Data Compression using SVD and Fisher Information for Radar Emitter Location

Mark L. Fowler,† Mo Chen, J. Andrew Johnson, and Zhen Zhou

Department of Electrical and Computer Engineering
State University of New York at Binghamton
Binghamton, NY 13902

Abstract: This paper presents a data compression method that can achieve a very large compression ratio for radar pulse trains that are to be used for time-difference-of-arrival/frequency-difference-of-arrival (TDOA/FDOA) multiple-platform emitter location; this method exploits pulse-to-pulse redundancy to get a compression ratio much higher than possible using standard compression methods. We show how to use (i) the ability of the singular value decomposition (SVD) to exploit redundancy between radar pulses, and (ii) a Fisher information-based distortion criterion to enable elimination of pulses that are negligible to the FDOA estimation tasks. To enable the SVD to effectively remove the redundancy it is necessary to first optimally gate the pulses and place them in the rows of a matrix and then align the pulses to arrive at a matrix that is close to having rank of one. Finally, we suggest reasonable coding schemes for the information to be sent and assess the achievable compression level.

The Fisher information-based removal of pulses is shown to have negligible impact on the FDOA accuracy but does degrade the TDOA accuracy from that achievable using the SVD-based compression without pulse elimination. However, we demonstrate that the SVD method includes an inherent denoising effect (common in SVD-based signal processing) that provides an improvement in TDOA accuracy over the case of no compression processing; thus, the overall impact on TDOA/FDOA accuracy is negligible while providing compression ratios up to 100:1 for typical radar signals.

Keywords: data compression, singular value decomposition, emitter location, time-difference-of-arrival, TDOA, frequency-difference-of-arrival, FDOA

† Correspondence: mfowler@binghamton.edu

I. Introduction

A common way to locate electromagnetic emitters is to measure the time-difference-of-arrival (TDOA) and the frequency-difference-of-arrival (FDOA) between pairs of signals received at geographically separated platforms [1],[2]. The measurement of TDOA/FDOA between these signals is done by coherently cross-correlating the signal pairs [3],[4]. This requires that the signal samples of the two signals are available at a common platform, which is accomplished by transferring the signal samples over a data link from one platform to the other.

An important aspect of this processing that was not widely addressed in the literature until recently is that the available data link rate often is insufficient to accomplish the transfer within the time requirement unless some form of lossy data compression is employed. To mitigate this, we have developed data compression methods tailored specifically for TDOA/FDOA emitter location systems which can be grouped into two main categories of approach: (i) exploiting redundancy between pulses when the emitter to be located is a radar [5],[6],[7] and (ii) a more general approach of exploiting the relative importance of specific time-frequency components of general signals (i.e. communication or radar signals) through the use of a Fisher information-based distortion measure [8],[9],[10].

In [5],[6],[7] we showed how to compress radar signals by "gating" around the detected pulses, putting the gated pulses into the rows of a "pulse matrix", and then using the singular value decomposition (SVD) to compress the signal; this approach employed a purely MSE distortion criterion. In [8],[9],[10] we have developed a method that uses Fisher information as a distortion measure that captures the true impact of compression on the TDOA/FDOA accuracy. This paper ties together the separate results we have obtained in those two areas. We apply the Fisher information-based distortion measure for FDOA accuracy [8],[9],[10] to find the optimal set of pulses to remove from the pulse matrix and then use the SVD for transform-based compression. The removal of pulses is shown to have negligible impact on the FDOA accuracy but does degrade the TDOA accuracy from that achievable using the SVD-based compression without pulse elimination. However, we demonstrate that the SVD method includes an inherent

de-noising effect (common in SVD-based signal processing) that provides an improvement in TDOA accuracy over the case of no compression processing; thus, the overall impact on TDOA/FDOA accuracy is negligible while providing compression ratios up to 100:1 for some typical radar signals.

In Section II we provide an overview of the compression method. In Section III we provide the details of the method: in Section III-A we describe an optimal method for gating the pulses and placing them into the pulse matrix so as to minimize the amount of side information that needs to be sent; in Section III-B we describe how to align the pulses to reduce the rank of the matrix; in Section III-C we show how to use the SVD to extract a prototype pulse and a set of complex amplitudes that allow reconstruction of the pulse train; in Section III-D we provide an analysis of the achievable compression ratio; in Section III-E we show how to use Fisher information to eliminate pulses that are negligible for the FDOA estimation. Section IV provides illustrative simulation results.

II. Overview of Compression Method

The emitter location system consists of three or more platforms, each outfitted with identical receiving and processing equipment. Once signal data is collected at all of the platforms, the SNR is estimated at each platform and the one with the highest SNR is chosen as the one to transmit its data to the others for subsequent correlation processing; this platform is called the transmitting platform (Tx platform) and the other platforms are called the correlating platforms (Cx platforms). The Tx platform is required to be at a high enough SNR to allow standard radar intercept signal processing to be done [11],[12]. Because modern radars can change modes we assume that preliminary subtrain-extraction (de-interleaving) processing has grouped the signal of interest into one or more subtrains, each having pulses from the same mode of operation – such processing is a standard part of typical electronic warfare systems (this processing also removes pulses from other emitters) [11],[12]. Here we consider compressing one such subtrain.

As part of this interception processing, the Tx platform detects the individual pulses of the emitter of interest, gates around them, and keeps only the signal samples that lie inside the pulse gates; the numbers

of samples removed between the pulses is sent as side information. The gated signal samples and numbers of samples removed are transmitted to each of the Cx platforms where this broadcast data is used to reconstruct the gated pulse train (zeros are inserted between pulses according to the side information). The reconstructed signal is then cross-correlated with the signal received locally at the Cx platform to allow ML estimation (via cross-correlation) of the TDOA/FDOA values [3]. The sets of TDOA/FDOA estimates from the various Cx platforms are then combined to estimate the emitter location [1]. This is one particular scenario although other related scenarios can also be handled.

We focus on the transferal from the Tx platform to one Cx platform as shown in Figure 1. The data from the Tx platform is compressed before transmitting via a data link where the two signals are cross-correlated to obtain the ML estimate of TDOA/FDOA. The two noisy intercepted signals are modeled as

$$\begin{aligned} \hat{s}(k) &= s(k) + n(k) \\ \hat{d}(k) &= d(k) + v(k) \end{aligned} \tag{1}$$

where $s(k)$ and $d(k)$ are the complex baseband signals of interest and $n(k)$ and $v(k)$ are complex white Gaussian noises. The signal $d(k)$ is a delayed and doppler shifted version of $s(k)$. The signal-to-noise ratios (SNR) for these two signals are denoted *SNR* and *DNR*, respectively[‡]. The Tx Platform signal is compressed, transferred to the other platform, and then decompressed before cross-correlation. After lossy compression/decompression the signal $\hat{s}_c(k)$ has SNR of SNR_c.

Once the signal $\hat{s}(k)$ has been intercepted, it undergoes data compression/de-compression as outlined in Figure 2. The first step in the data compression is to remove the unwanted samples between pulses by using pulse gating; all compression ratios stated here take the gated signal as the original, non-compressed signal (i.e., the reduction due to gating out the "dead spaces" is not included as part of the compression ratio). Pulse gating and pulse matrix formation are illustrated in Figure 3. In order to minimize the amount of side information that must be sent, the pulse gating method optimally chooses the integers *G*,

[‡] SNR (non-italic) represents an acronym for signal-to-noise ratio; *SNR* (italic) represents the SNR for $\hat{s}(k)$ and *DNR* (italic) represents the SNR for $\hat{d}(k)$.

W and *T* in units of "samples" (see Section III-A). After the Pulse Matrix is formed, the pulses in the rows are aligned using either fractional delay FIR filters or DFT-based processing in order to obtain an Aligned Pulse Matrix that has rank of nearly one (see Section III-B). The amount of alignment imparted to each pulse is sent as side information in the sequence $\Delta_1, \Delta_2, \Delta_3, \ldots$ as shown in Figure 2. Because *W* is typically larger than the true pulse width, after alignment any excess columns outside the pulse width can be trimmed. If desired, the Aligned Pulse Matrix can be reshaped in two ways (i) by putting multiple pulses per row – proper reshaping is shown to maximize the compression ratio (see Section III-D) and (ii) by removing pulses deemed to be negligible via a Fisher information-based distortion measure (see Section III-C). The resulting matrix is close to being rank one so it is then decomposed using the SVD to extract the most significant left and right singular vectors, $\mathbf{u}_1$ and $\mathbf{v}_1$ (see Section III-C), which are then efficiently coded and transmitted to the Cx platform where the process is reversed to reform an approximation to the original pulse train (where zeros have been inserted between the reconstructed pulses).

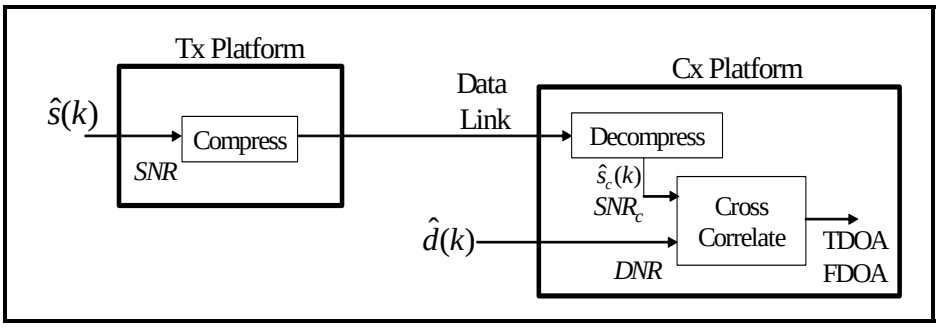

Figure 1: System Configuration for Compression

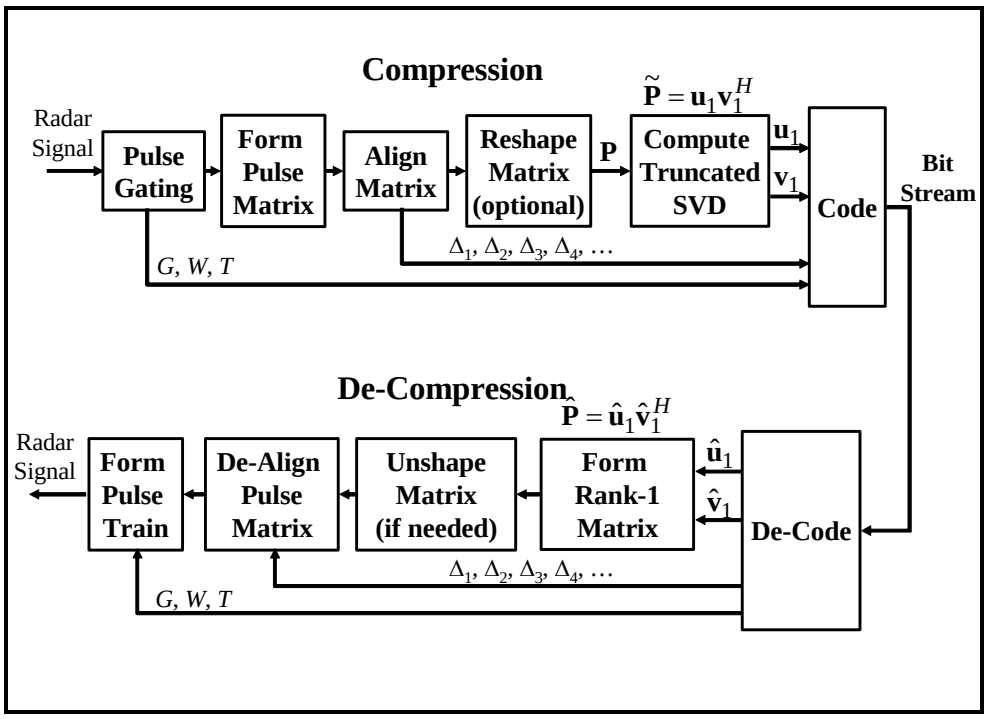

Figure 2: Overview of Compression and De-Compression Method

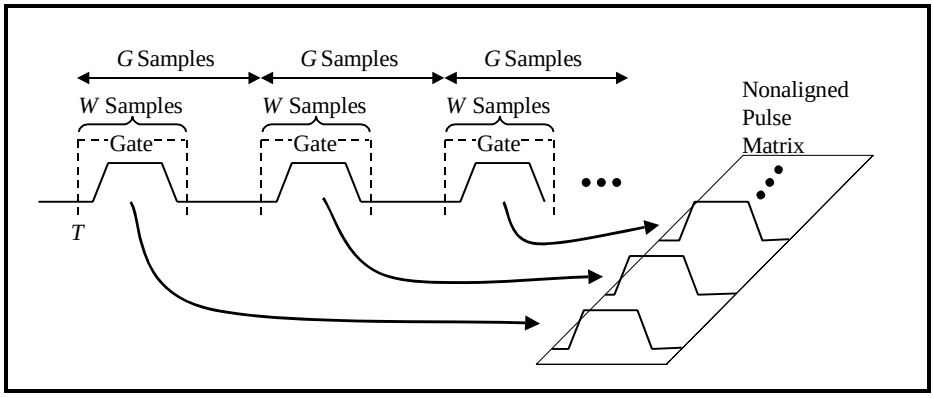

Figure 3: Pulse Gating and Pulse Matrix Formation

III. Details of Compression Method

A. Pulse Gating and Pulse Matrix Formation

In our original formulation of this method [5] the spacing between pulse gates was allowed to vary rather than having the constant *G* value shown in Figure 3; this required that a stream of G_i values needed to be sent as side information, which increased the amount of side information to be sent and hence decreased the achievable compression ratio. This section describes how to choose a single *G* value that reduces the amount of side information to be sent.

The gating method developed here constrains the gate width *W* and the gate interval *G* to be constants and gives an effective way to choose a minimal gate width *W* and the corresponding gate interval *G*; for maximum flexibility and performance we also include an initial gate offset *T*, thus the first gate starts at index *T*. We wish to find a *minimal* gate width to reduce the size of the resulting Pulse Matrix to achieve an effective level of compression.

The coding of *T* can be absorbed into the overhead that exists whether compression is used or not – namely that there is some minuscule amount of header information that describes the platform clock time of the first sample sent, which would be the time of the sample at index *T*. To send *G* would require no more than 32 bits to handle realistic pulse spacings at typical sampling rates; similarly, *W* could likely be coded with no more than 16 bits. In fact, fewer bits could be used for most systems, but allocating an excessive number of bits to this side information has a negligible impact on the resulting compression ratio. An implicit assumption has been made in the above analysis: there are no missing pulses (i.e., undetected due to low SNR, for example). The impact of this on the amount of side information must be addressed. While there are perhaps more efficient ways, we propose here to use a bit mask to indicate where there are missing pulses: the bit mask would have a “1” to indicate a pulse is present and a “0” to indicate that a pulse is missing; the occurrence of a missing pulse can be recognized from largely irregular spacing between leading edge times. Thus, for a given number of intercepted pulses *p*, the length of this bit mask will depend on the probability of a missing pulse P_{mp}. If there are *p* intercepted pulses, then the expected

number of transmitted pulses (and hence the length of the bit mask) is $p/(1 - P_{mp})$ bits. Thus, if we consider a some-what worst case scenario, with a value of $P_{mp} = 0.5$, then the expected number of bits in the bit mask would be $2p$ bits. Thus, we will use $2p + 32 + 16$ bits as a rough upper bound on the number of bits needed for the gating side-information.

The inputs to the gating algorithm are the stream of detected pulse leading edges and widths; these are parameters typically found as part of the standard processing of a typical electronic warfare system. Let t_k and w_k be the measured leading edge time and pulse width (integers in units of "samples"), respectively, for pulse k; here, it is assumed that within the algorithm these values are adjusted to give conservative values that ensure that t_k lies before the true start of the pulse and $t_k + w_k$ lies after the true stop of the pulse. This leads to specifying the following optimization problem:

$$\begin{aligned} &\text{minimize: } f(T,G,W) = W \\ &\text{such that for } k = \{0,1,\ldots p-1\}: \\ &\quad kG + T \le t_k \\ &\quad kG + T + W \ge t_k + w_k \\ &\quad G, W \ge 0 \\ &\quad T, G, W \quad \text{integers} \end{aligned} \tag{2}$$

To get a better perspective of this minimization problem, Figure 4 shows t_k and (t_k+w_k) vs. k along with two lines of slope G that bound these two sets of points. On the left side of the figure the corresponding pulse gates are represented by shaded rectangles. Thus, the minimization problem to be solved is to pick integers T, G, W (with G and W non-negative) so that the two lines enclose the two sets of points while minimizing the vertical distance between the lines. From the above problem formulation we see that we have reduced this problem to a traditional constrained linear optimization problem [13].

It is possible to modify the above problem so as to apply standard optimization methods [13],[14] such as simplex, simplex with convex hull, dual simplex, dual simplex with convex hull, and dual simplex with convex hull and cutting planes (see [6] for details). However, these standard methods are less efficient than the problem-specific approach described below.

We start by fixing G to some constant value, an integer. As stated above, the line $t = kG + T$ must pass though at least one of the leading edge points. If it did not, then W could be reduced by incrementing T until the line did pass through one of the leading edge points. From this, we find that if G is fixed, then T can be chosen such that this condition holds. In fact, it leads to this equation for selecting T, given G:

$$T = \min_k \left[t_k - kG \right] \tag{3}$$

A similar argument shows that for fixed G and T, W can be optimally chosen as:

$$W = \max_k \left[(t_k + w_k) - (kG + T) \right] \tag{4}$$

Note that in (3) and (4), if G is an integer, then so are T and W (since t_k and w_k are integers). Now it's a matter of finding the value of G that minimizes W.

It can easily be seen that W as a function of G is unimodal – where T is considered to take on its optimal value for each G as defined in (3). This arises from (2) as follows. Imagine the three dimensional space with axes W, G, and T. The first constraint in (2) specifies a series of vertical planes (i.e., parallel to the W axis) and Figure 5 shows the intersection of these constraint planes with the G-T plane; the shaded area identifies the region satisfying these constraints. From our previous considerations we know that the solution must lie on one of the lines forming the constraint region. The second constraint in (2) places a series of tilted planes above the G-T plane, each of which has a slope of –1 with respect to the T axis and the k^{th} plane has a slope of $-k$ with respect to the G axis. The conglomeration of these constraints plane creates a convex downward surface. Thus, W as a function of G is the values on this convex downward surface as traversed along the conglomeration of the constraint lines in Figure 5, which creates a unimodal function with a single minimum. A property of a unimodal function is that if a local mini-

mum is located, it will be the global minimum. Various well-known algorithms (such as bisection method, golden section search, etc.) can now be used to solve the optimization problem.

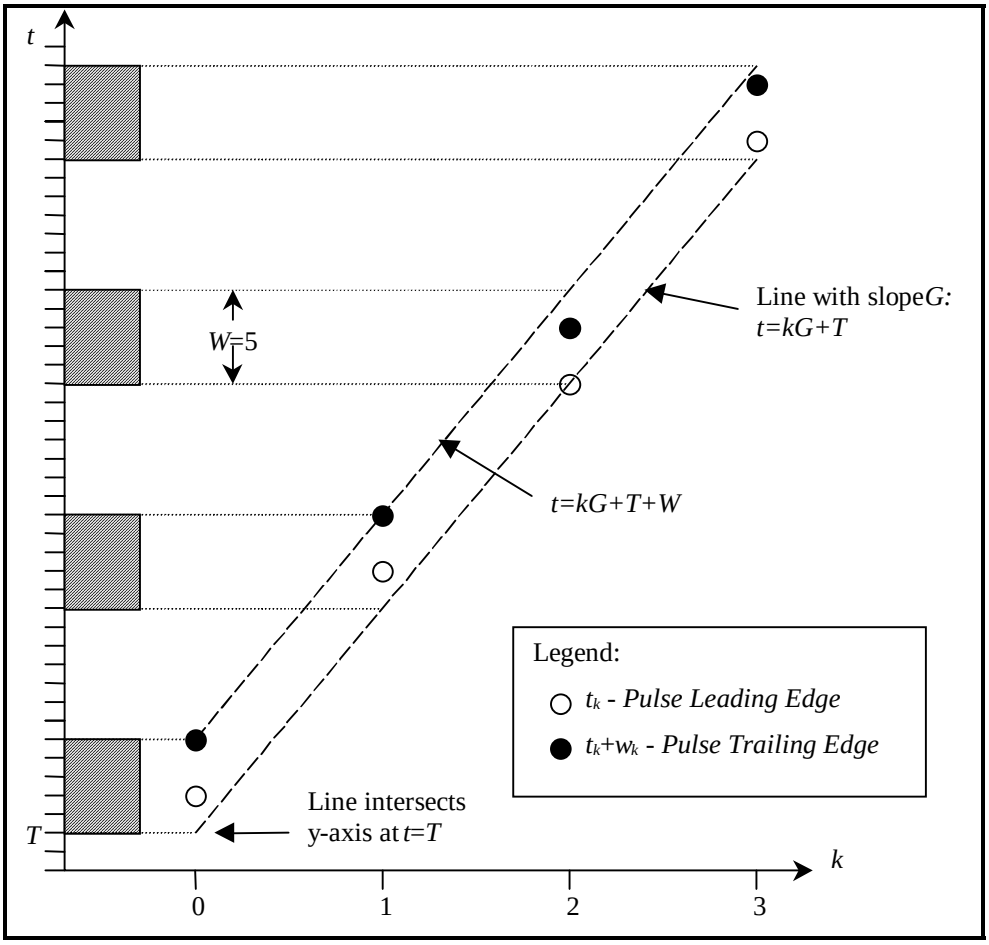

Figure 4: Pulse Leading and Trailing Edge Times vs. Pulse Index *k*

There are several different methods that can be used to select the initial trial value for *G*. The non-integer simplex method may be used. This tends to locate the optimal value for *G* within a fraction, but the method is complicated. (This method also places a lower bound on *W*). A least-square fit to the slope of the leading edge sample points may be used to estimate *G*. The method is simple, but may require more trials to reach the optimal *G*. A simple slope calculation between the leading edges of the first and last pulse may also be useful.

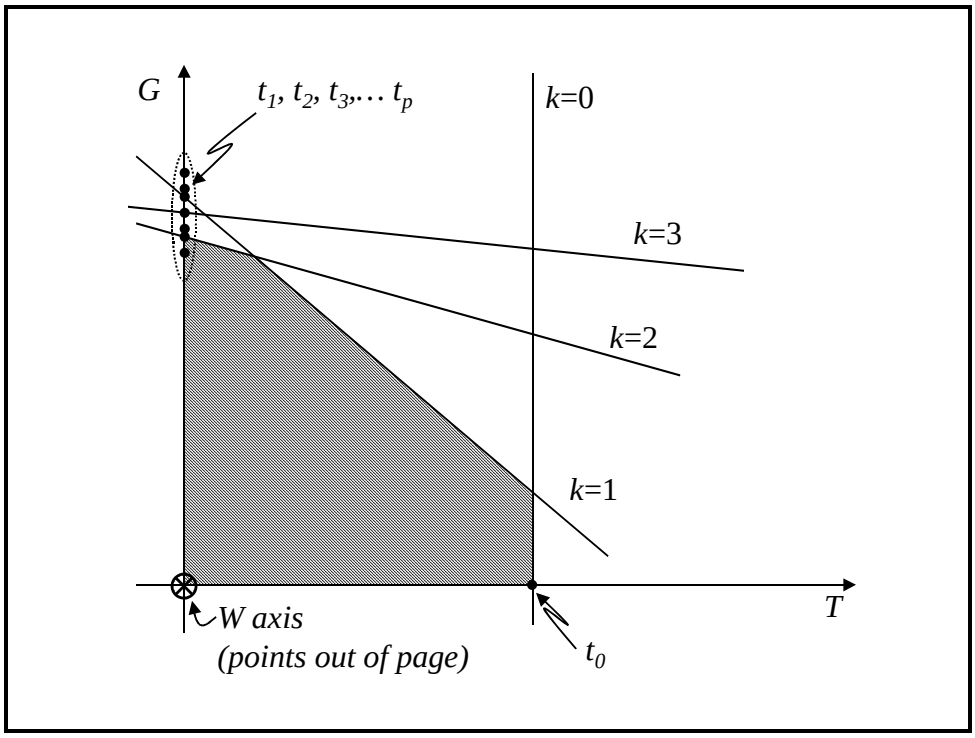

Figure 5: Example of constraints in the *T*-*G* plane

Our gating method (called "unimodal method") thus consists of finding an initial estimate for G (call it G_1) using any one of the methods mentioned in the previous paragraph and then updating to $G_2 = G_1 + 1$, solving (3) and (4) for each of these G estimates to get W_1 and W_2, from this information the unimodality can be used to determine the direction to the minimum of $W(G)$; G is then incremented in unit steps until $W(G)$ – checked using (3) and (4) – starts to increase, at which point the optimal G, T, and W have been found. The efficiency of this method depends on the accuracy of the initial estimate of G, but we have shown (see [6] for details) that it executes rapidly (about 2 orders of magnitude faster than the standard methods) and its run time grows very slowly with the number of pulses.

B. Pulse Matrix Alignment

The goal here is to yield a matrix that is as close to rank one as possible to enable compression via the SVD. Because the radar's PRI and the platform's sampling interval *T* are generally incommensurate (i.e., there is no *integer k* such that PRI = *kT*) the needed time alignment is not an integer number of samples. Therefore we need a method of shifting pulses by a fraction of a sampling interval.

Let $\mathbf{P}_{na}$ be the matrix whose rows hold the non-aligned pulses that are extracted by the gating procedure, as shown in Figure 3. Let $p_j(k)$ be the j^{th} pulse so that matrix $\mathbf{P}_{na}$ has its *j,k* element given by $\mathbf{P}_{na}(j,k) = p_j(k)$. We choose the pulse with the largest energy as the reference pulse, to which all the other pulses will be aligned; let this pulse be denoted as $p_m(k)$. Then to find the time alignment needed for the j^{th} pulse ($j \neq m$) we cross-correlate it with the reference pulse to give

$$C_j(k) = \sum_l p_m(l)\, p_j^*(l-k), \quad j \neq m \tag{5}$$

and then interpolate $C_j(k)$ to find the location of its peak, which is then taken as the time shift Δ_j to be applied to $p_j(k)$ to align it with $p_m(k)$. The time shift Δ_j can be written as $\Delta_j = D_j + \delta_j$ where D_j is an integer and $0 \le \delta_j < 1$. The time shift Δ_j values are coded for transmission using 8 bits each.

The integer part of the alignment can be handled separately according to $\tilde{p}_j(k) = p_j(k + D_j)$. Now $\tilde{p}_j(k)$ must be shifted by an amount that is less than a single sample. There are several ways to impart a fractional delay to a signal [5],[7],[15]. Which one is used depends partly on whether the signal to be delayed is available in its entirety as one block or is available sequentially, either sample-by-sample or on a subblock-by-subblock basis; that is, it depends on the level of latency that is acceptable. Other considerations are accuracy and complexity. Here we consider four different methods and assess them on these merits. The methods are (i) a full-block DFT method based on the delay property of the DFT [7], (ii) a subblock-based version of the DFT method [7], (iii) fixed FIR filters designed for fractional delay [15], and an adaptive FIR filter for fractional delay [7]. Zhou [5],[7] compared the accuracy on the basis of

magnitude vs. frequency, delay vs. frequency, and SNR vs. delay value. Because in this application the signals to be delayed (the individual pulses) can be quite short the adaptive FIR method is not suitable. The fixed FIR methods are also not the best choice because (i) the signals can be short, (ii) the signals are wideband so the frequency-specific characteristic is a major disadvantage, and (iii) the worst-case accuracy vs. delay may not be acceptable. That leaves the subblock DFT and the full-block DFT methods. For the current application there is no penalty in having to wait for an entire pulse to become available because the front-end processing must first identify all the pulses before any subsequent processing is done; furthermore, the required fractional delay can't be determined until the entire pulse is available. Therefore, for this application we use the full-block DFT approach [7].

To give a delay of $\delta \in (0,1)$ requires passing the signal through a system with frequency response given by $H(\Omega) = e^{-j\Omega\delta}, \quad \Omega \in [-\pi, \pi)$ where $\Omega = 2\pi fT$ is the discrete-time frequency for a sampling interval of *T*. This can be implemented using DFT properties as follows: (i) Compute the DFT of the signal using zero-padding to ensure the time shift is not circular; (ii) Multiply the DFT by $e^{-j\Omega\delta}$ evaluated at the DFT frequencies; (iii) Compute the inverse DFT of the result.

The effectiveness of the fractional delay method for this application can be seen by assessing its ability to reduce the effective rank of the pulse matrix. The effective rank of a matrix is best assessed via the SVD [16]. Let **P** be the matrix that is obtained from $\mathbf{P}_{na}$ after aligning its pulses as described above. If the alignment method is effective at creating a matrix with effective rank one, then all but the first singular value of **P** should be insignificant. Because the sum of the squares of the singular values gives the energy of the signal it makes more sense to plot the squares of the singular values; for each case, normalizing by the largest singular value improves the comparison between various cases (e.g., nonaligned, aligned, etc.). Figure 6 shows the squares of the normalized singular values as a function of singular value index for the unaligned matrix, the aligned matrix using only integer alignment, and also using fractional alignment; the case shown here is for a simulated linear FM radar signal sampled at an interval that

is incommensurate with the PRI. From this we see the effectiveness of the fractional alignment method – the effective rank of the fractionally aligned matrix can be seen to be close to one. This is the basis of the compression method developed here: compression is achieved because the fractionally-aligned matrix can be closely approximated in terms of a rank one matrix.

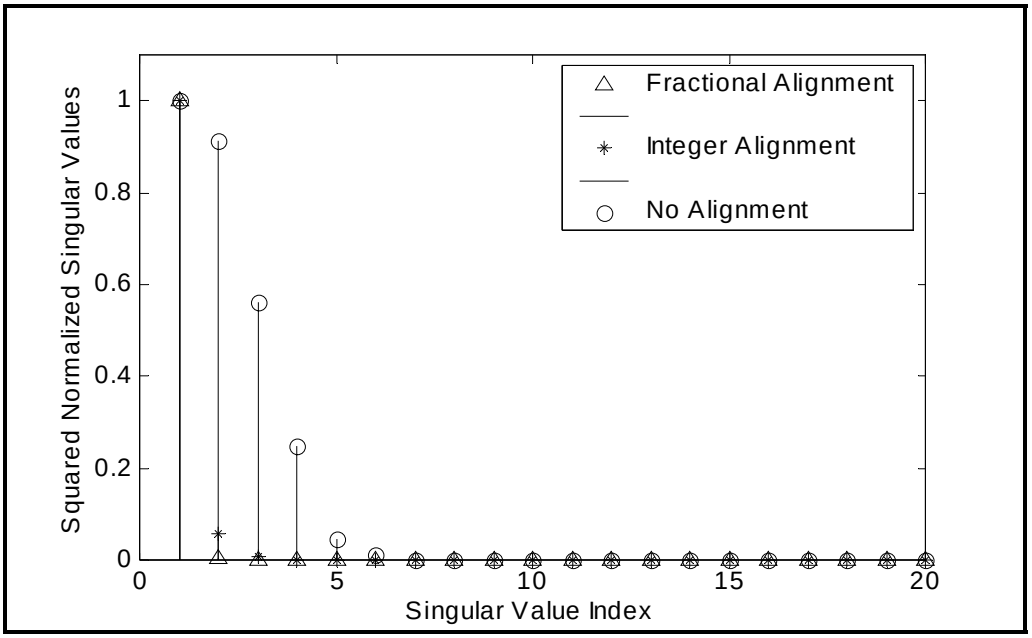

Figure 6: Squared Normalized Singular Values for Aligned and Unaligned Matrices

C. SVD-Based Extraction

In the actual processing the next step would be reshaping of the aligned pulse matrix; although it is optional, it is typically desirable to perform the reshaping step. However, we postpone discussion of this step until later for ease of discussion.

If we denote the $p \times n$ aligned pulse matrix by $\mathbf{P}$, its SVD is

$$\mathbf{P} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^H \ , \tag{6}$$

where p is the number of pulses, n is the number of samples retained per pulse after alignment, r is the rank of $\mathbf{P}$, $\mathbf{u}_i$ is the i^{th} left singular vector, $\mathbf{v}_i^H$ is Hermitian transpose of the i^{th} right singular vector, and σ_i is the i^{th} singular value, ordered such that $\sigma_i \geq \sigma_{i+1}$. Each term in the sum in (6) is a rank-one matrix. In general, truncating this sum to only $k < r$ terms we get a rank-k matrix $\mathbf{P}_k$ that best approximates $\mathbf{P}$ in the sense that the sum of the squares of the elements of $\mathbf{P} - \mathbf{P}_k$ is smaller than for any other rank-k matrix [16]. Note that in our case the matrix contains the pulses and therefore this approximation gives the smallest mean square error (MSE) between the original pulse train and the approximate pulse train formed by concatenating the de-aligned rows of $\mathbf{P}_k$. This minimum MSE property is the basis for using the SVD here. We limit our focus to the case of $k = 1$; the pulse alignment is done to enable this (see Figure 6). The effect of the noise on the singular values is uniformly spread across all the singular values – this is in fact a known result that is exploited in many applications of the SVD to signal processing problems [17]. Thus, SVD truncation reduces the noise. This simultaneous compression and noise reduction will be demonstrated in the simulations.

In particular, the approximating matrix $\mathbf{P}_1$ is formed from $\mathbf{P}_1 = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^H$, from which it is clear that each row in $\mathbf{P}_1$ is a complex-valued scalar multiple of $\mathbf{v}_1^H$, where the complex scalar for the i^{th} row is the i^{th} element in $\mathbf{u}_1$ times σ_1; it is also clear that σ_1 does nothing more than amplitude scale the <u>entire</u> reconstructed pulse train and can therefore be omitted. Thus, we can use

$$\tilde{\mathbf{P}}_1 = \mathbf{u}_1 \mathbf{v}_1^H , \tag{7}$$

from which we see that $\mathbf{u}_1$ holds the reconstruction magnitudes and phases. We can interpret vector $\mathbf{v}_1^H$ as a single prototype pulse that has been extracted from the original pulse train and the vector $\mathbf{u}_1$ holds the complex amplitudes that are multiplied by the prototype pulse to create the pulses in the reconstructed pulse train.

D. Coding and Compression Ratio Analysis

As discussed above, the gating parameters can be sent using 48 bits and we assume roughly $2p$ bits are needed for a bit mask to handle the effect of missing pulses. Thus, the side information coming from the pulse gating step in Figure 2 is $2p + 48$ bits. The rest of the data that must be coded consists of three parts: (i) the complex-valued right singular vector (RSV), (ii) the complex-valued left singular vector (LSV), and (iii) the time-shifts of the pulses.

To code the prototype pulse contained in the RSV we recognize that each of its samples is a complex number having magnitude and phase, both of which are changing from sample to sample. The cross-correlation processing will be much less sensitive to errors in the magnitude than in the phase, so we should ensure that the phase is coded with high accuracy whereas the magnitude can be coded with lower fidelity. We propose to code the phase of the prototype pulse using 8 bits/sample and to use a 1-bit differential PCM approach for the magnitudes of the prototype pulse. Thus, we use $B_{RSV} = 8+1 = 9$ bits to code each element of the RSV. It should be noted that this approach provides a fairly general approach that should work for virtually all cases; however, when the acquisition system identifies the radar as being a linear FM signal, the phase of the prototype should have fairly constant sample-to-sample phase differences, and then it may make more sense to use some form of differential coding there, too. To code the pulse magnitudes and phases contained in the LSV we again should allocate more bits to the phases than to the magnitudes. We use 4 bits per magnitude in the LSV and 8 bits per phase in the LSV. Thus, we use $B_{LSV} = 8+4 = 12$ bits to code each element of the LSV. Finally, each time shift is coded using $B_{TS} = 8$ bits.

How much compression can we get from this scheme? If no compression is used (other than gating) there are np complex samples to be sent and we use 8 bits/sample for the real part and 8 bits/sample for the imaginary part; thus, including the bits used to code the numbers of zeros to be inserted due to gating, the original signal is coded using

$$\text{Original Data} = 16pn + 2p + 48 \tag{8}$$

Here we consider the case where a single pulse is put into each row of $p \times n$ **P**, but we will see later that it is usually better to reshape the matrix to put multiple pulses per row. Given the number of bits used to code the SVD compressed data, the total number of bits used for the compressed data is summarized in Table 1. Therefore, when using one pulse per row we get a compression ratio of

$$CR_{\text{subopt}} = \frac{16pn + 2p + 48}{22p + 9n + 40}, \tag{9}$$

which is labeled as suboptimum because we will see below that putting multiple pulses per row can improve the compression ratio.

Table 1: Total Compressed Data with One Pulse per Row

Quantity to Code	**General Form of Coding**	**Specific Form of Coding**
$n \times 1$ RSV	$(n \times B_{RSV})$	$(8+1)n = 9n$
$p \times 1$ LSV	$(p \times B_{LSV})$	$(8+4)p = 12p$
$(p\text{-}1) \times 1$ time shifts	$(p\text{-}1) \times B_{TS}$	$8(p-1) = 8p - 8$
Gating Side Information	$2p + 48$	$2p + 48$
Compressed Data		$\mathbf{9n + 22p + 40}$

As mentioned above, it is possible to improve this compression ratio by putting more than one pulse per row (after alignment) such that we get an $r \times c$ matrix. This will require a few modifications, namely, we will need to normalize all the pulses so that even if we have significant pulse-to-pulse fading we will still be able to get a near rank one matrix. Thus, we won't have to code the magnitudes of the LSV, but we will now need p magnitude normalizers that can be coded using $B_{MN} = 2$ bits each. The $r \times 1$ LSV now only needs to have its phase coded, using $B_{\phi} = 8$ bits per element. This changes the results in Table 1 to those shown in Table 2.

Table 2: Total Compressed Data with Multiple Pulses per Row

Quantity to Code	**General Form of Coding**	**Specific Form of Coding**
(c×1) RSV Phases	$(c \times B_{LSV})$	$(8+4)c = 12c$
(r×1) LSV	$(r \times B_{\phi})$	$8r$
(p×1) Magnitude Normalizers	$(p \times B_{MN})$	$2p$
(p-1)×1 time shifts	$(p\text{-}1) \times B_{TS}$	$8(p-1) = 8p - 8$
Gating Side Information	$2p + 48$	$2p + 48$
Compressed Data		$\mathbf{8r + 12c + 12p + 40}$

The goal here is to find the optimal values of c and r. As a means of exploring this, for now assume that we can make any size pulse matrix for a given collection of pulses as long as the total number of elements equals the total number of samples np in the pulse train. Consider an $r \times c$ matrix with r and c chosen such that CR is maximized under the constraint that $rc = np$ (or equivalently that $r = np/c$). For this case the compression ratio becomes

$$\begin{aligned} CR &= \frac{16pn + 2p + 48}{8r + 12c + 12p + 40} \\ &= \frac{16pn + 2p + 48}{8r + \frac{12np}{r} + 12p + 40}, \end{aligned} \tag{10}$$

which should be maximized as a function of r for a given np. Thus, we must minimize the function

$$f(r) = 8r + \frac{12np}{r} + 12p + 40, \tag{11}$$

which is minimized when $r = \sqrt{3np/2}$ or equivalently when $c = \sqrt{2np/3}$; this is equivalent to making the pulse matrix such that $c = 2r/3$. From plots of CR vs. c, this peak is fairly broad so that hitting the exact value is not real crucial, so restricting r and c to be integers will not drastically reduce the CR from the theoretical maximum. Thus, we should put multiple pulses in a row in order to make the pulse matrix as

close as possible to having two-thirds as many rows as columns. Using these results in (10), the optimal compression ratio is

$$CR_{\text{opt}} = \frac{16pn + 2p + 16}{8\sqrt{6np} + 12p + 6} \tag{12}$$

which is plotted in Figure 7 as a function of n and p. From this plot we see that for low to medium number of samples per pulse that the optimal compression ratio becomes effectively independent of the number of pulses as the number of pulses gets large. As both n and p increase, the compression ratio increases without bound; thus, we see that the compression ratio increases as the number of samples increases – that is, larger compression ratios are achieved when more compression is needed. Specific compression ratio results for typical practical scenarios are given in Table 3; the pulse width (PW), pulse repetition interval (PRI), and bandwidth (BW) values are for typical radars; the samples-per-pulse values are dictated by practical sampling theory; the ranges of number-of-pulses is dictated by the need to achieve sufficient TDOA/FDOA accuracy under expected conditions. It should be noted that these compression ratio results are much lower than some preliminary results [18] because those earlier results did not consider the impact of coding the side information; nonetheless, even including the effect of the side information, as we have here, the compression ratios achieved are still very good.

E. Pulse Matrix Reshaping

Once the matrix is aligned it may be possible to reshape the matrix in two ways: (i) remove rows (i.e., pulses) that have negligible contribution to TDOA/FDOA accuracy and/or (ii) combine multiple pulses into a single row. The former reshaping is done on the basis of the fact that the Fisher information for FDOA estimation shows that pulses near the middle of the pulse train contribute less toward FDOA accuracy than do pulses at the beginning or end. The latter reshaping is done on the basis of the analysis given in the previous subsection showing that the compression ratio can be improved by properly putting multiple pulses per row in some cases. In this section we will focus on pulse removal via the Fisher information measure.

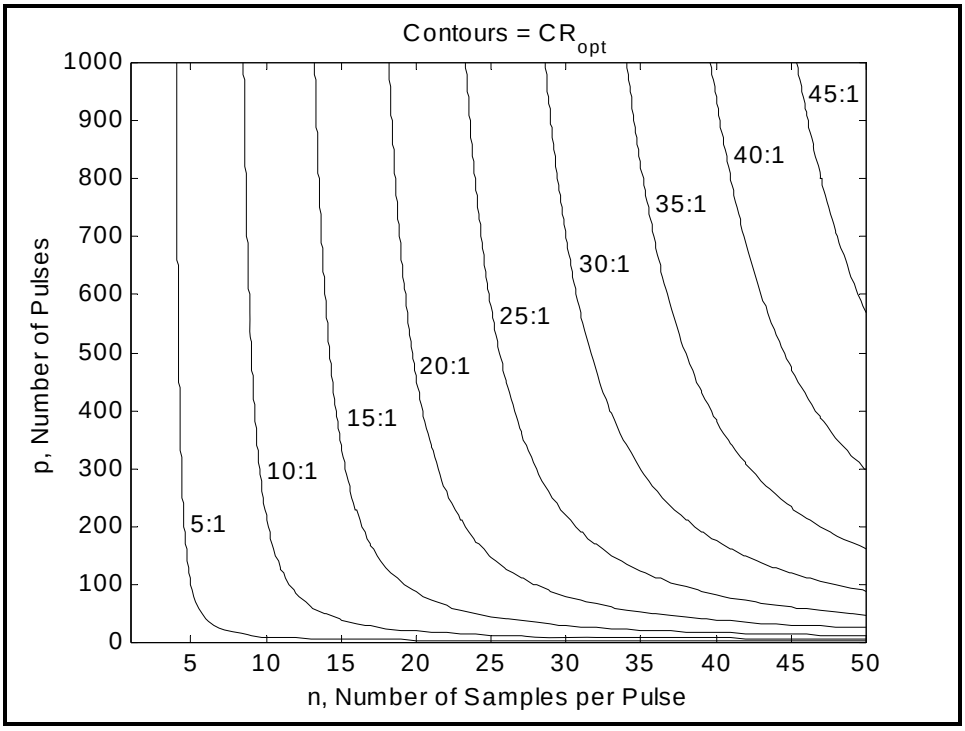

Figure 7: Contour plot showing optimal compression ratio as a function of the number of pulses and the number of samples per pulse.

Table 3: Compression Ratios for Typical Practical Scenarios

PW (μs)	**PRI** (μs)	**BW** (MHz)	**# of Pulses** *p*	**Samples/Pulse** *N*	CR_{subopt}	CR_{opt}
0.5	600	4.0	80 – 300	6	4.3 – 4.4	5.6 – 6.6
1.5	10	2.0	1,500 – 14,000	8	5.9 – 5.9	9.7 – 10.4
6.5	70	2.5	250 – 1,500	24	16.9 – 17.4	21.3 – 26.7
9.0	240	2.8	60 – 400	38	22.0 – 26.7	22.0 – 33.8

To choose an appropriate distortion criterion for compression for TDOA/FDOA applications it is important to understand the impact of compression on the TDOA/FDOA accuracies rather than its impact on the signal fidelity (e.g., MSE) as is commonly done in compression algorithms. We have used the Fisher information (FI) for TDOA/FDOA estimates to gain insight into what signal characteristics can be ex-

ploited when compressing signals for TDOA/FDOA estimation [8][9],[10]. As we have derived in [8], the FI measure for the data at the Tx platform for TDOA and FDOA are, respectively,

$$J_{TDOA}(\hat{s}) = \frac{2\pi^2 k^2 \left|\hat{S}[k]\right|^2}{N\sigma^2}, \quad k = -N/2, -N/2+1, \ldots, N/2-1. \tag{13}$$

and

$$J_{FDOA}(\hat{s}) = \frac{2\pi^2 n^2 \left|\hat{s}[k]\right|^2}{\sigma^2}, \quad k = -N/2, -N/2+1, \ldots, N/2. \tag{14}$$

where σ^2 is the noise variance of the data and $\hat{S}[k]$ are the DFT coefficients of the received signal data $\hat{s}[k]$. Note that the FI measure for TDOA depends on a measure of the frequency spread of the signal's DFT and that the FI measure for FDOA depends on a measure of the time spread of the signal itself. Thus, removal of pulses would reduce J_{FDOA}; but due to the quadratic temporal weighting in (14) some pulses will cause less reduction in J_{FDOA} than other pulses. While it is possible to apply the DFT to each pulse row and apply (13) across each row and apply (14) over the pulses, we consider only the later case here; namely, this is because it is common to have fairly few samples per pulse and the frequency resolution would then be too poor to reliably apply (6). From the structure of the FI for FDOA given in (14) one can see what parts of the signal are most important for estimating FDOA and use that as a guide for eliminating pulses. From (14) we see that the pulses that are near the ends of the collected signal interval are more important than those in the center of the data, since the pulses near $t = 0$ have little contribution to Fisher information. This insight motivates the following processing steps:

1. Choose the number of pulses, η, to be deleted (based on how much compression is desired).
2. Compute $\Omega_i = \sum_{n \in i^{th} \text{ pulse index set}} n^2 |p_i[n]|^2$ for each pulse.
3. Delete the η pulses having the smallest Ω_i.

Note that the index *n* for the <u>whole</u> pulse train should run over a set that is symmetric around 0.

To get a rough idea of the impact of pulse removal we analyze the effect under an equal-amplitude pulse assumption so we can write the pulse train of p pulses as

$$s(t) = \sum_{i=1}^{p} s_o(t - t_i) e^{j\phi_i} \ , \tag{15}$$

where $s_o(t)$ is the prototype pulse, t_i is the time position of the i^{th} pulse, and ϕ_i is the phase of the i^{th} pulse. Using this in the form for CRLB for FDOA [4] when $SNR \approx DNR$ and $SNR >> 1$ we get

$$\sigma_{FDOA} \geq \frac{C_1}{\sqrt{\int_{-T/2}^{T/2} t^2 |s(t)|^2 dt}} \approx \frac{C_1}{\sqrt{\sum_{i=1}^{p} t_i^2} \times \sqrt{\int_0^{\Delta} |s_o(t)|^2 dt}} = \frac{C_{FDOA}}{\sqrt{\sum_{i=1}^{p} t_i^2}} \ , \tag{16}$$

for some appropriately chosen constant C_1. This shows that under the equal pulse amplitude assumption, pulses should be eliminated starting with the center pulse and working bi-directionally outward.

We'd also like a result that allows us to understand the effect that pulse elimination has on the accuracy of the TDOA estimate. The CRLB for TDOA [4] when $SNR \approx DNR$ and $SNR >> 1$ is

$$\sigma_{TDOA} \geq \frac{C_2}{\sqrt{\int_{-W/2}^{W/2} f^2 |S(f)|^2 df}} \ . \tag{17}$$

From (15) and properties of the Fourier transform we have that

$$|S(f)| = \left| \sum_{i=1}^{N} S_o(f) e^{j(2\pi f t_i - \phi_i)} \right| \leq \sum_{i=1}^{N} |S_o(f)| = N |S_o(f)|,$$

where we have used the inequality $|X + Y| \leq |X| + |Y|$. Using this result in (17) then gives

$$\sigma_{TDOA} \geq \frac{C_2}{N\sqrt{\int_{-W/2}^{W/2} f^2 |S_o(f)|^2 df}} \ . \tag{18}$$

This shows that eliminating l of the N pulses would increase the bound in (18) to

$$\sigma_{TDOA} \geq \frac{C_2}{(N-l)\sqrt{\int_{-W/2}^{W/2} f^2 |S_o(f)|^2 df}} = \frac{C_{TDOA}}{(N-l)} \ . \tag{19}$$

This analysis gives some motivation to expect that the TDOA estimation error's standard deviation is likely to vary with l as $1/(N-l)$. Note that the effect on TDOA shown in (19) does not depend on which pulses are eliminated and the effect in (19) is simply due to the loss of processing gain due to elimination of pulses. On the other hand, the loss in FDOA accuracy does depend on which pulses are removed and that by choosing them carefully (via the FI-based measure used above) we would expect the loss in FDOA accuracy to be small.

IV. Simulation Results

To illustrate the potential of SVD-based compression with and without FI-based pulse elimination we present some simulation results using a simulated radar signal (complex baseband) chosen to satisfy the assumptions made above. The signal is a train of pulses having linear FM modulation within each pulse. The simulated signal had the following parameters: 80 pulses, pulse width (PW) of 40μsec, a pulse repetition interval (PRI) of 70μsec, a maximum FM frequency deviation of ±2 MHz, and a sampling rate of 4.92346 MHz. The PRI was set low for convenience to reduce the total number of samples used in the processing; this ensures that the time necessary to run simulations is not unreasonably long. The sampling rate was chosen so the sampling interval was incommensurate with the PRI to ensure that the sampled pulses were not perfectly aligned. Monte Carlo simulations were performed to evaluate the standard deviation of the TDOA/FDOA estimation errors, where 400 runs were done for each evaluation. The compression ratios stated here exceed those given by the rough estimate given by (9) because we did not consider the effect of missing pulses and we used better coding methods than were assumed in the development of (9).

The first set of simulation runs were performed to establish the baseline performance of the SVD-based compression method without pulse elimination. The results are shown in Figure 8 where the estimation accuracy is assessed as a function of *SNR* (with *SNR* = *DNR*) both with and without SVD-based compression. Surprisingly, even though the compression ratio is very high, the compressed signal actu-

ally yields *better* TDOA/FDOA accuracy at the lower SNR values (and equivalent accuracy at the higher SNR values). This is due to the fact that the SVD provides and inherent de-noising property; however, surprisingly the de-noising seems to have less effect on the FDOA accuracy – an effect we currently do not understand.

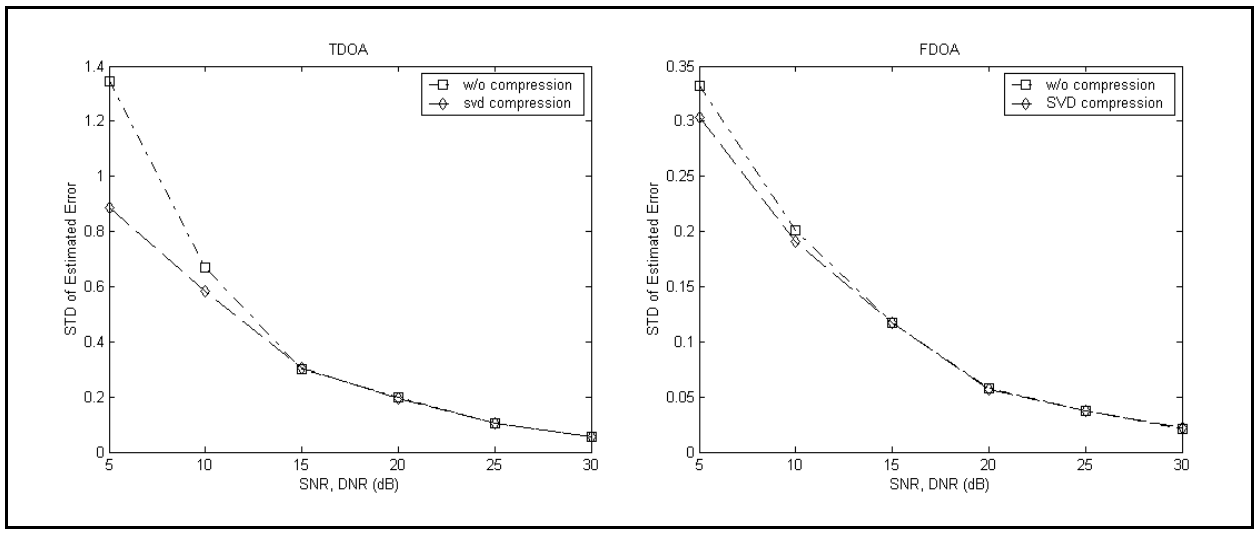

Figure 8: Simulation results showing standard deviation of TDOA/FDOA estimation error for a simulated radar signal when compressed 89:1 using SVD-based compression of the full pulse matrix (i.e., without pulse elimination); in each plot the SNRs of the two cross-correlated signals are set equal to each other as they are varied over a range. For comparison, results are shown for the case of no compression.

As we noted above, if pulses are eliminated according to the guidance of the Fisher information then we expect negligible degradation in the FDOA accuracy up to a point; however, we would expect the TDOA estimation error standard deviation to increase according to (19). Fortunately, the results in Figure 8 indicate that, due to the de-noising, we have TDOA accuracy to "spare"; this, then is a perfect setting for applying pulse elimination. To first see how many pulses can be eliminated we ran simulations at the point $SNR = DNR = 10$ dB and evaluated the error standard deviation after SVD compression with pulse elimination, as shown in Figure 9, where the standard deviation values are presented relative to the value achieved without compression. For this case we see that we can eliminate half of the 80 pulses without

suffering degradation in the FDOA accuracy, but there is some degradation of the TDOA when half of the pulses are eliminated. It should be noted that the shapes of the curves using "partial" compression roughly match the shapes predicted by (16) and (19).

Finally, for the case of eliminating half of the 80 pulses, we present results in Figure 10 that show how the TDOA/FDOA accuracy varies with SNR. Through elimination of half the pulses the compression ratio increases from 89:1 to 105:1 but the TDOA/FDOA accuracy is, remarkably, still roughly the same as when no compression is used.

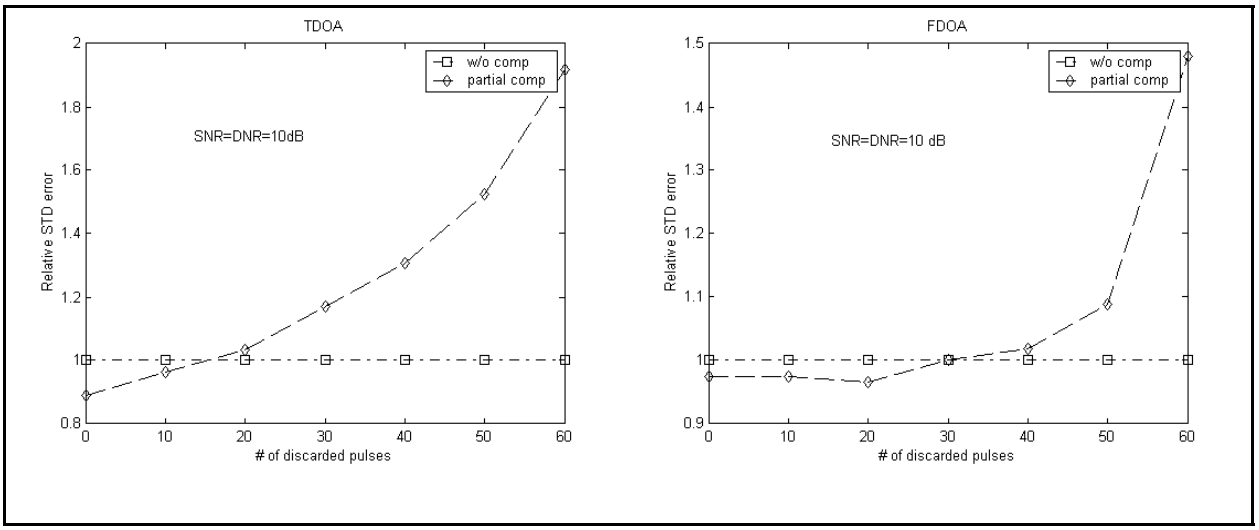

Figure 9: Simulation results showing the effect that discarding pulses has on the TDOA/FDOA estimation error standard deviation. The term "partial comp" means that SVD compression was done on a partial pulse matrix, after pulse elimination. For comparision, results are shown for the case without compression and pulse elimination.

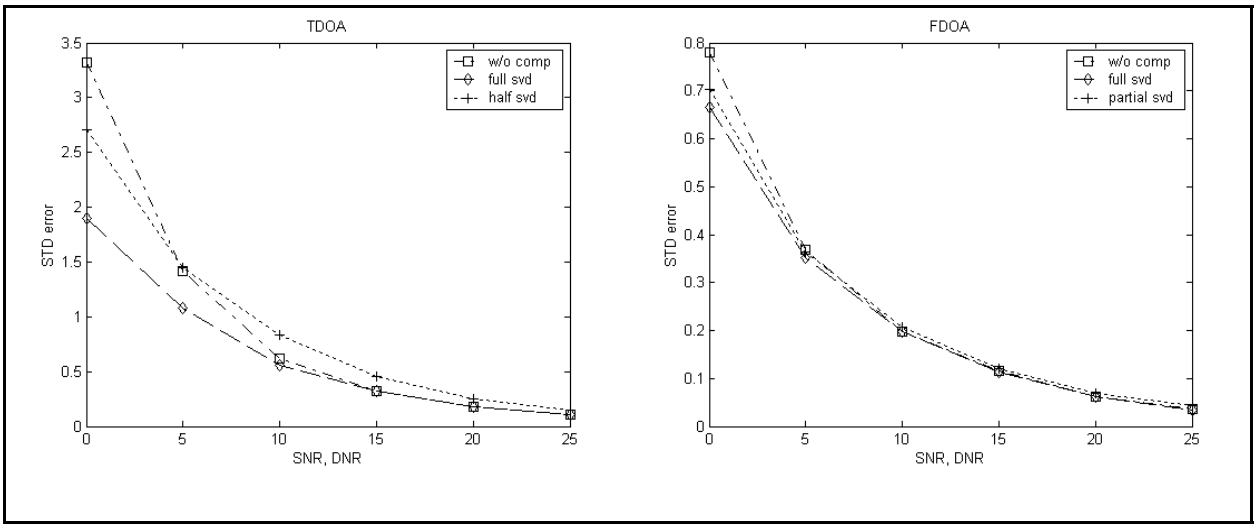

Figure 10: Simulation results showing standard deviation of TDOA/FDOA estimation error for a simulated radar signal when compressed using SVD-based compression. The three curves compare the cases (i) without compression, (ii) full SVD-based compression (i.e., no pulse elimination) with compression ratio of 89:1, and (iii) SVD-based compression where half of the pulses are eliminated to give a compression ratio of 105:1.

References

[1] P. C. Chestnut, "Emitter location accuracy using TDOA and differential doppler ," *IEEE Trans. Aero. and Electronic Systems*, vol. AES-18, pp. 214-218, March 1982.

[2] D. J. Torrieri, "Statistical theory of passive location system," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-20, no. 2, March 1984, pp. 183 – 198.

[3] S. Stein, "Differential delay/doppler ML estimation with unknown signals," *IEEE Trans. Sig. Proc.*, vol. 41, pp. 2717 - 2719, August 1993.

[4] S. Stein, "Algorithms for ambiguity function processing," *IEEE Trans. Acoust., Speech, and Signal Processing*, vol. ASSP-29, pp. 588 - 599, June 1981.

[5] M. L. Fowler and Z. Zhou, "Data compression via pulse-to-pulse redundancy for radar emitter location," ," in *Mathematics and Applications of Data/Image Coding, Compression, and Encryption IV*, Mark S. Schmalz, Editor, Proceedings of SPIE, Vol. 4475, San Diego, CA, July 29 – August 3, 2001, pp. 1 –12.

[6] J. A. Johnson and M. L. Fowler, "Handling side-information for data compression of radar pulse trains," in *Mathematics and Applications of Data/Image Coding, Compression, and Encryption V*, Mark S. Schmalz, Editor, Proceedings of SPIE, Vol. 4793, Seattle, WA, July 8 – 11, 2002, pp. 176 – 187.

[7] Z. Zhou, *Data Compression for Radar Signals: An SVD-Based Approach*, M.S. Thesis, State University of New York at Binghamton, May 2001.

[8] M. L. Fowler and M. Chen, "Fisher-information-based data compression for estimation using two sensors," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 3, July 2005, pp. 1131 - 1137.

[9] M. Chen and M. L. Fowler, "Data compression trade-offs for multiple parameter estimation in emitter location systems," submitted to *IEEE Transactions on Aerospace and Electronic Systems.*

[10] M. Chen, *Data Compression for Inference Tasks in Wireless Sensor Networks*, Ph.D. dissertation, Binghamton University, Binghamton, NY, 2005.

[11] R. G. Wiley, *Electronic Intelligence: The Interception of Radar Signals*. Artech House, 1985.

[12] R. G. Wiley, *Electronic Intelligence: The Analysis of Radar Signals*, 2nd Edition. Artech House, 1993.

[13] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, San Diego, CA: Academic Press, 1981.

[14] L. R. Foulds, *Combinatorial Optimization for Undergraduates*, New York: Springer-Verlag, 1984.

[15] T. I. Laakso, V. Valimaki, M. Karjalainen, and U. K. Laine, "Splitting the unit delay," *IEEE Sig. Proc. Mag.*, vol. 13, no. 1, pp. 30 - 60, January 1996.

[16] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing*. Prentice Hall, 2000.

[17] L. L. Scharf, "The svd and reduced-rank signal processing," *SVD and Signal Processing II, Algorithms, Analysis and Applications*, pp. 4–25, 1991.

[18] M. L. Fowler, Z. Zhou, and A. Shivaprasad, "Pulse Extraction for Radar Emitter Location," Conference on Information Sciences and Systems, Johns Hopkins University, March 21-23, 2001.