A Novel Normalization Algorithm Based on the Three-Dimensional Minimum Variance Spectral Estimator

Christopher P. Carbone
Naval Undersea Warfare Center
Division Newport
1176 Howell St.
Newport,RI 02841-1708
(email: ChristoperCarbone@navy.mil)

Steven M. Kay
University of Rhode Island
Department of Electrical,
Computer And Biomedical Engineering
4 East Alumni Ave., Kingston, RI 02881
(email: kay@ele.uri.edu)

December 1, 2009

Abstract

In reverberation limited environments, the non-target noise power in active SONAR data $\mathbf{x}$ is non-constant over range and bearing and from ping to ping. The goal of normalization is to make the noise only output of the detection test statistic $T_0(\mathbf{x})$ as constant as possible. Successful normalization makes SONAR signal processing much simpler. For example, in automatic tracking, normalizing increases the probability that true tracks get initiated and decreases the probability that false tracks get initiated. Normalizers work by estimating the background noise power, and using that estimate to divide $T_0(\mathbf{x})$ by. The split window normalizer (SWN) is a common method of normalization. It normalizes each range cell of the data by finding a local noise power estimate, and dividing the cell by that local estimate. Because the data is non-stationary the SWN normalizes a range cell by forming an estimate using bins near the cell to be estimated. This limits the total amount of data available to estimate any range cell.

The normalizer developed in this paper is based on the minimum variance spectral estimator (MVSE). The MVSE is a power spectral density (PSD) estimator that easily extends to multiple dimensions. The properties of spectral estimators will allow us to avoid the non-stationarity problem and use all available data to estimate each range cell. This allows the normalizer to use the data over the ranges, bearings and pings to perform a three dimensional (3D) estimate of the background power. This 3D estimate uses more data to normalize each element of the data than the SWN. Using more data should produce a better estimate. The SWN is compared to the new normalizer using a simple track initiation algorithm developed in this paper. Simulation results indicate the MVSE normalizer is more effective than the standard SWN for active SONAR

normalization.

Index Terms: Active SONAR, multiple target tracking, normalization, reverberation, track initiation.

1 Introduction

The goal of SONAR is to find a target of interest under the water. The underwater target could be a variety of things, for example: a school of fish, a mine, or a submarine. To accomplish this goal, active SONAR transmits sound in the water, and then receives the echo off the target of interest. The transmitted and reflected sound is referred to as a ping. The sound reflecting off the target arrives at the SONAR in a background of ambient noise and reverberation. Ambient noise is sound generated by anything in the water other than the intentional sound transmitted by the SONAR system. Some examples of things that can generate sound are: marine life, waves, and ships. Reverberation is sound transmitted by the SONAR system reflecting off anything other than the target [1]. Some examples of things that can reflect sound are: manmade objects, the surface and the bottom of the ocean, and marine life. Clutter in RADAR is electromagnetic radiation reflecting off anything other than the target [2], so the clutter problem in RADAR is similar to the reverberation problem in SONAR. In active SONAR, reverberation often dominates the effect of ambient noise so that the effect of ambient noise can be ignored [3]. When this is the case, the environment is referred to as a reverberation limited environment. This paper assumes an active SONAR in a reverberation limited environment and references to a noise background will mean reverberation only. (Note, sometimes reverberation is referred to as interference since it is correlated with the transmit signal.)

Reverberation power levels change with range and bearing and change from ping to ping. This causes SONAR data $\mathbf{x}$ to be non-white over range and bearing and non-stationary from ping to ping. The problem of detecting a target with one ping of data, is the problem of

detecting a target in non-white noise. The optimum solution to detecting a target in non-white noise is to prewhiten the data, and then apply a detection statistic $T(\mathbf{x})$ to the whitened data [2], [1], [4]. When the target in not in the data, the noise only output of the detection test statistic $T_0(\mathbf{x})$ is constant over range and bearing. The optimal method requires knowledge of the noise covariance, which in practical SONAR systems, is unavailable. If the data were stationary from ping to ping, we could get a good estimate of the noise covariance by using multiple pings of data. Unfortunately, SONAR data is not stationary from ping to ping. A common approach to this problem is to apply the detector to the data without prewhitening, resulting in a non-constant $T_0(\mathbf{x})$, and then applying a normalization algorithm to compensate for this non-constancy. The combination of a detector and a normalizer is a constant false alarm rate (CFAR) processor [5]. In addition to the challenge of the changing noise power background, normalizers are also with faced with the problem of multiple close targets and intentional interference (jamming). The problem of normalizing in the presence of jamming is not investigated in this paper.

Since practical SONARs cannot do the optimal prewhitening, normalization is an important part of SONAR signal processing. For instance normalization is an important part of multiple target tracking (MTT) systems. MTT is the estimation of the state of moving targets, using the outputs of one or more sensors [6]. MTT is used in RADAR and SONAR systems [7], [6], and [8]). A simple MTT system consists of the following elements: data from a sensor, a normalizer, an algorithm to initiate tracks, and a tracker. A block diagram for the simple MTT system discussed above is shown in Figure 1.

In an MTT system, the data from a sensor is processed by a detector. The output of the detector is then normalized to compensate for the non-white and non-constant noise

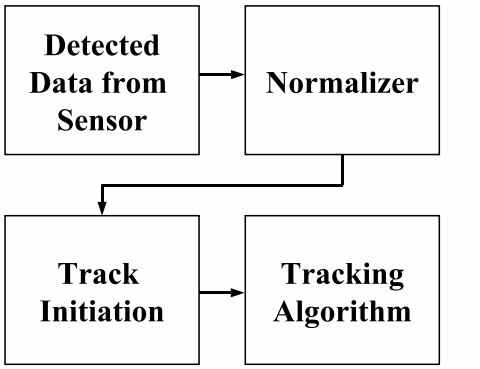


Figure 1: Simple MTT Block Diagram

background [9]. A brief overview of normalization can be found in [10]. A description of normalization as applied to SONAR can be found in Chapter 4 of [9]. There, the author describes a variation of the SWN, the Two-Pass Split-Window normalizer (TPSWN). In [11] the authors compare several types cell averaging (CA) CFAR processors (normalizers) commonly used in RADAR.

After normalization, data is then processed by a track initiation algorithm. A track initiation algorithm should initialize a track, whenever a new target appears in the scene, while minimizing the number of false tracks due to false alarms from the detector (see Hu [12]). In Hu [12] and Leung [13], the authors provide a description and performance analysis of track initiation algorithms. Track initiation becomes a more difficult problem when the background is non-white and non-constant [6].

In this paper we use the minimum variance spectral estimator (MVSE) as the basis of the 3D MVSE normalizer (3DMN). We propose the 3DMN as an improvement to the standard TPSWN. We compared the two normalizers by finding each normalizer's probability of correct track initiation (PCI), versus signal to interference ratio (SIR). This is done for

the both single target case and the multiple target case. For other methods of comparing normalizers see [14], [15], and [16]. The track initialization algorithm used in this paper was derived based on the generalized likelihood ratio test (GLRT). See [4] for a discussion of the GLRT.

Section 2 finds the GLRT statistic on which the track initiation algorithm is based on. Section 3 describes the TPSWN and the 3DMN. Section 4 contains simulation results.

2 Track Initiation Algorithm

2.1 The Problem Statement

In this section we use the generalized likelihood ratio test (GLRT) to derive a track initiation algorithm. The algorithm developed initiates tracks using an arbitrary number of pings. In deriving the algorithm we motivate the need for normalization and also develop a way to compare normalizers.

2.2 Description of the data

The data used in this paper is multiple discrete range-bearing snapshots (multiple pings of discrete range-bearing data) processed by a matched filter and an envelope detector (taking the magnitude squared). The range-bearing-ping cells prior to the envelope detector are denoted by $\widetilde{x}(m,n,t)$ (the "$\sim$" denotes complex data) and the data after the envelope detector are denoted by $x(m,n,t)$, where m is the range index, n is the beam index, and t is the ping index. The indexes take on the following values: m-range lines, $m = 0,\ldots,M-1$, n-beams, $n = 0,\ldots,N-1$, and, t-pings, $t = 0,\ldots,T-1$. $\widetilde{x}(m,n,t)$ is the sum of many independent scatterers. Most cells will contain only non-target scatters, while other cells will contain target and non-target scatters. Because $\widetilde{x}(m,n,t)$ is the sum of many independent scatters it can be assumed to be complex Gaussian noise (CN) with zero mean [1]. In cells with only reverberation the variance is $P_r(m,n,t)$. $P_r(m,n,t)$ will be referred to as the reverberation background or just background. In cells with reverberation and a target the variance is $P_r(m,n,t) + P_A(m,n,t)$. Note, $P_r(m,n,t)$ and $P_A(m,n,t)$ are not covariance matrices, they are the variance (power) of $\widetilde{x}$ at (m,n,t). Also note, if $v \sim$ CN with zero mean and variance

σ^2 and $u = |v|^2$ then $v \sim$ Exponential with parameter σ^2 [4], which we will write Exp(σ^2). Since $x(m,n,t) = |\widetilde{x}(m,n,t)|^2$ it is distributed as,

$$x(m,n,t) \sim \begin{cases} \text{Exp}(P_r(m,n,t)) & \text{when } x(m,n,t) \text{ is a cell with no target} \\ \text{Exp}(P_r(m,n,t) + P_A(m,n,t)) & \text{when } x(m,n,t) \text{ is a cell with a target} \end{cases} \tag{1}$$

As an example, Figure 2 contains 4 pings of range-bearing data. Each ping of data contains 64 range lines (the rows) and 32 beams (the columns) of independent and non-identically distributed exponential noise.

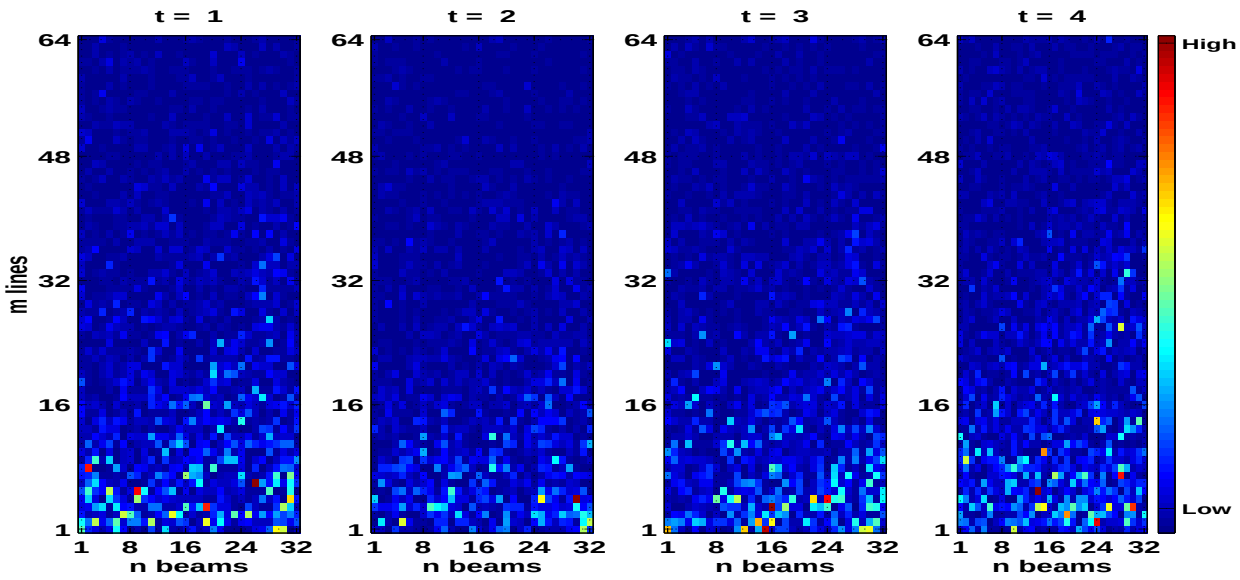


Figure 2: Four pings making up 3D data. Color bar indicates power level.

2.3 Derivation of the GLRT Track Initiation Algorithm

The track initiation algorithm was derived using the following assumptions: there is exactly one target, target extent in range and bearing are known, known reverberation background P_r, target initial position is unknown, target signal to interference ratio (SIR) is unknown but is the same in each cell with a target, and the target velocity is unknown but limited to 1 of K known constant velocities. The combination of the target extent and velocity are referred to as a "target model". Because the SIR is the same in each target cell, $P_A(m,n,t) = 10^{(SIR/10)}P_r(m,n,t)$, where SIR is in dB. This algorithm could be extended to the case of unknown target extent and non-constant velocities, but this would require more complex notation and is not done in this paper.

We set up the following hypothesis test to derive the track initialization algorithm. Under the no target hypothesis $\mathcal{H}_0$, the data has the probability density function (PDF)

$$p(x(m,n,t)) = \frac{1}{P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{P_r(m,n,t)}\right). \tag{2}$$

Under the target hypothesis $\mathcal{H}_0$, the data has the PDF

$$p(x(m,n,t)) = \begin{cases} \frac{1}{c_1 P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{c_1 P_r(m,n,t)}\right) & \text{when } (m,n,t) \in A_i(m_0,n_0) \\ \\ \frac{1}{P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{P_r(m,n,t)}\right) & \text{otherwise.} \end{cases} \tag{3}$$

where $c_1 = 1 + 10^{(SIR/10)}$, $A_i(m_0,n_0)$ is the set of range-bearing-ping cells associated with target model i and whose starting location is at m_0, n_0 in ping one. Since $x(m,n,t) = |\widetilde{x}(m,n,t)|^2$, $x(m,n,t) \geq 0$ so that this restriction is implicit in equations (2) and (2).

$A_i(m_0, n_0)$ is defined as follows,

$$A_i(m_0, n_0) = \{a_i(t=0),\ a_i(t=1),\ \ldots a_i(t=T-1)\} \tag{4}$$

where, $a_i(t) = [m_0 + tv_i^m, m_0 + e^m + tv_i^m] \times [n_0 + tv_i^n, n_0 + e^n + tv_i^n]$, m_0 is the start of the target range cells, n_0 is the start of the target bearing cells, e^m is the line extent, e^n is the beam extent, v_i^m is the speed in lines, and v_i^n is the speed in beams. Because target's velocity is limited to 1 of K known velocities, v_i^m and v_i^n are constrained, but not known. Figure 3 is an example of equation (4) when $m_0 = 8$,$n_0 = 16$, $v_i^m = 0$, $v_i^n = 2$, $e^m = 1$ and $e^n = 1$.

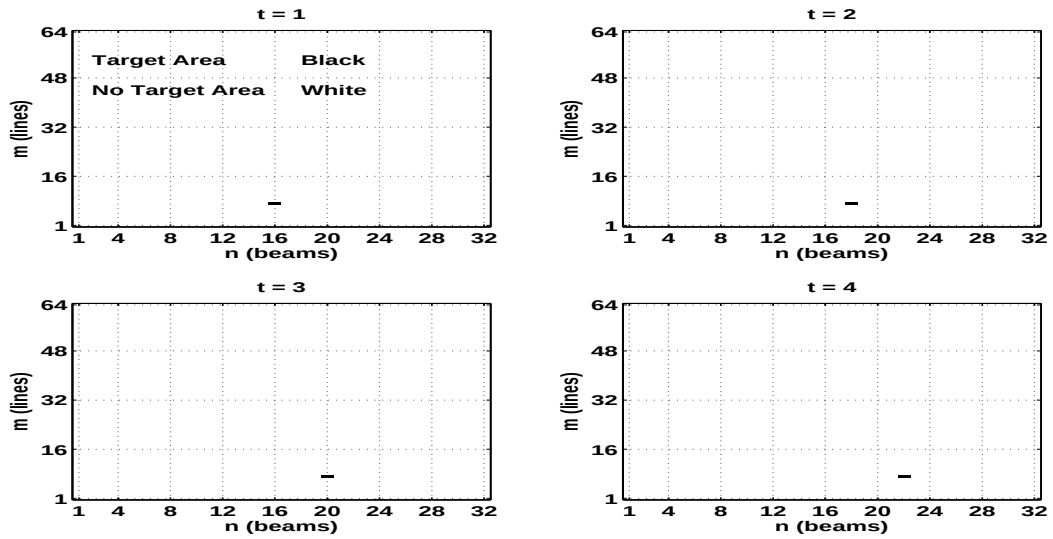


Figure 3: Target Model Example

When setting up the GLRT, we have the following unknowns: target starting location (m_0, n_0), and target model i. Because the PDF under $\mathcal{H}_0$ is completely known, the GLRT

can be written as (see [4])

$$L_G(\mathbf{x}) = \max_{m_0,n_0,i} \frac{p(\mathbf{x}; m_0, n_0, i, c_1 P_r; \mathcal{H}_1)}{p(\mathbf{x}; P_r; \mathcal{H}_0)}. \tag{5}$$

The PDF under $\mathcal{H}_1$ is

$$p(\mathbf{x}; m_0, n_0, i, c_1, P_r, \mathcal{H}_1) = \prod_{(m,n,t)\in A_i(m_0,n_0)} \prod \prod \frac{1}{c_1 P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{c_1 P_r(m,n,t)}\right) \cdot \prod_{(m,n,t)\notin A_i(m_0,n_0)} \prod \prod \frac{1}{P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{P_r(m,n,t)}\right).$$

The PDF under $\mathcal{H}_0$ is

$$p(\mathbf{x}; P_r, \mathcal{H}_0) = \prod_{(m,n,t)\in A_i(m_0,n_0)} \prod \prod \frac{1}{P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{P_r(m,n,t)}\right) \cdot \prod_{(m,n,t)\notin A_i(m_0,n_0)} \prod \prod \frac{1}{P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{P_r(m,n,t)}\right).$$

Because the two PDF are the same for cells without a target we can write the GLRT as

$$L_G(\mathbf{x}) = \max_{m_0,n_0,i} \frac{\prod \prod \prod_{(m,n,t)\in A_i(m_0,n_0)} \frac{1}{c_1 P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{c_1 P_r(m,n,t)}\right)}{\prod \prod \prod_{(m,n,t)\in A_i(m_0,n_0)} \frac{1}{P_r(m,n,t)} \exp\left(\frac{-x(m,n,t)}{P_r(m,n,t)}\right)}. \tag{6}$$

Rewriting (6) we have

$$L_G(\mathbf{x}) = \max_{m_0,n_0,i} \frac{\left[\prod \prod \prod_{(m,n,t)\in A_i(m_0,n_0)} \frac{1}{c_1 P_r(m,n,t)}\right] \exp\left(\sum \sum \sum_{(m,n,t)\in A_i(m_0,n_0)} \frac{-x(m,n,t)}{c_1 P_r(m,n,t)}\right)}{\left[\prod \prod \prod_{(m,n,t)\in A_i(m_0,n_0)} \frac{1}{P_r(m,n,t)}\right] \exp\left(\sum \sum \sum_{(m,n,t)\in A_i(m_0,n_0)} \frac{-x(m,n,t)}{P_r(m,n,t)}\right)}. \tag{7}$$

Regrouping terms together equation (7) becomes

$$L_G(\mathbf{x}) = \max_{m_0,n_0,i} \prod\prod\prod_{(m,n,t)\in A_i(m_0,n_0)} \frac{P_r(m,n,t)}{c_1 P_r(m,n,t)} \exp\left(\sum\sum\sum_{(m,n,t)\in A_i(m_0,n_0)} \frac{x(m,n,t)}{P_r(m,n,t)} - \right.$$

$$\left. \sum\sum\sum_{(m,n,t)\in A_i(m_0,n_0)} \frac{x(m,n,t)}{c_1 P_r(m,n,t)} \right). \tag{8}$$

We can drop $\prod\prod\prod_{(m,n,t)\in A_i(m_0,n_0)} \frac{P_r(m,n,t)}{c_1 P_r(m,n,t)}$ since it is a positive constant and will not affect the maximization. After dropping the constant and simplifying equation (8) we get

$$L_G(\mathbf{x}) = \max_{m_0,n_0,i} \exp\left(\sum\sum\sum_{(m,n,t)\in A_i(m_0,n_0)} x(m,n,t) \left(\frac{1}{P_r(m,n,t)} - \frac{1}{c_1 P_r(m,n,t)} \right) \right).$$

Taking the natural logarithm, we have

$$l_G(x) = \ln(L_G(\mathbf{x})) = \max_{m_0,n_0,i} \sum\sum\sum_{(m,n,t)\in A_i(m_0,n_0)} x(m,n,t) \left(\frac{1}{P_r(m,n,t)} - \frac{1}{c_1 P_r(m,n,t)} \right). \tag{9}$$

Combining terms from equation (9) produces

$$l_G(x) = \max_{m_0,n_0,i} \sum\sum\sum_{(m,n,t)\in A_i(m_0,n_0)} x(m,n,t) \left(\frac{c_1 - 1}{c_1 P_r(m,n,t)} \right). \tag{10}$$

Taking the constant out of the sums yields

$$l_G(x) = \max_{m_0,n_0,i} \frac{c_1 - 1}{c_1} \sum\sum\sum_{(m,n,t)\in A_i(m_0,n_0)} \frac{x(m,n,t)}{P_r(m,n,t)}. \tag{11}$$

Finally, we note that the since $(c_1 - 1)/c_1$ is greater than zero the test statistic in equation (11) can be modified to

$$l(x) = \max_{m_0,n_0,i} \sum\sum\sum_{(m,n,t)\in A_i(m_0,n_0)} \frac{x(m,n,t)}{P_r(m,n,t)}. \tag{12}$$

Equation (12) could also be written as

$$T(\mathbf{x}) = \max_{m_0,n_0,i} \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1} I_i(m_0,n_0)\frac{x(m,n,t)}{P_r(m,n,t)} \tag{13}$$

where,

$$I_i(m_0,n_0) = \begin{cases} 1 & (m,n,t) \in A_i(m_0,n_0) \\ 0 & \text{otherwise.} \end{cases}$$

Note that equation (13) is a "correlation" between the target models $I_i(m_0, n_0)$ and the given range-bearing-ping data $x(m, n, t)$ divided by the known background power P_r. The combination of m_0,n_0, and i that produces the largest test statistic is the maximum likelihood estimator (MLE) for target model i, and initial position (m_0, n_0). Equation (13) is similar to a "time domain" version of the 3D matched filter discussed in [17] and [18]. However, unlike earlier work this method does not assume a known target intensity distribution or target velocity.

Equation (13) requires knowledge of P_r which in practical problems is not available. So, to use equation (13), we will first have to find an estimate of P_r. This estimate will come from the normalizers described in the next section. It is assumed that the better an estimate is for P_r, the better the results when equation (13) is used to initialize tracks.

3 Description of Two Normalization Algorithms

3.1 Two Pass Split Window Normalizer (TPSWN)

The basic SWN (or CA CFAR processor) finds a local power estimate around a range cell, then uses that power estimate to normalize the cell. The local power estimate is found by averaging range cells in windows around either side of the cell to be normalized. The windows around the cell to be normalized should be big enough to get a good (low variance) estimate of the background power, but small enough to not be too biased by the non-stationarity of the data. When using a SWN, this is an unavoidable tradeoff and limits the amount of data used to estimate each range-bearing-ping cell power. The windows on either side of the cell are separated by a gap window which is bigger than the biggest likely target. This keeps the target from biasing the background power estimate. But, the gap window shouldn't too large, because it excludes the cells nearest to the cell to be normalized. The windows and gap are then moved down the beam until all the range cells in the beam have been normalized. This is repeated for all the beams in the data for each ping.

There are many variations of SWN employed in SONAR systems. The variations try to improve the SWN's ability to estimate the background in the presents of non-uniform noise and multiple close targets. Variations on the SWN can be found in [9],[14] and [15]. We used the TPSWN because: it is in common use and it has good performance in the presences of non-uniform noise background. Below is a description of the TPSWN that is used in this paper.

A local first-pass mean $\overline{z}$ is found for each range cell m and for all beams n in each ping

t. This will require $n \times t$ first-pass means, given by

$$\overline{z}_{n,t}(m) = \frac{1}{k} \sum_{i=m-(W_s+(W_g-1)/2)}^{m+(W_s+(W_g-1)/2)} \Omega(i) x(i,n,t) \tag{14}$$

where W_s is number of range cells on one side of the cell to be normalized (the sum window), W_g is the odd number of skipped range cells in the middle of the sum windows (the gap window), $k = 2W_s$ and

$$\Omega(i) = \begin{cases} 0 & \text{if } |i-m| \leq (W_g-1)/2 \\ 1 & \text{otherwise .} \end{cases}$$

The gap window (0's) in Ω is chosen to have more range cells than the expected range extent of the target. The sum window should have enough range cells to provide adequate averaging, but be small enough so the noise power is approximately constant over Ω.

Each first-pass mean is then compared to its respective range cell for all beams in each ping.

$$y_{n,t}(m) = \begin{cases} x(m,n,t) & \text{if } x(m,n,t) < r\overline{z}_{n,t}(m) \\ \overline{z}_{n,t}(m) & \text{if } x(m,n,t) \geq r\overline{z}_{n,t}(m) \end{cases}$$

With the clip threshold r chosen to be a compromise between downward bias of the signal estimate and the probability that high level is part of the signal [19]. Some methods to choose r are presented in [14] and [15].

A local second-pass mean is found for each range cell from the sequence $y_{n,t}(m)$.

$$\overline{y}_{n,t}(m) = \frac{1}{k} \sum_{i=m-(W_s+(W_g-1)/2)}^{m+(W_s+(W_g-1)/2)} \Omega(i) y_{n,t}(i) \tag{15}$$

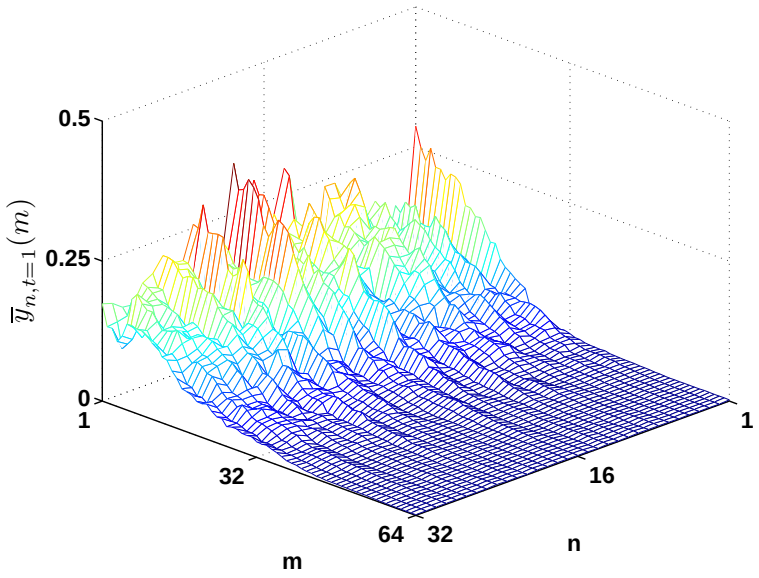
Once we have $n \times t$ second-pass means $\overline{y}_{n,t}(m)$ we can normalize the data as follows,

$$\overline{x}_{SW}(m,n,t) = x(m,n,t)/\overline{y}_{n,t}(m). \tag{16}$$

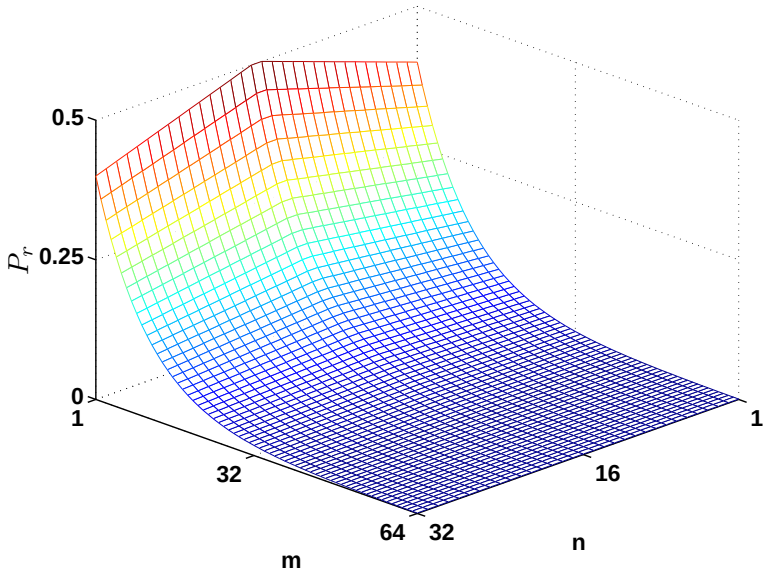
Note $\overline{y}_{n,t}(m)$ in equation (15) can be thought of as an estimate of the exponential PDF parameter P_r in equation (1). Ping one of $\overline{y}_{n,t}(m)$ is plotted in Figure 4a along with the exponential PDF reverberation parameter $P_r(m,n,t)$ (Figure 4b). The normalized data, $\overline{x}_{SW}$, is then used in the track initialization algorithm.

3.2 3D MVSE Normalizer (3DMN)

The 3DMN is based on a 3D power spectral density (PSD) estimator. At the output of a detector, SONAR data is positive and therefore it can be thought of as a PSD [20]. So, a sequence of range-bearing plots stacked on top of each other can be treated as a 3D PSD. A PSD estimator will alow us to estimate $x(m,n,t)$ as it changes over range, bearing, and ping. Since a PSD is stationary by definition, we can estimate a changing $x(m,n,t)$ and avoid non-stationary problem. The PSD estimator used in the 3DMN is the 3D minimum variance spectral estimator (MVSE). The MVSE is also know as Capon's method [21]. The MVSE was chosen because, unlike autoregressive (AR) estimation, it easy extends to three dimensions and it has better resolution than classical spectral estimators [22]. A preliminary



(a) TPSWN background estimate.



(b) True reverberation background.

Figure 4: TPSWN estimate vs. true background.

version of this normalizer was presented in [23]. A detailed discussion of the MVSE and how it can be extended to multiple dimensions can be found in [24] and [25]. The 3DMN is

implemented as follows:

1) First find the estimated autocorrelation matrix $\widehat{\mathbf{R}}_{xx}$. The elements of $\widehat{\mathbf{R}}_{xx}$ are a subset of the elements from the 3D autocorrelation function (ACF). In this paper, elements from the 3D ACF are referred to as autocorrelation function elements (ACE). In the 3DMN, the ACE are found by performing a 3D inverse Fourier transform (IFT) on a mirror imaged (made even) version of the 3D data $x(m, n, t)$. The data are mirror imaged to reduce bias in the MVSE. (See Appendix A for a discussion of mirror imaging.) Because the data can be thought of as a 3D PSD, the 3D inverse Fourier transform results in a 3D ACF [20]. The 3D ACF function contains the ACE needed for the $\widehat{\mathbf{R}}_{xx}$. The ACE are arranged in $\widehat{\mathbf{R}}_{xx}$ in exactly the same way as an autocorrelation matrix of a first quadrant (quarter plane in 2D) AR model [24]. We want to select enough ACE to get a good estimate of the reverberation background but not so many that we introduce too much variance to the background estimate. Appendix B has further details on the construction of the autocorrelation matrix see.

2) Next, the 3D MVSE is found as

$$\widehat{P}(m, n, t) = \frac{(p_m + 1)(p_n + 1)(p_t + 1)}{\mathbf{e}^H \widehat{\mathbf{R}}_{xx}^{-1} \mathbf{e}} \tag{17}$$

where p_m is the largest range line lag, p_n is the largest beam lag, p_t is the largest ping lag, H is the hermitian transpose, $\mathbf{e}^H =$

$$[z_1^0 z_2^0 z_3^0 \;\; z_1^0 z_2^0 z_3^1 \ldots z_1^0 z_2^0 z_3^{p_t} \;\; z_1^0 z_2^1 z_3^0 \ldots z_1^0 z_2^{p_n} z_3^0 \ldots z_1^0 z_2^{p_n} z_3^{p_t} \;\; z_1^1 z_2^0 z_3^0 \ldots z_1^{p_m} z_2^{p_n} z_3^{p_t}]$$

with $z_1 = exp(j2\pi m/M)$, $z_2 = exp(j2\pi n/N)$, and $z_3 = exp(j2\pi t/T)$. A one ping slice $\widehat{P}_{m,n,t=1}$ is plotted (see Figure 5a) along with the exponential PDF reverberation parameter

$P_r(m,n,t)$ (see Figure 5b).

3) Finally, once we have $\widehat{P}(m,n,t)$ we can normalize the data as

$$\overline{x}_{3DMN}(m,n,t) = x(m,n,t)/\widehat{P}(m,n,t). \tag{18}$$

The normalized data, $\overline{x}_{3DMN}$, is then used in the track initialization algorithm.

3.3 Bias and Variance in the Normalizers

The bias and variance of the two normalizers were compared using computer simulations. The first step was to generate the simulated data. The data consisted of 4 pings of range-bearing plots, for each trial of the simulation. Each range-bearing plot has 64 range lines and 32 beams. This results in 8192 range-bearing-ping cells ($64 \cdot 32 \cdot 4$). Each cell consists of independent non-identically distributed exponential data. The cells consist of reverberation power only. The cells are distributed as exponential with parameter $P_r(m,n,t)$, where $P_r(m,n,t)$ is decaying in range and varies slowly with bearing and ping. This model was chosen because it is reasonable for reverberation. Figure 6 is a plot of $P_r(m,n,t)$ used in the simulation. Figure 7 is a single realization of the noise background $x(m,n,t)$ distributed as exponential with the parameter $P_r(m,n,t)$, as seen in Figure 6. A total of fifty realizations were generated.

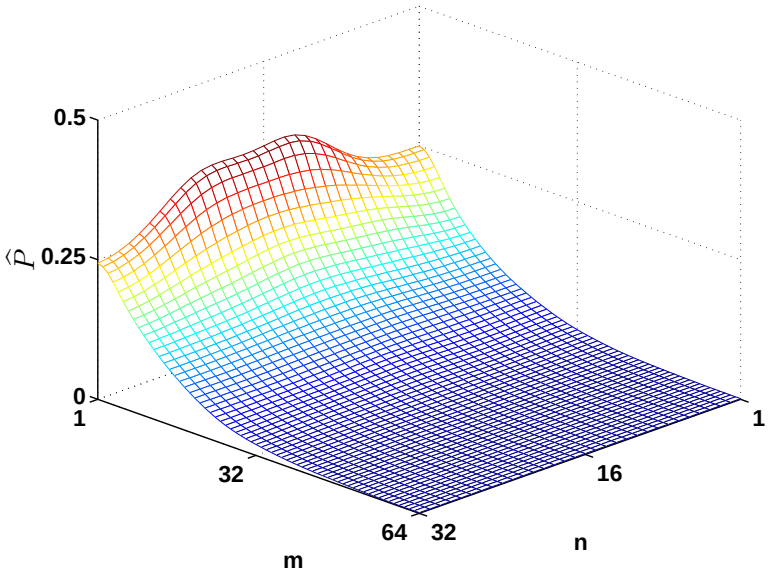
After the data was generated, all fifty realizations were normalized by both normalizers. The TPSWN had the following parameter settings: the sum window $W_s = 10$, the gap window $W_g = 3$, and the clip threshold $r = 3$. These parameters were chosen by trial and error to give the best probability of correct initiation (PCI) for a probability of false

initiation (PFI) of 0.01. For details on this see section 4. We were able to pick the smallest, and best gap window because we know that in Section 4, the computer simulation section uses a single cell target, so a gap of one matches the target. In practice the TPSWN does not enjoy this advantage. The MVSE in the 3DMN normalizer used ACE with the following lags: thirteen range lags ($l_m = 0, 1, 2, ..., 12$), seven beam lags ($l_n = 0, 1, 2, ..., 6$) and three ping lags ($l_t = 0, 1, 2$). The ACE were chosen by trial and error, but knowledge of the target was not used to help pick the parameters.

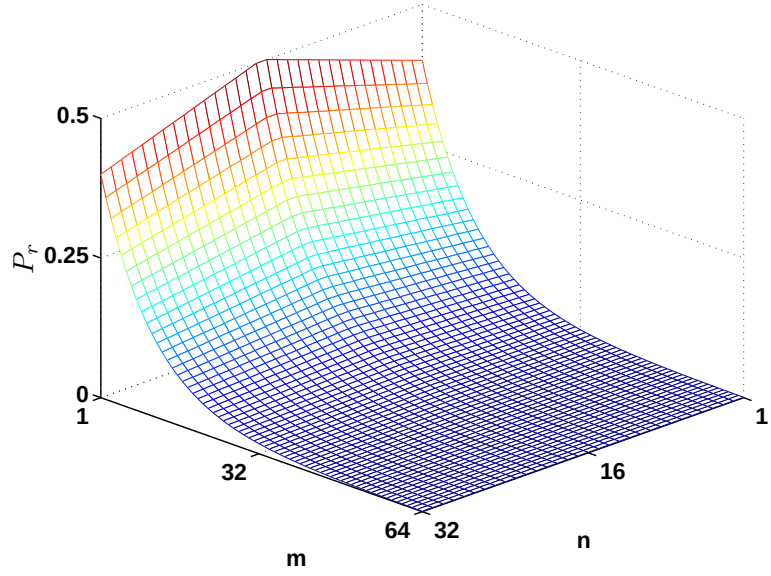
In order to more easily view the estimator variance, beam sixteen of ping one was chosen. The other beams in other pings looked similar. Figure 8 consists of fifty overlaid realizations of $\overline{y}_{16,1}(m)$ (for details on $\overline{y}_{n,t}(m)$ see equation (15)) from the TPSWN and $P_r(m, 16, 1)$. Figure 9 consists of fifty overlaid realizations of $\widehat{P}_{m,16,1}$ from 3DMN and $P_r(m, 16, 1)$. For details on $\widehat{P}_{m,n,t}$ see equation (18). As can be seen from Figures 8 and 9 the 3DMN has less variance than the TPSWN. Figure 10 is a plot of the average of 50 realizations of both estimates. As can be seen from Figure 13 the 3DMN estimate has less bias than the TPSWN estimate.

Because the parameters for both the normalizers have been picked by trial and error using the noise backgrounds made with the exponential PDF parameter $P_r(m, n, t)$ from Figure 6, one might wonder how well they will work with another noise background. So, we repeat the above procedure but with the exponential PDF parameter in Figure 11. Figure 12 consists of fifty overlaid realizations of $\overline{y}_{16,1}(m)$ from the TPSWN and $P_r(m, 16, 1)$. Figure 13 consists of fifty overlaid realizations of $\widehat{P}_{m,16,1}$ from the 3DMN and $P_r(m, 16, 1)$. As can be seen from Figures 12 and 13 the 3DMN has less variance than the TPSWN. Figure 14 plots averages of the 50 realizations of both normalizers. As can be seen from Figure 14 the

3DMN has less bias than the TPSWN.



(a) Reverberation background estimate from the 3DMN.



(b) True reverberation background.

Figure 5: 3DMN estimate vs. true background.

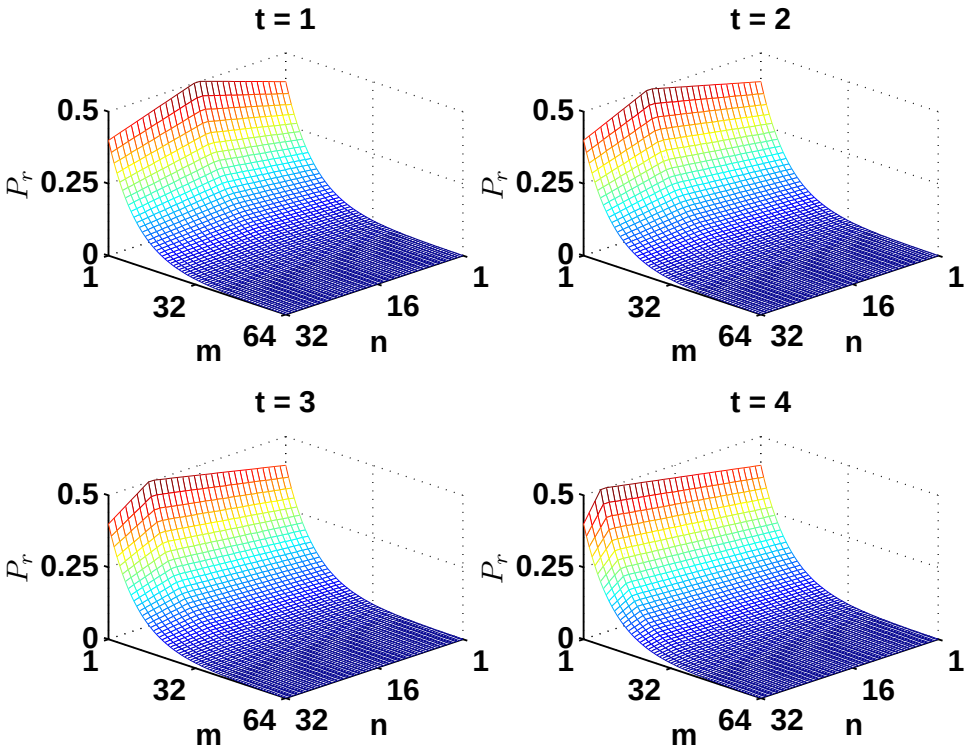


Figure 6: Plot of $P_r(m, n, t)$.

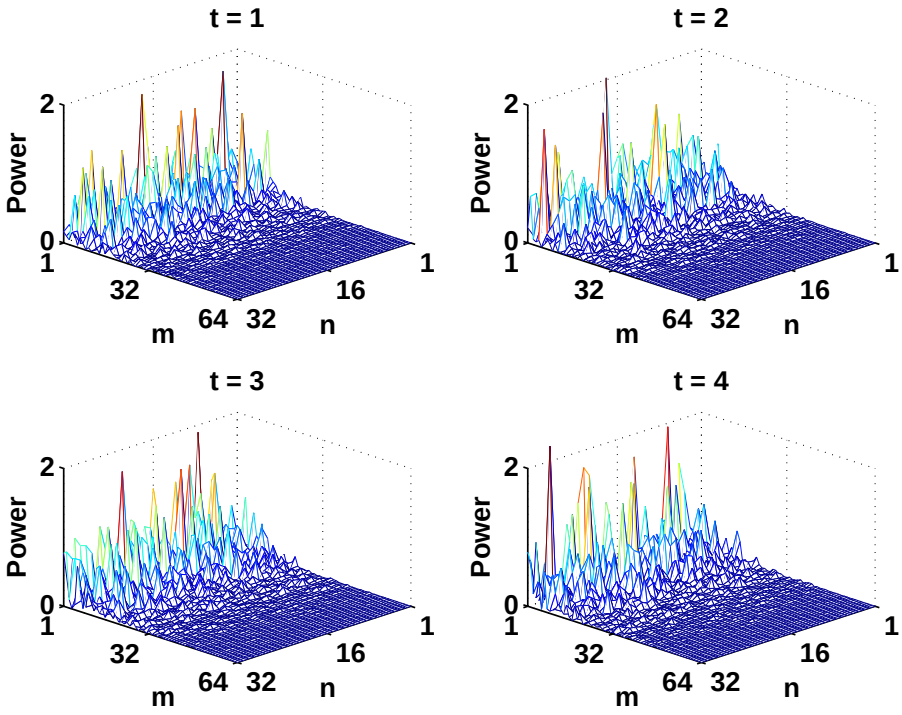


Figure 7: A plot of a single realization of $x(m, n, t)$.

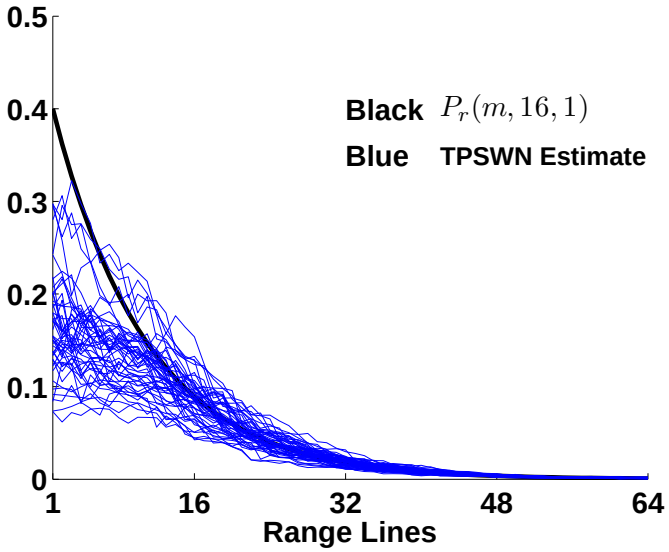


Figure 8: Overlaid realizations of $\overline{y}_{16,1}(m)$.

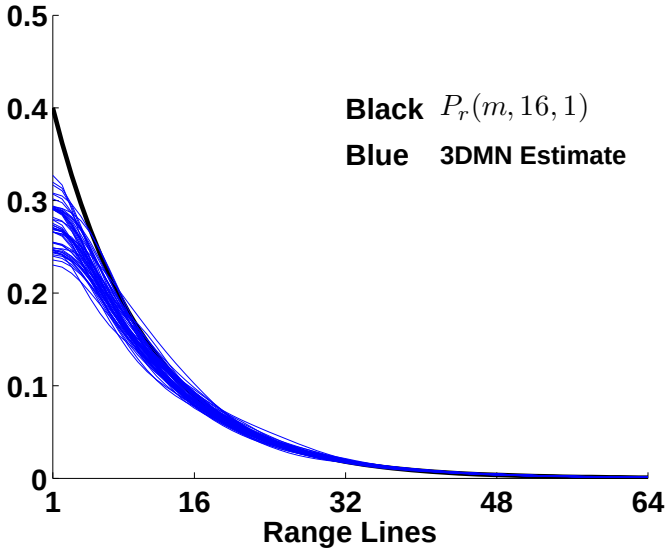


Figure 9: Overlaid realizations of $\widehat{P}_{m,16,1}$.

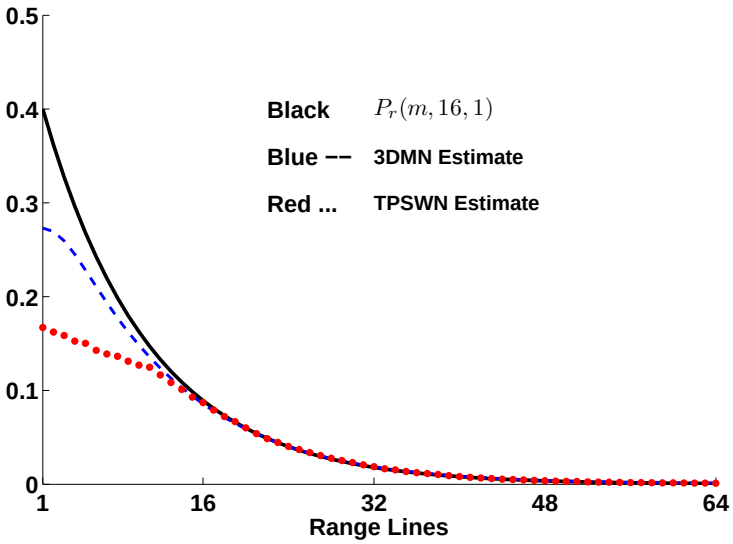


Figure 10: Average of Realizations

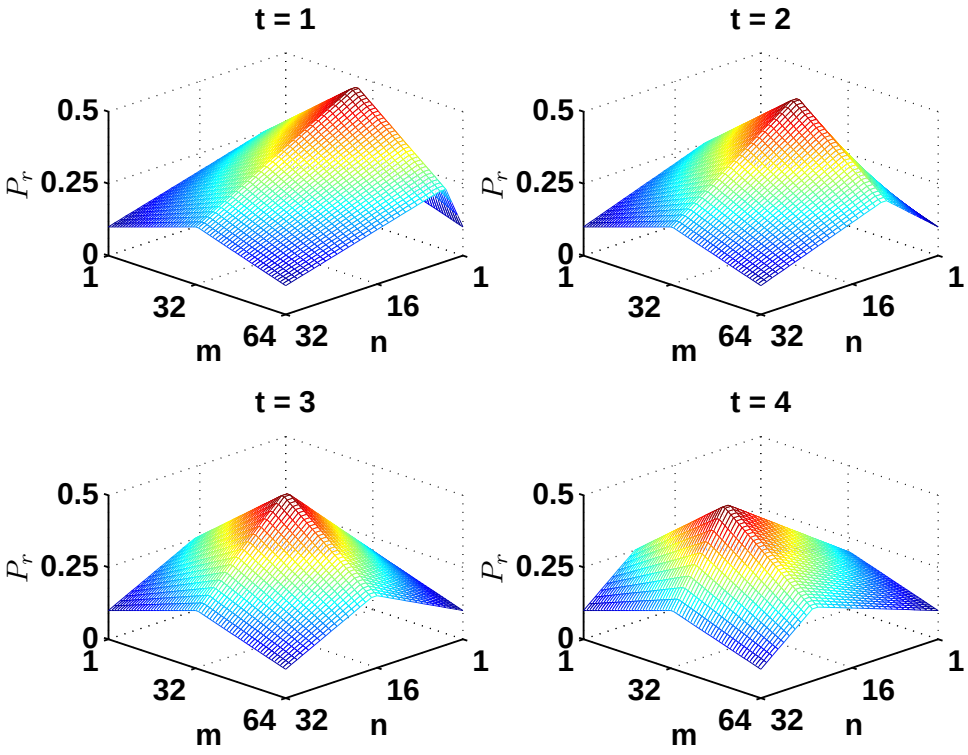


Figure 11: Plot of another $P_r(m, n, t)$.

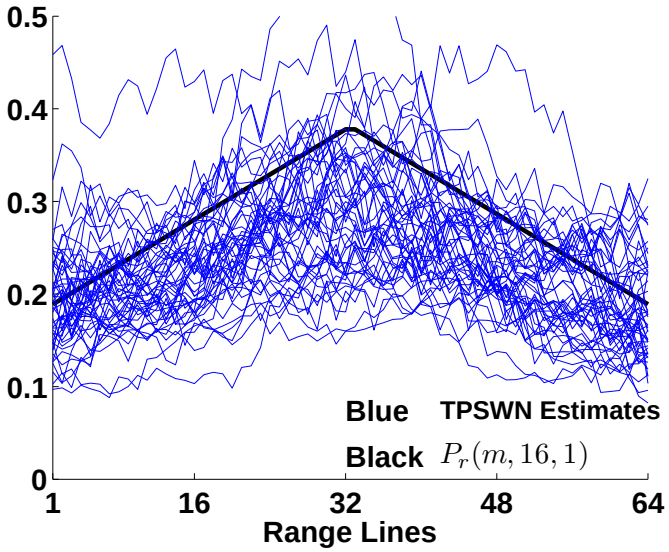


Figure 12: Overlaid realizations of $\overline{y}_{16,1}(m)$.

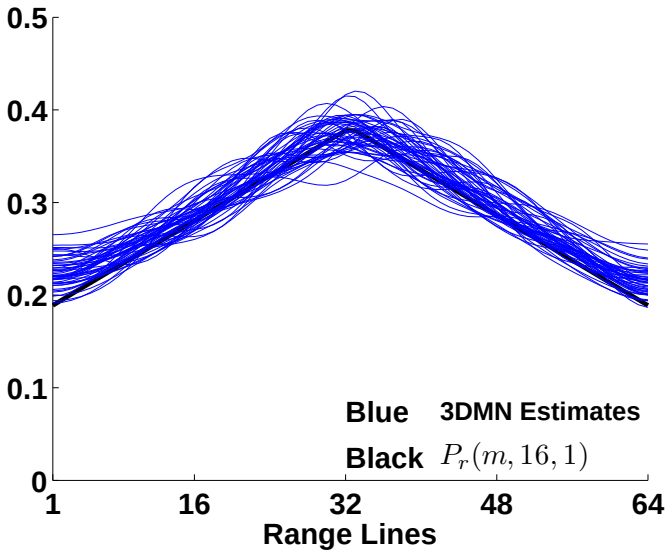


Figure 13: Overlaid realizations of $\widehat{P}_{m,16,1}$.

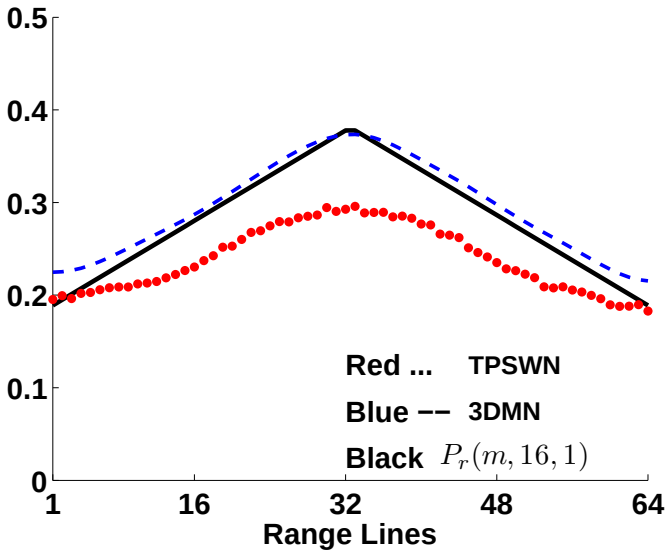


Figure 14: Average of Realizations

4 Track Initiation Computer Simulations

In this section we plot the results of the PCI versus the SIR for the 3DMN and the TPSWN. The first step was to generate the simulated exponentially distributed data. Next the simulated data was normalized by both the 3DMN and the TPSWN. The two sets of normalized data were then processed with the GLRT track initialization algorithm. After the track initiation was completed, the PCI/SIR curves were composed. See Figure 15 for overview.

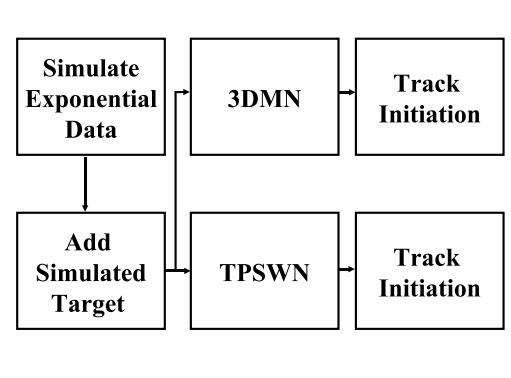


Figure 15: Simulation Block Diagram

4.1 Simulate Detector Output

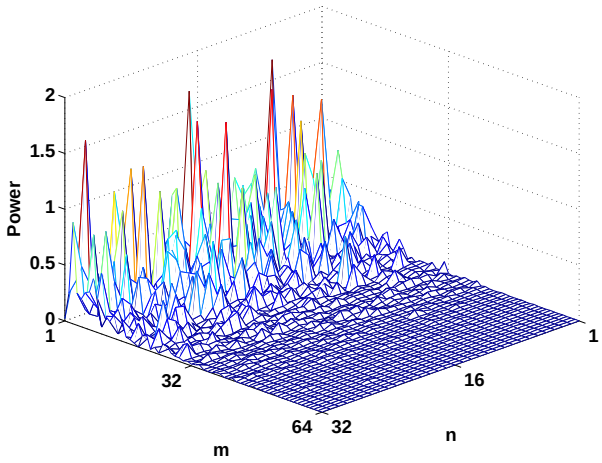
We generated 4 pings of range-bearing plots for each trial of the simulations. Each range-bearing plot has 64 range lines and 32 beams. Each cell consists of independent non-identically distributed exponentially distributed data. The cells consist of reverberation only or reverberation plus a target. The reverberation only cells, used in generating Figures 19, to 22, are distributed as $\text{Exp}(P_r(m,n,t))$, where $P_r(m,n,t)$ is a decaying in range and

varies slowly with bearing and ping. Figure 6 is a plot of $P_r(m,n,t)$ used in the simulation and Figure 7 is a realization of the noise background made using the exponential parameter in Figure 6. The reverberation only cells, used in generating Figure 23, are distributed as $\text{Exp}(P_r(m,n,t))$, where $P_r(m,n,t)$ is increases in range and varies slowly with bearing and ping. Figure 11 is a plot of $P_r(m,n,t)$ used in the simulation generating Figure 23.

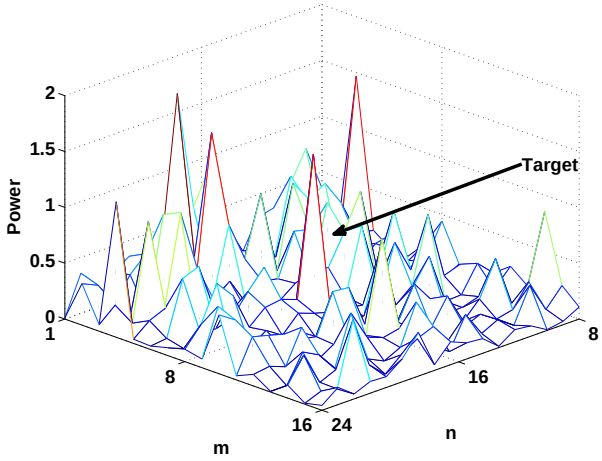
The cells with a target and reverberation are distributed as $\text{Exp}(P_A(m,n,t)+P_r(m,n,t))$, where $P_A(m,n,t) = 10^{(SIR/10)}P_r(m,n,t)$ in a cell with a target and 0 otherwise. SIR is the signal to interference ratio in dB. The multiple $10^{(SIR/10)}$ is constant over the cells with a target. So, the simulated data $x(m,n,t)$ has the distribution,

$$x(m,n,t) \sim \begin{cases} \text{Exp}(P_A(m,n,t) + P_r(m,n,t)) & \text{if cell } (m,n,t) \text{ contains a target} \\ \text{Exp}(P_r(m,n,t)) & \text{otherwise.} \end{cases} \tag{19}$$

In all the simulations, only one range-bearing cell per ping has power from a target. Because there are four pings per realization of $x(m,n,t)$ there are four cells that contain power form a target in each realization of $x(m,n,t)$. For example in the simulations that made Figures 19 and 20, the target was at a constant range and moving up in beams as shown in Figure 3. To be specific, the data cells $x(8,16,1)$, $x(8,18,2)$, $x(8,20,3)$, and $x(8,22,4)$ had a target present. With the target at range 8, it is in the high reverberation portion of the data. Figure 16 is an example of the range-bearing plot for a single ping of a simulated 9 dB target when injected into the high reverberation region of the background noise data.



(a) Target injected into the background noise.



(b) Zoomed in view of target.

Figure 16: Example of target injected into the background noise.

4.2 Normalize the Simulated Detector Output

Each realization $x(m,n,t)$ was normalized by the two normalizers as described in Section 3. This resulted in normalized data $\overline{x}(m,n,t)$. In the simulations the TPSWN had the following parameter settings; $W_s = 10$, $W_g = 3$, and $r = 3$. The MVSE in the 3DMN normalizer used ACE with the following lags: thirteen range lags ($l_m = 0,1,2,...,12$), seven beam lags ($l_n = 0,1,2,...,6$) and three ping lags ($l_t = 0,1,2$).

4.3 GLRT Track Initialization

To perform the track initiation algorithm as derived in equation (13) we would need P_r, which we don't have. So we replace $x(m,n,t)/P_r(m,n,t)$ with the normalized data $\overline{x}(m,n,t)$. Now we can find the "correlation" between the normalized data $\overline{x}(m,n,t)$ and the target models, $I_i(m_0,n_0)$, as

$$\max_{m_0,n_0,i} \sum\sum\sum_{(m,n,t)} I_i(m_0,n_0)\overline{x}(m,n,t). \tag{20}$$

The simulations used eight target models $I_i(m_0,n_0)$. The target models were all single cell models and matched in speed to the simulated target. The target models consist of eight combinations of moving zero or two beams left or right beam and zero or two lines up and down in line (see Figure 17).

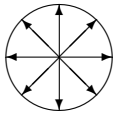


Figure 17: The Eight target model trajectories

The maximum in equation (20) occurs at a particular m_0, n_0 and i_0 that provides the

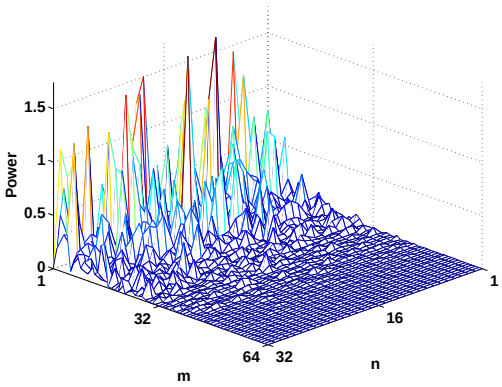
initial location and subsequent trajectory of a possible target. m_0 and n_0 gives the location of the target in ping one. i_0 gives the direction and speed of the target. Therefore, the location of the target in pings two, three, and four can be calculated using the target model i_0.

Figure 18 is a graphic depiction of the process of MVSE normalization followed by the track initiation in (20). Figure 18a is one realization of one ping of the data $x(m, n, t = 1)$. Figure 18b is one ping of the normalized data $\overline{x}_{3DMN}(m, n, t = 1)$. Figure 18c shows the location of the maximum of equation (20). Figure 18c was made by keeping i fixed to one, and letting m and n vary over their ranges. The lower 3 sub-Figures of Figure 18 are a zoomed view of 20.

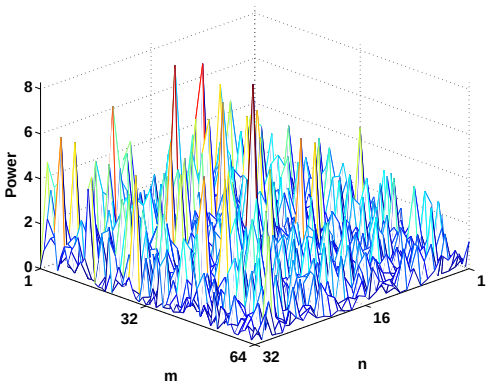
4.4 PCI/SIR Curves

Thresholds that result in a probability of false initiation (PFI) of 0.01 were found for the 3DMN and TPSWN. (A 3DMN threshold for a PFI of 0.001 was also found.) These were found by using $10,000$ realizations of reverberation only data . The data was normalized with both of the above normalizers. Both normalized outputs were operated on by the track initiation test statistic (equation (20)). A threshold was chosen for each normalizer/track initiator that produced the desired PFI. The thresholds used in this paper are summarized in Table 1.

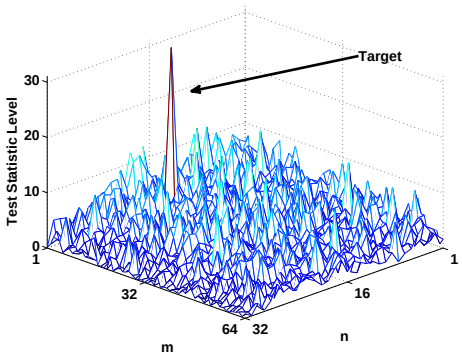
After the thresholds were found, the PCI/SIR curves were generated by first generating 1,000 realizations of $x(m, n, t)$ with a SIR dB target for SIR $= \{5, 6, \ldots 14\}$. Then each $x(m, n, t)$ is normalized with both the normalizers. Then, both normalized data sets are operated on by equation (20), which produces a maximum at a particular (m_0, n_0, i_0) for



(a) One ping slice of the data.

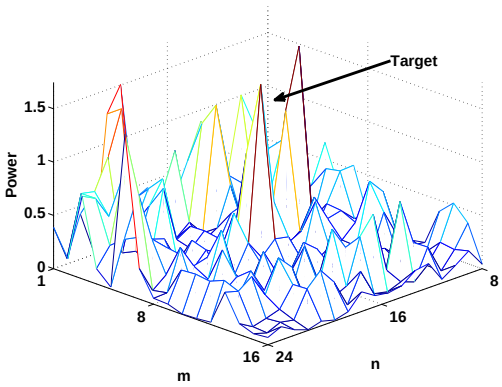


(b) Normalized data.

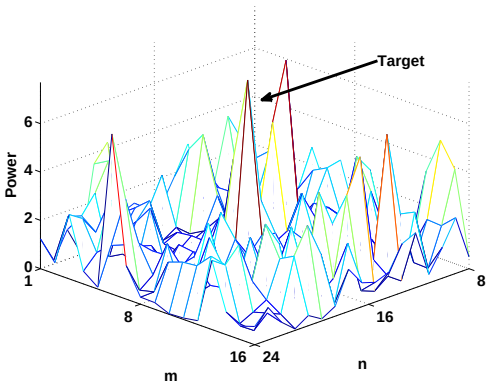


(c) Maximum test statistic.

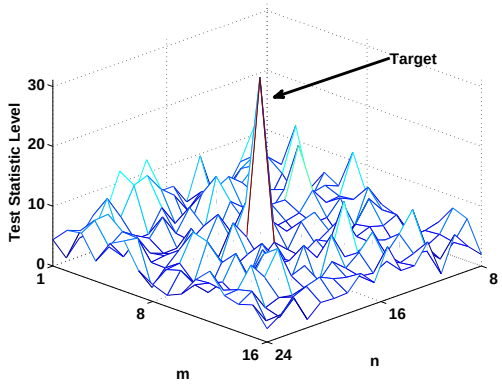
Figure 18: Example of processing.



(d) One ping slice of the data.



(e) Normalized data.



(f) Maximum test statistic.

Figure 18: Example of processing (Zoomed in).

Normalizer	$P_r(m,n,t)$	PFI 10^{-2}	PFI 10^{-3}
3DMN	Figure 6	23.9	26.8
TPSWN	Figure 6	48.4	xxx
3DMN	Figure 11	22.5	xxx
TPSWN	Figure 11	34.5	xxx

Table 1: Thresholds for the normalizers.

each normalizer. If (m_0, n_0) is the location of the true target in ping 1 and if i_0 is the correct target model and the maximum exceeds the thresholds above, then the target was correctly initiated, otherwise it is not correctly initiated.

The procedures described above, with the target located at $x(32, 16, 1)$, $x(32, 18, 2)$, $x(32, 20, 3)$, and $x(32, 22, 4)$ and using the reverberation background in Figure 6, produces the curves in Figure 19. As can be seen the in Figure 19 there is more than a 3 dB improvement in track initiation performance by using the 3DMN over using the TPSWN.

The 3 dB improvement in correct initiation can be used to reduce the false initiation probability by rasing the threshold for the 3DMN. As can be seen in Figure 20 a ten fold reduction in false initiation probability was achieved while still having a better PCI than the TPSWN.

As mentioned in the introduction normalizers are also with faced with the problem of multiple close targets. A second target was added to the data to test the 3DMN verses the TPSWN with more than one target. So, in addition to the target at $x(8, 16, 1)$, $x(8, 18, 2)$, $x(8, 20, 3)$, and $x(8, 22, 4)$, there was also a target with the same SIR at $x(12, 16, 1)$, $x(12, 18, 2)$, $x(12, 20, 3)$, and $x(12, 22, 4)$. With two targets, the location and target model of both targets must be correct to be considered a corrice initialization. As can be seen in Figure 21 both normalizers perform much worse but the 3DMN still has a 3 dB advantage.

Sofar the target has been put in the high reverberation part of the data, so now we will put the target in lower reverberation parts of the data. The data that generated Figure 22 had the target put at $x(32, 16, 1)$, $x(32, 18, 2)$, $x(32, 20, 3)$, and $x(32, 22, 4)$, which is the lower reverberation power area of the data.

The last simulation was done with a different reverberation power background, the one in Figure 11 instead of the one in Figure 6. In this case, to keep the target in the high reverberation area it was put at $x(16, 16, 1)$, $x(16, 18, 2)$, $x(16, 20, 3)$, and $x(16, 22, 4)$. Because the number of ACE used in the 3DMN were picked by trial and error, using the reverberation background in Figure 6, changing the background will give an idea of the robustness of the normalizer.

This performance improvement comes at the cost of increased computational complexity. In the above simulations, the 3DTIN took about 34 times as long to normalize the data as the TPSWN. If computational speed is critical, then fast algorithms such as the ones in [26] or [27], might be modified for this application.

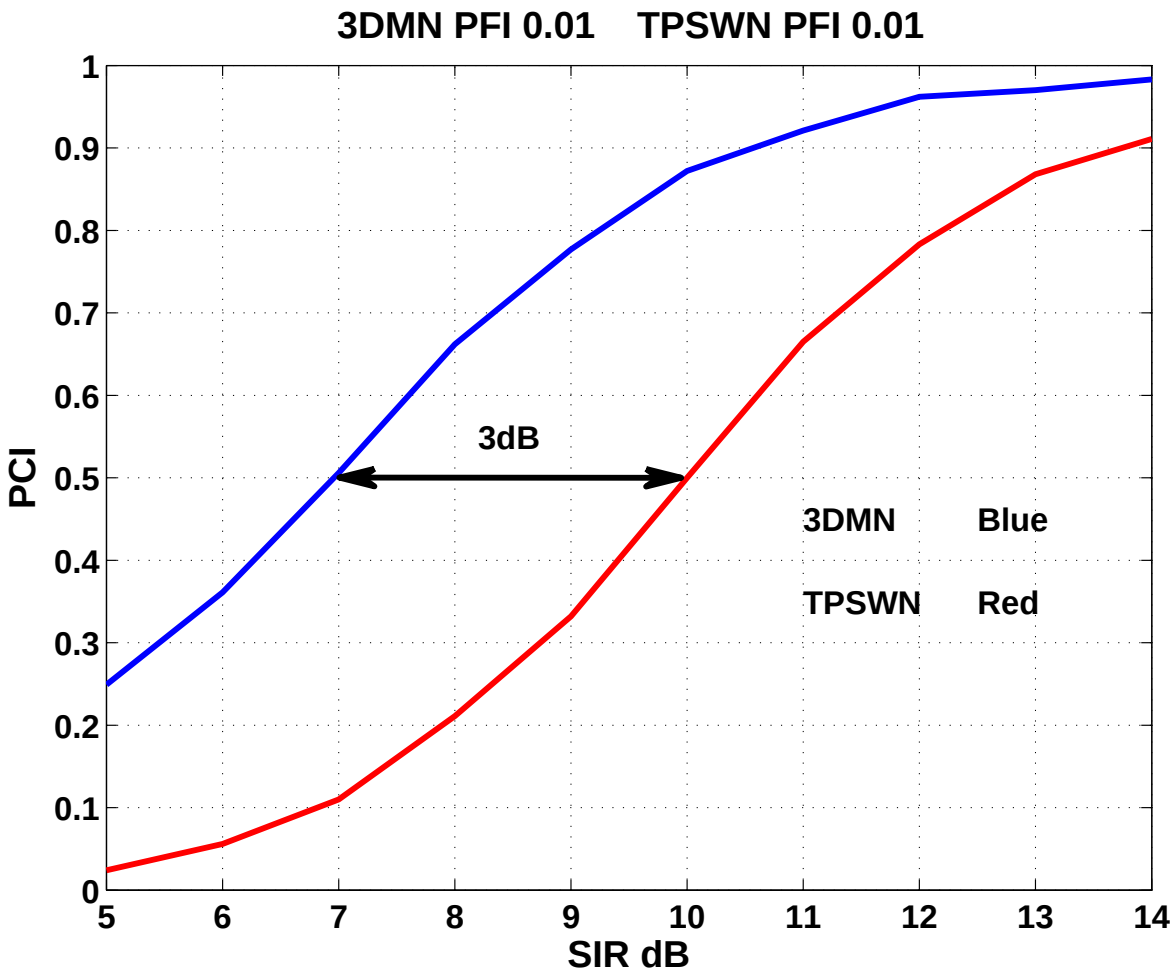


Figure 19: Single target in high reverberation area.

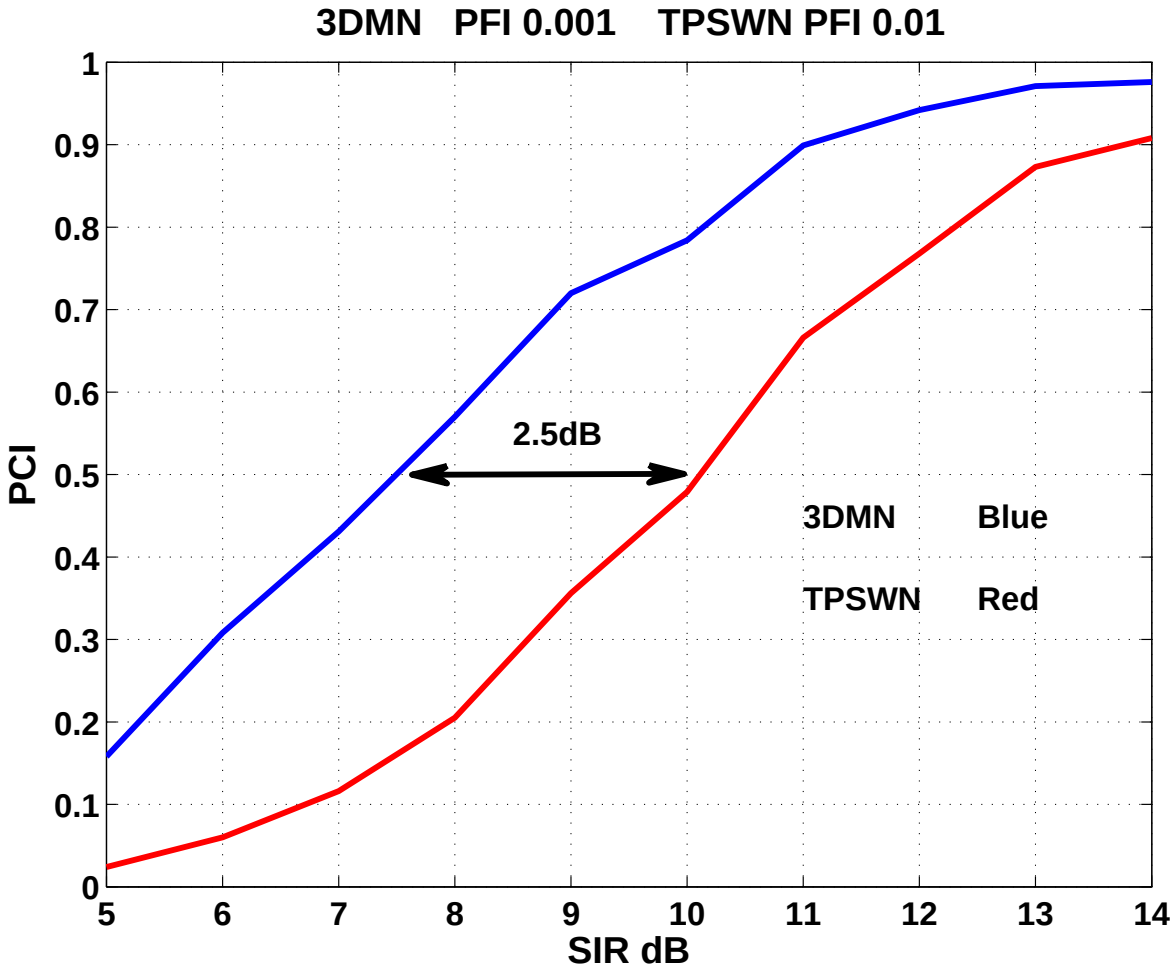


Figure 20: A factor of 10 lower PFI for the 3DMN than the TPSWN with a single target in high reverberation area.

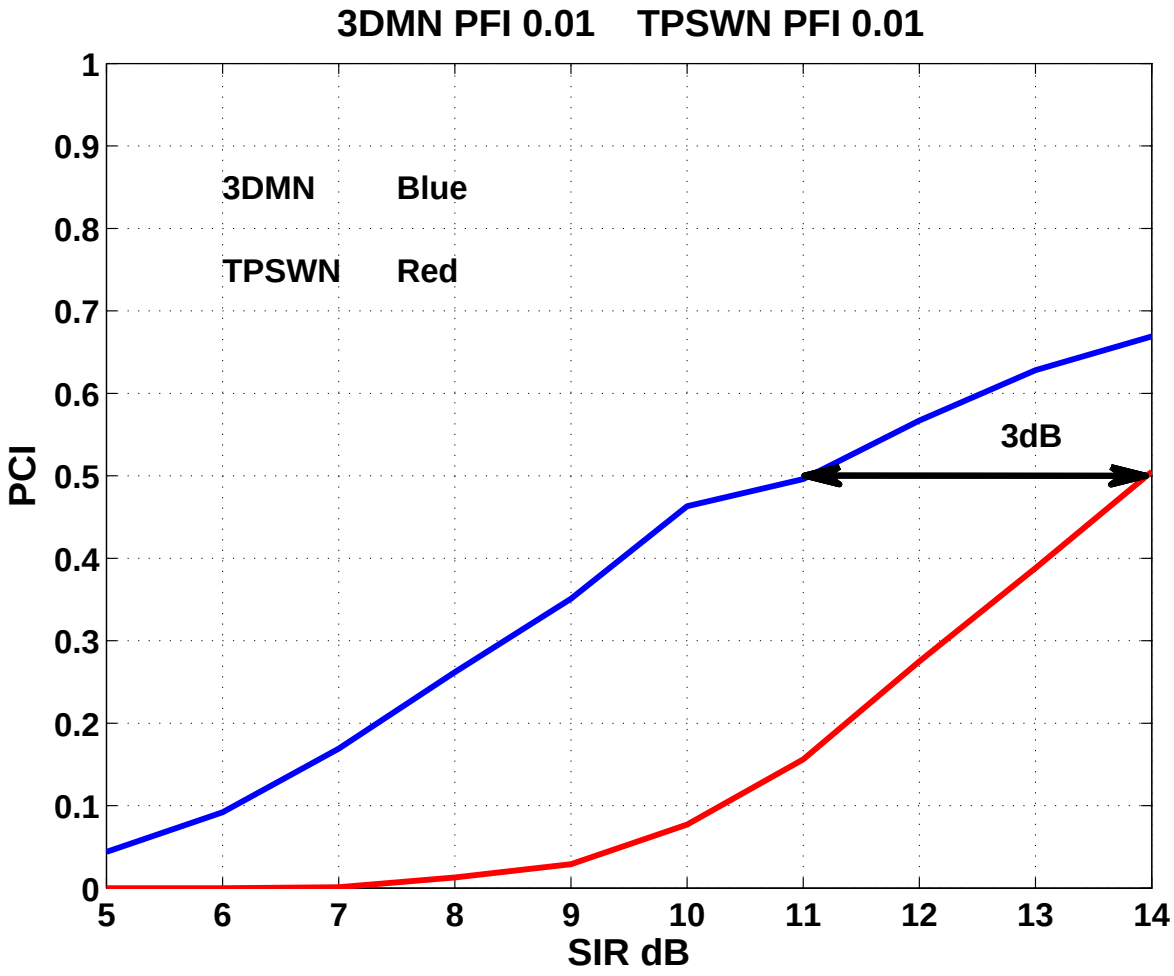


Figure 21: Two close targets in high reverberation area.

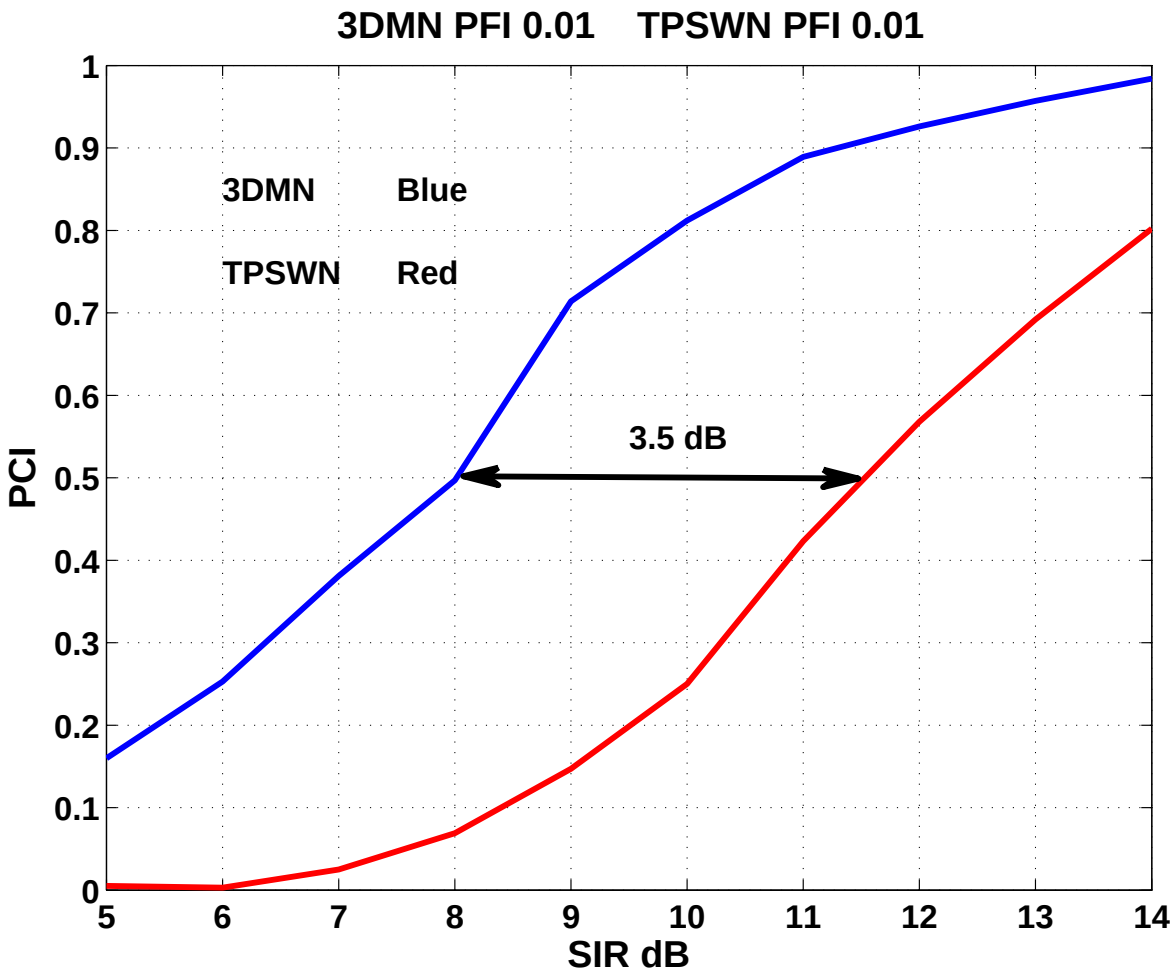


Figure 22: Single target in low reverberation area.

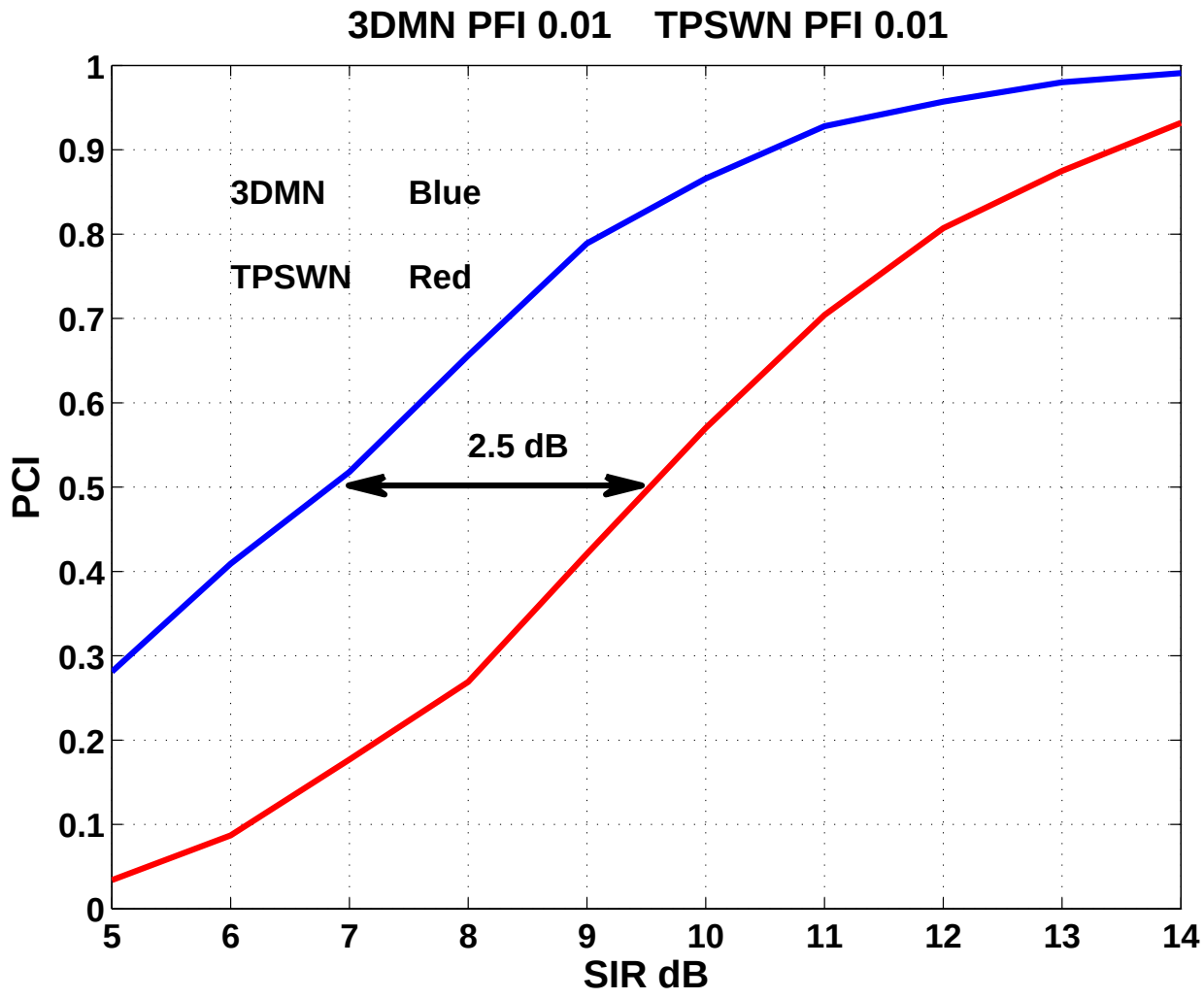


Figure 23: Single target in different reverberation background.

5 Conclusion

A normalization algorithm based on the MVSE has been developed. This normalizer inputs multiple range-bearing plots from an active SONAR detector and outputs a normalized version of those range-bearing plots. This algorithm differs from more common normalizers in that it uses a spectral estimation technique to circumvent the non-stationarity problem, primarily caused by reverberation. Because non-stationarity is no longer a problem, the 3DMN can use more data to normalize each point in the range-bearing-ping space than traditional normalizers. Also, a target initialization algorithm based on the GLRT, has been derived. This target initialization algorithm was used to motivate the need for normalization, and was used to compare the 3DMN to the TPSWN. The 3DMN was then shown to be more effective than the TPSWN in initializing active SONAR tracks. We observed the 3DMN achieve a better PCI verses SIR than the TPSWN, in the following circumstances: high reverberation areas, low reverberation areas, and two close targets. We were able to use this improvement in PCI to achieved a ten fold reduction in false alarms while having the 3DMD have a better PCI than the TPSWN.

References

[1] W. S. Burdic, *Underwater Acoustic System Anlysis.* Los Altos Hills, CA: Peninsula Publishing, 2002.

[2] M. I. Skolnik, *Introduction To RADAR Systems.* New York, NY: McGraw-Hill Book Company, 1962.

[3] R. J. Urick, *Principles of Underwater Sound.* New York, NY: McGraw-Hill Book Company, 1983.

[4] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory.* Upper Saddle River, NJ: Prentice-Hall, 1993.

[5] R. Nitzberg, "Analysis of the arithmetric mean cfar normalizer for fluctuating targets," *IEEE Transactions on Areospace and Electronic Systems*, no. 1, pp. 44–47, January 1978.

[6] Y. K. Bar-Shalom, *Multitarget-Multisensor Tracking.* Danvers, MA: YBS, 1995.

[7] S. S. Blackman, *Multiple-Target Tracking with Radar Application.* Norwood, MA: Artech House, 1986.

[8] E. Brookner, *Tracking and Kalman Filtering Made Easy.* New York: Wiley-Interscience, 1998.

[9] R. O. Nielsen, *Sonar Signal Processing.* Boston, MA: Artech House, 1991.

[10] W. C. Knight, R. G. Pridham, and S. M. Kay, "Digital signal processing for sonar," *Proceedings of the IEEE*, vol. 69, no. 11, pp. 1451–1506, November 1981.

[11] P. P. Gandhi and S. A. Kassam, "Analysis of cfar processors in nonhomogeneous background," *IEEE Transactions on Areospace and Electronic Systems*, vol. 24, no. 4, pp. 427–445, July 1988.

[12] Z. Hu, H. Leung, and M. Blanchette, "Statistical performance analysis of track initiation techniques," *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 445–456, February 1997.

[13] H. Leung, Z. Hu, and M. Blanchette, "Evaluation of multiple target track initiation techniques in real readar treacking enviroments," *IEE Proceedings Radar*, vol. 143, no. 4, pp. 246–254, August 1996.

[14] W. A. Struzinski and E. D. Lowe, "A performance comparison of four noise background normalization schemes proposed for signal detection systems," *Journal Acoustical Society of America*, vol. 76, no. 6, pp. 1738–1742, December 1984.

[15] J. H. Shapiro and T. J. G. JR., "Performance of split-window multipass-mean noise spectral estimators," *IEEE Transactions on Areospace and Electronic Systems*, vol. 36, no. 4, pp. 1360–1370, October 2000.

[16] F. Khan, "Simulation-based performance analysis of several types of normalizers under correlated stationary noise conditions," NUWC Techinical Memo, New London, CT, Tech. Rep., January 1993.

[17] I. S. Reed, R. M. Gagliardi, and H. M. Show, "Application of three-dimensional filtering to moving target detection," *IEEE Transactions on Areospace and Electronic Systems*, no. 6, pp. 898–905, 1983.

[18] I. S. Reed, R. M. Gagliardi, and L. B. Stotts, "Optical moving target detection with 3-D matched filtering," *IEEE Transactions on Areospace and Electronic Systems*, vol. 24, no. 4, pp. 327–336, July 1988.

[19] F. Khan, "Private comunication," January 2007.

[20] A. Papoulis, *Probability, Random Variables and Stochastic Processes.* Boston, MA: McGraw-Hill, 1991.

[21] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proceedings of the IEEE*, vol. 57, no. 8, pp. 1408–1418, August 1969.

[22] S. M. Kay and J. Stanley Lawrence Marple, "Spectrum analysis-a modern perspective," *Proceedings of the IEEE*, vol. 69, no. 11, pp. 1380–1419, November 1981.

[23] C. P. Carbone and S. M. Kay, "Model based classification using multi-ping data," in *Oceans 2006 Conference Proceedings*, Boston, MA, September 2006.

[24] S. M. Kay, *Modern Spectral Estimation.* Englewood Cliffs, NJ: Prentice-Hall, 1988.

[25] D. E. Dudgeon and R. M. Mersereau, *Multidimensional Digital Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

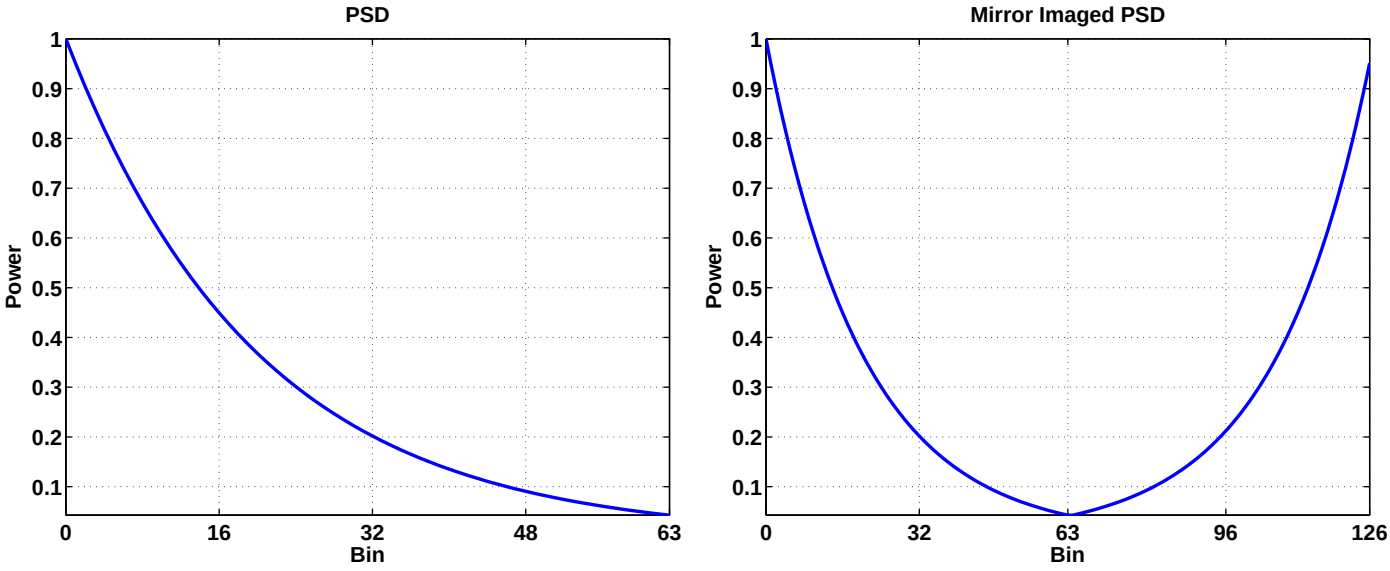
[26] A. Jakobsson, J. S. Lawrence Marple, and P. Stoica, "Computationally efficient two-dimensional capon spectrum analysis," *IEEE Transactions on Signal Processing*, vol. 48, no. 9, pp. 2651–2661, September 2000.

[27] L. Wei, "Fast algorithms and applications for multi-dimensional least-squares-based minimum variance spectral estimation," Ph.D. dissertation, Oregon State University, Corvallis, OR, 2007.

[28] H. P. Hsu, *Signals and Systems*. New York: McGraw-Hill, 1995.

A Appendix

When the estimated PSD not symmetric around zero, the MVSE can produce a severely biased estimate. So, the data are mirror imaged before finding the ACE with the IFT. Figure 24a is an example of a PSD and Figure 24b is the same PSD mirror imaged. Figure



(a) (b)

Figure 24: A PSD and its mirror imaged PSD version.

25 is an example of a one dimensional (1D) PSD and two MVSE estimates of that 1D PSD. One MVSE was performed using a $\widehat{\mathbf{R}_{\mathbf{xx}}}$ made from ACE found with an IFT on a mirror imaged PSD. The other one was estimated without mirror imaging the PSD. Figure 25 shows, the MVSE done with a mirror imaged PSD has less bias. In this paper, all the MVSEs are performed using 3D data that are mirror imaged. Even though it is difficult to visualize how 3D data are mirror imaged, it is easy to check if the mirror imaging was done

correctly. To check if the mirror imaging was done correctly perform an IFT on the mirror imaged data. If the IFT has only real parts, within roundoff error, then the mirror imaging was done correctly. This is because, a PSD is symmetric if and only it has an IFT that is real [28].

B Appendix

The 3D $\widehat{\mathbf{R}_{\mathbf{xx}}}$ is constructed as follows:

1) Select the maximum range line lag p_m, the maximum beam lag p_n, and the maximum ping lag p_t.

2) Find the find the needed ACE using a 3D IFT as

$$\widehat{r}_{xx}(l_m, l_n, l_t) = \frac{1}{MNT} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} x(m,n,t) \exp(j2\pi(ml_m/M + nl_n/N + tl_t/T))$$

for $l_m = 0, \pm 1, \pm 2 \ldots, \pm p_m$, $l_n = 0, \pm 1, \pm 2 \ldots, \pm p_n$, and $l_t = 0, \pm 1, \pm 2 \ldots, \pm p_t$.

3) Arrange the ACE into a autocorrelation matrix that has the same form as an autocorrelation matrix from a first quadrant AR model. For example if $p_m = 1$,$p_n = 1$, and $p_t = 1$ then $\widehat{\mathbf{R}_{\mathbf{xx}}} =$

$$\begin{bmatrix}
\widehat{r}_{xx}(0,0,0) & \widehat{r}_{xx}(0,0,-1) & \widehat{r}_{xx}(0,-1,0) & \widehat{r}_{xx}(0,-1,-1) & \widehat{r}_{xx}(-1,0,0) & \widehat{r}_{xx}(-1,0,-1) & \widehat{r}_{xx}(-1,-1,0) & \widehat{r}_{xx}(-1,-1,-1) \\
\widehat{r}_{xx}(0,0,1) & \widehat{r}_{xx}(0,0,0) & \widehat{r}_{xx}(0,-1,1) & \widehat{r}_{xx}(0,-1,0) & \widehat{r}_{xx}(-1,0,1) & \widehat{r}_{xx}(-1,0,0) & \widehat{r}_{xx}(-1,-1,1) & \widehat{r}_{xx}(-1,-1,0) \\
\widehat{r}_{xx}(0,1,0) & \widehat{r}_{xx}(0,1,-1) & \widehat{r}_{xx}(0,0,0) & \widehat{r}_{xx}(0,0,-1) & \widehat{r}_{xx}(-1,1,0) & \widehat{r}_{xx}(-1,1,-1) & \widehat{r}_{xx}(-1,0,0) & \widehat{r}_{xx}(-1,0,-1) \\
\widehat{r}_{xx}(0,1,1) & \widehat{r}_{xx}(0,1,0) & \widehat{r}_{xx}(0,0,1) & \widehat{r}_{xx}(0,0,0) & \widehat{r}_{xx}(-1,1,1) & \widehat{r}_{xx}(-1,1,0) & \widehat{r}_{xx}(-1,0,1) & \widehat{r}_{xx}(-1,0,0) \\
\widehat{r}_{xx}(1,0,0) & \widehat{r}_{xx}(1,0,-1) & \widehat{r}_{xx}(1,-1,0) & \widehat{r}_{xx}(1,-1,-1) & \widehat{r}_{xx}(0,0,0) & \widehat{r}_{xx}(0,0,-1) & \widehat{r}_{xx}(0,-1,0) & \widehat{r}_{xx}(0,-1,-1) \\
\widehat{r}_{xx}(1,0,1) & \widehat{r}_{xx}(1,0,0) & \widehat{r}_{xx}(1,-1,1) & \widehat{r}_{xx}(1,-1,0) & \widehat{r}_{xx}(0,0,1) & \widehat{r}_{xx}(0,0,0) & \widehat{r}_{xx}(0,-1,1) & \widehat{r}_{xx}(0,-1,0) \\
\widehat{r}_{xx}(1,1,0) & \widehat{r}_{xx}(1,1,-1) & \widehat{r}_{xx}(1,0,0) & \widehat{r}_{xx}(1,0,-1) & \widehat{r}_{xx}(0,1,0) & \widehat{r}_{xx}(0,1,-1) & \widehat{r}_{xx}(0,0,0) & \widehat{r}_{xx}(0,0,-1) \\
\widehat{r}_{xx}(1,1,1) & \widehat{r}_{xx}(1,1,0) & \widehat{r}_{xx}(1,0,1) & \widehat{r}_{xx}(1,0,0) & \widehat{r}_{xx}(0,1,1) & \widehat{r}_{xx}(0,1,0) & \widehat{r}_{xx}(0,0,1) & \widehat{r}_{xx}(0,0,0)
\end{bmatrix}. \quad (21)$$

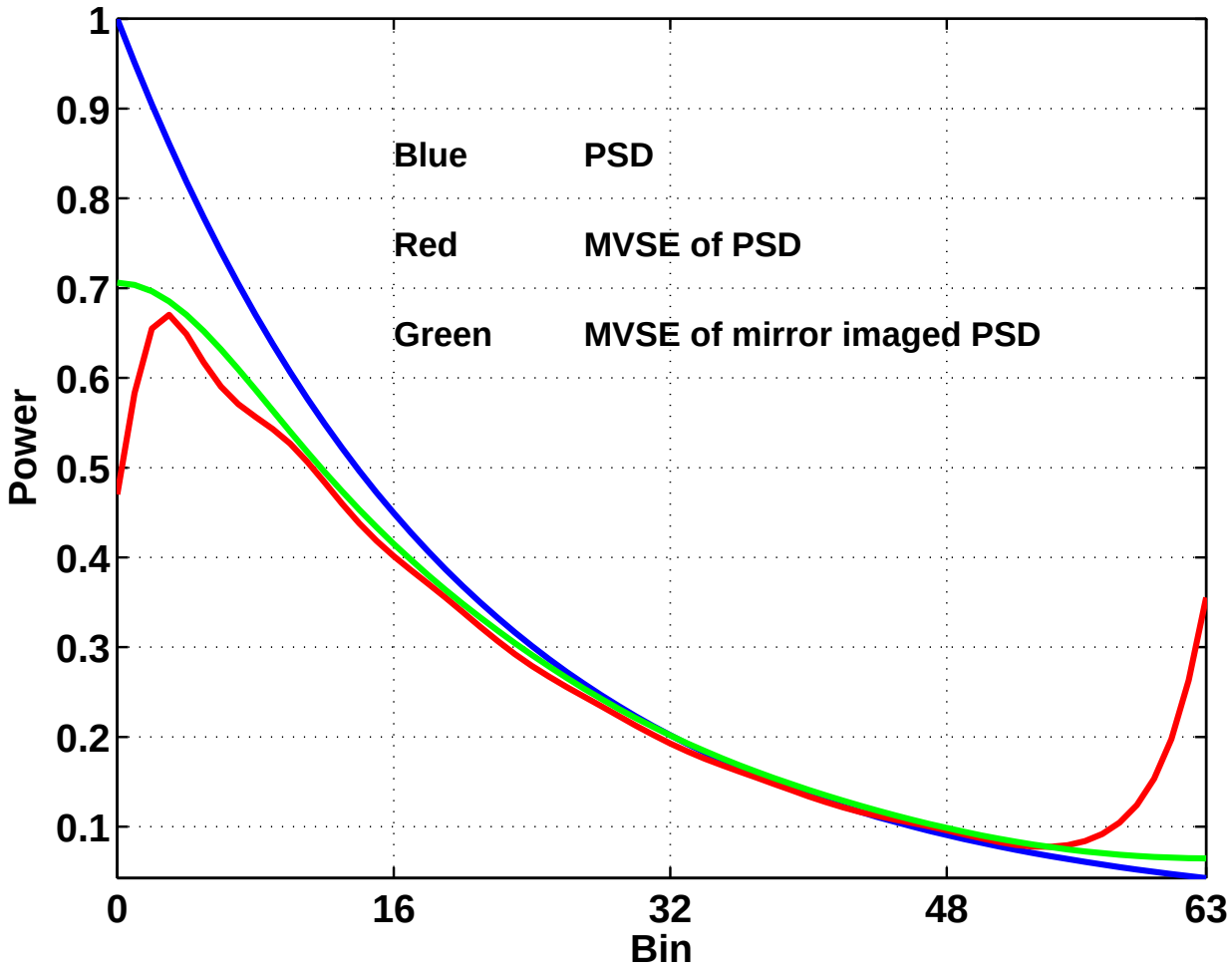


Figure 25: MVSE done with and without mirror imaged PSD