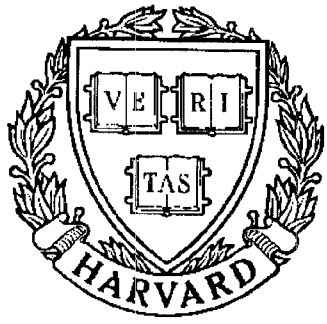
TECHNICAL RESEARCH REPORT

SYSTEMS RESEARCH CENTER



Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

An ESPRIT Algorithm for Tracking Time-Varying Signals

by K.J.R. Liu, D.P. O'Leary, G.W. Stewart, and Y-J. Wu

TR 92-54

An ESPRIT Algorithm for Tracking Time-Varying Signals

K. J. R. Liu*and Dianne P. O'Leary†andG. W. Stewart‡andYuan-Jye Wu §

April 24, 1992

Abstract

ESPRIT is a successful algorithm for determining the constant directions of arrival of a set of narrowband signals on an array of sensors. Unfortunately, its computational burden makes it unsuitable for real time processing of signals with time-varying directions of arrival. In this work we develop a new implementation of ESPRIT that has potential for real time processing. It is based on a rank-revealing URV decomposition, rather than the eigendecomposition or singular value decomposition used in previous ESPRIT algorithms. We demonstrate its performance on simulated data representing both constant and time-varying signals. We find that the URV-based ESPRIT algorithm (total least squares variant) is effective for time-varying directions-of-arrival using either rectangular or exponential windowing techniques to diminish the effects of old information.

*Department of Electrical Engineering and Systems Research Center, University of Maryland, College Park, MD 20742

†Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742. This work was supported in part by the Institute for Mathematics and Its Applications at the University of Minnesota, AFOSR Grant 87-0158, and the Graduate School General Research Board of the University of Maryland.

‡Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742.

§Applied Mathematics Program, University of Maryland, College Park, MD 20742. This work was supported by AFOSR Grant 87-1058.

1 Introduction

The ESPRIT algorithm [8] is a very successful means of determining directions-of-arrival (DOA) of a set of narrowband signals impinging on an array of sensors with translational invariance. It handles array geometries almost as general as those of the MUSIC algorithm [9] at a significant computational savings.

A key limitation of both the MUSIC and ESPRIT algorithms is that they require $O(nm^2)$ work to process n data samples from an array of m sensors. Unfortunately, at the heart of each algorithm is the solution of an eigenvalue problem or singular value problem, and there is no good way to economically update the previous information in order to reduce this computational cost when new signal information arrives, so processing one new data sample also requires $O(nm^2)$ or $O(m^3)$ work. This burden makes both algorithms unsuitable for real-time computation. Even using parallel processing $O(m^2)$ processors are required to achive $O(m)$ time.

Some attempts have been made to reduce the updating complexity by maintaining an approximate singular value decomposition (e.g., [5]), but we believe that better results can be obtained using an alternate decomposition.

Recently, Stewart [10] has introduced a new matrix decomposition, the rank-revealing URV decomposition, that requires somewhat less computation than either a singular value decomposition or an eigendecomposition but reveals much of the same information. A key advantage of the URV decomposition is that it can be updated in time $O(m)$ on an array of m processors. This means that algorithms that previously depended on eigendecomposition or singular value decomposition might now be practical for real time applications if the URV decomposition can be successfully substituted. Boman, Griffen, and Stewart [1] have already exploited this fact in accelerating the MUSIC algorithm. The purpose of the present work is to investigate the use of the

URV algorithm in time-varying signal processing using ESPRIT. Roy and Kaliath [8] noted that the use of the eigendecomposition or singular value decomposition was in some sense unnecessary for ESPRIT, because all that is required is a basis for the range space of a certain matrix. They lacked a more economical matrix decomposition that reliably provided this information. We show in this work that the rank-revealing URV decomposition is the appropriate tool.

The following two sections give brief descriptions of the total least squares ESPRIT algorithm and the URV decomposition. Section 4 discusses the algorithm TV-ESPRIT , a time-varying implementation of the ESPRIT algorithm. Experimental results are summarized in §5, and computational details of the URV decomposition are given in an appendix.

2 The ESPRIT Algorithm

Consider d narrow-band plane waves simultaneously incident on a planar array of m sensors, arranged in $m/2$ doublet pairs, where m is an even integer. The displacement Δ between sensors in a pair is a constant, but the location of each pair is arbitrary. The wave sources are assumed to be located in the same plane, and the location of each source is specified by a single parameter $\theta_i \in [0, 2\pi]$, the DOA of the ith source.

For notational convenience, we will denote data related to the first sensor in each pair by a subscript X, and data related to the second by Y, and all vectors will be column vectors.

Given data from the array of sensors, the DOA estimation problem is to locate the directions of the sources of radiating energy that is being detected by the sensors. If the signals are assumed to be narrow-band processes, with the same known center frequency ω_0, then the DOA problem can be described by a simple model: the relationship between the unknown signal $\underline{s}(t) \in \mathcal{C}^d$ and

the sensor output $\underline{z}_X(t) \in \mathcal{C}^{m/2}$ and $\underline{z}_Y(t) \in \mathcal{C}^{m/2}$ is given by

$$\underline{z}_X(t) = A\underline{s}(t) + \underline{\epsilon}_X(t), \tag{1}$$

$$\underline{z}_Y(t) = A\Phi\underline{s}(t) + \underline{\epsilon}_Y(t), \tag{2}$$

or

$$\underline{z}(t) = \begin{pmatrix} A \\ A\Phi \end{pmatrix} \underline{s}(t) + \underline{\epsilon}(t) \tag{3}$$

where $\underline{\epsilon}(t)$ is the measurement noise, and $A \in \mathcal{C}^{m/2 \times d}$ is the unknown matrix of *array responses* or *array steering vectors*, dependent on the directions of arrival. The diagonal matrix Φ is also unknown, and is related to the phase delays between the sensors in each doublet pair:

$$\phi_i = e^{j\omega_0 \Delta \sin\theta_i / c}, \quad i = 1, ..., d. \tag{4}$$

Our task is to estimate the number of signals d and the directions-of-arrival, $\theta_i, i = 1, \cdots, d$. For this, it is sufficient to estimate the matrix Φ, and this is the underlying idea in ESPRIT .

2.1 ESPRIT for constant source directions

The ESPRIT algorithm exploits the array geometry in the following way. In the absence of noise (i.e., $\underline{\epsilon} = \underline{0}$), the range of the matrix A is the same as the range of the matrix Z_X formed from n columns of sensor outputs (n sufficiently large) and that is the same as the range of Z_Y, taken over the same time interval. From the information in the basis vectors for the range, it is possible to construct a matrix that is similar to the unknown diagonal matrix Φ, and thus determine Φ from the eigenvalues. Determining the basis vectors and computing the matrix similar to Φ can be done in many different ways, but each variant of ESPRIT has these two phases:

A. Obtain a basis for the range space of A.

B. Compute Φ by finding the eigenvalues of a matrix similar to Φ, and compute the DOAs from this.

Phase A contains the bulk of the computational work. Two different approaches have been taken to determining the range space basis: computing a singular value decomposition of the data matrix Z^H or computing an eigendecomposition of the estimated covariance matrix

$$\hat{R}_{ZZ} = \frac{1}{n}\sum_{t=1}^{n} \underline{z}(t)\underline{z}^H(t) = \frac{1}{n} Z Z^H \ . \tag{5}$$

For comparison, we describe one previous version of the ESPRIT algorithm [8], one based on the singular value decomposition. The SVD of the matrix Z^H is

$$Z^H = U \Sigma V^H,$$

where U is an $n \times m$ matrix and V is a $m \times m$ matrix, both having orthonormal columns. The matrix Σ is diagonal containing the singular values in descending order: $\sigma_1 \geq \cdots \geq \sigma_m$. The first d columns of V form the required basis. Note that U, which has the same dimensions as Z^H, is not needed in the analysis and can be discarded.

The first four steps correspond to Phase A; the remaining ones process the range-space basis. Our URV-based algorithm, presented in §4, differs in the steps marked with asterisks.

Algorithm: ESPRIT(SVD)

1) Obtain the data measurements Z.

2*) Compute the singular value decomposition of Z^H:

$$Z^H = U \Sigma E^H.$$

3*) Estimate the number of sources d (the rank of Z^H) using An Information Criterion (AIC) or the Minimum Description Length (MDL) criterion.

4) The basis for the signal subspace (the range of Z^H) is E_Z, equal to the first d columns of E.

5*) Partition E_Z into $m/2 \times d$ blocks as

$$E_Z = \begin{pmatrix} E_X \\ E_Y \end{pmatrix}$$

and compute the singular value decomposition of (E_X, E_Y):

$$(E_X, E_Y) = W \Psi V^H.$$

6) Partition V into $d \times d$ blocks as

$$V = \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix}$$

and calculate ϕ_i, the eigenvalues of $-V_{12}V_{22}^{-1}$.

7) Estimate the DOAs from ϕ_i using (4).

Computing the SVD of the data matrix becomes impractical if the matrix is too large. An alternate approach is to work with the sample covariance matrix $\widehat{R}_{ZZ}$, which is $m \times m$. The eigenvectors corresponding to the d largest eigenvalues of $\widehat{R}_{ZZ}$ form a basis for the approximate rangespace.

A different approach which might be used is to compute a rank-revealing QR factorization of the data matrix Z^H [3]. The goal of this algorithm is to factor Z^H as

$$Z^H = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}$$

where R_{22} is $(m-d) \times (m-d)$ and small in norm. This is done using a two-phase procedure, in which the standard QR algorithm (perhaps using pivoting) is used to compute an initial factorization. Then the presence of a small singular value in R_{11} is detected by condition estimation

algorithms. If such a value is found, the dimension of R_{22} is increased and a permutation and update of R is performed to create small elements and move them to R_{22}. The condition estimation process is repeated as necessary. The columns of Q form the basis for the range space of Z^H, but the orthogonal basis for the range of Z is not directly available.

Note that the basis for the null space of a matrix is not really needed in Step 6 of the algorithm. Since $V^H V = I$, we know that $V_{12}^H V_{11} + V_{22}^H V_{21} = 0$, and $-V_{12}V_{22}^{-1} = V_{11}^{-H} V_{21}^H$, so a basis for the range space suffices.

The SVD , eigendecomposition, and QR approaches can all yield reliable estimates of the range space, but they have drawbacks. None is suitable for time-varying computation, since none of the decompositions can be updated efficiently as new data arrive. Parallelization of the methods seems to require an array of $O(m^2)$ processors in order to achieve $O(m)$ or $O(n)$ time. Recently, less computationally expensive approximations to the singular value decomposition have been proposed, but it is not clear how they behave with transient data [2, 5, 6, 7]. Moreover, these approximations still require $O(m^2)$ processors to update in parallel.

The main goal of our work is to develop a technique for Phase A that is similar to the singular value decomposition in reliability but faster, more parallel, and easier to update.

2.2 Requirements for a Time-Varying ESPRIT algorithm

In the time dependent problem, the sensors receive a new data sample at each time unit. There are two common approaches to discounting the old data in order to develop reliable estimates of the current DOAs.

The first is the *(rectangular) windowing method.* In this method, only the most recent n data samples are retained, and earlier ones are discarded as being irrelevant. One way to express this is to multiply the data z_i, the data collected at time i, by a rectangular *window function* of fixed

dimension n defined as

$$w_{n,N}(i) = \begin{cases} 1 & i = N, N-1, \ldots, N-n+1 \\ 0 & \text{otherwise} \end{cases} .$$

This function works like a window frame that only admits n pieces of data, and we shift it forward every trial to use the most recent n samples of data. At time N, we work with the data samples

$$(z_{N-n+1}, ..., z_{N-1}, z_N).$$

The second approach uses *forgetting factors* to discount the data in a more gradual way. As each new data sample is received, the old data is multiplied by a number μ between 0 and 1. In this method, all preceeding data is multiplied by an exponential window function, so at time N we work with the data samples

$$(\mu^{N-1} z_1, ..., \mu\, z_{N-1}, z_N)$$

Since the dimension of this matrix is growing, it is essential that our numerical technique works with a compressed form of the data, a matrix whose dimension is independent of N.

Updates to the the range space cannot be computed in real time using either the eigendecomposition or the singular value decomposition, so we turn to a new decomposition, the rank-revealing URV.

3 The URV Decomposition

The determination of an approximate range or nullspace for an $n \times m$ matrix Z^H is a recurring problem in many areas of signal processing. We will assume that Z^H is a matrix consisting of true values plus measurement errors, and that in the absence of error, Z^H would have rank $d < m$.

The rank revealing URV decomposition expresses Z^H in the form

$$Z^H = (U_Z \; U_\perp) \begin{pmatrix} R & F \\ 0 & G \end{pmatrix} \begin{pmatrix} V_Z^H \\ V_\perp^H \end{pmatrix} \equiv U \bar{R} V^H,$$

where the columns of U and V are orthonormal, R and G are upper triangular of orders d and $m - d$, and

$$\|F\|_F^2 + \|G\|_F^2 \leq \epsilon^2.$$

This decomposition reveals that Z^H is within ϵ of a matrix of rank d (distance measured in the Frobenius norm). From this it is seen that $\|Z^H V_\perp\|_F \leq \epsilon$, so that V_Z is a basis for the approximate rangespace of the true (errorless) data matrix and $V_\perp$ is a basis for the approximate null space. Moreover, it is not necessary to carry U along to compute and update the decomposition, so that the storage requirements are modest.

The rank-revealing URV decomposition is not unique; the singular value decomposition and the rank-revealing QR factorization are both special cases, and tools such as plane rotations and condition estimators are common to the entire family. The usefulness of the URV approach hinges on choices that make it rank-revealing and inexpensive to update, requiring only the use of a fixed number of plane rotations. The presence of the matrix V (distinguishing URV from QR) ensures that an explicit basis for the approximate nullspace is available. The flexibility of using a triangular factor rather than a diagonal one (distinguishing URV from SVD) greatly improves the efficiency of the update.

The URV decomposition has the following properties [10]:

- It requires $O(m^2)$ storage.
- It furnishes the matrix $V_\perp$ explicitly.
- It can be updated in $O(m^2)$ time.

- With m processors, it can be updated in $O(m)$ time.

Given a rank revealing URV procedure, we can implement an ESPRIT algorithm based either on manipulation of Z^H or of $\hat{R}_{ZZ}$. Updating the factorization when new data arrive proceeds in three steps, described in detail in [10]. We add the new row to the $\bar{R}$ matrix and determine whether the matrices F or G have become too large in norm (based on a tolerance γ). If so, we tentatively increase our estimate of the rank and perform some rotations that change $\bar{R}$ and V. We then reduce $\bar{R}$ to upper triangular form without changing V. We need to decide whether R has become rank deficient; this is the standard *condition estimation problem.* A tolerance of $\tau\gamma$ is used to test for possible deficiency here. If there is a potential rank deficiency, then a pivoting procedure is performed, modifying $\bar{R}$ and V, to make the last column of R small. Then a refinement step is performed in order to bring the triangular matrix closer to diagonal form. It suffices to perform the rank-deficiency check at most twice per update.

For the forgetting factor method, the current matrix Z^H is replaced by

$$\begin{pmatrix} \mu Z^H \\ z^H \end{pmatrix},$$

where μ is the forgetting factor and z is the new data vector. Once we have a rank-revealing URV decomposition of the data matrix Z^H, it is an $O(m^2)$ process to compute the rank-revealing URV decomposition of the modified matrix, We expect the columns of F and G to have size approximately

$$\gamma \equiv \sqrt{\frac{(m-d)}{1-\mu^2}}\sigma_N,$$

(measured in the 2-norm), where σ_N is the standard deviation of the noise.

If the windowing method is used, one additional step is needed. We first downdate the decomposition by deleting the first data row in Z^H, and then perform the updating algorithm

with $\mu = 1$. The details of downdating are discussed in the appendix. Again, the total work is $O(m^2)$, but the most recent n data samples must be saved, increasing the storage requirements to $O(nm)$. We expect the columns of F and G to have size approximately

$$\gamma \equiv \sqrt{(m-d)n}\sigma_N,$$

so again one should choose a good estimate for σ_N for the rank test tolerance.

4 The TV-ESPRIT Algorithm

Algorithm: Time Varying ESPRIT(URV) Suppose that we already have a rank revealing URV decomposition of the data matrix from the previous time. For windowing, we also save the most recent n data samples.

1) Obtain the new data sample z.

2) Update the previous rank revealing URV decomposition by downdating and updating the previous factors if windowing is used, or updating the previous factors if forgetting factors are used.

3) Estimate the number of sources d (i.e., the rank of $\bar{R}$) using a tolerance of τ times the standard deviation of the noise. (The parameter τ is chosen by the user.)

4) The basis for the signal subspace (the range of Z^H) is E_Z, equal to the first d columns of the V factor in the URV decomposition.

5) Partition E_Z into $m/2 \times d$ blocks corresponding to the X and Y sensors:

$$E_Z = \begin{pmatrix} E_X \\ E_Y \end{pmatrix}$$

and compute a rank-revealing URV decomposition of (E_X, E_Y):

$$(E_X, E_Y) = W\Psi V^H.$$

6) Partition V into $d \times d$ blocks as

$$V = \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix}$$

and calculate ϕ_i, the eigenvalues of $-V_{12}V_{22}^{-1}$ (or, equivalently, the eigenvalues of $V_{11}^{-H}V_{21}^{H}$).

7) Estimate the DOAs from ϕ_i using (4).

In Step 3, we chose τ to be 3 or 6. In Step 5, we set $\tau = 1$, $\gamma = .5$ if the signal sources were separated by at least 10^o, and $\tau = 10$, $\gamma = .05$ to force more refinement of the subspace in case the signals were close.

The URV decomposition in Step 5 requires $O(d^3)$ time since we will get a different E_{XY} every trial and can not apply the updating technique to reduce the required operations. Thus the total time per step is $O(m^2 + d^3)$, and in practice the number of signal sources d is often much less than the number of sensors m. Even if d is large, Steps 1-3 require only $O(m^2)$, and thus it should be possible to keep up with the incoming data, and if necessary update the DOA estimates at less frequent intervals.

In comparison with the SVD algorithm, the basis for the signal subspace changes less frequently: only if the algorithm believes that the subspace is changing dimension. Thus if the signals are moving slowly compared with the sampling rate, we expect the URV-based algorithm to require many fewer updates of the DOAs than the SVD algorithm.

5 Experimental Results

In this section, we will present some simulation results that illustrate the performance of the two algorithms: ESPRIT(SVD) and ESPRIT(URV).

We use a five-pair ($m = 10$) linear array with pair spacing $\lambda/4$. The pairs are placed on a line of length 4λ with relative location $[0, 1, 3, 5.5, 7]$ $(\lambda/2)$. The two signals are narrow-band with signal to noise ratio (SNR) 23dB and 20dB, respectively. The noise is Gaussian, and the algorithms were tested with duplicate data samples in order to make a fair comparison.

We say that an algorithm *failed* at a particular time if it estimated more or fewer than two signals.

The first example concerned two fixed signal sources located at 24° and 29° and with 50% initial correlation. For each trial, we estimated the DOAs based on 100 data samples, and we ran 2000 trials. Figure 6.1 shows a histogram and tabular summary of the results. Both algorithms were quite successful. This shows that we are not sacrificing much accuracy by substituting the more economical URV for the SVD .

The second example tracked two signals with time-varying DOA using an exponential forgetting factor of 0.9. We considered two cases:

- close sources: periodic DOA of $10° + 5° \sin(2\pi n/360)$ and $20° + 5° \sin(2\pi n/240)$,
- well-separated sources: $-10° + 10° \sin(2\pi n/360)$ and $40° + 5° \sin(2\pi n/240)$,

where $n = 1, 3, \ldots, 719$. We ran 100 trials for each case. The *average error* was defined to be the mean of the differences between estimated and actual DOAs.

Figure 2 and 3 show that the DOA estimations from the two algorithms are quite similar. The error histogram in Figures 4 and 5 show that the variance of the relative error between actual and estimated DOAs for ESPRIT(URV) is quite small. When $n \in [375, 425]$, the DOA s of the close

sources are almost equal, and both algorithms were fooled into believing that there was a single source. In the figures, we plotted the results from the most recent successful trial, so we have straight lines for both tracks in this interval. Note that the SVD based algorithm is not a practical computational tool, since the data matrix grows far too large, but it provides a useful standard of comparison to assess the quality of the URV-based algorithm. Figure 6 shows the number of floating point operations as a function of the number of samples in a single trial. The work for the URV-based algorithm grows as m^2, while the work for the SVD-based algorithm grows as nm^2.

As a third example, we used the same data as in the second, but a rectangular window of size 20 rather than forgetting. The window shifted 2 points for each trial. Again, the results given in Figure 7 and 8 and error histograms given in Figure 9 and 10 show that the estimations are acceptable. Figure 11 plots the number of floating point operations as a function of the number of samples. Again it is clear that the work for the URV-based algorithm is quadratic, while that for the SVD algorithm grows cubicly.

As a final example, to demonstrate tracking of instantaneously changing signals, we assumed that there were two signal sources located at 24° and 29°, each with SNR 23 dB, and that the signals alternatively appear and disappear. We took a rectangular window of size 10. Figure 12 shows the similar performance of the two algorithms.

These experimental results lead us to believe that the URV-based ESPRIT algorithm can be successfully used for real-time tracking of time-varying signals.

6 Summary

We have presented a new variant of the ESPRIT algorithm that has potential for real-time tracking of moving signals. It has the following features:

- The storage requirement is $O(m^2)$ (plus mn for rectangular windowing).
- The work per update is $O(m^2 + d^3)$.
- It performs nearly as well as SVD-based algorithms, at greatly reduces computational cost, and admits an efficient parallel realization.

References

[1] E. C. Boman, M. F. Griffen, and G. W. Stewart. Direction of arrival and the rank-revealing URV decomposition. Technical report, Inst. for Advanced Computer Studies Report TR-91-166, Computer Science Department Report TR-2813, University of Maryland, College Park, 1991.

[2] J. R. Bunch and C. P. Nielsen. Updating the singular value decomposition. *Numerische Mathematik*, 31:111–129, 1978.

[3] T.F. Chan. Rank-revealing QR factorization. *Lin. Alg. and Its Applics.*, 88/89:67–82, 1987.

[4] J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart. *LINPACK User's Guide.* SIAM, Philadelphia, 1979.

[5] W. Ferzali and J. Proakis. Adaptive SVD algorithm with application to narrowband signal tracking. In R. J. Vaccaro, editor, *SVD and Signal Processing, II*, pages 149–160. Elsevier Science Publishers, Amsterdam, 1990.

[6] M. Moonen. *Jacobi-Type Updating Algorithms for Signal Processing, Systems Identification and Control.* PhD thesis, Katholieke Universiteit Leuven, 1990.

[7] M. Moonen, P. Van Dooren, and J. Vandewalle. Combined Jacobi-type algorithms in signal processing. In R. J. Vaccaro, editor, *SVD and Signal Processing, II*, pages 177–188, Amsterdam, 1990. Elsevier Science Publishers.

[8] R. Roy and T. Kailath. ESPRIT – estimation of signal parameters via rotational invariance techniques. In F. A. Grünbaum, J. W. Helton, and P. Khargonekar, editors, *Signal Processing Part II: Control Theory and Applications*, pages 369–411. Springer-Verlag, New York, 1990.

[9] R.O. Schmidt. A signal subspace approach to multiple emitter location and spectral estimation. Technical report, Ph.D. dissertation, Stanford Univ., 1981.

[10] G. W. Stewart. An updating algorithm for subspace tracking. *IEEE Trans. ASSP*, to appear.

6.1 Appendix: Downdating the URV

The computations related to *updating* the URV decomposition when new data appear have been described in [10], so we concentrate in this section on the *downdating* algorithm when data are deleted. This downdating algorithm takes $O(m^2)$ time and computes the URV factorization when the first data row in the window is deleted.

Suppose that we already have a URV factorization of an $n \times m$ data matrix Z^H with $n > m$. Specifically, only the $\bar{R}$ and V matrices are known. Then

$$Z^H = U \begin{pmatrix} \bar{R} \\ 0 \end{pmatrix} V,$$

where U is an unknown $n \times n$ unitary matrix. Then

$$Z^H V^H = \begin{pmatrix} \underline{z}_1^H V^H \\ \hat{Z}^H V^H \end{pmatrix} = \begin{pmatrix} \underline{u}_1^H \\ \tilde{U} \end{pmatrix} \begin{pmatrix} \bar{R} \\ 0 \end{pmatrix}, \tag{6}$$

where $\underline{z}_1^H$ and $\underline{u}_1^H$ are the first rows of Z^H and U respectively and Z^H has rank d. Now, we want a URV factorization of the matrix $\hat{Z}^H$. Referring to (6), our problem is equivalent to downdating a QR factorization of $Z^H V^H$ [4].

First, we compute the vector $\underline{u}_1^H$. The first m components of this vector are determined by solving the linear system

$$\underline{w}_1^H \bar{R} = \underline{z}_1^H V^H. \tag{7}$$

Although the block G of $\bar{R}$ is small, the corresponding subvector of $\underline{z}_1^H V^H$ is also small, and the process is well determined.

Columns $m+1$ through n of U can be chosen to be any basis for the null space of Z^H, and without loss of generality, we assume that columns $m+2$ through m have a zero in the first entry. Therefore, the $(m+1)$st entry of $\underline{u}_1^H$ is $\bar{\mu} = 1 - \|\underline{w}_1\|$, and the remaining ones are zero.

Now, we determine a sequence $J_m, \ldots, J_1$ of left rotations that rotate $\underline{u}_1$ into a multiple of the first unit vector. Rotation J_k modifies the kth and $(k+1)$st rows in order to zero out the $(k+1)$st element. That is

$$J_1 \cdots J_m \, \underline{u}_1 = \alpha \underline{e}_1 \tag{8}$$

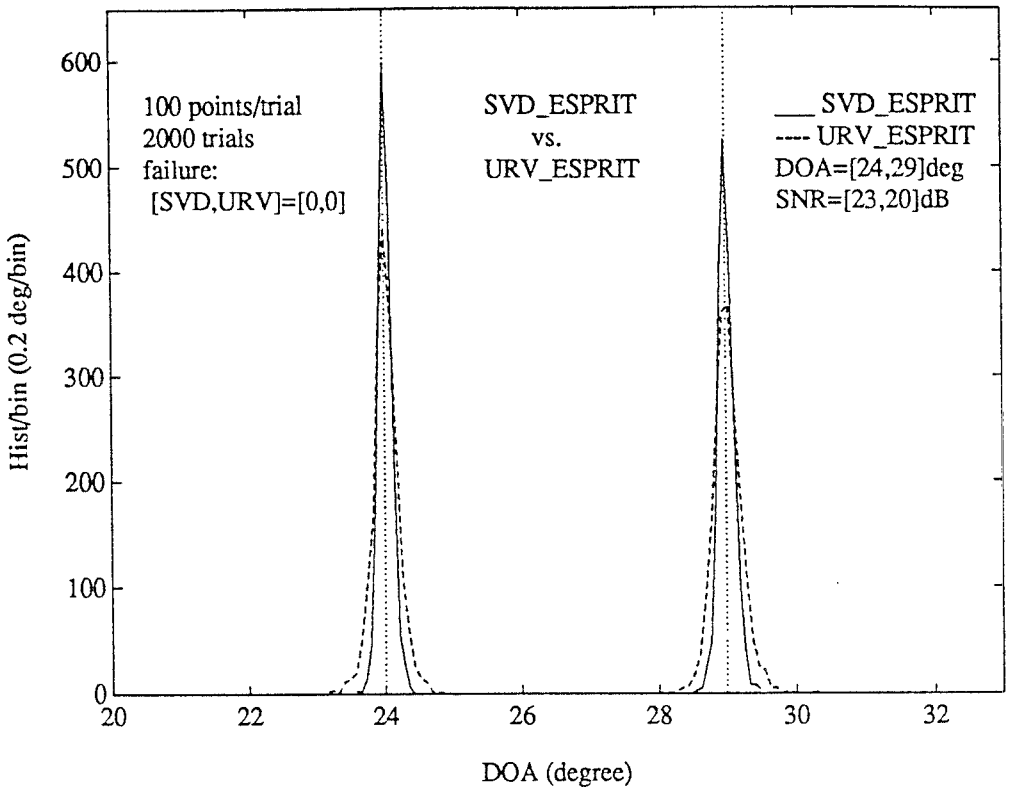
Applying these rotations to the triangular factor in (6) from the left will yield a Hessenberg matrix $\tilde{H}$, a matrix with zeros below the first subdiagonal. We now have the relation

$$\begin{pmatrix} \underline{z}_1^H V^H \\ \hat{Z}^H \, V^H \end{pmatrix} = \begin{pmatrix} \alpha & \underline{0}^T \\ \underline{0} & \hat{U} \end{pmatrix} \tilde{H} \, . \tag{9}$$

Note that the first column of the unitary matrix must be a multiple of the first unit vector since $|\alpha| = 1$, which forces all of the other entries in the column to be zero. Therefore we have a decomposition of the matrix $\hat{Z}^H \, V^H$ as a matrix with orthonormal columns times a triangular matrix $\tilde{R}$ consisting of rows 2 through $m+1$ of $\tilde{H}$, and our task is completed.

It requires about $4m$ multiplications and $2m$ additions to perform one rotation. Since we apply m rotations in the downdating processing, the total time required is $O(m^2)$.

In order to preserve the rank-revealing properties of the URV decomposition, we now need to check whether the triangular matrix has dropped in rank, a procedure described in [10].



Algorithm	signal 1	signal 2
SVD -ESPRIT	24.0031 ± 0.1002	28.9981 ± 0.1172
URV -ESPRIT	24.0021 ± 0.1915	29.0056 ± 0.2069

Figure 1: Histogram,results means and variances of ESPRIT(SVD) and ESPRIT(URV) for fixed sources

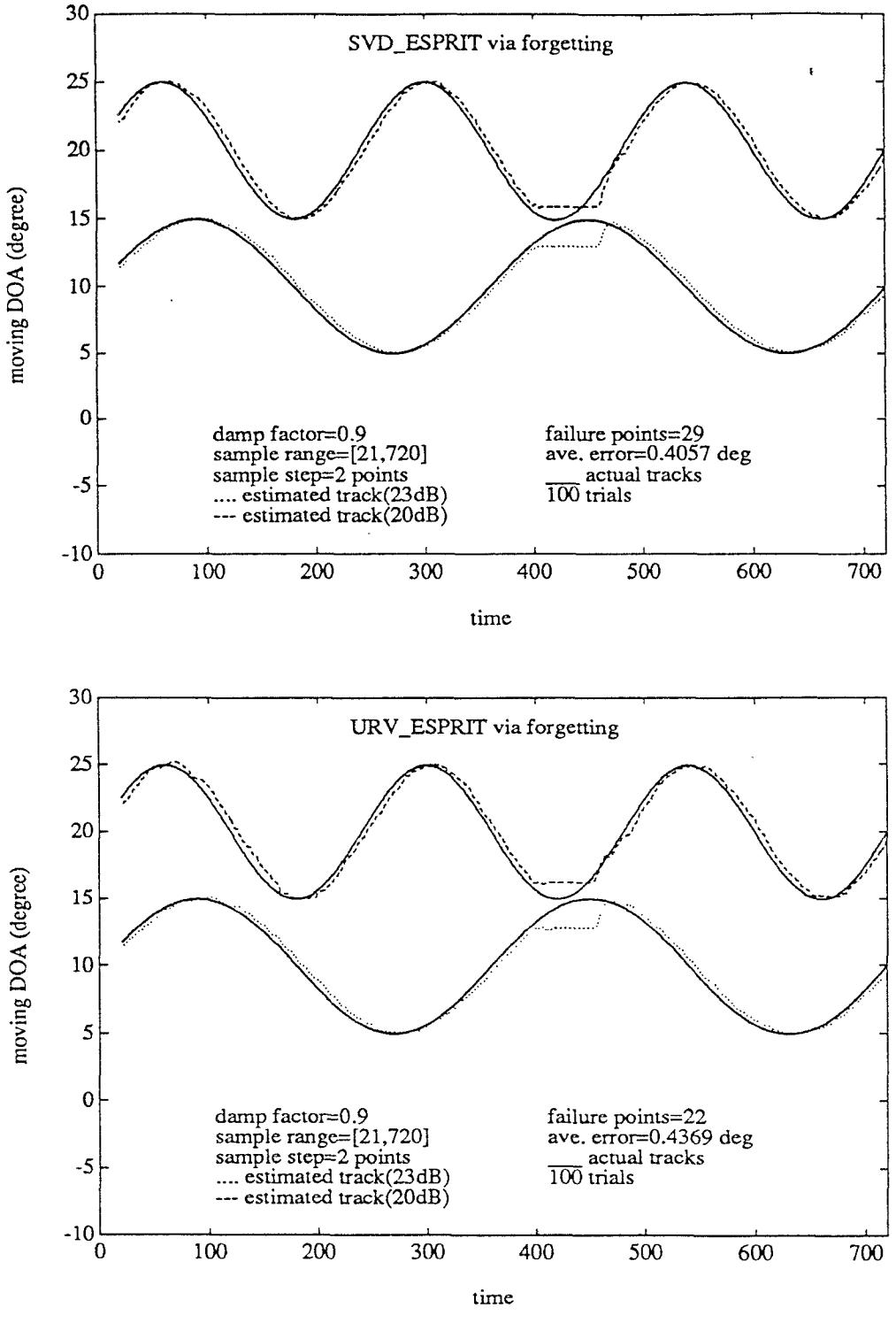


Figure 2: Estimated time-varying DOAs for close sources using exponential windowing.

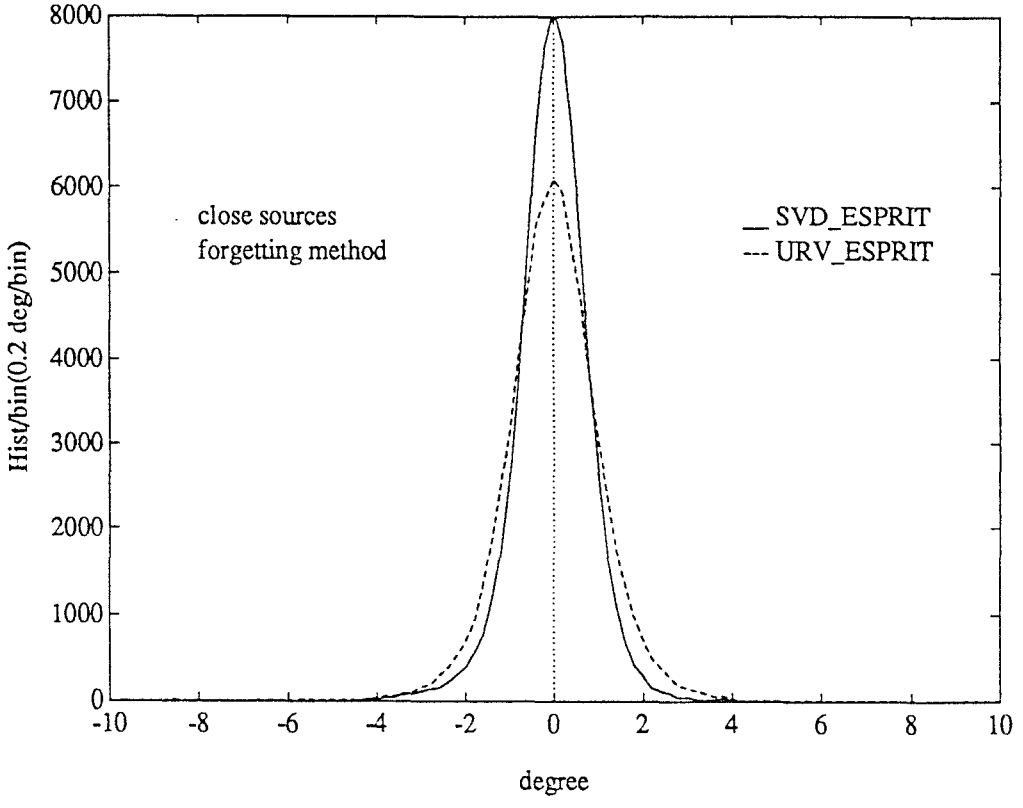


Figure 3: Error histogram for DOAs for close sources using exponential windowing.

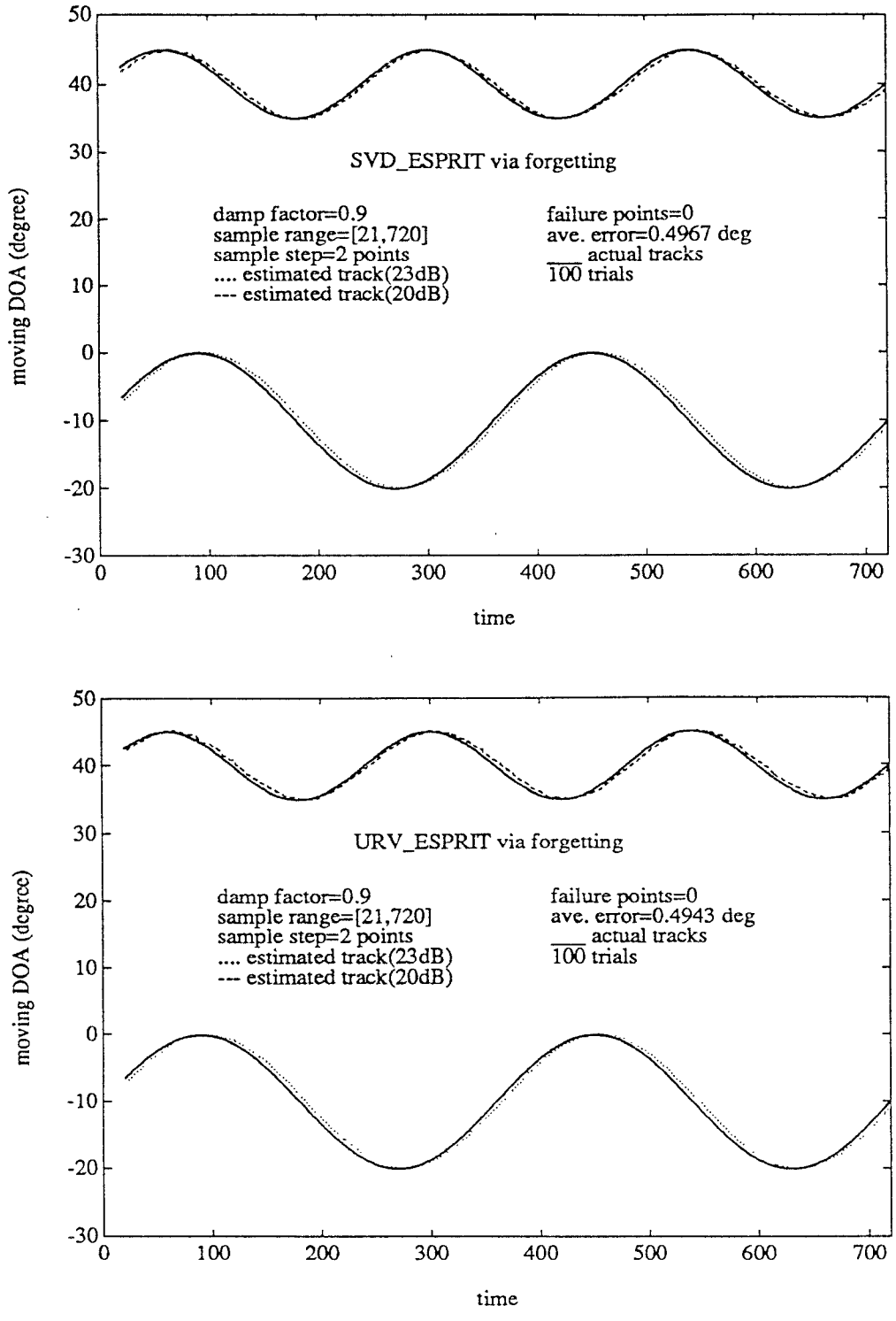


Figure 4: Estimated time-varying DOAs for well-separated sources using exponential windowing.

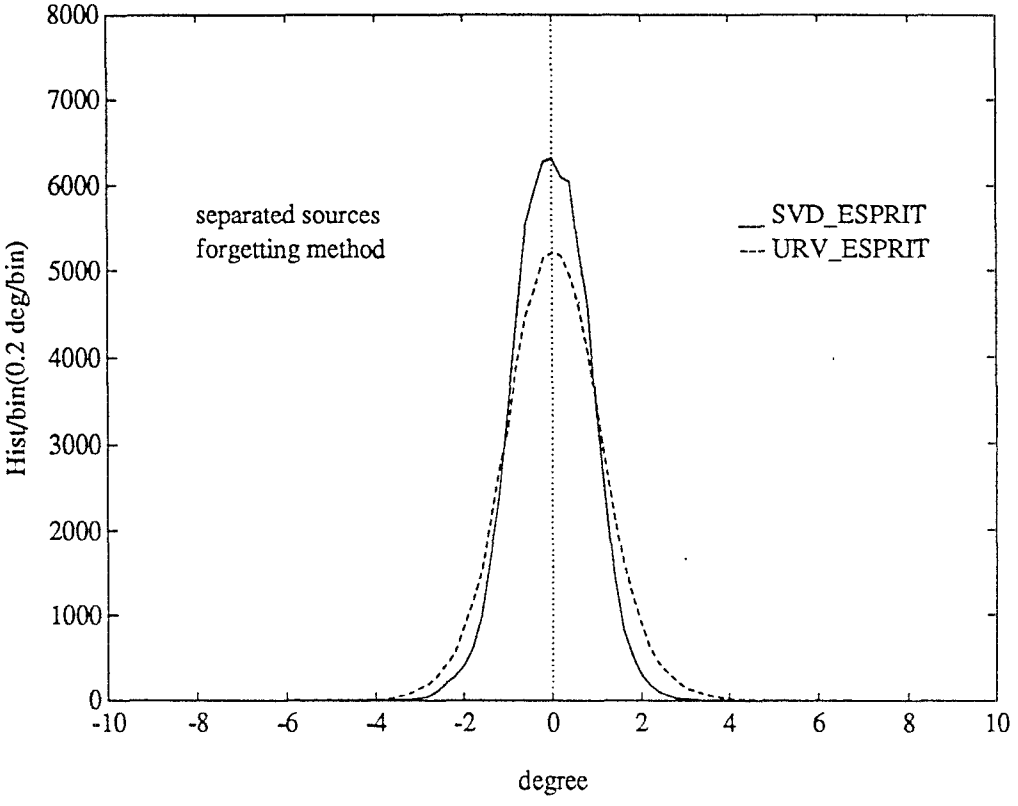


Figure 5: Error histogram for DOAs for well-separated sources using exponential windowing.

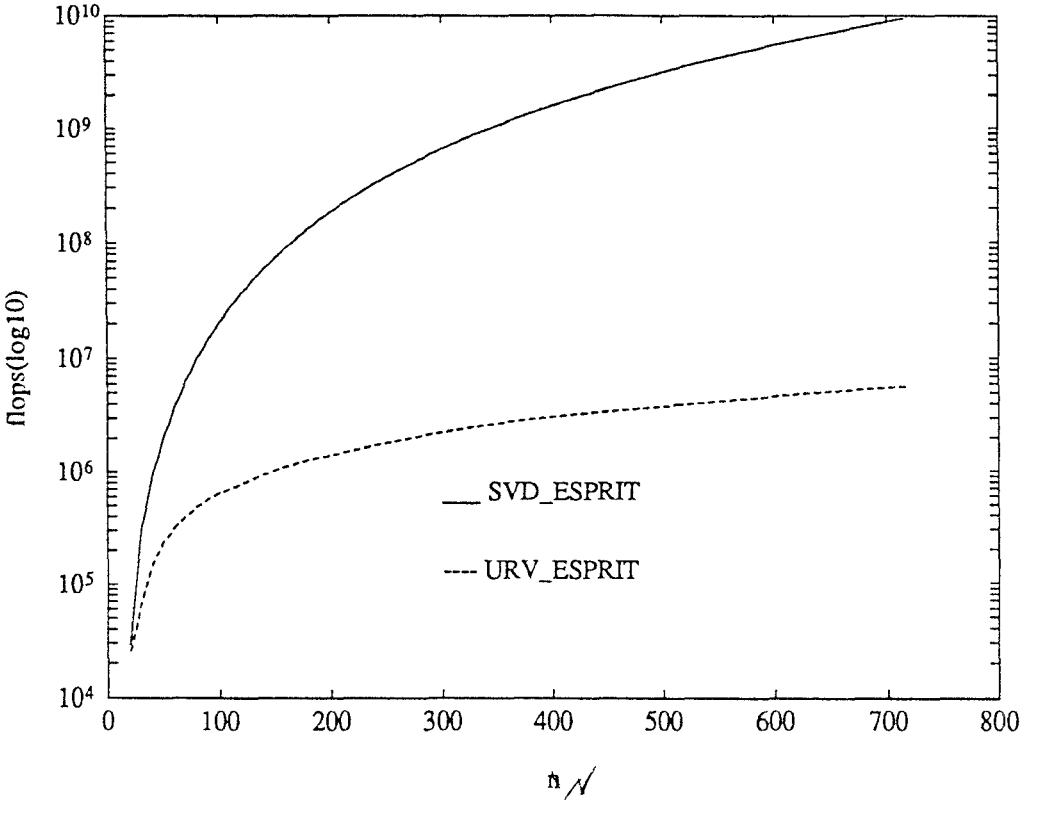


Figure 6: Number of floating point operations for one trial of the two algorithms using exponential windowing.

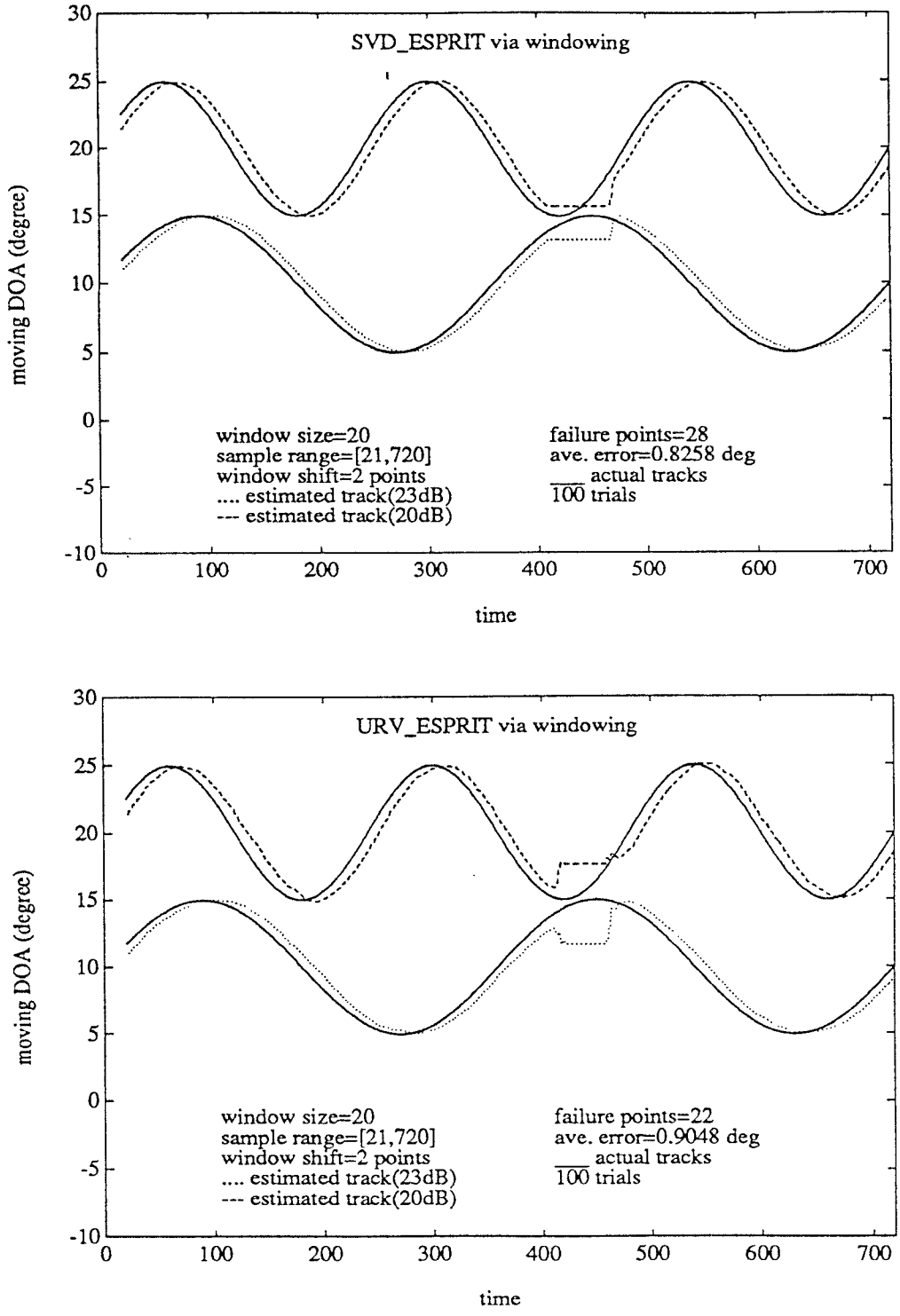


Figure 7: Estimated time-varying DOAs for close sources using rectangular windowing.

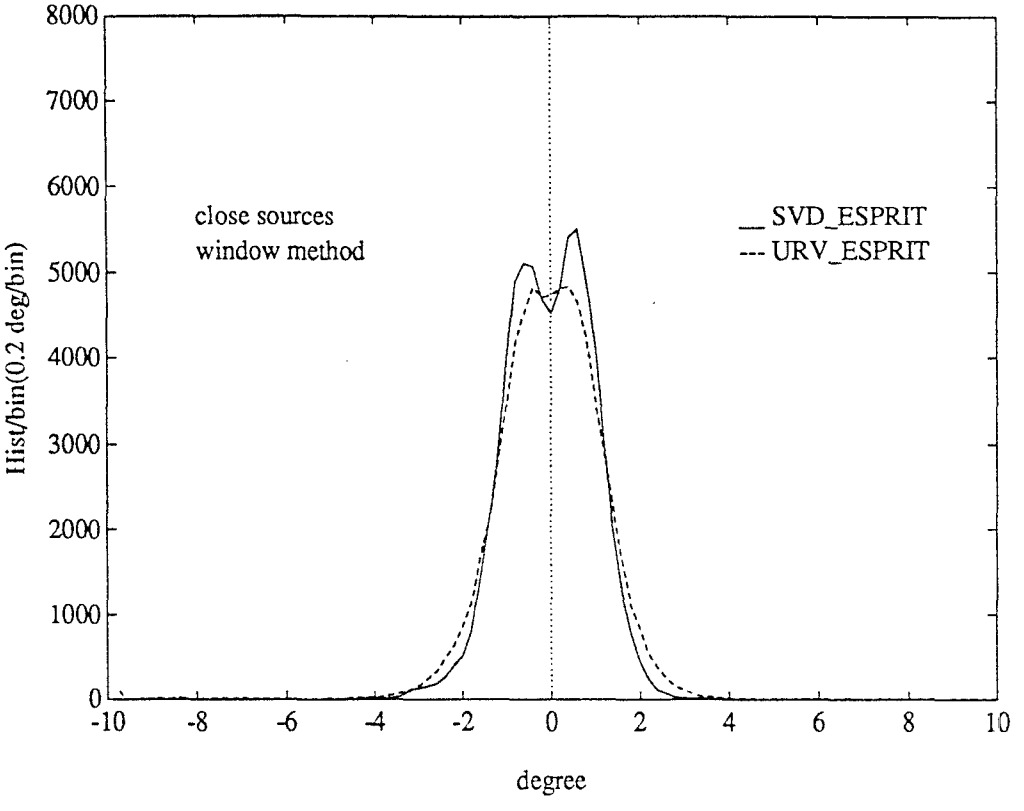


Figure 8: Error histogram for close sources using rectangular windowing.

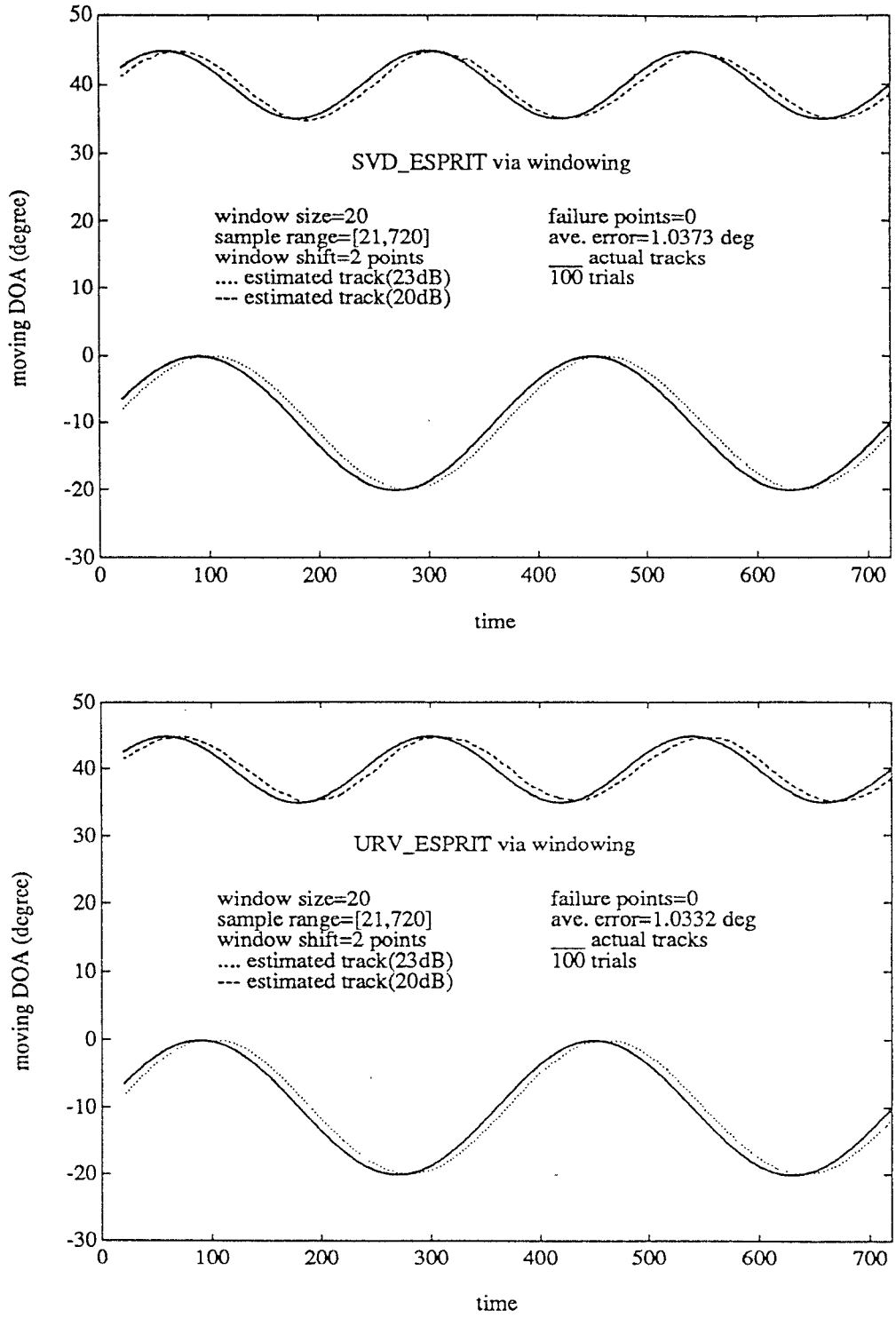


Figure 9: Estimated time-varying DOAs for well-separated sources using rectangular windowing.

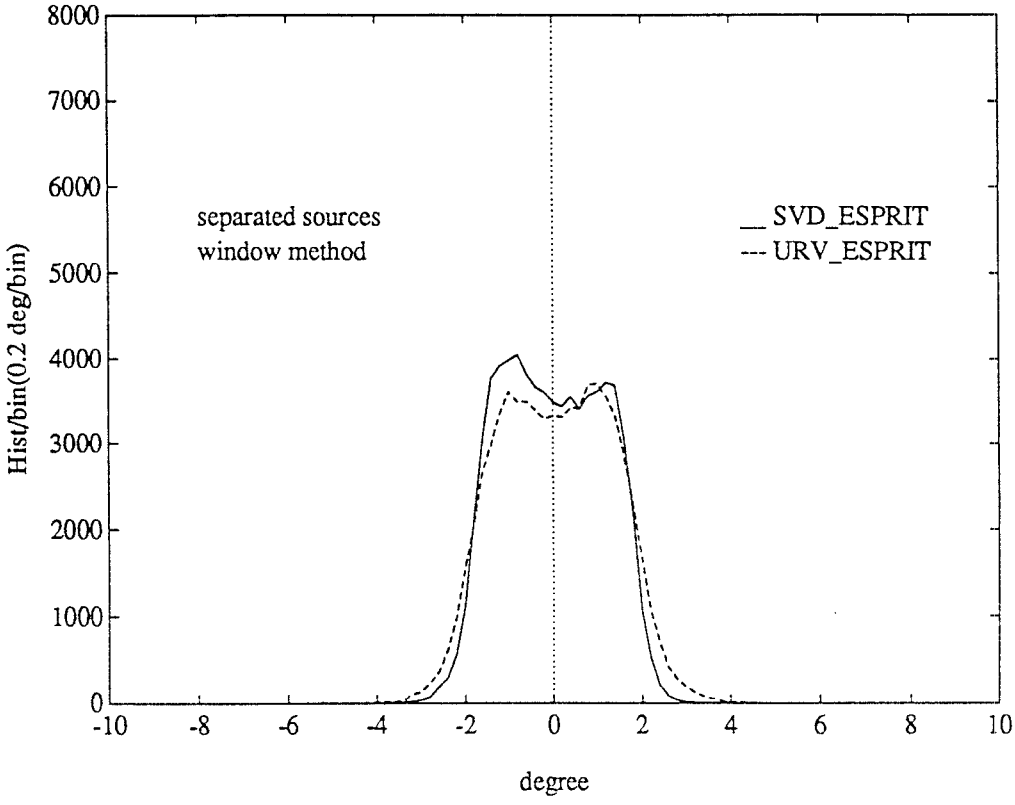


Figure 10: Error histogram for well-separated sources using rectangular windowing.

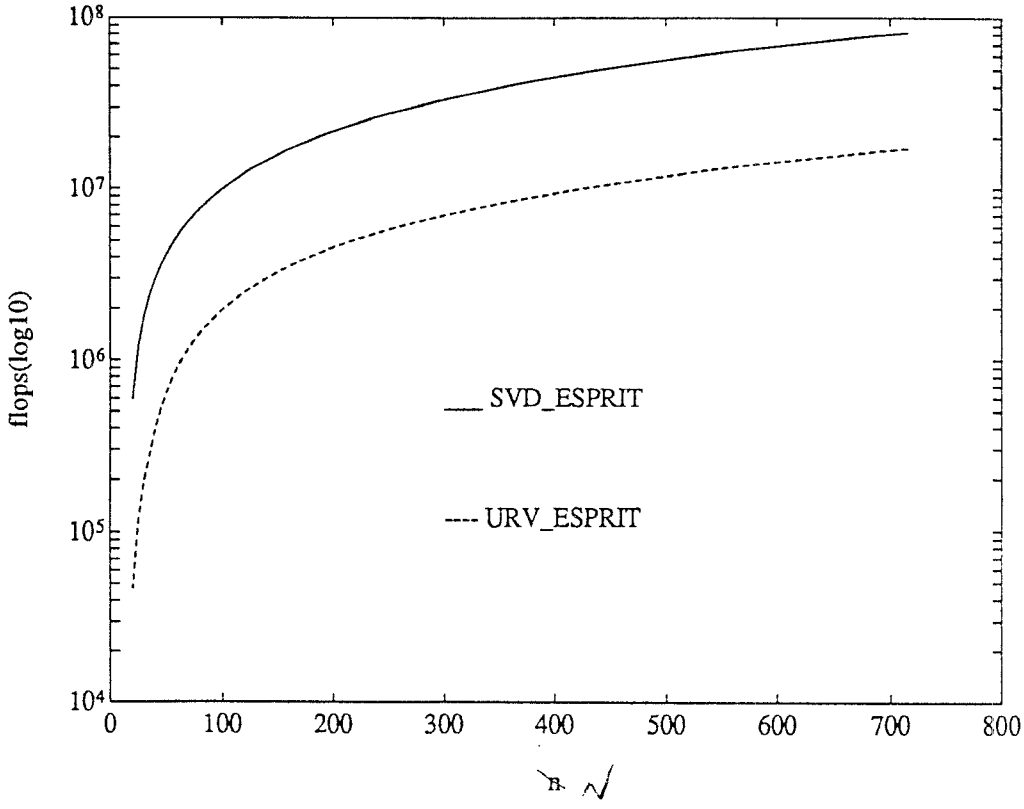


Figure 11: Number of floating point operations for one trial of the two algorithms using rectangular windowing.

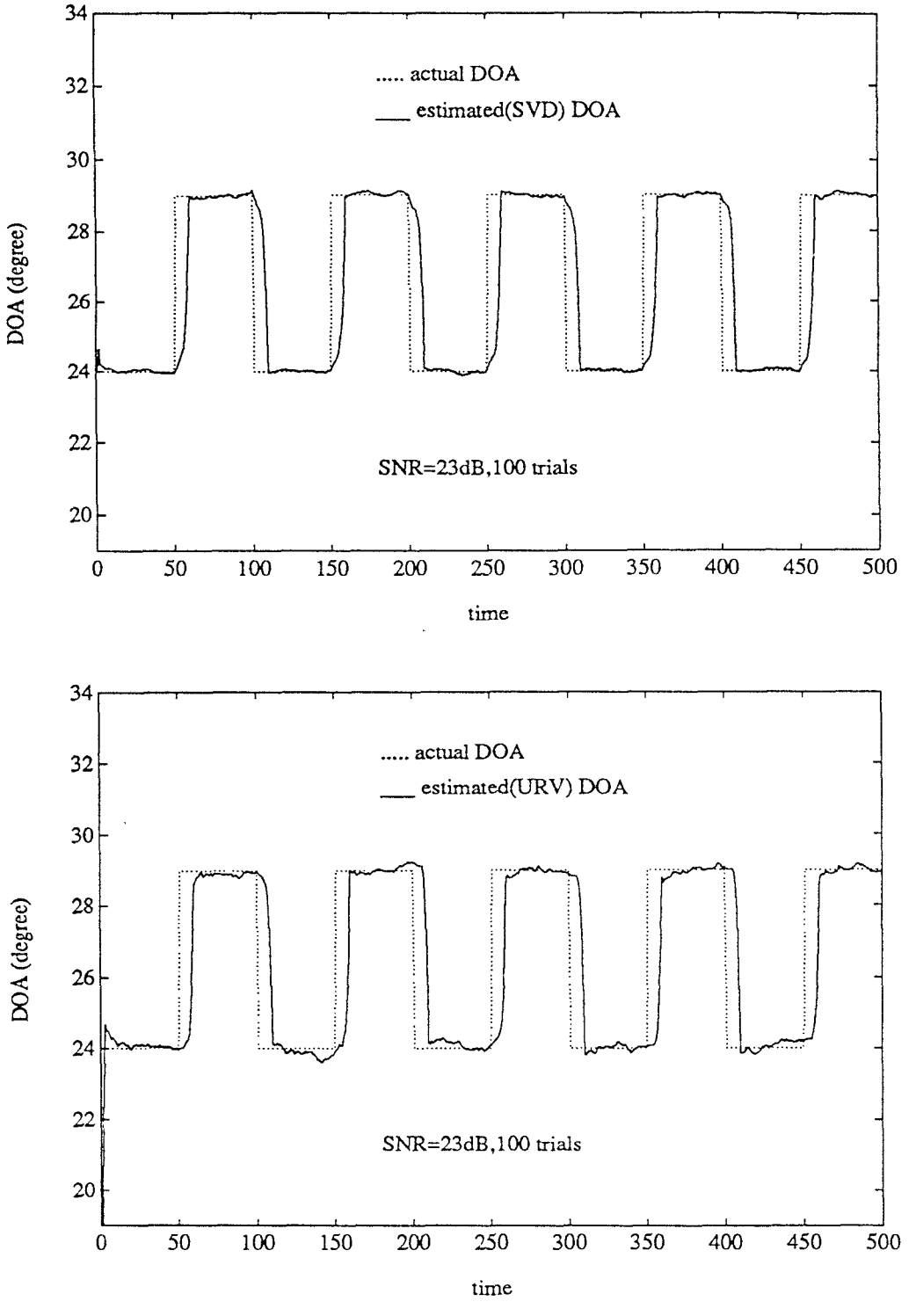


Figure 12: Estimated time-varying DOAs for instantaneously changing signals using rectangular windowing.