

NAVAL POSTGRADUATE SCHOOL Monterey, California

AD-A257 562



DTIC
ELECTE
DEC 1 1992
S C D

THESIS

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

92-30447

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY Unclassified			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE					
4. PERFORMING ORGANIZATION REPORT NUMBER(S)			5. MONITORING ORGANIZATION REPORT NUMBER(S)		
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School		6b. OFFICE SYMBOL (If applicable) EC	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School		
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000		7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000			
8a. NAME OF FUNDING/SPONSORING ORGANIZATION		8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c. ADDRESS (City, State, and ZIP Code)		10. SOURCE OF FUNDING NUMBERS			
		PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) APPLICATIONS OF THE TOTAL LEAST SQUARES TECHNIQUE TO FREQUENCY ESTIMATION					
12. PERSONAL AUTHOR(S) Chung-Jung Chiang					
13a. TYPE OF REPORT Master's Thesis		13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Year,Month,Day) September 1992		15. PAGE COUNT 98
16. SUPPLEMENTARY NOTATION 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.					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	TLS Algorithm, Constrained TLS Algorithm, Linear Constraint TLS Algorithm, and Recursive TLS Algorithm		
19. ABSTRACT (Continue on reverse if necessary and identify by block number) 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 <i>Total Least Squares</i> (<i>TLS</i>) technique which is a refinement of the <i>Least Squares</i> (<i>LS</i>) technique. In this thesis, we first describe the origin of the <i>TLS</i> 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 <i>TLS</i> extensions such as the <i>Constrained Total Least Squares</i> (<i>CTLS</i>) technique and the <i>Linear Constraint Total Least Squares</i> (<i>LCTLS</i>) technique. The <i>CTLS</i> addresses the case where the noise components are related and the <i>LCTLS</i> 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 <i>Recursive Total Least Squares</i> (<i>RTLS</i>) technique and apply it to the case of a signal with a fixed frequency together with a signal with a time-varying frequency.					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION Unclassified		
22a. NAME OF RESPONSIBLE INDIVIDUAL Monique P. Fargues			22b. TELEPHONE (Include Area Code) (408) 646 - 2859		22c. OFFICE SYMBOL EC/Fa

DD Form 1473, JUN 86

Previous editions are obsolete.
S/N 0102-LF-014-6603SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

Approved for public release; distribution is unlimited.

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

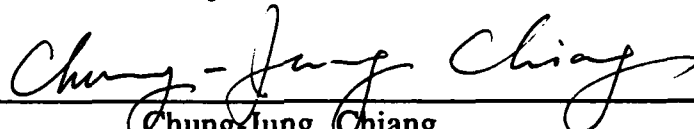
Submitted in partial fulfillment of the requirements for
the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1992

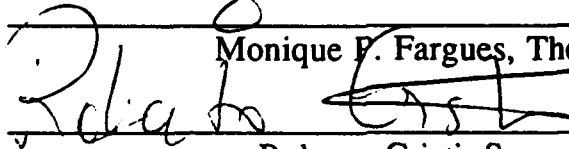
Author:


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.

Accession For	
NTIS Grant	☒
DWID TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Avail and/or	
Dist	Special
A-1	

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 (CTLTS) ALGORITHM.....	14
1. Introduction.....	14
2. Algorithm Derivation	14
E. LINEARLY CONSTRAINT TOTAL LEAST SQUARES (LCTLTS) 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, $f_1=0.23$, $f_2=0.27$, undamped sinusoids.....	44
Figure 3.1.2	TLS solution for SNR=12dB, P=8, N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids.....	44
Figure 3.1.3	TLS solution for SNR=5dB, P=8, N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids.....	45
Figure 3.1.4	TLS solution for SNR=3dB, P=8, N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids.....	45
Figure 3.1.5	TLS solution for SNR=1dB, P=8, N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids.....	46
Figure 3.1.6	TLS solution for SNR=0dB, P=8, N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids.....	46
Figure 3.2.1	TLS solution for SNR=3dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids.....	47
Figure 3.2.2	TLS solution for SNR=12dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids.....	47
Figure 3.2.3	TLS solution for SNR=3dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, $\phi_1=\pi/4$, $\phi_2=-\pi/4$, damped sinusoids.....	48
Figure 3.2.4	TLS solution for SNR=12dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, $\phi_1=\pi/4$, $\phi_2=-\pi/4$, damped sinusoids.....	48
Figure 3.3.1	TLS solution for SNR=12dB, P=2, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids.....	49
Figure 3.3.2	TLS solution for SNR=12dB, P=4, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids.....	49
Figure 3.3.3	TLS solution for SNR=12dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids.....	50

Figure 3.3.4	TLS solution for SNR=12dB, P=18, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids.....	50
Figure 3.3.5	TLS solution for SNR=12dB, P=28, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids.....	51
Figure 3.3.6	TLS solution for SNR=12dB, P=38, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids.....	51
Figure 3.4.1	TLS solution for SNR=12dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids, signal and noise subspaces mixed by 3 columns	52
Figure 3.4.2	TLS solution for SNR=12dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids, signal and noise subspaces mixed by 6 columns	52
Figure 3.5.1	TLS solution for SNR=5dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, damped sinusoids.....	53
Figure 3.5.2	TLS solution for SNR=20dB, P=8, N=50, $f_1=0.25$, $f_2=0.27$, damped sinusoids.....	53
Figure 3.5.3	TLS solution for SNR=5dB, P=18, N=50, $f_1=0.25$, $f_2=0.27$, damped sinusoids.....	54
Figure 3.5.4	TLS solution for SNR=20dB, P=18, N=50, $f_1=0.25$, $f_2=0.27$, damped sinusoids.....	54
Figure 4.1.1	RTLS solution for SNR=12dB, P=8, N=100, $\alpha=0.25$, weight=3%, f_1 is time-varying, $f_2=0.27$, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'	55
Figure 4.1.2	averaged eigenvalues of updated spherical matrix 'S' SNR=12dB, P=8, N=100, $\alpha=0.25$, weight=3%, f_1 is time-varying, $f_2=0.27$, solution derived by using the signal subspace information	55
Figure 4.1.3	RTLS solution for SNR=12dB, P=8, N=100, $\alpha=0.95$, weight=3%, f_1 is time-varying, $f_2=0.27$, using the signal subspace information & performing the eigendecomposition of spherical matrix 'S'	56

Figure 4.1.4	averaged eigenvalues of updated spherical matrix 'S' SNR=12dB, P=8, N=100, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $\alpha=0.95$, 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, $P=8$, $N=100$, $\alpha=0.95$, weight=10%, f_1 is time-varying, $f_2=0.27$, 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 $m \times n$ and $m \times 1$ 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 $\mathbf{x}_{\text{TLS}}$ by simultaneously minimizing the perturbations $\Delta\mathbf{A}$ and $\Delta\mathbf{b}$ which exist in the data matrix $\mathbf{A}$ and the observation vector $\mathbf{b}$, respectively.

When there exists some linear dependence algebraic relationship among the correlated noise perturbation components in $\mathbf{A}$ and $\mathbf{b}$, 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 $\Delta\mathbf{A}$ and $\Delta\mathbf{b}$ are algebraically correlated. By employing a new matrix $\mathbf{F}$ with a smaller dimension and a white noise random vector $\mathbf{w}$, it minimizes the *Frobenius norm* of $\Delta\mathbf{C}$ to derive the solution $\mathbf{x}_{\text{CTLS}}$. 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 $\mathbf{x}_{\text{RTLS}}$. 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 occurs 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:

$$\min_{x, \Delta b} \|\Delta b\| \quad (2.1)$$

subject to

$$Ax = b + \Delta b \quad (2.2)$$

which leads to

$$x_{LS} = (A^H A)^{-1} A^H b = A^+ b \quad (2.3)$$

where A^+ is the *pseudoinverse* of A , $A^H A$ 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:

$$\begin{bmatrix} \xi_0 & \xi_1 & \cdots & \xi_{P-1} \\ \xi_1 & \xi_2 & \cdots & \xi_P \\ \vdots & \vdots & \ddots & \vdots \\ \xi_{N-P-1} & \xi_{N-P} & \cdots & \xi_{N-2} \end{bmatrix} \begin{bmatrix} x_P \\ x_{P-1} \\ \vdots \\ x_1 \end{bmatrix} = \begin{bmatrix} \xi_P \\ \xi_{P+1} \\ \vdots \\ \xi_{N-1} \end{bmatrix} \quad (2.4)$$

ξ_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

$$\| \Delta C \| = \| [\Delta A | \Delta b] \| \quad (2.5)$$

subject to

$$(A + \Delta A) x = b + \Delta b \quad (2.6)$$

which leads to

$$x_{TLS} = (A^H A - \sigma^2 I)^{-1} A^H b \quad (2.7)$$

where ΔA and Δb are two independent noise perturbations and σ^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 $\mathbf{w}$ and computes $\mathbf{R} = \mathbf{E}[\mathbf{w}\mathbf{w}^*] = \mathbf{Z}\mathbf{Z}^*$, using a *Cholesky decomposition*. This leads to the definition of a new white noise vector δ where

$$\delta = \mathbf{Z}^{-1}\mathbf{w}. \quad (2.8)$$

The *whitening procedure* can be performed by using a prediction-error filter which offers a linear prediction equation same as $\mathbf{A}\mathbf{x}=\mathbf{b}$. 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 $\mathbf{x}_{\text{TLS}}$ is obtained by using information obtained from the *singular value decomposition* (*SVD*) of the matrix $\mathbf{C}=[\mathbf{A}|\mathbf{b}]$ with a dimension $(N-P) \times (P+1)$, i.e.,

$$\mathbf{C} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H \quad (2.9)$$

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

one on the upper-left-handed corner. The matrices U , V and Σ can be partitioned as

$$U = [u_1 | u_2 | \dots | u_{N-P}] \quad (2.10)$$

$$V = [v_1 | v_2 | \dots | v_{P+1}] \quad (2.11)$$

and
$$\Sigma = \text{diag}(\sigma_1, \dots, \sigma_P, \sigma_{P+1}). \quad (2.12)$$

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

2. Algorithm Derivation

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

$$([A|b] + [\Delta A|\Delta b]) \begin{pmatrix} x \\ -1 \end{pmatrix} = 0 \quad (2.13)$$

or

$$(C + \Delta C) z = 0 \quad (2.14)$$

where

$$C = [A|b], \quad \Delta C = [\Delta A|\Delta b], \quad z = \begin{pmatrix} x \\ -1 \end{pmatrix} \quad (2.15)$$

From the homogeneous equation (2.13), we see that the *TLS* problem finds the minimum norm of the perturbation matrix ΔC such that $(C + \Delta C)$ is

rank deficient. In other words, the *TLS* solution can be formulated as seeking a solution vector $\mathbf{x}$ [6] to the following problem

$$\min_{\Delta \mathbf{C}, \mathbf{x}} \|\Delta \mathbf{C}\|_F^2, \quad (2.16)$$

subject to

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

where

$$\|\Delta \mathbf{C}\|_F^2 = \text{tr}\{\Delta \mathbf{C}^* \Delta \mathbf{C}\}, \quad (2.18)$$

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

Next, the matrix $\mathbf{C}$ is decomposed in the following form

$$\mathbf{C} = [\mathbf{U}_1 \quad \mathbf{U}_2] \begin{bmatrix} \boldsymbol{\Sigma}_1 & 0 \\ 0 & \boldsymbol{\Sigma}_2 \end{bmatrix} \begin{bmatrix} \mathbf{V}_1^H \\ \mathbf{V}_2^H \end{bmatrix} \quad (2.19)$$

where $\mathbf{U}_1$, $\boldsymbol{\Sigma}_1$ and $\mathbf{V}_1$ correspond to the signal subspace; $\mathbf{U}_2$, $\boldsymbol{\Sigma}_2$ and $\mathbf{V}_2$ 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 σ_{p+1} in the diagonal matrix $\boldsymbol{\Sigma}$ is the smallest singular value and it corresponds to the last vector $\mathbf{v}_{p+1}$ in the unitary matrix $\mathbf{V}$. Then we get

$$\mathbf{x}_{\text{TLS}} = -\frac{1}{(v_{P+1})_{P+1}} \begin{bmatrix} v_{(P+1)_1} \\ \vdots \\ v_{(P+1)_r} \end{bmatrix} \quad (2.20)$$

where $(v_{P+1})_{P+1}$ is the last element of the last singular vector $\mathbf{v}_{P+1}$ 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.,

$$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_M > \sigma_{M+1} = \dots = \sigma_{P+1}, \quad (2.21)$$

and set $\mathbf{V}_2$ to be a column partition of $\mathbf{V}$ where

$$\mathbf{V}_2 = (\mathbf{v}_{M+1}, \dots, \mathbf{v}_{P+1}) \quad (2.22)$$

or

$$\mathbf{V}_2 = \begin{pmatrix} \mathbf{c}^T \\ \mathbf{V}_2' \end{pmatrix} \quad (2.23)$$

where $\mathbf{c}^T$ represents the first row of the matrix $\mathbf{V}_2$.

Next, we introduce the *Householder (reflection) matrix* $\mathbf{Q}$ 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]:

$$\mathbf{Q} \equiv \mathbf{I}_p - 2 \frac{\mathbf{v}\mathbf{v}^*}{\mathbf{v}^* \mathbf{v}} \quad (2.24)$$

where

$\mathbf{I}$ is an $m \times m$ identity matrix,

$\mathbf{v}$ is an $m \times 1$ complex vector which corresponds to the complex vector $\mathbf{v}_i$ in

$\mathbf{V}_2$ defined in (2.22) and has the *Euclidean norm*

$$\|\mathbf{v}\| = (\mathbf{v}^H \mathbf{v})^{\frac{1}{2}}. \quad (2.25)$$

The vector $\mathbf{v}$ can be defined below for analysis:

$$\mathbf{v} = \mathbf{c}^* - \alpha^* \mathbf{e}_p \quad \Rightarrow \quad \mathbf{v}^H = \mathbf{c}^T - \alpha^T \mathbf{e}_p^H \quad (2.26)$$

$$\alpha^* = \pm \frac{c_p^*}{|c_p|} \|\mathbf{c}\| \quad \Rightarrow \quad \alpha = \pm \frac{c_p}{|c_p|} \|\mathbf{c}\| \quad (2.27)$$

$$\mathbf{e}_p^T = [0, \dots, 0, 1]_{1 \times p} \Rightarrow \mathbf{v}_p^* = c_p - \alpha. \quad (2.28)$$

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:

$$\mathbf{V}_2 \mathbf{Q} = \begin{bmatrix} 0 & \alpha \\ \mathbf{W} & \mathbf{y} \end{bmatrix} \quad (2.29)$$

$$\mathbf{c}^T \mathbf{Q} = [0, \dots, 0, \alpha]. \quad (2.30)$$

Now we express $\mathbf{V}_2 \mathbf{Q}$ to extract the last column, which is used in the solution:

$$\mathbf{V}_2 \mathbf{Q} = \mathbf{V}_2 [\mathbf{q}_1 | \mathbf{q}_2 | \mathbf{q}_3 | \dots | \mathbf{q}_p] \quad (2.31)$$

and let

$$\mathbf{V}_2 \mathbf{q}_p = \begin{bmatrix} \alpha \\ y \end{bmatrix}. \quad (2.32)$$

Note that

$$\begin{aligned} \mathbf{v}^H \mathbf{v} &= (\mathbf{c}^T - \alpha^T \mathbf{e}_p^H) (\mathbf{c}^* - \alpha^* \mathbf{e}_p) \\ &= \|\mathbf{c}\|^2 - \alpha^* c_p - \alpha c_p^* + \|\alpha\|^2 \quad (\because \alpha^* c_p = \alpha c_p^*) \\ &= \|\mathbf{c}\|^2 - 2\alpha^* c_p + \|\alpha\|^2 \quad \left(\because \alpha^* = \pm \frac{c_p^*}{|c_p|} \|\mathbf{c}\} \right) \\ &= \|\mathbf{c}\|^2 - 2 \frac{|c_p|^2}{|c_p|} \|\mathbf{c}\} + \frac{|c_p|^2}{|c_p|^2} \|\mathbf{c}\|^2 \quad (\because \|\alpha\|^2 = \alpha \alpha^*) \\ &= 2 \left(\|\alpha\|^2 - \frac{c_p c_p^*}{|c_p|} \|\mathbf{c}\} \right) \quad (\because \|\alpha\|^2 = \|\mathbf{c}\|^2 = \mathbf{c}^H \mathbf{c}) \\ &= 2(\alpha \alpha^* - \alpha^* c_p). \end{aligned} \quad (2.33)$$

The last column of the matrix $\mathbf{Q}$ is given by

$$\begin{aligned} \mathbf{q}_p &= \mathbf{e}_p - \frac{2 \mathbf{v}_p^* \mathbf{v}}{\mathbf{v}^H \mathbf{v}} \\ &= \mathbf{e}_p - \frac{2(c_p - \alpha)}{2\alpha^*(\alpha - c_p)} (\mathbf{c}^* - \alpha^* \mathbf{e}_p) \\ &= \mathbf{e}_p - \frac{c_p \mathbf{c}^* - c_p \alpha^* \mathbf{e}_p - \alpha \mathbf{c}^* + \alpha \alpha^* \mathbf{e}_p}{\alpha^*(\alpha - c_p)} \\ &= \frac{\mathbf{c}^*(\alpha - c_p)}{\alpha^*(\alpha - c_p)} = \frac{\mathbf{c}^*}{\alpha^*}. \end{aligned} \quad (2.34)$$

Thus,

$$y = V_2' q_p = \frac{V_2' c^*}{\alpha^*} \quad (2.35)$$

and

$$x_{\text{TLS}} = -\frac{y}{\alpha} = -\frac{V_2' c^*}{\alpha \alpha^*} = -\frac{V_2' c^*}{c^H c} \quad (2.36)$$

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

$$\begin{bmatrix} g^H & c^H \\ V_1' & V_2' \end{bmatrix} \begin{bmatrix} g & V_1'^H \\ c & V_2'^H \end{bmatrix} = I \quad (2.37)$$

which leads to

$$\begin{aligned} g^H g + c^H c &= 1 & \Rightarrow & c^H c = 1 - g^H g \\ V_1' g + V_2' c &= 0 & \Rightarrow & V_2' c = -V_1' g \end{aligned} \quad (2.38)$$

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

$$x_{\text{TLS}} = \frac{V_1' g^*}{1 - g^H g} \quad (2.39)$$

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 ΔA and Δb both as linearly constrained.

$$\Delta A = [F_1 w \mid F_2 w \mid \cdots \mid F_P w] \quad (2.40)$$

$$\Delta b = F_{P+1} w \quad (2.41)$$

Then the augmented error matrix ΔC is formed by

$$\Delta C = [\Delta A \mid \Delta b] = [F_1 w \mid F_2 w \mid \cdots \mid F_P w \mid F_{P+1} w] \quad (2.42)$$

or

$$\Delta C = [\Delta C_1 | \dots | \Delta C_p | \Delta C_{p+1}] = Fw, \quad (2.43)$$

$$F = [F_1 | F_2 | \dots | F_p | F_{p+1}] \quad (2.44)$$

where F is a matrix whose elements are related to those of independent variables ΔC 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 ΔC can satisfy and to accomplish the minimization faster. So the *CTLS* solution can be formulated as follows:

$$\min_{w,x} \|w\|^2 \quad (2.45)$$

subject to

$$(C + \Delta C) \begin{pmatrix} x \\ -1 \end{pmatrix} = 0. \quad (2.46)$$

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

$$C \begin{pmatrix} x \\ -1 \end{pmatrix} + \Delta C \begin{pmatrix} x \\ -1 \end{pmatrix} = 0 \quad (2.47)$$

where

$$\begin{aligned}\Delta C \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} &= [\mathbf{F}_1 \mathbf{w} \mid \mathbf{F}_2 \mathbf{w} \mid \dots \mid \mathbf{F}_P \mathbf{w} \mid \mathbf{F}_{P+1} \mathbf{w}] \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} \\ &= \sum_{i=1}^P \mathbf{x}_i \mathbf{F}_i \mathbf{w} - \mathbf{F}_{P+1} \mathbf{w} = \left(\sum_{i=1}^P \mathbf{x}_i \mathbf{F}_i - \mathbf{F}_{P+1} \right) \mathbf{w}.\end{aligned}\quad (2.48)$$

Let

$$\mathbf{H}_x \equiv \sum_{i=1}^P \mathbf{x}_i \mathbf{F}_i - \mathbf{F}_{P+1}, \quad (2.49)$$

then we have

$$C \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} + \mathbf{H}_x \mathbf{w} = 0 \quad \Rightarrow \quad \mathbf{w} = -\mathbf{H}_x^+ C \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} \quad (2.50)$$

where $\mathbf{H}_x^+$ is the *pseudoinverse* of $\mathbf{H}_x$ defined as

$$\mathbf{H}_x^+ \equiv \mathbf{H}_x^* (\mathbf{H}_x \mathbf{H}_x^*)^{-1}. \quad (2.51)$$

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

$$\min_{\mathbf{w}} \|\mathbf{w}\|^2 = \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix}^* C^* \mathbf{H}_x^+ \mathbf{H}_x C \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix}. \quad (2.52)$$

Next, let us define the analytic function $\mathbf{J}(\mathbf{x})$

$$\mathbf{J}(\mathbf{x}) \equiv \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix}^* C^* (\mathbf{H}_x^+)^* \mathbf{H}_x^+ C \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} \quad (2.53)$$

$$\mathbf{x} = (x_1, \dots, x_p).$$

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

$$\begin{aligned} \mathbf{J}(\mathbf{x}) &= \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix}^* \mathbf{C}^* \mathbf{H}_x (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \mathbf{H}_x^* (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \mathbf{C} \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix}^* \mathbf{C}^* [(\mathbf{H}_x \mathbf{H}_x^*)^{-1} \mathbf{H}_x \mathbf{H}_x^*] (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \mathbf{C} \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix}^* \mathbf{C}^* (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \mathbf{C} \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix}. \end{aligned} \quad (2.54)$$

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

$$\frac{\partial \mathbf{J}}{\partial \mathbf{x}} = \left(\frac{\partial \mathbf{J}}{\partial x_1} \dots \frac{\partial \mathbf{J}}{\partial x_p} \right) = 0. \quad (2.55)$$

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 $\mathbf{J}(\mathbf{x})$ to update the values of $\mathbf{x}$. The *Newton iteration (recursion)* is briefly described by

$$\mathbf{x}(n+1) = \mathbf{x}(n) + (\overline{\mathbf{A}}\mathbf{B}^{-1}\mathbf{A} - \overline{\mathbf{B}})^{-1}(\overline{\mathbf{a}} - \overline{\mathbf{A}}\mathbf{B}^{-1}\mathbf{a}) \quad (2.56)$$

where

$$\begin{aligned}
\mathbf{a} &= \frac{\partial \mathbf{J}^T}{\partial \mathbf{x}} = \left(\frac{\partial \mathbf{J}}{\partial x_1} \dots \frac{\partial \mathbf{J}}{\partial x_p} \right)^T = \text{complex gradient of } \mathbf{J} \\
\mathbf{A} &= \frac{1}{2} \left(\frac{\partial^2 \mathbf{J}}{\partial \mathbf{x}^2} + \frac{\partial^2 \mathbf{J}^T}{\partial \mathbf{x}^2} \right) = \text{unconjugated complex Hessian of } \mathbf{J} \\
\mathbf{B} &= \frac{\partial^2 \mathbf{J}}{\partial \bar{\mathbf{x}} \partial \mathbf{x}} = \text{conjugated complex Hessian of } \mathbf{J}.
\end{aligned} \tag{2.57}$$

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

$$\begin{aligned}
\mathbf{a} &= (\mathbf{h} * \tilde{\mathbf{B}})^T \\
\mathbf{A} &= -\tilde{\mathbf{F}}^* \mathbf{H}_x^* (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \tilde{\mathbf{B}} - (\tilde{\mathbf{F}}^* \mathbf{H}_x^* (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \tilde{\mathbf{B}})^T \\
\mathbf{B} &= [\tilde{\mathbf{B}}^* (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \tilde{\mathbf{B}}]^T + \tilde{\mathbf{F}}^* [\mathbf{H}_x^* (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \mathbf{H}_x - \mathbf{I}] \tilde{\mathbf{F}}
\end{aligned} \tag{2.58}$$

where

$$\begin{aligned}
\mathbf{h} &= (\mathbf{H}_x \mathbf{H}_x^*)^{-1} \mathbf{C} \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} \\
\tilde{\mathbf{B}} &= \mathbf{C} \mathbf{I}_{p+1, p} - [\mathbf{F}_1 \mathbf{H}_x^* \mathbf{h} | \dots | \mathbf{F}_p \mathbf{H}_x^* \mathbf{h}] \\
\tilde{\mathbf{G}} &= [\mathbf{F}_1^* \mathbf{h} | \dots | \mathbf{F}_p^* \mathbf{h}].
\end{aligned} \tag{2.59}$$

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

$$x = (x_0, x_1, \dots, x_p)^T \quad (2.60)$$

(b).compute :

$$Z_Y = \sum_{j=1}^N Y_j^H (X^H X)^{-1} Y_j \quad (2.61)$$

where

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

y_i being the observations, and

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

c).solve the Quadratic Minimization Problem :

$$\min_{x(n+1)} (x_{(n+1)}^H Z_Y^{(n)} x_{(n+1)}) \quad (2.64)$$

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

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

(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 *Linearly 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 :

$$(A + \Delta A) x = b + \Delta b. \quad (2.65)$$

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

$$(b + \Delta b) \in \text{Range}(A + \Delta A). \quad (2.66)$$

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

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

2. Algorithm Derivation

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

$$\Psi x = \Gamma \quad (2.68)$$

where Ψ , $\mathbf{x}$ and Γ have dimensions $p \times (n+1)$, $(n+1) \times 1$ and $p \times 1$, respectively.

Equation (2.69) can also be expressed as below:

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

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:

$$\zeta(n) = A_1 e^{jn\omega_1} + A_2 e^{jn\omega_2} + w(n), \quad n = 1, \dots, N \quad (2.70)$$

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

Assume that ω_1 is known and ω_2 is unknown. Create a signal source vector $\lambda(\omega)$ defined as:

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

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

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

The constraint forces $e^{j\omega_2}$ to be one of the roots of the prediction-error polynomial $e(\omega)$ defined as:

$$e(\omega) = \lambda^H(\omega) \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} = 0. \quad (2.73)$$

Then, the angle of the next closest root on the unit circle is chosen as the estimate of the unknown angular frequency ω_2 .

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

$$\begin{pmatrix} \Psi & \Gamma \\ A + \Delta A & b + \Delta b \end{pmatrix} \begin{pmatrix} x \\ -1 \end{pmatrix} = 0. \quad (2.74)$$

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

$$[\Psi \ \Gamma] = [\Omega_1 \ 0] \Lambda^H = \Omega_1 \Lambda_1^H \quad (2.75)$$

$$\Lambda^H = \begin{pmatrix} \Lambda_1^H \\ \Lambda_2^H \end{pmatrix} \quad (2.76)$$

where Ω_1 is an upper triangular matrix with a dimension $(p \times p)$ and Λ^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 Λ_2 with a dimension $(n+1) \times (n+1-p)$ for the null space of $[\Psi \ \Gamma]$ and an arbitrary non-zero vector η with a dimension $(n+1-p) \times 1$, i.e.,

$$z = \Lambda_2 \eta. \quad (2.77)$$

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

$$\tilde{C} = C\Lambda_2\Lambda_2^H \quad (2.78)$$

with

$$C = [A | b]. \quad (2.79)$$

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

$$\tilde{C} = \tilde{U}\tilde{\Sigma}\tilde{V}^H = \sum_{i=1}^{n+1-p} \tilde{\sigma}_i \tilde{u}_i \tilde{v}_i^H \quad (2.80)$$

or

$$\tilde{C} = \begin{bmatrix} \tilde{U}_1 & \tilde{U}_2 & \tilde{U}_3 \end{bmatrix} \begin{bmatrix} \tilde{\Sigma}_1 & 0 & 0 \\ 0 & \tilde{\Sigma}_2 & 0 \\ 0 & 0 & \tilde{\Sigma}_3 \end{bmatrix} \begin{bmatrix} \tilde{V}_1^H \\ \tilde{V}_2^H \\ \tilde{V}_3^H \end{bmatrix} \quad (2.81)$$

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

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

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

$\tilde{\Sigma}_3 = 0.$

Using the fact that

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

we get

$$C = C \Lambda_1 \Lambda_1^H + C \Lambda_2 \Lambda_2^H. \quad (2.83)$$

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 exists ($s=n+1-p-r$) 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)

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

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

$$\mathbf{x}_{\text{LCTLS}} = -\frac{\mathbf{V}_2' \mathbf{c}}{\mathbf{c}^H \mathbf{c}}. \quad (2.85)$$

Further using the following identity

$$\begin{bmatrix} \mathbf{g}^H & \mathbf{c}^H & \mathbf{f}^H \\ \mathbf{V}_1' & \mathbf{V}_2' & \mathbf{V}_3' \end{bmatrix} \begin{bmatrix} \mathbf{g} & \mathbf{V}_1'^H \\ \mathbf{c} & \mathbf{V}_2'^H \\ \mathbf{f} & \mathbf{V}_3'^H \end{bmatrix} = \mathbf{I}, \quad (2.86)$$

which leads to the following two equations:

$$\begin{aligned} \mathbf{g}^H \mathbf{g} + \mathbf{c}^H \mathbf{c} + \mathbf{f}^H \mathbf{f} &= 1 \Rightarrow \mathbf{c}^H \mathbf{c} = 1 - \mathbf{g}^H \mathbf{g} - \mathbf{f}^H \mathbf{f} \\ \mathbf{V}_1' \mathbf{g} + \mathbf{V}_2' \mathbf{c} + \mathbf{V}_3' \mathbf{f} &= 0 \Rightarrow \mathbf{V}_2' \mathbf{c} = -\mathbf{V}_1' \mathbf{g} - \mathbf{V}_3' \mathbf{f} \end{aligned} \quad (2.87)$$

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

$$\mathbf{x}_{LCTLS} = \frac{\mathbf{V}_1' \mathbf{g}^* + \mathbf{V}_3' \mathbf{f}^*}{1 - \mathbf{g}^H \mathbf{g} - \mathbf{f}^H \mathbf{f}}. \quad (2.88)$$

Note that the minimum *LCTLS* solution should be based on the subspace $\bar{\mathbf{V}}_2$ or

$\begin{bmatrix} \tilde{\mathbf{V}}_1 & \tilde{\mathbf{V}}_3 \end{bmatrix}$, 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 $f_s=1\text{Hz}$:

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

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 σ^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 σ^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 $10 \cdot \log(\frac{1}{2\sigma^2})$ 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 :

$$ESD = \sqrt{\frac{1}{T} \sum_{i=1}^T \left| \hat{f}_i - f \right|^2} \quad (3.2)$$

where

T : the total number of independent trials

$\hat{f}_i$: the estimated frequency at the i^{th} realization

f : 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 $f_1=0.25\text{Hz}$, $f_2=0.27\text{Hz}$; 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 $f_1=0.25\text{Hz}$, $f_2=0.27\text{Hz}$, $\text{SNR}=12\text{dB}$, $N=50$, 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 $SNR=12dB$ 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 $SNR=12dB$, $P=8$, $N=50$ 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 :

$$x(n) = e^{(-\alpha_1 n + j(2\pi f_1 n + \phi_1))} + e^{(-\alpha_2 n + j(2\pi f_2 n + \phi_2))} + w(n), \quad (3.3)$$

$$n = 0, 1, \dots, N-1$$

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 α , called the *exponential weighting factor*, which is similar to λ (*forgetting factor*) in the *RLS* algorithm.

$$C(n) = D_{\alpha}^{(n)} [A \mid b] \quad (4.1)$$

where

$$D_{\alpha}^{(n)} = \sqrt{\text{diag}(\alpha^n, \dots, \alpha^0)}. \quad (4.2)$$

It is worthy to investigate the impact of the *forgetting weight factor* α on the tracking capability. The factor α 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 α 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 α 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

$$R_{(n)} = \alpha R_{(n-1)} + (1 - \alpha) y_{(n)} y_{(n)}^H \quad (4.3)$$

where $R_{(n)}$ is defined as

$$R_{(n)} = C_{(n)}^H C_{(n)}. \quad (4.4)$$

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

$$R_{(n-1)} = U_{(n-1)} D_{(n-1)} U_{(n-1)}^H \quad (4.5)$$

$$\mathbf{R}_{(n)} = \mathbf{U}_{(n)} \mathbf{D}_{(n)} \mathbf{U}_{(n)}^H \quad (4.6)$$

where for all present time n

$$\mathbf{U}_{(n)} \mathbf{U}_{(n)}^H = \mathbf{U}_{(n)}^H \mathbf{U}_{(n)} = \mathbf{I} \quad (4.7)$$

and $\mathbf{D}_{(n)}$ 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* $\mathbf{R}_{(n)}$ under the condition of forced *sphericity* of the eigenstructure. To determine the right singular subspace of $\mathbf{C}_{(n)}$, we need to obtain the eigenspaces of $\mathbf{R}_{(n)}$ corresponding to the right singular subspaces of $\mathbf{C}_{(n)}$. For later simulation, we get the vector $\mathbf{y}_{(n)}$ extracted from the last row of matrix $[\mathbf{A}|\mathbf{b}]$. 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 $\tilde{\mathbf{R}}$ for the rank-one updating recursion

$$\tilde{\mathbf{R}}_{(n)} = \alpha \mathbf{R}_{(n)} + \mathbf{y}_{(n)} \mathbf{y}_{(n)}^H. \quad (4.8)$$

By substituting (4.6) into (4.8) and letting

$$\boldsymbol{\chi} = \mathbf{U}^H \mathbf{y}, \quad (4.9)$$

we further get

$$\mathbf{y} = \mathbf{U} \boldsymbol{\chi} \quad (4.10)$$

and derive the following form

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

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

$$\mathbf{x}_{\text{RTLS}} = \frac{\mathbf{V}_1' \mathbf{g}^*}{1 - \mathbf{g}^H \mathbf{g}}, \quad (4.12)$$

for the noise subspace, the solution vector is

$$\mathbf{x}_{\text{RTLS}} = -\frac{\mathbf{V}_2' \mathbf{c}^*}{\mathbf{c}^H \mathbf{c}}. \quad (4.13)$$

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 0.23Hz 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 α , 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 α 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 α 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 $\text{SNR}=12\text{dB}$ 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 $\mathbf{b}$ of a linear prediction equation $\mathbf{Ax}=\mathbf{b}$. Since the assumption is not realistic, we present the *Total Least Squares* (*TLS*) algorithm which considers the errors or perturbations in the observation vector $\mathbf{b}$ and the data matrix $\mathbf{A}$. When there exists some linear dependence algebraical relationship among the correlated noise perturbation components in $\mathbf{A}$ and $\mathbf{b}$, 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 1. EFFECTIVE STANDARD DEVIATIONS OBTAINED
BY USING THE TLS ALGORITHM**

SNR (dB)	Estimated Deviation for $f_1=0.23\text{Hz}$	Estimated Deviation for $f_2=0.27\text{Hz}$
20	0.0020	8.5083e-04
18	0.0030	9.1698e-04
16	0.0049	0.0046
14	0.0047	0.0033
12	0.0097	0.0073
10	0.0058	0.0065
8	0.0054	0.0203
7	0.0189	0.0133
6	0.0198	0.0164
5	0.0073	0.0166
4	0.0217	0.0388
3	0.0106	0.0461
2	0.0231	0.0333
1	0.0325	0.0108
0	0.0684	0.1816

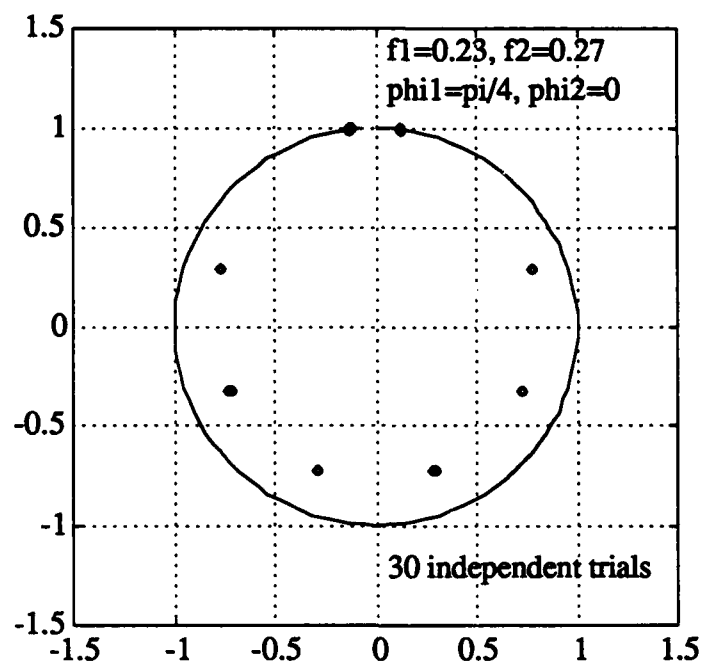


Figure 3.1.1 TLS solution for SNR=20dB, $P=8$
 $N=50$, $f_1=0.23$, $f_2=0.27$, undamped sinusoids

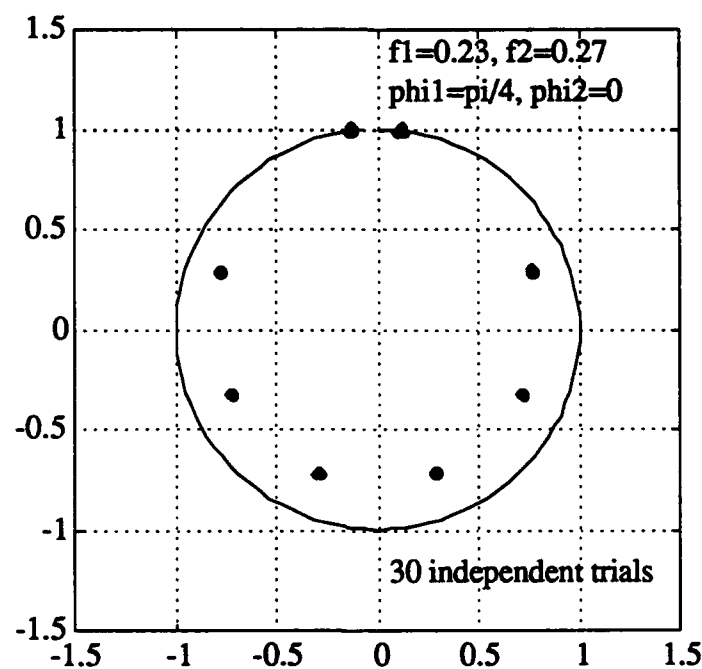


Figure 3.1.2 TLS solution for SNR=12dB, $P=8$
 $N=50$, $f_1=0.23$, $f_2=0.27$, undamped sinusoids

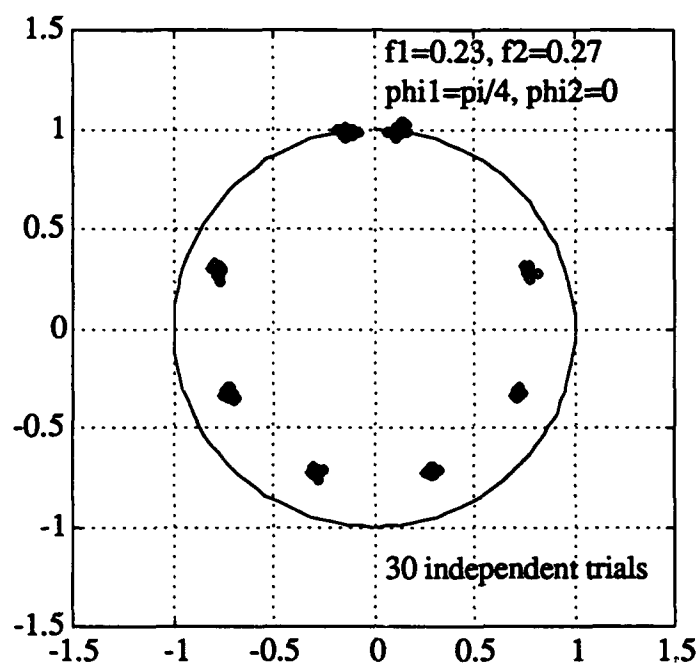


Figure 3.1.3 TLS solution for SNR=5dB, P=8
N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids

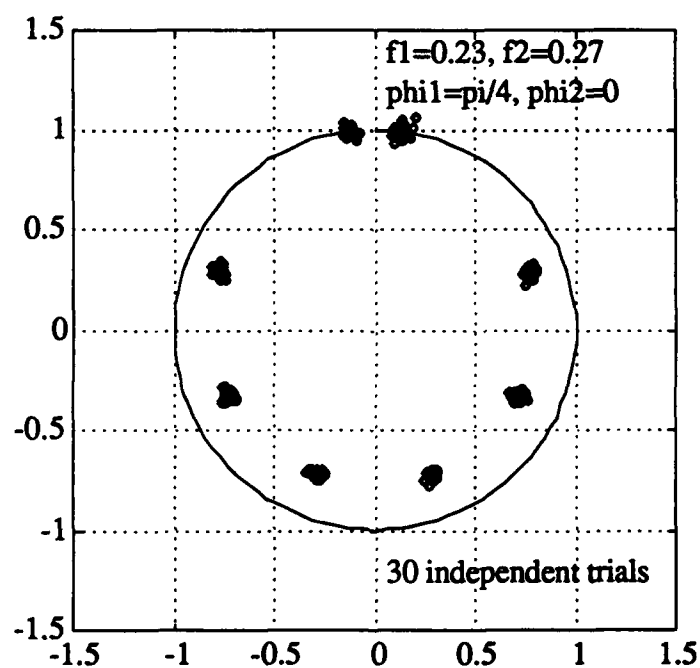


Figure 3.1.4 TLS solution for SNR=3dB, P=8,
N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids

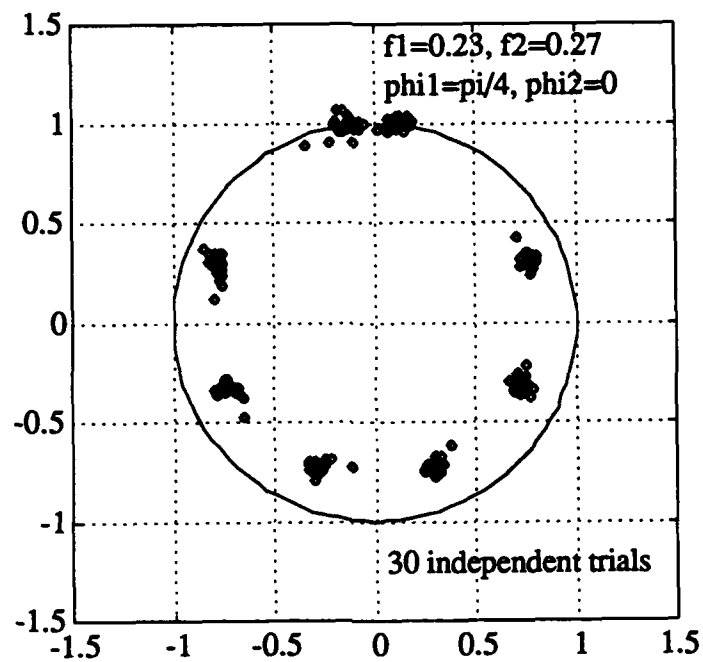


Figure 3.1.5 TLS solution for SNR=1dB, P=8
N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids

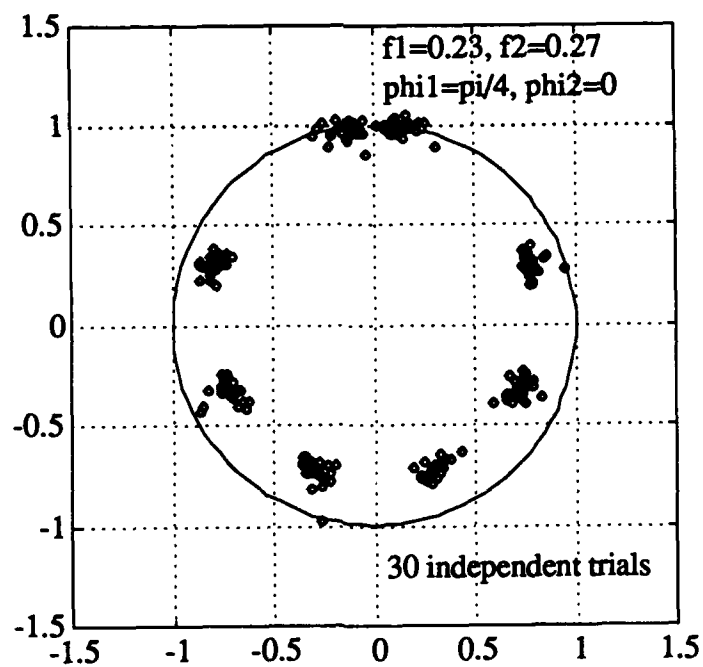


Figure 3.1.6 TLS solution for SNR=0dB, P=8
N=50, $f_1=0.23$, $f_2=0.27$, undamped sinusoids

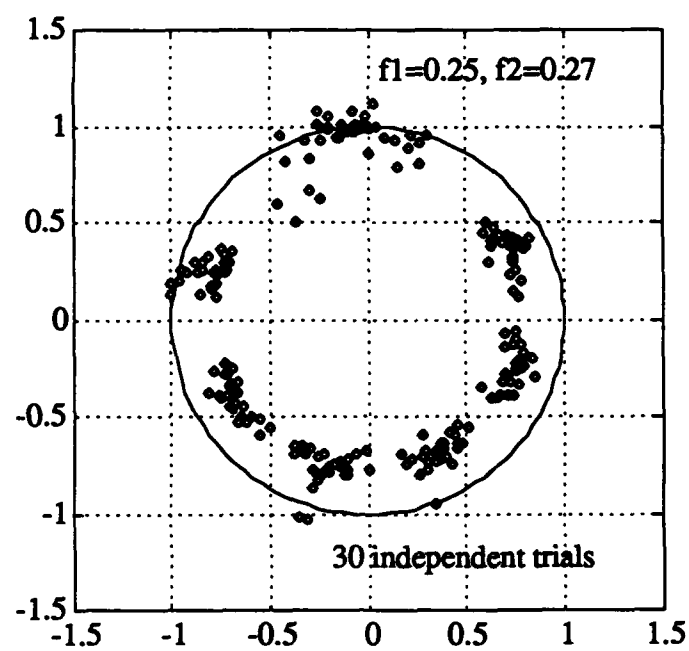


Figure 3.2.1 TLS solution for SNR=3dB, P=8
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids

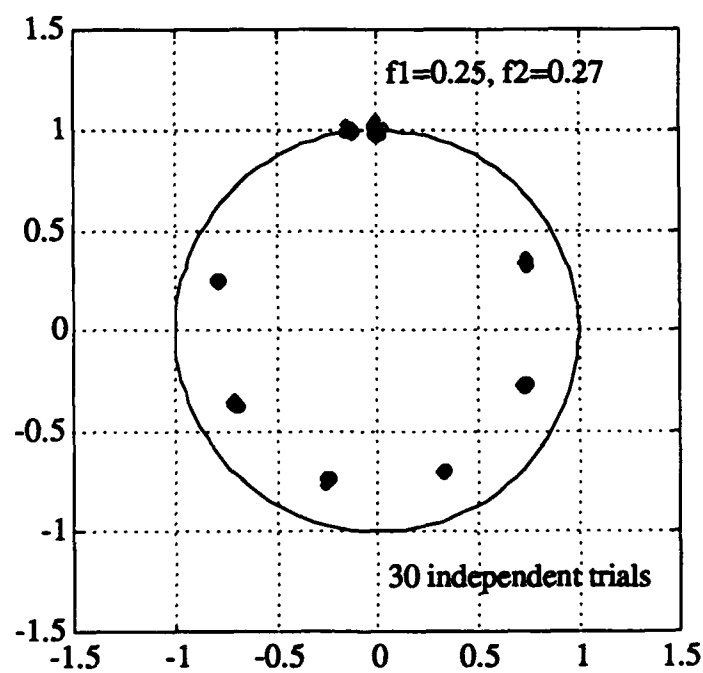


Figure 3.2.2 TLS solution for SNR=12dB, P=8
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids

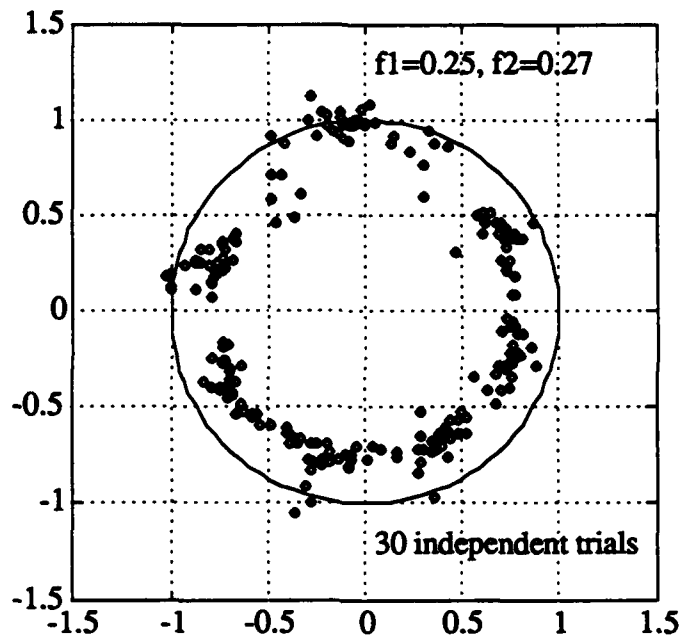


Figure 3.2.3 TLS solution for SNR=3dB, P=8, N=50
 $f_1=0.25$, $f_2=0.27$, $\phi_1=\pi/4$, $\phi_2=-\pi/4$, undamped sinusoids

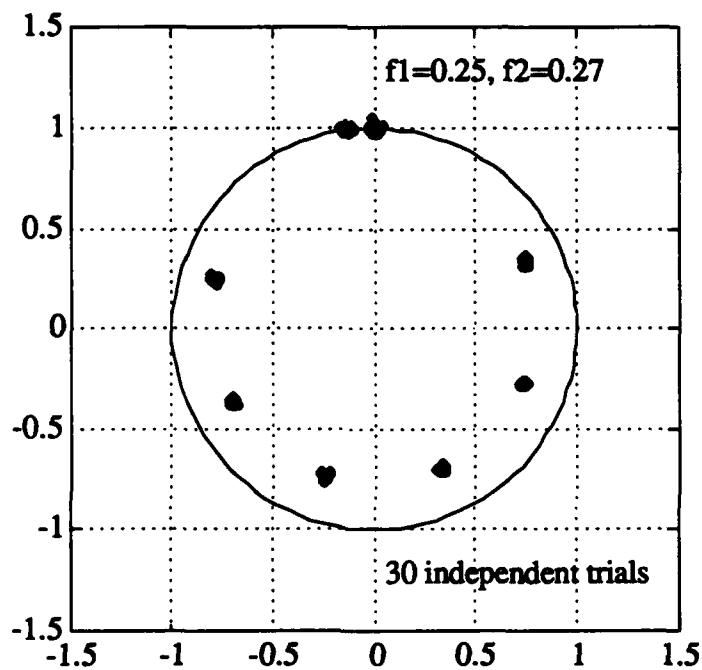


Figure 3.2.4 TLS solution for SNR=12dB, P=8, N=50
 $f_1=0.25$, $f_2=0.27$, $\phi_1=\pi/4$, $\phi_2=-\pi/4$, undamped sinusoids

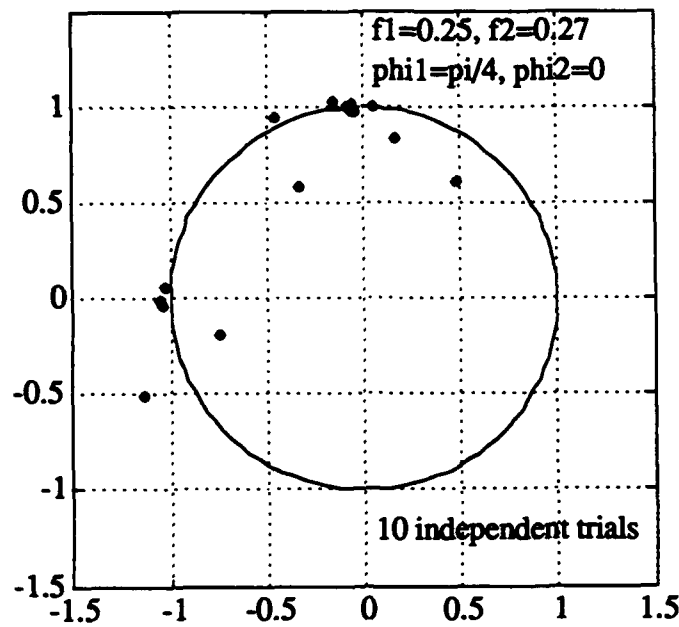


Figure 3.3.1 TLS solution for SNR=12dB, P=2
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids

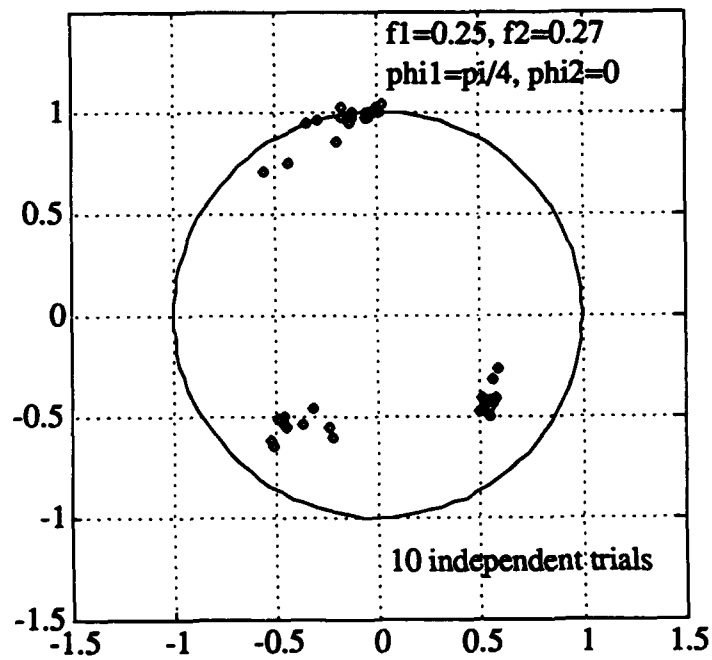


Figure 3.3.2 TLS solution for SNR=12dB, P=4
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids

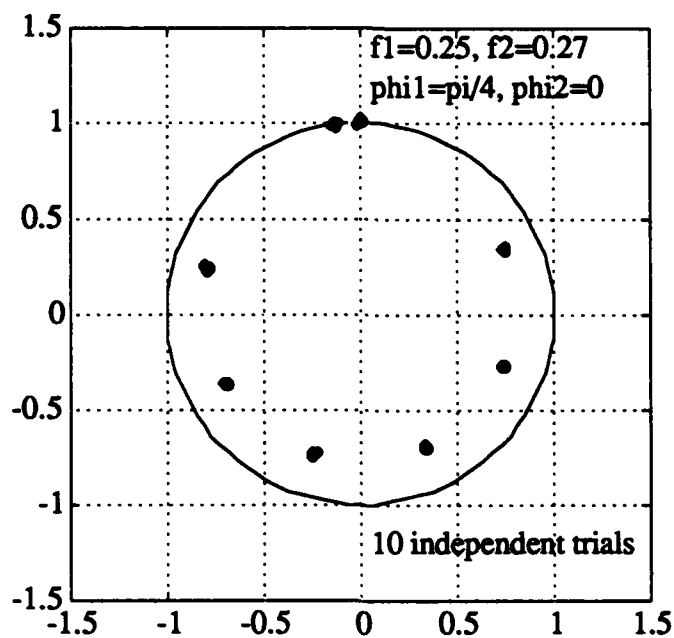


Figure 3.3.3 TLS solution for SNR=12dB, P=8
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids

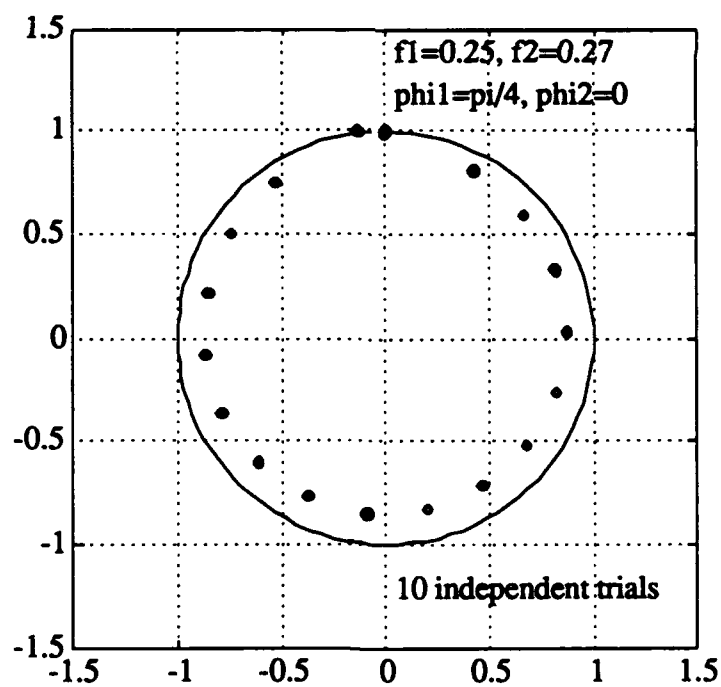


Figure 3.3.4 TLS solution for SNR=12dB, P=18
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids

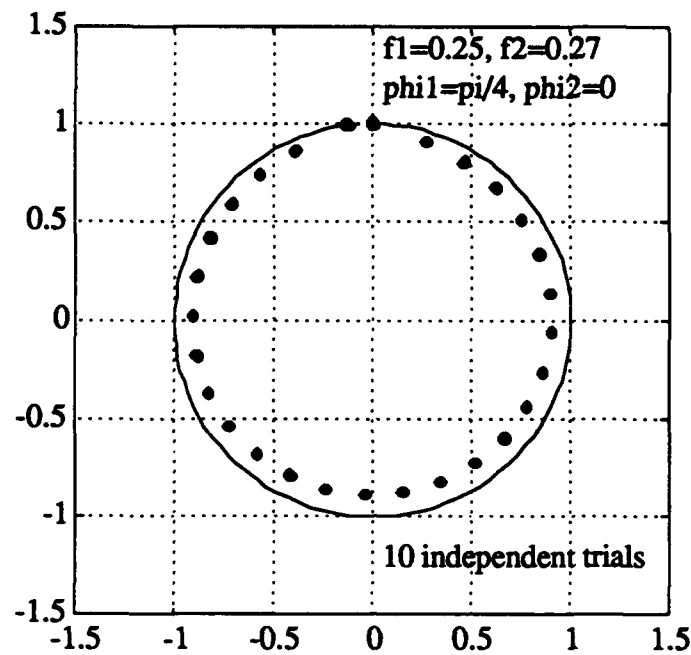


Figure 3.3.5 TLS solution for SNR=12dB, P=28
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids

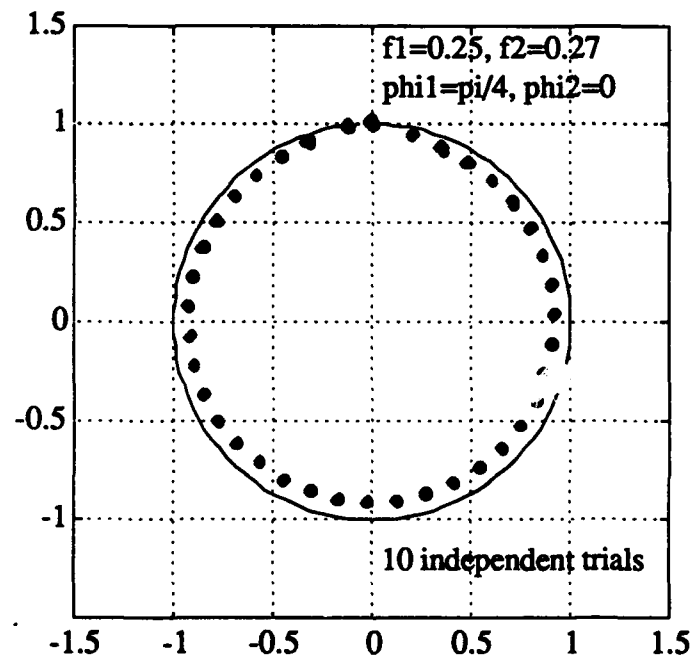


Figure 3.3.6 TLS solution for SNR=12dB, P=38
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids

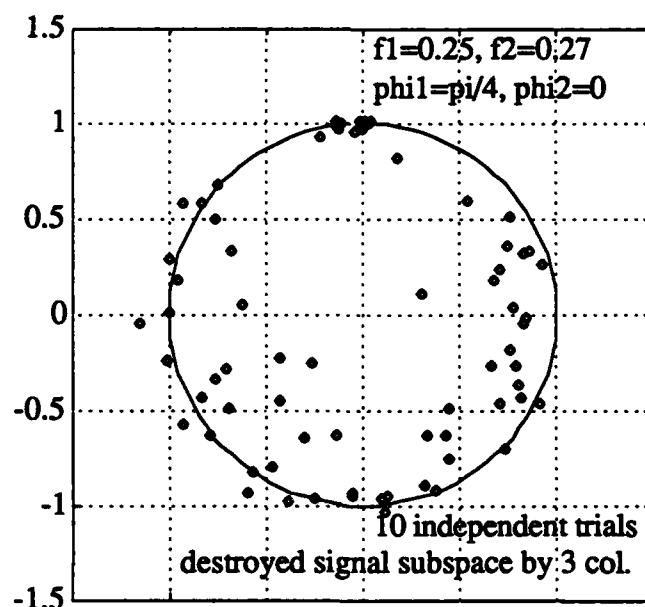


Figure 3.4.1 TLS solution for SNR=12dB, P=8
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids
signal and noise subspaces mixed by 3 columns

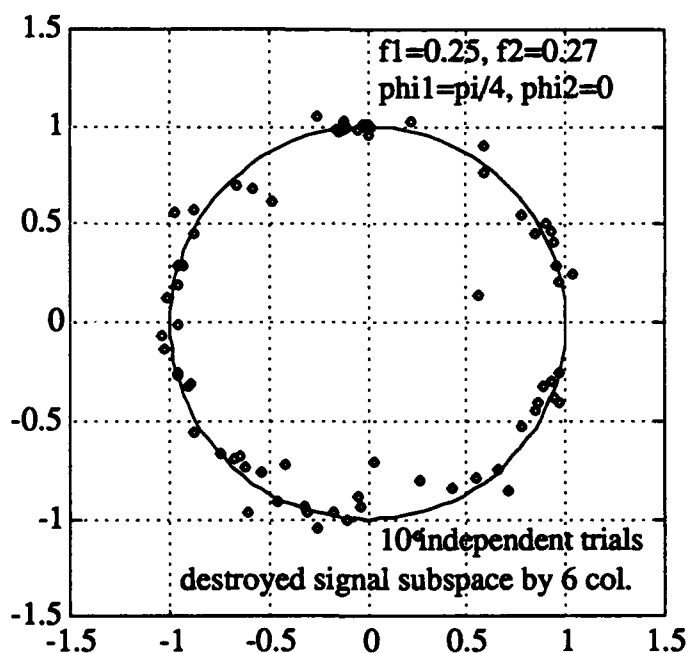


Figure 3.4.2 TLS solution for SNR=12dB, P=8
N=50, $f_1=0.25$, $f_2=0.27$, undamped sinusoids
signal and noise subspaces mixed by 6 columns

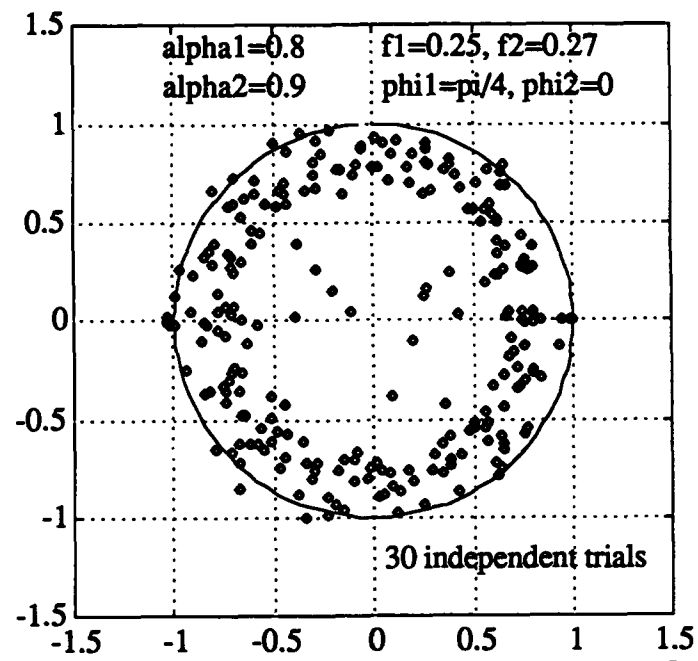


Figure 3.5.1 TLS solution for SNR=5dB, $P=8$
 $N=50, f_1=0.25, f_2=0.27$, damped sinusoids

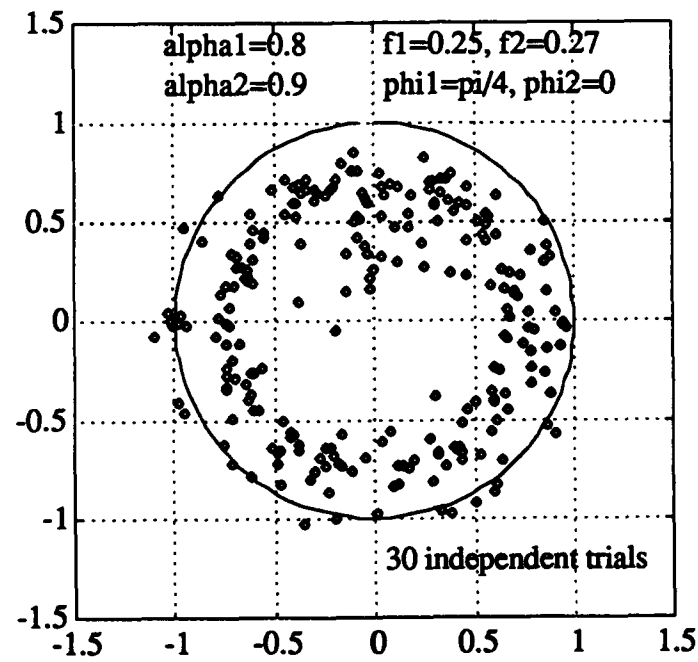


Figure 3.5.2 TLS solution for SNR=20dB, $P=8$
 $N=50, f_1=0.25, f_2=0.27$, damped sinusoids

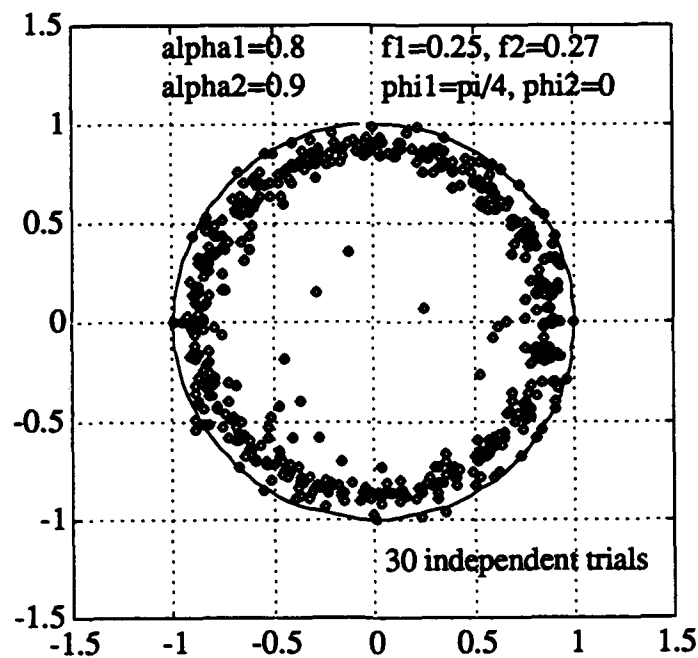


Figure 3.5.3 TLS solution for SNR=5dB, P=18
N=50, $f_1=0.25$, $f_2=0.27$, damped sinusoids

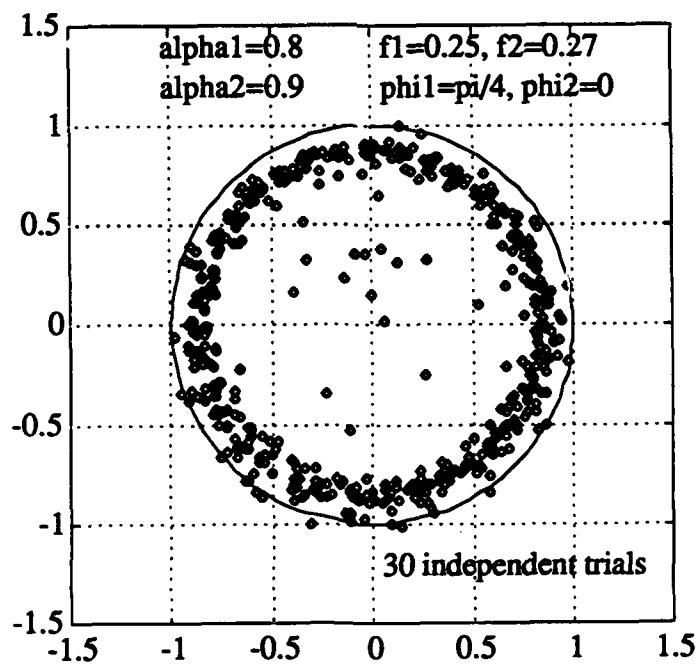


Figure 3.5.4 TLS solution for SNR=20dB, P=18
N=50, $f_1=0.25$, $f_2=0.27$, damped sinusoids

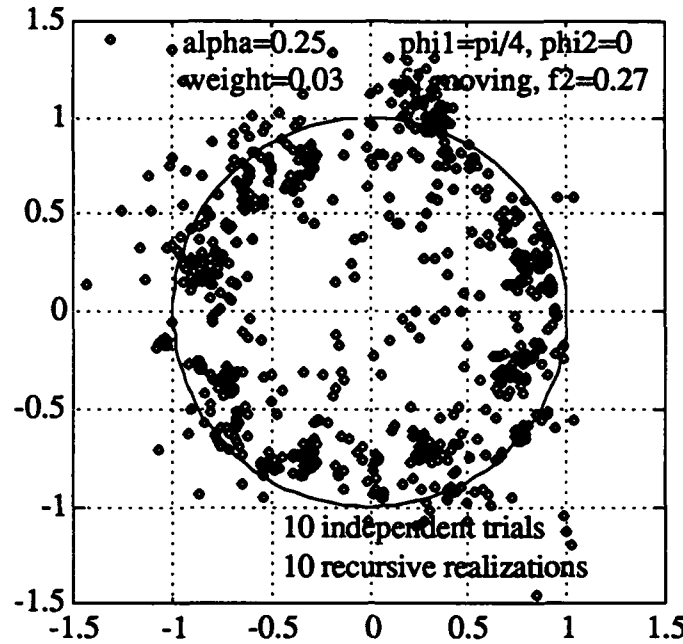


Figure 4.1.1 RTLS solution for SNR=12dB, P=8, N=100
 $\alpha=0.25$, weight=3%, f_1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'S'

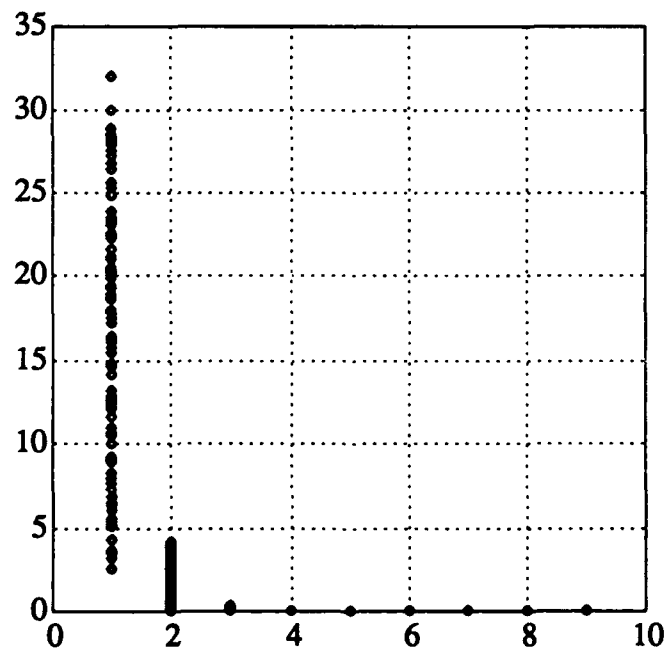


Figure 4.1.2 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, $\alpha=0.25$, weight=3%, f_1 is time-varying
solution derived by using the signal subspace information

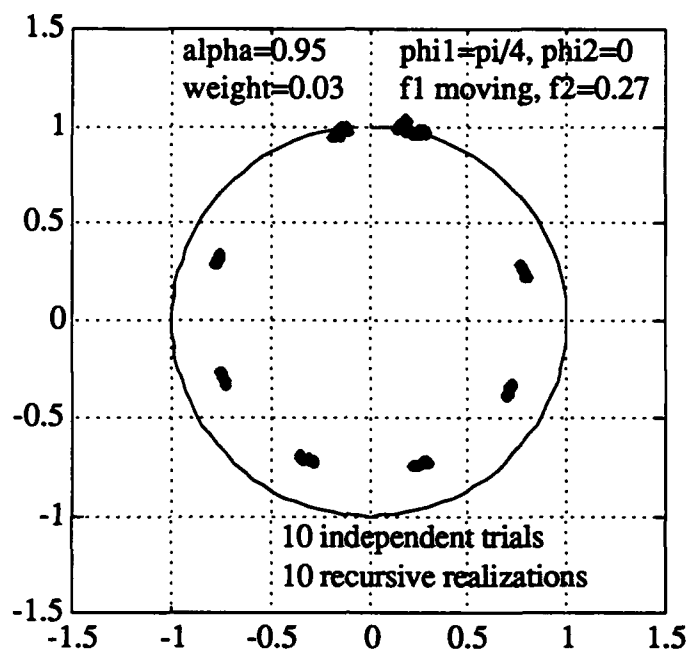


Figure 4.1.3 RTL solution for SNR=12dB, P=8, N=100
 $\alpha=0.95$, weight=3%, f_1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'S'

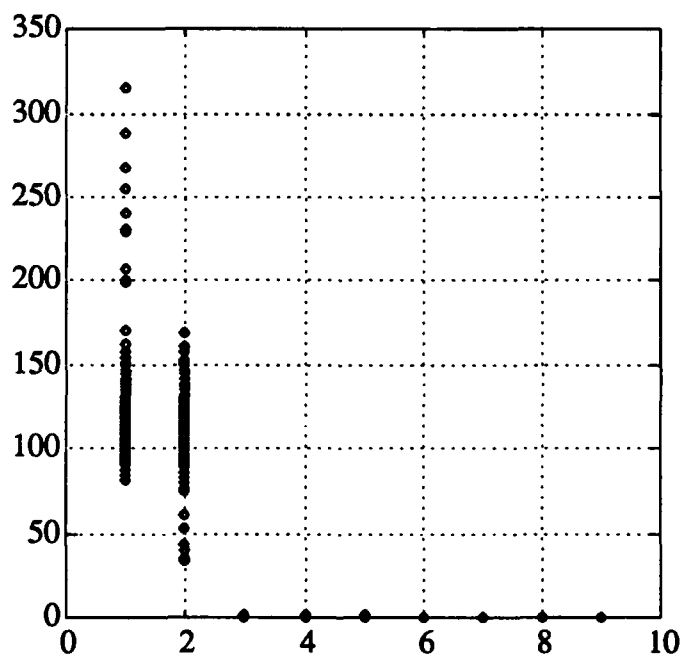


Figure 4.1.4 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, $\alpha=0.95$, weight=3%, f_1 is time-varying
solution derived by using the signal subspace information

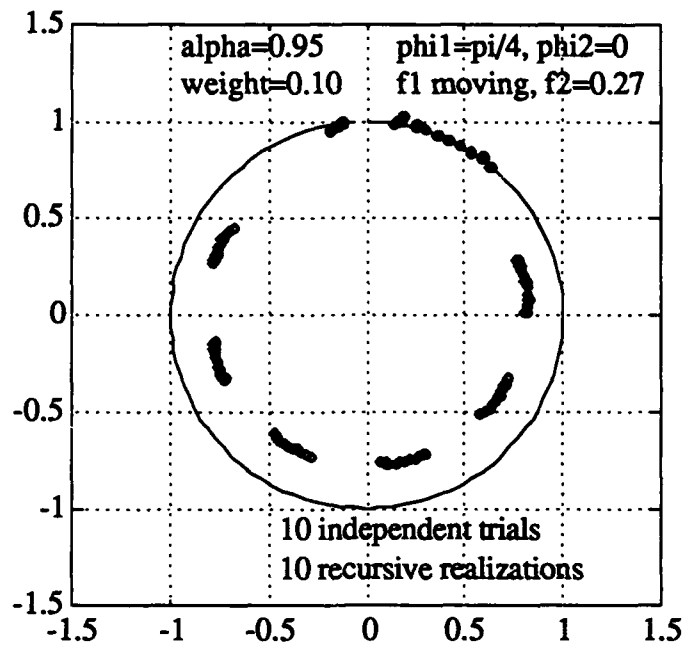


Figure 4.2.1 RTLS solution for SNR=12dB, P=8, N=100
 $\alpha=0.95$, weight=10%, f_1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'S'

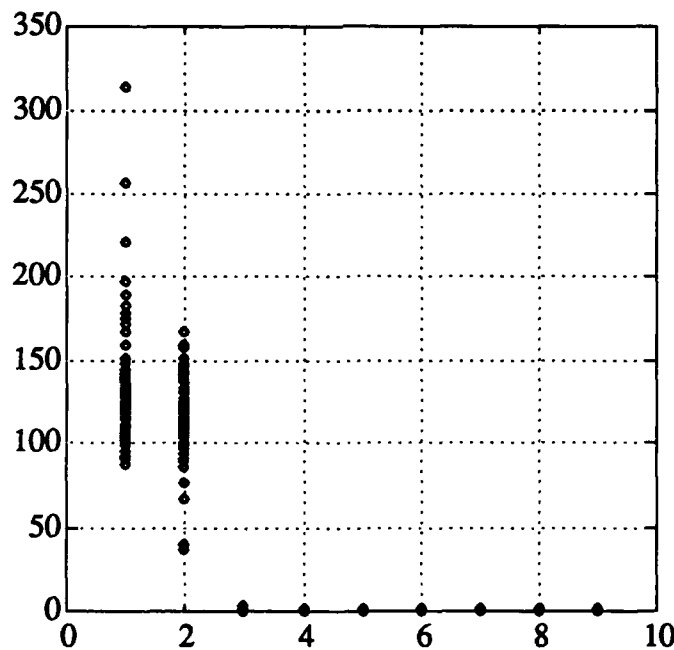


Figure 4.2.2 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, $\alpha=0.95$, weight=10%, f_1 is time-varying
solution derived by using the signal subspace information

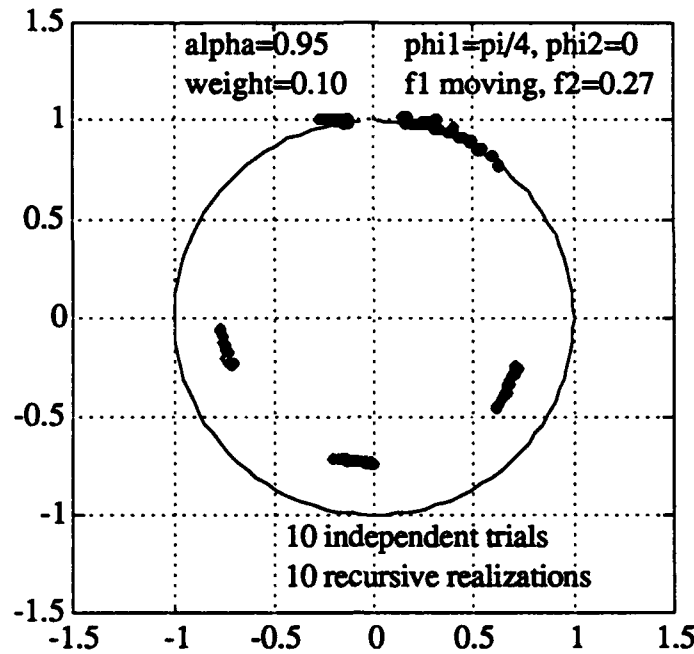


Figure 4.3.1 RTLS solution for SNR=12dB, P=5, N=100
 $\alpha=0.95$, weight=10%, f_1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'S'

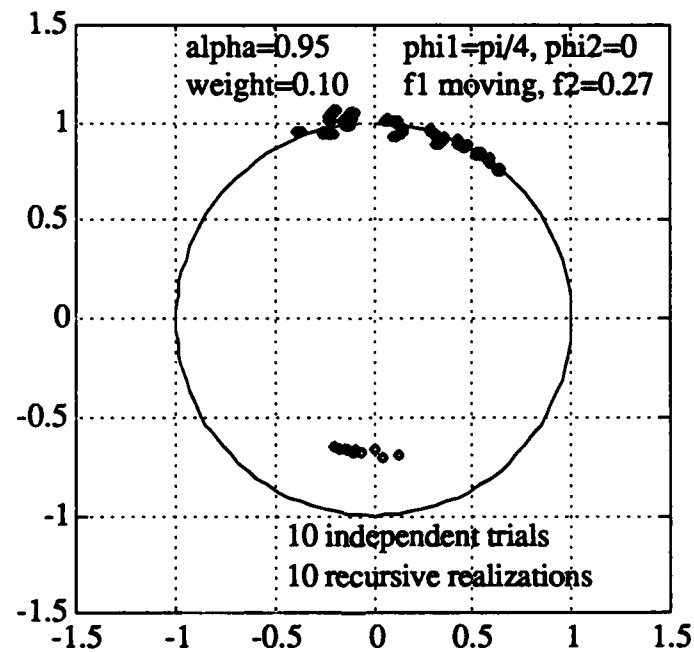


Figure 4.3.2 RTLS solution for SNR=12dB, P=3, N=100
 $\alpha=0.95$, weight=10%, f_1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'S'

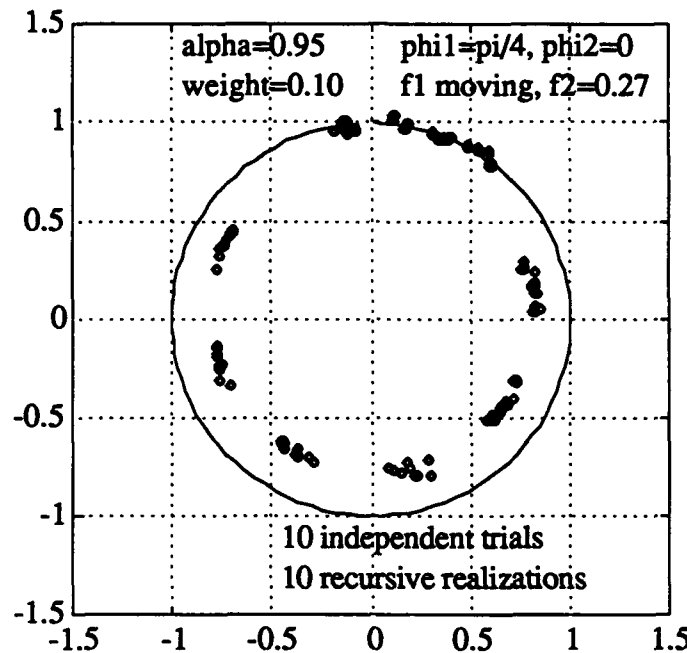


Figure 4.4.1 RTLS solution for SNR=0dB, P=8, N=100
 $\alpha=0.95$, weight=10%, f1 is time-varying, using
 the signal subspace information & performing
 the eigendecomposition of spherical matrix 'S'

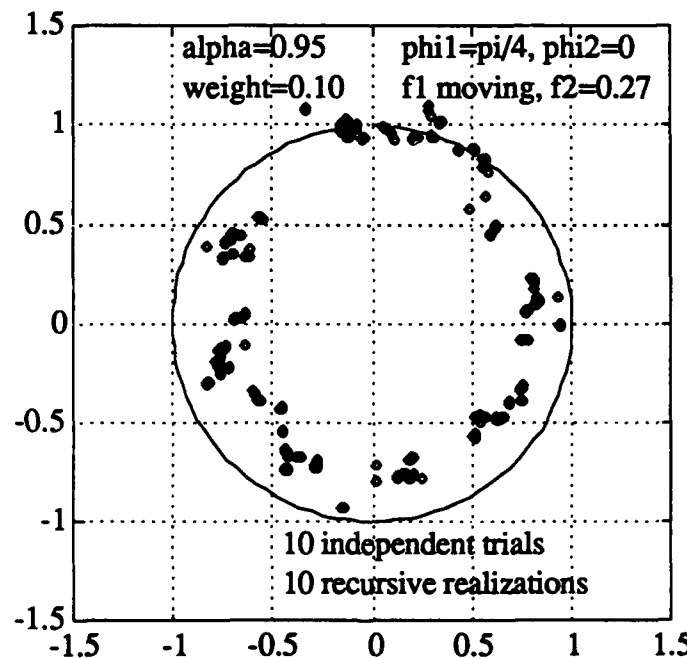


Figure 4.4.2 RTLS solution for SNR=-5dB, P=8, N=100
 $\alpha=0.95$, weight=10%, f1 is time-varying, using
 the signal subspace information & performing
 the eigendecomposition of spherical matrix 'S'

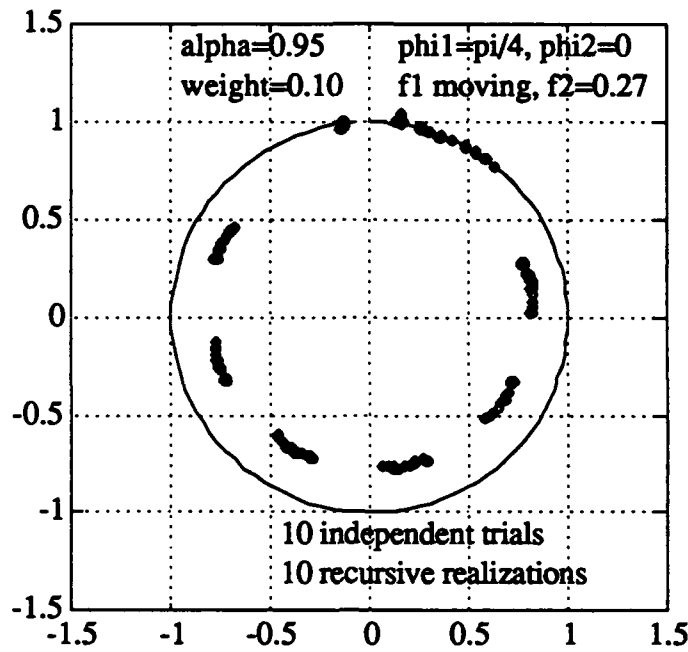


Figure 4.5.1 RTLS solution for SNR=12dB, P=8, N=100
 $\alpha=0.95$, weight=10%, f_1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'R'

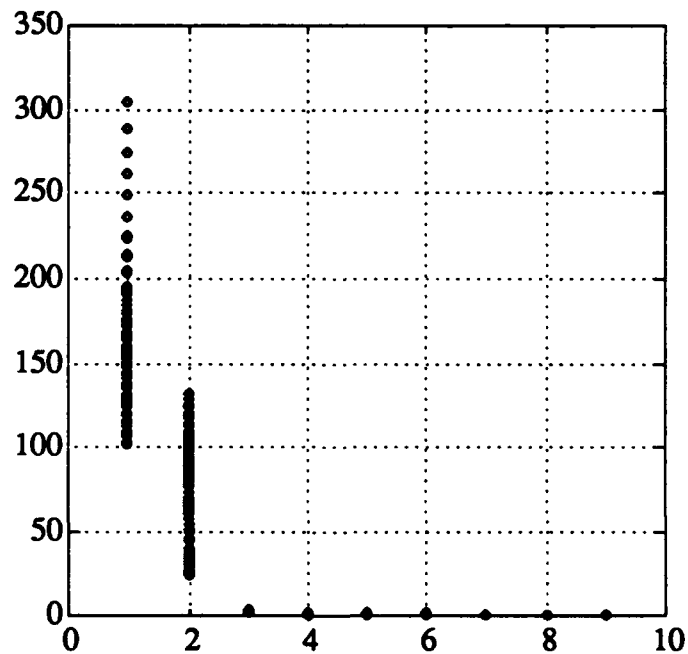


Figure 4.5.2 non-averaged eigenvalues of correlation matrix 'R'
SNR=12dB, P=8, N=100, $\alpha=0.95$, weight=10%, f_1 is time-varying
solution derived by using the signal subspace information

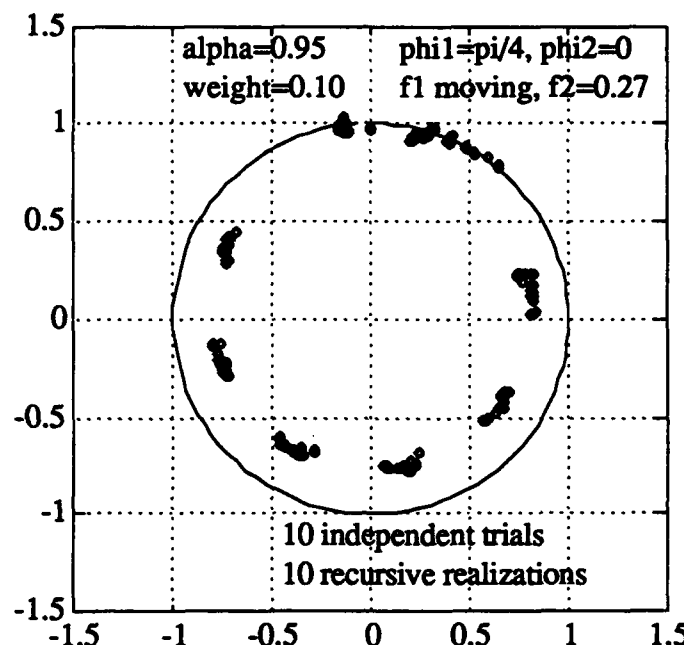


Figure 4.6.1 RTLS solution for SNR=0dB, P=8, N=100
 $\alpha=0.95$, weight=10%, f1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'R'

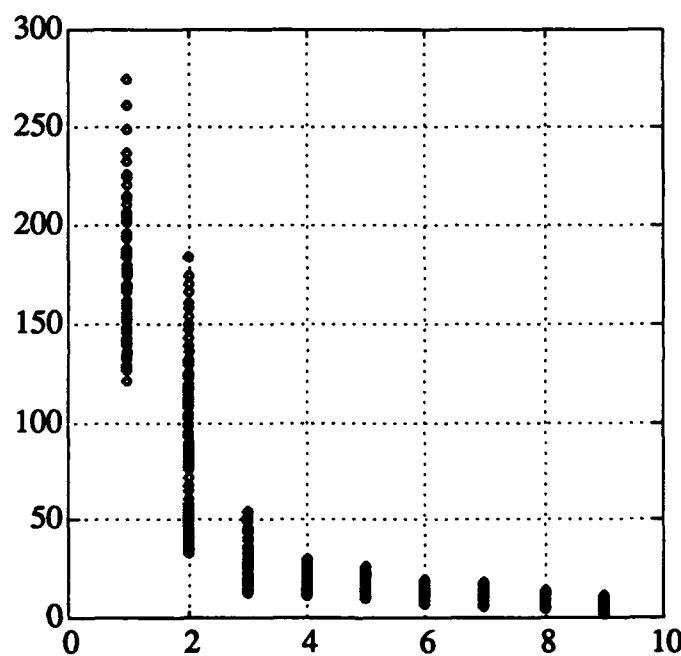


Figure 4.6.2 non-averaged eigenvalues of correlation matrix 'R'
SNR=0dB, P=8, N=100, $\alpha=0.95$, weight=10%, f1 is time-varying
solution derived by using the signal subspace information

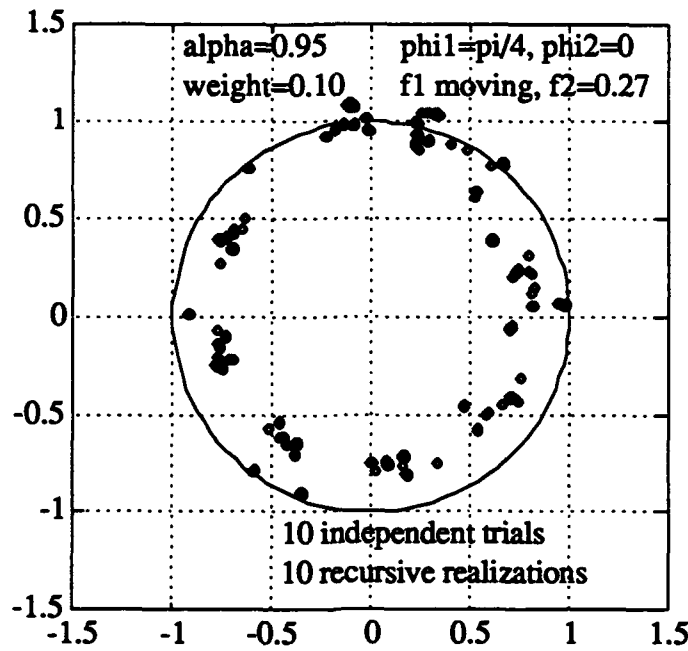


Figure 4.6.3 RTLS solution for SNR=-5dB, P=8, N=100
 $\alpha=0.95$, weight=10%, f_1 is time-varying, using
the signal subspace information & performing
the eigendecomposition of spherical matrix 'R'

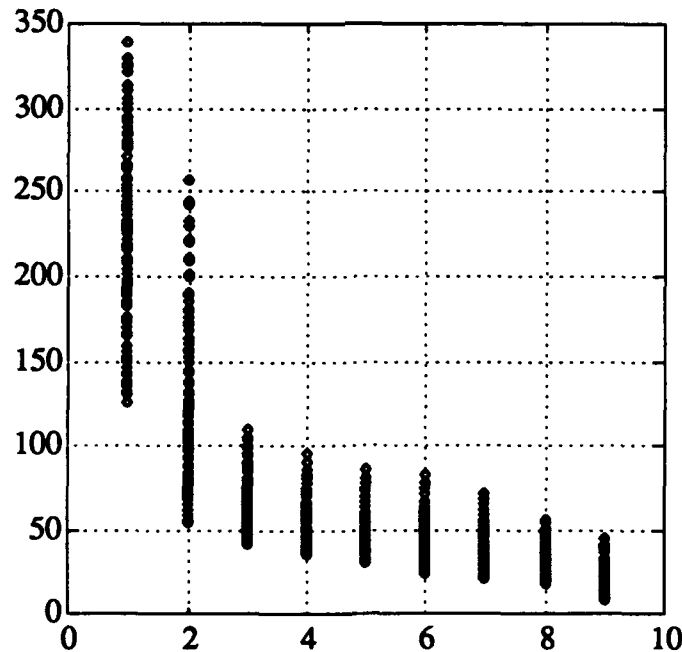


Figure 4.6.4 non-averaged eigenvalues of correlation matrix 'R'
SNR=-5dB, P=8, N=100, $\alpha=0.95$, weight=10%, f_1 is time-varying
solution derived by using the signal subspace information

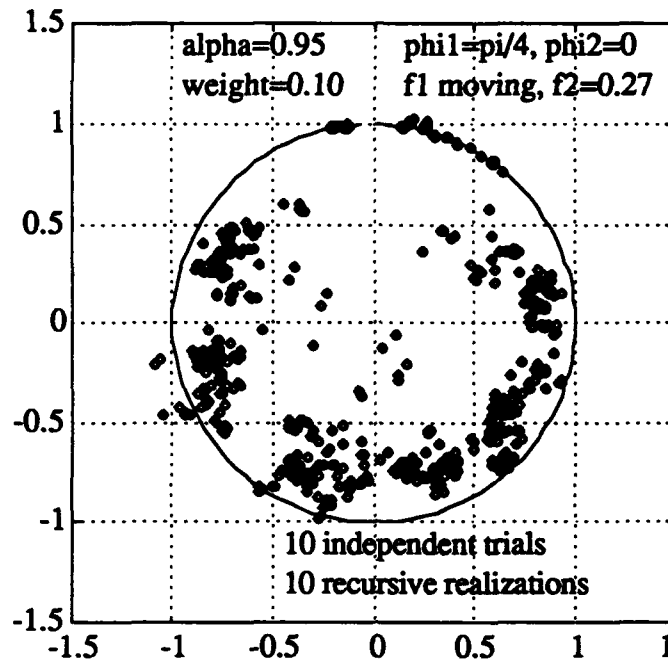


Figure 4.7.1 RTLS solution for SNR=12dB, P=8, N=100
 $\alpha=0.95$, weight=10%, f1 is time-varying, using
the noise subspace information & performing
the eigendecomposition of spherical matrix 'S'

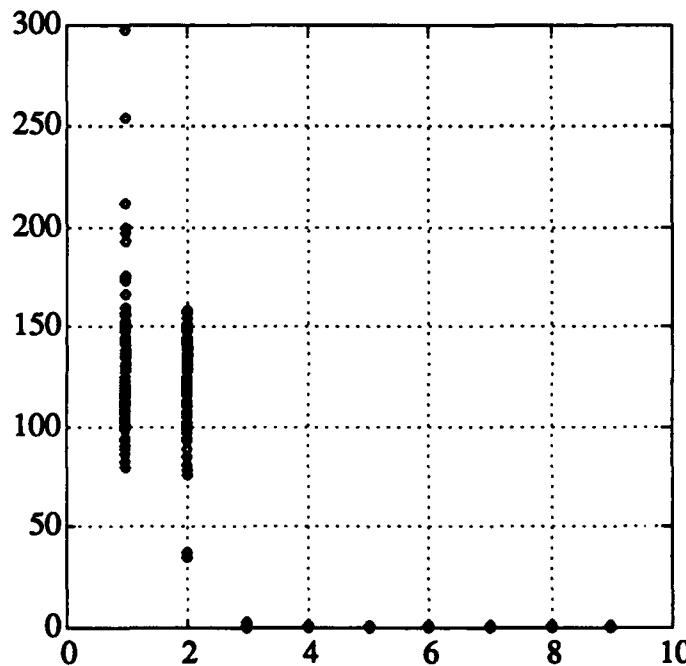


Figure 4.7.2 averaged eigenvalues of updated spherical matrix 'S'
SNR=12dB, P=8, N=100, $\alpha=0.95$, weight=10%, f1 is time-varying
solution derived by using the noise subspace information

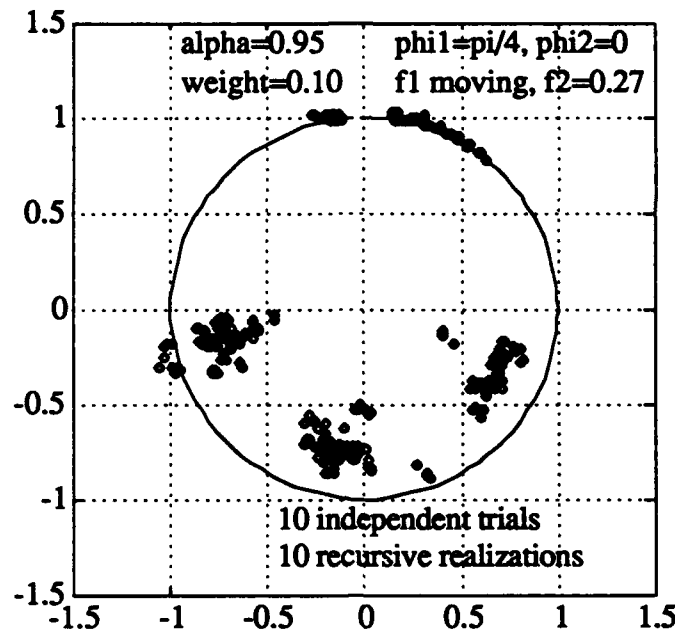


Figure 4.8.1 RTLS solution for SNR=12dB, $P=5$, $N=100$
 $\alpha=0.95$, weight=10%, f_1 is time-varying, using
the noise subspace information & performing
the eigendecomposition of spherical matrix 'S'

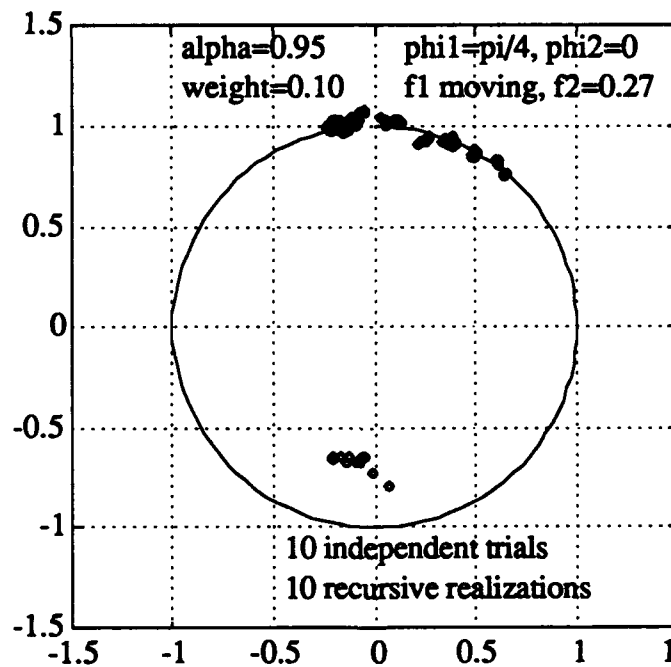


Figure 4.8.2 RTLS solution for SNR=12dB, $P=3$, $N=100$
 $\alpha=0.95$, weight=10%, f_1 is time-varying, using
the noise subspace information & performing
the eigendecomposition of spherical matrix 'S'

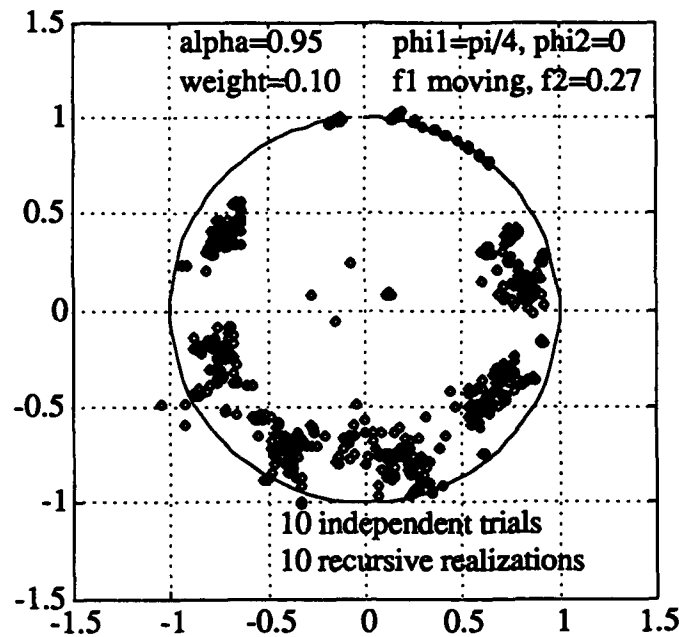


Figure 4.9.1 RTLS solution for SNR=15dB, $P=8$, $N=100$
 $\alpha=0.95$, $\text{weight}=10\%$, f_1 is time-varying, using
the noise subspace information & performing
the eigendecomposition of spherical matrix 'S'

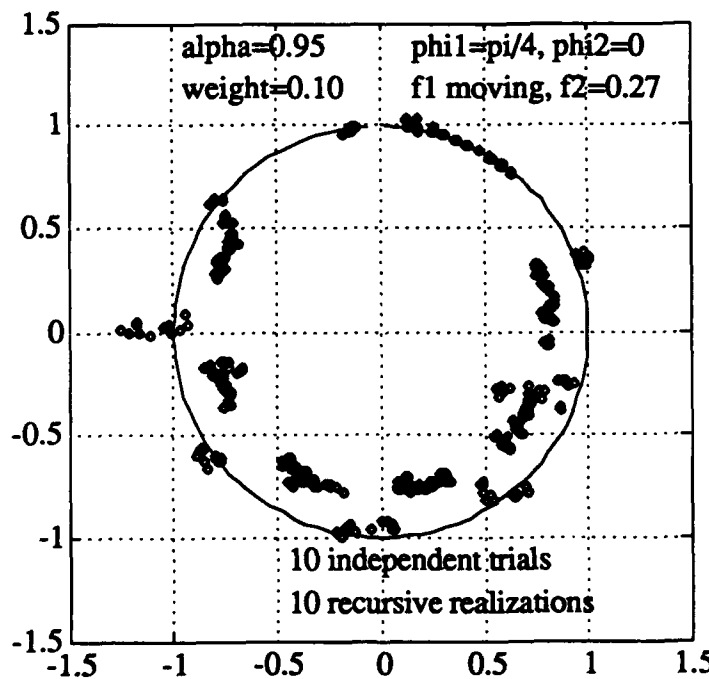


Figure 4.9.2 RTLS solution for SNR=30dB, $P=8$, $N=100$
 $\alpha=0.95$, $\text{weight}=10\%$, f_1 is time-varying, using
the noise subspace information & performing
the eigendecomposition of spherical matrix 'S'

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

```
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;
```

```

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)); %create two undamped signals
s2=exp(j*(2*pi*f2*n+phi2));
s1=exp(-alpha1*n+j*(2*pi*f1*n+phi1)); %create two damped signals
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); %generate observation vector
    bb=conj(b)';
end
C=[bb A]; %create augmented matrix
[u,s,v]=svd(C); %perform singular value decom.
sigma=diag(s);
ct=v(1,nsin+1:p+1); %extract the desired vector
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(1,h)=new_y11(k1,h);
        estd1=sqrt(1/nn*(sum(abs(esti_y11(1,h))...
            -(0.1253+j*0.9921) .*ones(1,nn)) .^2)));
    else
        y2(n1,h)=rt(n1,h);
        y22=sort(y2);
        y22=flipud(y22);

```

```

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(esti_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_y11),imag(new_y11),'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 ');

```

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

```

clg
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;           %make f1 time-varying moving
    f2=0.27;
    phi1=pi/4;
    phi2=0;

```

```

j=sqrt(-1);
wn=w1+j*w2;                                %independent complex noise
s1=exp(j*(2*pi*f1*n1+phi1));                %create two undamped signals
s2=exp(j*(2*pi*f2*n1+phi2));
ss=s1+s2;
xn=ss+sqrt(1/(2*10^(SNR/10))).*wn;          %received resulted signal
nsin=2;                                     %for two sinusoidal sources
n2=50;                                     %window data sample number
if n2>p+2,
    for k=1:n2-p,
        A(k,1:p)=xn(k:p+k-1);              %generate data matrix
    end
    b=xn(1,p+1:n2);                         %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;                                     %create augmented matrix
R=C'*C;                                   %define estd. correlation matrix
[U,D]=eig(R);                             %perform eigendecomposition
for t=1:ni,                                %recursively update subspace
    y=xn(1,n2-p+t:n2+t);
    x=y';

```

```

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;
gt=Unew(1,1:nsin);                        %extract the desired vector
g=conj(gt)';
u1p=Unew(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 D2'];

```

```

sig=[sig s_diaspa];
noi=[noi n_diaspa];
U=Unew;                                %assign new U & D for updating
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 singularvalues 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 ' R '

Note that $R = \alpha * R + (1 - \alpha) * x * x^H$

```
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=pi/4;
    phi2=0;
    j=sqrt(-1);
```

wn=w1+j*w2;	%independent complex noise
s1=exp(j*(2*pi*f1*n1+phi1));	%create two undamped signals
s2=exp(j*(2*pi*f2*n1+phi2));	
ss=s1+s2;	
xn=ss+sqrt(1/(2*10^(SNR/10))).*wn;	%received resulted signal
nsin=2;	%for two sinusoidal sources
n2=50;	%window data sample number
if n2>p+2,	
for k=1:n2-p,	
A(k,1:p)=xn(k:p+k-1);	%generate data matrix
end	
b=xn(1,p+1:n2);	%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;	%create augmented matrix
R=C'*C;	%define estd. correlation matrix
[U,D]=eig(R);	%perform eigendecomposition
for t=1:ni,	%recursively update subspace
y=xn(1,n2-p+t:n2+t);	
x=y';	
R=alpha.*R+(1-alpha).*x*x';	%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 '

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

```

clg
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;           %make f1 time-varying moving
    f2=0.27;
    phi1=pi/4;
    phi2=0;

```

```

j=sqrt(-1);
wn=w1+j*w2;                                %independent complex noise
s1=exp(j*(2*pi*f1*n1+phi1));                %create two undamped signals
s2=exp(j*(2*pi*f2*n1+phi2));
ss=s1+s2;
xn=ss+sqrt(1/(2*10^(SNR/10))).*wn;          %received resulted signal
nsin=2;                                     %for two sinusoidal sources
n2=50;                                     %window data sample number
if n2>p+2,
    for k=1:n2-p,
        A(k,1:p)=xn(k:p+k-1);              %generate data matrix
    end
    b=xn(1,p+1:n2);                         %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;                                     %create augmented matrix
R=C'*C;                                   %define estd. correlation matrix
[U,D]=eig(R);                             %perform eigendecomposition
for t=1:ni,                                %recursively update subspace
    y=xn(1,n2-p+t:n2+t);
    x=y';

```

```

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 -fliplr(conj(xtls)')];
rt=[rt roots(z)];
esp=[esp D2'];

```

```

sig=[sig s_diaspa];
noi=[noi n_diaspa];
U=Unew; %assign new U & D for updating
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 singularvalues status of updated matrix
clg
axis;
n3=1:p+1;
plot(n3,esp,'o'),grid

```

REFERENCES

- [1] Charles W. Therrien, Discrete Random Signals and Statistical Signal Processing, Prentice Hall, March 1992.
- [2] Gene H. Golub and Charles F. Van Loan, "*An Analysis of the Total Least Squares Problem*," SIAM J. Numer. Anal., Vol. 17, Dec. 1980, pp.883-893.
- [3] Eric M. Dowling and Ronald D. DeGroat, "*The Equivalence of the Total Least Squares and Minimum Norm Methods*," IEEE Trans on Signal Processing, Vol. 39, Aug. 1991, pp.1891-1892.
- [4] Simon Haykin, Adaptive Filter Theory, 2nd edition, Prentice Hall, 1991.
- [5] Md. Anisur Rahman and Kai-Bor Yu, "*Total Least Squares Approach for Frequency Estimation using Linear Prediction*," IEEE trans. ASSP, Vol. 35, Oct. 1987, pp.1440-1454.
- [6] Theagenis J. Abatzoglou, Jerry M. Mendel and Gail A. Harada, "*The Constrained Total Least Squares Technique and its Applications to Harmonic Superresolution*," IEEE ASSP, Vol. 39 No.5, May 1991, pp.1070-1086.
- [7] C.M. Rader and A.O. Steinhardt, "*Hyperbolic Householder Transformations*," IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-34, pp.1589-1602, Dec. 1986.

AD-A257 562

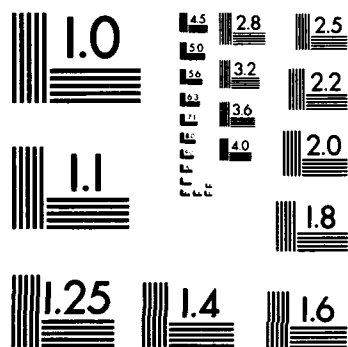
APPLICATION OF THE TOTAL LEAST SQUARES TECHNIQUE TO
FREQUENCY ESTIMATION(U) NAVAL POSTGRADUATE SCHOOL
MONTEREY CA C CHIANG SEP 92 KB-NPS

272

UNCLASSIFIED

NL

END
FILMED
DTIC



- [8] David Starer and Arye Nehorai, "*Maximum Likelihood Estimation of Exponential Signals in Noise using a Newton Algorithm*," in Proc. 4th ASSP Workshop Spectrum Estimation Modeling (Minneapolis, MN), Aug. 1988, pp.240-245.
- [9] Yoram Bresler and Albert Macovski, "*Exact Maximum Likelihood Parameter Estimation of Superimposed Exponential Signals in Noise*," IEEE Trans. Acoust. Speech Signal Processing, Vol. ASSP-34, pp.1081-1089, Oct. 1986.
- [10] Eric M. Dowling, Ronald D. DeGroat, and Darel A. Linebarger, "*Total Least Squares with Linear Constraints*," IEEE-ICASSP-92 proceedings, May 1992, V.341-V344.
- [11] E.M. Dowling and R.D. DeGroat, "*Recursive Total Least Squares Adaptive Filtering*," Proceedings of the SPIE Internat. Symposium on Optical Science and Engineering, July 1991.
- [12] R.D. DeGroat and E.M. Dowling, "*Non-iterative subspace updating*," Proceedings of the SPIE Internat. Symposium on Optical Applied Science and Engineering, San Diego CA, July 1991.

INITIAL DISTRIBUTION LIST

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