \( R(n-1) \) to get the updated decomposition at time \( n \). Thus, we have the following updating

\[
R_{(n-1)} = U_{(n-1)}D_{(n-1)}U^H_{(n-1)}
\]  \hspace{1cm} (4.5)
\[ R_{(n)} = U_{(n)}D_{(n)}U_{(n)}^H \]  

(4.6)

where for all present time \( n \)

\[ U_{(n)}^H U_{(n)} = U_{(n)} U_{(n)}^H = I \]  

(4.7)

and \( D(n) \) is still a diagonal matrix consisting of the eigenvalues.

This updating algorithm is developed to track the smaller averaged eigenspace of the exponentially weighted correlation matrix \( R(n) \) under the condition of forced sphericity of the eigenstructure. To determine the right singular subspace of \( C(n) \), we need to obtain the eigenspaces of \( R(n) \) corresponding to the right singular subspaces of \( C(n) \). For later simulation, we get the vector \( y(n) \) extracted from the last row of matrix \( Aib \). Next, for the purpose of simplifying the derivation of the subspace updating, we neglect the factor \((1 - \alpha)\) and adopt a new time-varying correlation matrix \( \tilde{R} \) for the rank-one updating recursion

\[ \tilde{R}_{(n)} = \alpha R_{(n)} + y_{(n)}y_{(n)}^H. \]  

(4.8)

By substituting (4.6) into (4.8) and letting

\[ \chi = U^H y, \]  

(4.9)

we further get

\[ y = U\chi. \]  

(4.10)
and derive the following form

$$\tilde{R} = U(\alpha D + XX^H)U^H. \quad (4.11)$$

Since both of the RTLS algorithm and the TLS algorithm use the SVD-based partition method, they have the same solution for the signal subspace and the noise subspace, respectively. Namely, we can rewrite (2.36) and (2.39) from Chapter II below:

for the signal subspace, the solution vector is

$$x_{RTLS} = \frac{V_g^*}{1 - g^*g}, \quad (4.12)$$

for the noise subspace, the solution vector is

$$x_{RTLS} = \frac{V'_c^*}{c^H c}. \quad (4.13)$$

In simulations considered later in this chapter, we use two signal frequencies: one frequency is fixed and the other one is linear time-varying, so the smaller subspace is the signal subspace. We assume that these two complex sinusoidal signals have frequency components $f_2=0.27\text{Hz}$ and $f_1$ moving around $0.23\text{Hz}$ with a moving weight of 3% and 10%; the phase components are $\phi_1=\frac{\pi}{4}$, $\phi_2=0$; 10 independent trials and 10 recursive realizations are used in this simulation. Note that we have four parameters to vary: the filter order $P$, the exponential weighting factor $\alpha$, the signal-to-noise ratio (SNR) and the moving weight of the time-varying signal frequency.
3. Simulations

First, we apply the RTLS algorithm and use the spherical subspace updating technique. Next, we compare the performance of the RTLS algorithm with spherical updating with that of the RTLS algorithm without spherical updating. A data length of 100 points and a window length of 50 points are used in the following simulations. In addition, we assign the forgetting factor $\alpha$ to be one of the two values: 0.25 and 0.95. Next, we adopt one of the two moving frequency weights: 3% and 10%. We would also like to see when the resolution of signal zeros breakdowns by selecting various filter orders. Finally, by fixing $\alpha=0.95$ and the moving weight=10%, we decrease SNR to 0dB and -5dB so as to observe the influence of low SNR on the resolution. The cases studied are listed as follows:

a. Case (1)
   Fix SNR=12dB, $P=8$, moving weight=3%
   Change $\alpha=0.25$, 0.95.

b. Case (2)
   Fix SNR=12dB, $P=8$, moving weight=10%
   Use $\alpha=0.95$

c. Case (3)
   Fix SNR=12dB, $\alpha=0.95$, moving weight=10%
   Choose $P=3$, 5.

d. Case (4)
   Fix $P=8$, $\alpha=0.95$, moving weight=10%
   Decrease SNR=0dB, -5dB.
B. EXPERIMENTAL OBSERVATIONS

1. Estimated Signal Subspace Obtained with Averaging the Eigenvalues

The signal subspace information is used to estimate the two signal frequencies. By using the data in case (1), for $\alpha=0.25$, we observe that the locations of the zeros are randomly distributed. It is very difficult to distinguish signal zeros from noise zeros, as shown in Figure 4.1.1. The averaged eigenvalues of the spherical updated matrix are presented in Figure 4.1.2. If $\alpha$ is chosen much higher, say $\alpha=0.95$, we observe that the resolution becomes very good and we can discern the fixed signal and the moving signal. It is noted that the noise zeros just congregate forming a small cluster and are distributed inside the unit circle, as shown in Figures 4.1.3 and 4.1.4. Hence, the experiments show that the case of $\alpha=0.95$ has much better resolution since it keeps more data samples as shown in Figure 4.2.1. Next, we observe that it is quite obvious to distinguish the fixed frequency (a slight shift) and the time-varying frequency (countable small circles) using the data given in case (2). The averaged eigenvalues are also displayed in Figure 4.2.2. Thus, results show that the case with larger moving weight and higher forgetting factor (approaching to one) has the best zeros resolution and tracking capability. Further speaking, the forgetting factor plays a more important role than the moving weight. By using the data in case (3), we observe that in the case of $P=5$ the "fixed" signal estimate shows some some variations. However, we are still able to differentiate signal zeros from noise zeros, as shown in Figure 4.3.1. The performance gets worse when we use $P=3$, as shown in Figure 4.3.2. Usually the filter order number ($P$) is chosen larger than the true signal
number \( M \)), so we do not consider the case \( P=M \) here. Using the data in case (4), we observe that the resolution becomes worse at very low SNR since some noise zeros are very close to or on the unit circle and can be mistaken for true signal zeros, as shown in Figures 4.4.1 & 4.4.2.

2. Estimated Signal Subspace Obtained without Averaging Eigenvalues

Next, we consider the estimated correlation matrix \( R(n) \) obtained without using averaged eigenvalues. Using the data in case (2), we obtain very good resolution of the fixed frequency and the time-varying frequency, as shown in Figure 4.5.1. The non-averaged eigenvalues of diagonal elements of the correlation matrix are displayed in Figure 4.5.2. Next, using the data in case (4) shows that the resolution becomes worse with very lower SNR, as shown in Figure 4.6 to be compared with Figure 4.4. Besides, from the plot of non-averaged eigenvalues, we see the signal eigenvalues are still much higher than the noise eigenvalues as shown in Figures 4.6.2 & 4.6.4.

3. Estimated Noise Subspace Obtained with Averaging the Eigenvalues

Here, we use the spherical subspace updating method and choose the noise subspace to estimate the frequencies. The data are generated using the characteristics of case (2) given in section 3. The results are quite different from those obtained when choosing the signal subspace. Some of the noise zeros appear on the unit circle and can potentially be mistaken for signal zeros, as shown in Figure 4.7 to be compared with Figure 4.2. Next, using the data defined in case (3) earlier, we still obtain poor resolution, as shown in Figure 4.8 to be compared with Figure 4.3. Next, we investigate the effects of the SNR on
the resolution. Since the resolution of SNR=12dB plotted in Figure 4.7.1 is not good, we raise the SNR to 15dB and 30dB. We observe that the resolution does not improve, as shown in Figures 4.9.1 & 4.9.2. Therefore, we conclude that it is much better and more efficient to adopt the signal subspace in tracking two closely-spaced signal sources (targets) instead of adopting the noise subspace, when using the spherical property.
V. CONCLUSIONS

In this thesis, we first introduce the Least Squares (LS) algorithm which only considers the errors or perturbations in the observation vector \( b \) of a linear prediction equation \( Ax=b \). Since the assumption is not realistic, we present the Total Least Squares (TLS) algorithm which considers the errors or perturbations in the observation vector \( b \) and the data matrix \( A \). When there exists some linear dependence algebraical relationship among the correlated noise perturbation components in \( A \) and \( b \), we need to modify the TLS algorithm by taking into the dependence account, which leads to the Constrained Total Least Squares (CTLS) algorithm. Next, we present the Linear Constraint Total Least Squares (LCTLS) algorithm. The LCTLS can be viewed as a special case of the CTLS algorithm and is used to resolve closely-spaced frequencies with a given known frequency component. Then, we present numerical implementations of the TLS algorithm. We investigate the performances of the TLS algorithm to resolve two damped / undamped complex sinusoidal signals in additive white noise. Next, we present a numerical implementation of the Recursive Total Least Squares (RTLS) algorithm to locate a fixed signal frequency and to track a time-varying signal frequency. The RTLS algorithm uses an eigendecomposition technique instead of a singular value decomposition technique. We apply a spherical subspace updating method in order to efficiently reduce the computational load and compare the performances of the RTLS with and without spherical updating.

From the simulation, we observe the case with larger moving weight and higher exponential weighting factor (forgetting factor) has much better
resolution and tracking capability, when the spherical signal subspace is used. Furthermore, we show that the exponential weighting factor plays a more important role than the moving weight. In addition, we see that the resolution becomes worse at very low SNR because some noise zeros are approaching to the unit circle and can be potentially mistaken for signal zeros. When we use the signal subspace without using averaged eigenvalues, we observe that the results do not have any significant difference from the case using averaged eigenvalues. However, if we use the noise subspace with spherical averaged eigenvalues, poor resolution is obtained. Therefore, it is better to adopt the signal subspace for resolving two closely-spaced signal sources.
<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>Estimated Deviation for $f_1=0.23$Hz</th>
<th>Estimated Deviation for $f_2=0.27$Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.0020</td>
<td>8.5083e-04</td>
</tr>
<tr>
<td>18</td>
<td>0.0030</td>
<td>9.1698e-04</td>
</tr>
<tr>
<td>16</td>
<td>0.0049</td>
<td>0.0046</td>
</tr>
<tr>
<td>14</td>
<td>0.0047</td>
<td>0.0033</td>
</tr>
<tr>
<td>12</td>
<td>0.0097</td>
<td>0.0073</td>
</tr>
<tr>
<td>10</td>
<td>0.0058</td>
<td>0.0065</td>
</tr>
<tr>
<td>8</td>
<td>0.0054</td>
<td>0.0203</td>
</tr>
<tr>
<td>7</td>
<td>0.0189</td>
<td>0.0133</td>
</tr>
<tr>
<td>6</td>
<td>0.0198</td>
<td>0.0164</td>
</tr>
<tr>
<td>5</td>
<td>0.0073</td>
<td>0.0166</td>
</tr>
<tr>
<td>4</td>
<td>0.0217</td>
<td>0.0388</td>
</tr>
<tr>
<td>3</td>
<td>0.0106</td>
<td>0.0461</td>
</tr>
<tr>
<td>2</td>
<td>0.0231</td>
<td>0.0333</td>
</tr>
<tr>
<td>1</td>
<td>0.0325</td>
<td>0.0108</td>
</tr>
<tr>
<td>0</td>
<td>0.0684</td>
<td>0.1816</td>
</tr>
</tbody>
</table>
Figure 3.1.1 TLS solution for SNR=20dB, P=8, N=50, f1=0.23, f2=0.27, undamped sinusoids

Figure 3.1.2 TLS solution for SNR=12dB, P=8, N=50, f1=0.23, f2=0.27, undamped sinusoids
Figure 3.1.3  TLS solution for SNR=5dB, P=8, \(N=50, f_1=0.23, f_2=0.27\), undamped sinusoids

Figure 3.1.4  TLS solution for SNR=3dB, P=8, \(N=50, f_1=0.23, f_2=0.27\), undamped sinusoids
Figure 3.1.5 TLS solution for SNR=1dB, P=8
N=50, f1=0.23, f2=0.27, undamped sinusoids

Figure 3.1.6 TLS solution for SNR=0dB, P=8
N=50, f1=0.23, f2=0.27, undamped sinusoids
Figure 3.2.1 TLS solution for SNR=3dB, P=8
N=50, f1=0.25, f2=0.27, undamped sinusoids

Figure 3.2.2 TLS solution for SNR=12dB, P=8
N=50, f1=0.25, f2=0.27, undamped sinusoids
Figure 3.2.3 TLS solution for SNR=3dB, P=8, N=50
f_1=0.25, f_2=0.27, \phi_1=\pi/4, \phi_2=-\pi/4, undamped sinusoids

Figure 3.2.4 TLS solution for SNR=12dB, P=8, N=50
f_1=0.25, f_2=0.27, \phi_1=\pi/4, \phi_2=-\pi/4, undamped sinusoids
Figure 3.3.1 TLS solution for SNR=12dB, P=2
N=50, f1=0.25, f2=0.27, undamped sinusoids

Figure 3.3.2 TLS solution for SNR=12dB, P=4
N=50, f1=0.25, f2=0.27, undamped sinusoids
Figure 3.3.3 TLS solution for SNR=12dB, P=8
N=50, f1=0.25, f2=0.27, undamped sinusoids

Figure 3.3.4 TLS solution for SNR=12dB, P=18
N=50, f1=0.25, f2=0.27, undamped sinusoids
Figure 3.3.5 TLS solution for SNR=12dB, P=28
N=50, f1=0.25, f2=0.27, undamped sinusoids

Figure 3.3.6 TLS solution for SNR=12dB, P=38
N=50, f1=0.25, f2=0.27, undamped sinusoids
Figure 3.4.1 TLS solution for SNR=12dB, P=8
N=50, f1=0.25, f2=0.27, undamped sinusoids
signal and noise subspaces mixed by 3 columns

Figure 3.4.2 TLS solution for SNR=12dB, P=8
N=50, f1=0.25, f2=0.27, undamped sinusoids
signal and noise subspaces mixed by 6 columns
Figure 3.5.1 TLS solution for SNR=5dB, P=8
N=50, f1=0.25, f2=0.27, damped sinusoids

Figure 3.5.2 TLS solution for SNR=20dB, P=8
N=50, f1=0.25, f2=0.27, damped sinusoids
Figure 3.5.3 TLS solution for SNR=5dB, P=18
N=50, f1=0.25, f2=0.27, damped sinusoids

Figure 3.5.4 TLS solution for SNR=20dB, P=18
N=50, f1=0.25, f2=0.27, damped sinusoids
1.5

alpha=0.25, phi1=pi/4, phi2=0
weight=0.03

Figure 4.1.1 RTLS solution for SNR=12dB, P=8, N=100
\( \alpha=0.25, \text{ weight}=3\%, f_1 \text{ is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'} \)

\begin{figure}
\includegraphics[width=\textwidth]{figure4.1.1}
\caption{RTLS solution for SNR=12dB, P=8, N=100, \( \alpha=0.25, \text{ weight}=3\%, f_1 \text{ is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'} \)}
\end{figure}

35

Figure 4.1.2 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, \( \alpha=0.25, \text{ weight}=3\%, f_1 \text{ is time-varying solution derived by using the signal subspace information} \)

\begin{figure}
\includegraphics[width=\textwidth]{figure4.1.2}
\caption{Averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, \( \alpha=0.25, \text{ weight}=3\%, f_1 \text{ is time-varying solution derived by using the signal subspace information} \)}
\end{figure}
1.5

$\alpha = 0.95, \phi_1 = \pi/4, \phi_2 = 0$

weight = 0.03

$f_1$ moving, $f_2 = 0.27$

-1.5 -1 -0.5 0 0.5 1 1.5

10 independent trials
10 recursive realizations

Figure 4.1.3 RTLS solution for SNR=12dB, P=8, N=100
$\alpha = 0.95$, weight=3%, $f_1$ is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'

Figure 4.1.4 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, $\alpha = 0.95$, weight=3%, $f_1$ is time-varying solution derived by using the signal subspace information
1.5
alpha=0.95 phi1=\pi/4, phi2=0
weight=0.10 f1 moving, f2=0.27

Figure 4.2.1 RTLS solution for SNR=12dB, P=8, N=100
\alpha=0.95, weight=10\%, f_1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'S'

350
300
250
200
150
100
50
0

Figure 4.2.2 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, \alpha=0.95, weight=10\%, f_1 is time-varying
solution derived by using the signal subspace information

57
Figure 4.3.1 RTLS solution for SNR=12dB, P=5, N=100
\( \alpha=0.95 \), weight=10\%, f1 is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'

Figure 4.3.2 RTLS solution for SNR=12dB, P=3, N=100
\( \alpha=0.95 \), weight=10\%, f1 is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'
Figure 4.4.1 RTLS solution for SNR=0dB, P=8, N=100
\( \alpha=0.95 \), weight=10%, \( f_1 \) is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'.

Figure 4.4.2 RTLS solution for SNR=-5dB, P=8, N=100
\( \alpha=0.95 \), weight=10%, \( f_1 \) is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'.
1. alpha = 0.95, weight = 0.10, f1 moving, f2 = 0.27

Figure 4.5.1 RTLS solution for SNR = 12dB, P = 8, N = 100
\( \alpha = 0.95 \), weight = 10\%, f1 is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'R'.

Figure 4.5.2 non-averaged eigenvalues of correlation matrix 'R'
SNR = 12dB, P = 8, N = 100, \( \alpha = 0.95 \), weight = 10\%, f1 is time-varying solution derived by using the signal subspace information.
Figure 4.6.1 RTLS solution for SNR=0dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'R'.

Figure 4.6.2 non-averaged eigenvalues of correlation matrix 'R'
SNR=0dB, P=8, N=100, α=0.95, weight=10%, f1 is time-varying solution derived by using the signal subspace information.
Figure 4.6.3 RTLS solution for SNR=-5dB,P=8,N=100
$\alpha=0.95$, weight=10%, $f_1$ is time-varying, using the signal subspace information & performing the eigendecomposition of spherical matrix 'R'.

Figure 4.6.4 non-averaged eigenvalues of correlation matrix 'R'
SNR=-5dB,P=8,N=100,$\alpha=0.95$,weight=10%,f1 is time-varying solution derived by using the signal subspace information
Figure 4.7.1 RTLS solution for SNR=12dB, P=8, N=100
\[\alpha=0.95, \text{weight}=10\%, f_1 \text{ is time-varying, using the noise subspace information & performing the eigendecomposition of spherical matrix } 'S' \]

Figure 4.7.2 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, \[\alpha=0.95, \text{weight}=10\%, f_1 \text{ is time-varying solution derived by using the noise subspace information} \]
Figure 4.8.1 RTLS solution for SNR=12dB, P=5, N=100
\( \alpha=0.95 \), weight=10\%, \( f_1 \) is time-varying, using the noise subspace information & performing the eigendecomposition of spherical matrix 'S'.

Figure 4.8.2 RTLS solution for SNR=12dB, P=3, N=100
\( \alpha=0.95 \), weight=10\%, \( f_1 \) is time-varying, using the noise subspace information & performing the eigendecomposition of spherical matrix 'S'.
Figure 4.9.1 RTLS solution for SNR=15dB, P=8, N=100
\( \alpha=0.95 \), weight=10\%, \( f_1 \) is time-varying, using
the noise subspace information & performing
the eigendecomposition of spherical matrix 'S'.

Figure 4.9.2 RTLS solution for SNR=30dB, P=8, N=100
\( \alpha=0.95 \), weight=10\%, \( f_1 \) is time-varying, using
the noise subspace information & performing
the eigendecomposition of spherical matrix 'S'.

65
APPENDIX A: MATLAB SOURCE CODE

This appendix consists of MATLAB codes used to implement the Total Least Squares (TLS) algorithm mentioned in Chapter III and the Recursive Total Least Squares (RTLS) algorithm mentioned in Chapter IV.

1. IMPLEMENTATION OF THE TLS ALGORITHM

```matlab
clear
%initialize the set of roots
rt=[];
nn=input(' trial number ');
SNR=input(' S/N(dB) = ');
p=input(' prediction filter order ');
y1=zeros(p,1);y2=zeros(p,1);
%construct the received signal and begin the trial
for h=1:nn,
    n=0:49;
    rand('normal');
    w1=rand(1,50);
    rand('normal');
    w2=rand(1,50);
    f1=0.25;                %signal frequencies
    f2=0.27;                %f1=0.23 for option
    phi1=pi/4;              %signal phase angles
    phi2=0;
```

```matlab
end
```

66
alpha1=0.8; %decaying constants
alpha2=0.9;
j=sqrt(-1);
wn=w1+j*w2; %independent complex noise
s1=exp(j*(2*pi*f1*n+phi1));
s2=exp(j*(2*pi*f2*n+phi2));
s1=exp(-alpha1*n+j*(2*pi*f1*n+phi1));
s2=exp(-alpha2*n+j*(2*pi*f2*n+phi2));
ss=s1+s2;

xn=ss+sqrt(1/(2*10^((SNR/10))).*wn; %received resulted signal
nsin=2; %for two sinusoidal sources
n=50; %window data sample number
if n>p+2,
    for k=1:n-p,
        A(k,1:p)=xn(k:p+k-1); %generate data matrix
    end
    b=xn(1,p+1:n);
    bb=conj(b)';
end
C=[bb A]; %create augmented matrix
[u,s,v]=svd(C); %perform singular value deom.
sigma=diag(s);
ct=v(1,nsin+1:p+1);
c=conj(ct)';
vp=v(2:p+1,nsin+1:p+1);
v2=[ct;vp];
xtls=-(vp*conj(c))/(conj(ct)*c); %solution from noise subspace
z=[1 -fliplr(conj(xtls))]; %prediction filter polynomial
rt=[rt roots(z)];
z1=0.3+j*0.7; %create sector regions vs. origin
z2=-0.3+j*0.7;
real_rt=real(rt);
for n1=1:p, %calculate estimated deviations
    if real_rt(n1,h)>0,
        y1(n1,h)=rt(n1,h);
        y11=sort(y1);
        y11=flipud(y11);
        new_y11(1:p/2,h)=y11(1:p/2,h);
        k1=find((angle(new_y11(1:p/2,h)) > angle(z1))... & (angle(new_y11(1:p/2,h)) < pi/2));
        if length(k1)==1,
            error('More than 1 element of new_y11 is nonzero.');
        end
        esti_y11(n1,h)=new_y11(k1,h);
        estd1=sqrt(1/nn*(sum(abs(esti_y11(n1,h)... -0.1253+j*0.9921) *ones(1,nn).^2)));
    else
        y2(n1,h)=rt(n1,h);
        y22=sort(y2);
        y22=flipud(y22);
    end
end
new_y22(1:p/2,h)=y22(1:p/2,h);
k2=find((angle(new_y22(1:p/2,h)) > pi/2)...
    & (angle(new_y22(1:p/2,h)) < angle(z2)));
if length(k2)==1,
    error('More than 1 element of new_y22 is nonzero.');
end
esti_y22(1,h)=new_y22(k2,h);
estd2=sqrt(1/nn*(sum(abs(est_i_y22(1,h)...
    -(-0.1253+j*0.9921).*ones(1,nn)).^2)));
end
end
end

%plot the zeros distribution situations
axis('square')
axis([-1.5 1.5 -1.5 1.5])
plot(sin(0:2*pi/100:2*pi),cos(0:2*pi/100:2*pi),'-w',...  
real(new_y22),imag(new_y22),'o',real(new_y22),imag(new_y22),'o');
grid
title('TLS solution for SNR= , P= , N= ');
text(-1.1,1.3,' alpha1= ');
text(-1.1,1.1,' alpha2= ');
text(0,1.3,' f1= , f2= ');
text(0,1.1,' phi1= , phi2= ');
text(0,-1.3,' 30 independent trials ');
text(-1,-1.4,' destroyed signal subspace by # columns ');

69
%REMARKS:

%(1). When SNR down to 1 dB, we need change $z_2$ to $-0.4+j*0.8$
% in order to make $k_2$ for new $y_{22}$ satisfy 'find' condition.

%(2). For SNR down to 0 dB, we have to change $z_1$ to $0.4+j*0.7$
% in order to make $k_1$ for new $y_{11}$ satisfy 'find' condition.
2. IMPLEMENTATION OF THE RTLS ALGORITHM USING THE SIGNAL SUBSPACE AND PERFORMING THE EIGEN-DECOMPOSITION OF MATRIX 'S'

Note that $S = \alpha \Delta + (1-\alpha) \beta \beta^H$  \hspace{1cm} $\beta = U^H \alpha$

clear
% initialization of conditions
rt=[]; esp=[]; sig=[]; noi=[];
nt=input(' trial number= ');
ni=input(' iteration number= ');
p = input(' prediction order= ');
SNR = input(' signal-to-noise ratio, S/N(dB)= ');
weight = input(' the weight of moving frequency= ');
alpha = input(' exponential weighting factor, alpha= ');
% construct received signal and begin independent trial
for h=1:nt,
    n1=1:100;
    rand('normal');
    w1=rand(1,100);
    rand('normal');
    w2=rand(1,100);
    f1=0.23-(h-1)*weight/nt; \hspace{1cm} % make f1 time-varying moving
    f2=0.27;
    phi1=pi/4;
    phi2=0;
\( j = \sqrt{-1}; \)
\( \text{wn} = \text{w1} + j \times \text{w2}; \)
\( \text{s1} = \exp(j \times (2 \times \pi \times f_1 \times n_1 + \phi_1)); \)
\( \text{s2} = \exp(j \times (2 \times \pi \times f_2 \times n_1 + \phi_2)); \)
\( \text{ss} = \text{s1} + \text{s2}; \)
\( \text{xn} = \text{ss} + \sqrt{\frac{1}{2 \times 10^{\text{SNR}/10}}} \times \text{wn}; \)
\( \text{nsin} = 2; \)
\( \text{n2} = 50; \)
\( \text{if n2} > \text{p} + 2, \)
\( \text{for k = 1 : n2} - \text{p}, \)
\( \text{A(k, 1 : p)} = \text{xn(k : p + k - 1);} \)
\( \text{end} \)
\( \text{b} = \text{xn(1, p + 1 : n2)}; \)
\( \text{bb} = \text{conj(b)';} \)
\( \text{end} \)
\( \text{B} = [\text{bb A}]; \)
\( \text{aph} = \text{alpha} \times n_1; \)
\( \text{aph} = \text{fliplr(aph(1 : n2 - p))}; \)
\( \text{T} = \text{sqrt(diag(aph))}; \)
\( \text{C} = \text{T} \times \text{B}; \)
\( \text{R} = \text{C} \times \text{C}; \)
\( \text{[U, D]} = \text{eig}(\text{R}); \)
\( \text{for t} = 1 : \text{ni}, \)
\( \text{y} = \text{xn(1, n2 - p + t : n2 + t)}; \)
\( \text{x} = \text{y'}; \)
\[
\begin{align*}
\beta &= U'x; \\
D_1 &= \text{sort}((\text{diag}(D))); \\
D_2 &= \text{flipud}(D_1); \\
D_2 &= D_2'; \\
s_{\text{eigval}} &= \text{mean}(D_2(1:nsin)); \\
n_{\text{eigval}} &= \text{mean}(D_2(nsin+1:p+1)); \\
s_{\text{eigspa}} &= s_{\text{eigval}} \cdot \text{eye}(nsin); \\
n_{\text{eigspa}} &= n_{\text{eigval}} \cdot \text{eye}(p-nsin+1); \\
s_{\text{diaspa}} &= \text{diag}(s_{\text{eigspa}}); \\
n_{\text{diaspa}} &= \text{diag}(n_{\text{eigspa}}); \\
D_{\text{hat1}} &= [s_{\text{eigspa}} \text{zeros}(nsin,p-nsin+1)]; \\
D_{\text{hat2}} &= [\text{zeros}(p-nsin+1,nsin) n_{\text{eigspa}}]; \\
D_{\text{hat}} &= [D_{\text{hat1}}; D_{\text{hat2}}]; \\
S &= \alpha \cdot D_{\text{hat}} + (1-\alpha) \cdot \beta \cdot \beta; \quad \% \text{our major concern} \\
[UU,DD] &= \text{eig}(S); \\
U_{\text{new}} &= U \cdot UU; \\
D_{\text{new}} &= DD; \\
gt &= U_{\text{new}}(1,1:nsin); \quad \% \text{extract the desired vector} \\
g &= \text{conj}(gt)'; \\
u_{1p} &= U_{\text{new}}(2:p+1,1:nsin); \\
u_1 &= [gt; u_{1p}]; \\
xtls &= (u_{1p} \cdot \text{conj}(g)) / (1-\text{conj}(gt) \cdot g); \quad \% \text{solution from signal subspace} \\
z &= [1 - \text{flip}(\text{conj}(\text{xtls}'))]; \\
rt &= [\text{rt roots}(z)]; \\
esp &= [\text{esp} \ D_2']; \\
\end{align*}
\]
% assign new U & D for updating
U = Unew;
D = Dnew;
end
end

% plot zeros in complex z-plane
axis('square');
axis([-1.5 1.5 -1.5 1.5]);
plot(sin(0:2*pi/100:2*pi),cos(0:2*pi/100:2*pi),'-w','...
real(rt),imag(rt),'o'),grid
title('RTLS solution for SNR= dB, P= , N= ');
text(-1,1.3,' alpha= ');
text(-1,1.1,' weight= ');
text(0.1,1.1,' f1 moving, f2=0.27 ');
text(0.1,1.3,' phi1=pi/4, phi2=0 ');
text(-0.5,-1.2,' 10 independent trials ');
text(-0.5,-1.4,' 10 recursive realizations ');

% plot spherical singular values status of updated matrix
clg
axis;
n3 = 1:p+1;
plot(n3,esp,'o'),grid
3. IMPLEMENTATION OF THE RTLS ALGORITHM USING THE SIGNAL SUBSPACE AND PERFORMING THE EIGEN-DECOMPOSITION OF MATRIX $\mathbf{R}$

Note that $\mathbf{R} = \alpha \mathbf{R} + (1-\alpha) \mathbf{x} \mathbf{x}^H$

clear
%initialization of conditions
rt=[];esp=[];
nt=input(' trial number= ');
ni=input(' iteration number= ');
p =input(' prediction order= ');
SNR=input(' signal-to-noise ratio, S/N(dB)= ');
weight=input(' the weight of moving frequency= ');
alpha=input(' exponential weighting factor, alpha= ');
%construct received signal and begin independent trial
for h=1:nt,
    n1=1:100;
    rand('normal');
    w1=rand(1,100);
    rand('normal');
    w2=rand(1,100);
    f1=0.23-(h-1)*weight/nt; %make f1 time-varying moving
    f2=0.27;
    phi1=+j;
    phi2=0;
    j=sqrt(-1);
wn = w1 + j*w2; \hspace{1cm} % independent complex noise
s1 = exp(j*(2*pi*f1*n1 + phi1)); \hspace{1cm} % create two undamped signals
s2 = exp(j*(2*pi*f2*n1 + phi2));
ss = s1 + s2;
xn = ss + sqrt(1/(2*10^(SNR/10))).*wn; \hspace{1cm} % received resulted signal
nsin = 2;
n2 = 50;
if n2 > p + 2,
    for k = 1:n2-p,
        A(k,1:p) = xn(k:p+k-1); \hspace{1cm} % generate data matrix
    end
    b = xn(1,p+1:n2);
    bb = conj(b)';
end
B = [bb A];
aph = alpha .^ n1;
aph = fliplr(aph(1:n2-p));
T = sqrt(diag(aph));
C = T*B; \hspace{1cm} % create augmented matrix
R = C'*C; \hspace{1cm} % define estd. correlation matrix
[U,D] = eig(R); \hspace{1cm} % perform eigendecomposition
for t = 1:ni,
    y = xn(1,n2-p+t:n2+t);
    x = y';
    R = alpha .* R + (1-alpha) .* x'*x'; \hspace{1cm} % our major concern
[UU,DD]=eig(R);
D1=flipud(sort((diag/DD)));
gt=UU(1,1:nsin); %extract the desired vector
g=conj(gt);
u1p=UU(2:p+1,1:nsin);
u1=[gt;u1p];
xtls=(u1p*conj(g))/(1-conj(gt)*g); %solution from signal subspace
z=[1 -fliplr(conj(xtls))];
rt=[rt roots(z)];
esp=[esp D1];
end
end
%plot zeros in complex z-plane
axis('square');
axis([-1.5 1.5 -1.5 1.5]);
plot(sin(0:2*pi/100:2*pi),cos(0:2*pi/100:2*pi),'-w',...
real(rt),imag(rt),','o'),grid
title(' RTLS solution for SNR dB, P= , N= ');
text(-1,1.3,' alpha= ');
text(-1,1.1,' weight-- ');
text(0.1,1.1,' f1 moving, f2=0.27 ');
text(0.1,1.3,' phi1=pi/4, phi2=0 ');
text(-0.5,-1.2,' 10 independent trials ');
text(-0.5,-1.4,' 10 recursive realizations ');
clg
axis;

n3=1:p+1;

plot(n3,esp,'o'), grid
4. IMPLEMENTATION OF THE RTLS ALGORITHM USING THE NOISE SUBSPACE AND PERFORMING THE EIGEN-DECOMPOSITION OF MATRIX 'S'

Note that $S = \alpha \tilde{D} + (1-\alpha)\beta^H \beta \beta^H \beta = U^H x$

cig
clear
%initialization of conditions
rt=[];esp=[];sig=[];noi=[];
nt=input(' trial number= ');
i=
pt=input(' prediction order= ');
SNR=input(' signal-to-noise ratio, S/N(dB)= ');
weight=input(' the weight of moving frequency= ');
alpha=input(' exponential weighting factor, alpha= ');
%construct received signal and begin independent trial
for h=1:nt,
n1=1:100;
rand('normal');
w1=rand(1,100);
rand('normal');
w2=rand(1,100);
f1=0.23-(h-1)*weight/nt; %make f1 time-varying moving
f2=0.27;
phi1=pi/4;
phi2=0;
j=sqrt(-1);
wn=w1+j*w2;  \text{\textcolor{red}{\% independent complex noise}}
s1=exp(j*(2*pi*f1*n1+phi1));  \text{\textcolor{red}{\% create two undamped signals}}
s2=exp(j*(2*pi*f2*n1+phi2));
ss=s1+s2;
xn=ss+sqrt(1/(2*10^(SNR/10))).*wn;  \text{\textcolor{red}{\% received resulted signal}}
nsin=2;  \text{\textcolor{red}{\% for two sinusoidal sources}}
n2=50;  \text{\textcolor{red}{\% window data sample number}}
if n2>p+2,
  for k=1:n2-p,
    A(k,1:p)=xn(k:p+k-1);  \text{\textcolor{red}{\% generate data matrix}}
  end
  b=xn(1,p+1:n2);  \text{\textcolor{red}{\% generate observation vector}}
  bb=conj(b)';
end
B=[bb A];
aph=alpha .^n1;
aph=fliplr(aph(1:n2-p));
T=sqrt(diag(aph));
C=T*B;  \text{\textcolor{red}{\% create augmented matrix}}
R=C*C;
[U,D]=eig(R);  \text{\textcolor{red}{\% perform eigendecomposition}}
for t=1:ni,
  y=xn(1,n2-p+t:n2+t);
  x=y';
\text{\textcolor{red}{\% recursively update subspace}}
beta=U'*x;
D1=sort(diag(D));
D2=flipud(D1);
D2=D2';
s_eigval=mean(D2(1:nsin));
n_eigval=mean(D2(nsin+1:p+1));
s_eigspa=s_eigval.*eye(nsin);
n_eigspa=n_eigval.*eye(p-nsin+1);
s_diaspa=diag(s_eigspa);
n_diaspa=diag(n_eigspa);
Dhat1=[s_eigspa zeros(nsin,p-nsin+1)];
Dhat2=[zeros(p-nsin+1,nsin) n_eigspa];
Dhat=[Dhat1;Dhat2];
S=alpha.*Dhat+(1-alpha).*beta*beta;  %our major concern
[UU,DD]=eig(S);
Unew=U*UU;
Dnew=DD;
ct=Unew(1,nsin+1:p+1);  %extract the desired vector
c=conj(ct)';
vp=Unew(2:p+1,nsin+1:p+1);
v2=[ct;vp];
xtls=-(vp*conj(c))./(1-conj(ct)*c);  %solution from noise subspace
z=[1 -fliprl(conj(xtls)')];
rt=[rt roots(z)];
esp=[esp D2'];
\[
\text{sig} = [\text{s.diaspa}];
\text{noi} = [\text{n.diaspa}];
\text{U} = \text{U}\text{new}; \quad \%\text{assign new U & D for updating}
\text{D} = \text{D}\text{new};
\text{end}
\text{end}
\]

\%plot zeros in complex z-plane
\text{axis('square');}
\text{axis([-1.5 1.5 -1.5 1.5]);}
\text{plot(sin(0:2*pi/100:2*pi),cos(0:2*pi/100:2*pi),'-w',...}
\text{real(rt),imag(rt),'o'),grid}
\text{title(' RTLS solution for SNR= dB, P= , N= ');}
\text{text(-1,1.3,' alpha= ');}
\text{text(-1,1.1,' weight= ');}
\text{text(0.1,1.1,' f1 moving, f2=0.27 ');}
\text{text(0.1,1.3,' phi1=pi/4, phi2=0 ');}
\text{text(-0.5,-1.2,' 10 independent trials ');}
\text{text(-0.5,-1.4,' 10 recursive realizations ');}
\%plot spherical singularvalues status of updated matrix
\text{clg}
\text{axis;}
\text{n3=1:p+1;}
\text{plot(n3,esp,'o'),grid}
REFERENCES


<table>
<thead>
<tr>
<th>No.</th>
<th>Distribution List</th>
<th>No. Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Defense Technical Information Center</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Cameron Station</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Alexandria, Virginia 22304-6145</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>Library, Code 52</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5100</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Department Chairman, Code EC</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Department of Electrical and Computer Engineering</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5000</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Professor Monique P. Fargues, Code EC/Fa</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Department of Electrical and Computer Engineering</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5000</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Professor Roberto Cristi, Code EC/Cx</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Department of Electrical and Computer Engineering</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5000</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>Professor Charles W. Therrien, Code EC/Ti</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Department of Electrical and Computer Engineering</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5000</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>Chung-Cheng Institute of Technology, Main Library</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>P.O. BOX 8243, Ta-Hsi, 33500, Tao-Yuan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Taiwan, R.O.C.</td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>Chiang, Chung-Jung</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>No.3, Alley 27, Lane 50,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Kuang-Min Street, Tainan, 70105</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Taiwan, R.O.C.</td>
<td></td>
</tr>
</tbody>
</table>