AN ANALYSIS OF MINIMUM ENTROPY TIME-FREQUENCY DISTRIBUTIONS

by

Paul Bradley

BS, Grove City College, 2004

Submitted to the Graduate Faculty of

the Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Master of Science in Electrical Engineering

University of Pittsburgh

2008

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This thesis was presented

by

Paul Bradley

It was defended on

April 2, 2008

and approved by

Luis F. Chaparro, Associate Professor, Electrical & Computer Engineering Department

Patrick J. Loughlin, Associate Professor, Electrical & Computer Engineering Department

Thesis Advisor: Amro El-Jaroudi, Associate Professor, Electrical & Computer Engineering Department

AN ANALYSIS OF MINIMUM ENTROPY TIME-FREQUENCY DISTRIBUTIONS

Paul Bradley, M.S.

University of Pittsburgh, 2008

The subject area of time-frequency analysis is concerned with creating meaningful representations of signals in the time-frequency domain that exhibit certain properties. Different applications require different characteristics in the representation. Some of the properties that are often desired include satisfying the time and frequency marginals, positivity, high localization, and strong finite support. Proper time-frequency distributions, which are defined as distributions that are manifestly positive and satisfy both the time and frequency marginals, are of particular interest since they can be viewed as a joint time-frequency density function and ensure strong finite support. Since an infinite number of proper time-frequency distributions exist, it is often necessary to impose additional constraints on the distribution in order to create a meaningful representation of the signal. A significant amount of research has been spent attempting to find constraints that produce meaningful representations.

Recently, the idea was proposed of using the concept of minimum entropy to create time-frequency distributions that are highly localized and contain a large number of zero-points. The proposed method starts with an initial distribution that is proper and iteratively reduces the total entropy of the distribution while maintaining the positivity and marginal properties. The result of this method is a highly localized, proper TFD.

This thesis will further explore and analyze the proposed minimum entropy algorithm. First, the minimum entropy algorithm and the concepts behind the algorithm will be introduced

and discussed. After the introduction, a simple example of the method will be examined to help gain a basic understanding of the algorithm. Next, we will explore different rectangle selection methods which define the order in which the entropy of the distribution is minimized. We will then evaluate the effect of using different initial distributions with the minimum entropy algorithm. Afterwards, the results of the different rectangle selection methods and initial distributions will be analyzed and some more advanced concepts will be explored. Finally, we will draw conclusions and consider the overall effectiveness of the algorithm.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 INTRODUCTION

The subject area of time-frequency analysis is concerned with creating meaningful representations of signals in the time-frequency domain that exhibit certain properties. Different applications require different characteristics in the representation. Some of the properties that are often desired include satisfying the time and frequency marginals, positivity, high localization, and finite support. Currently, many of the existing distributions satisfy most of the properties, but not all of them satisfactorily. For example, the spectrogram distribution is manifestly positive, but does not satisfy the marginals.

Proper time-frequency distributions (TFDs), which are defined as distributions that are manifestly positive and satisfy the time and frequency marginals, are of particular interest for several reasons. First, the TFD can be viewed as a joint time-frequency density function [1] and [2]. As a result, the mathematical tools that are applicable to joint density functions can be applied to these distributions with reasonable, interpretable results. Using these techniques with distributions that have negative values, such as the Wigner distribution, or do not satisfy the marginals, such as the spectrogram, often produce results that are difficult to interpret. These properties guarantee that the TFD exhibits the correct spectral, temporal, and total signal energies [1]. Secondly, non-negativity and satisfying the marginals results in strong finite support. Since the distribution cannot go negative, the distribution must be zero at any location where the time or frequency marginal is zero [1].

It was shown by Wigner that bilinear distributions (i.e. signal independent distributions) cannot be both manifestly positive and satisfy the marginals for most signals. For example, the Wigner distribution can only have these properties for a Gaussian chirp [4].

Cohen-Posch showed that an infinite number of distributions exist that are both positive and satisfy the marginals, but they are not bilinear [4]. In other words, proper TFDs must be signal dependent for the general case. The Cohen-Posch distribution is defined as follows:

$$P(t,f;\Omega) = |s(t)|^2 |S(f)|^2 \Omega(u(t), v(f); s(t))$$

$$\int_0^1 \Omega(u,v)dv = 1 \text{ and } \int_0^1 \Omega(u,v)du = 1$$

$$u = u(t) = \int_{-\infty}^{t} |s(t')|^2 dt' \text{ and } v = v(f) = \int_{-\infty}^{f} |S(f')|^2 df' \quad [1]$$

An example of a kernel that would satisfy these conditions is as follows:

$$\Omega(u,v) = 1 - (nu^{n-1} - 1)(mv^{m-1} - 1)$$

Since there are an infinite number of kernels for the Cohen-Posch distribution, it is important to find the $\Omega(u,v)$ that produces the most meaningful results [6]. Thus, it is often necessary to apply additional constraints.

Several methods have been proposed to create consistently meaningful Cohen-Posch distributions. Loughlin, Pitton, and Atlas proposed a method to create a proper TFD by selecting an initial positive TFD that does not satisfy the marginals and using cross-entropy minimization with the marginals (and possibly higher order moments) as constraints. This method has been shown to provide meaningful, proper TFDs for a wide variety of signals [1] and [2]. Throughout this paper, this distribution will be referred to as the Minimum Cross-Entropy Positive Time-

Frequency Distribution (MCE-PTFD). Several variations on this method have been proposed to achieve different representations that are proper [2], [5], and [6].

Alternatively, El-Jaroudi proposed the idea of using the concept of minimum entropy to create time-frequency distributions that are highly localized and contain a large number of zero-points. The proposed method starts with an initial distribution that is manifestly positive and satisfies the marginals (i.e. proper) and iteratively reduces the total entropy of the distribution while maintaining the positivity and marginal properties. The result of this method is a highly localized, proper TFD [3]. This thesis will further explore and analyze this method.

2.0 MINIMUM ENTROPY TFDS: BACKGROUND

The idea of maximum entropy has been used extensively in creating power spectral densities and, to a slightly lesser extent, in time-frequency analysis. It is often argued that using maximum entropy in these areas results in the flattest possible spectrum or the TFD that uses the fewest assumptions about the original signal. Since the goal in time-frequency analysis is to create a highly localized representation of the signal in the time-frequency plane, the concept of maximum entropy contradicts the desired results for a TFD. Instead, the idea of minimum entropy seems to be the better fit for time-frequency analysis. The minimum entropy algorithm proposed by El-Jaroudi attempts to create a highly localized TFD by minimizing the entropy of the distribution using an iterative algorithm [3].

The proposed algorithm starts with an initial TFD that is manifestly positive and satisfies the time and frequency marginals (i.e. a proper TFD). An example of the initial TFD would be $P(n,\omega) = |x(n)|^2 \, |X(\omega)|^2$ which happens to be the maximum entropy distribution of the signal. Any proper TFD can be utilized as the starting point [3]. If satisfying the marginals is not a mandatory constraint, the algorithm can start with a distribution that does not satisfy the marginals such as the spectrogram. The final distribution will have the same marginal properties as the initial distribution. Regardless of the marginal properties, the initial distribution must be manifestly positive.

The algorithm takes four points that create a rectangle and attempts to modify them so that the total entropy of the TFD is reduced while continuing to satisfy the marginals. First, the four points of the selected rectangle are defined as $P_{11} = P(n_1,\omega_1)$, $P_{12} = P(n_1,\omega_2)$, $P_{21} = P(n_2,\omega_1)$, and $P_{22} = P(n_2,\omega_2)$. A value, which will be referred as Δ, is added to P_{11} and P_{22} and subtracted from P_{12} and P_{21}. This process modifies the TFD while continuing to satisfy the marginals [3].

The value Δ is calculated as follows. First, entropy is defined using the Shannon entropy:

$$E = -\sum_n \int_\omega P(n,\omega) \ln P(n,\omega)$$

The loss function is then defined as:

$$\begin{aligned} \text{Loss} &= E_{before} - E_{after} \\ &= -P_{11}\ln P_{11} - P_{12}\ln P_{12} - P_{21}\ln P_{21} - P_{22}\ln P_{22} \\ &\qquad + (P_{11} - \Delta)\ln(P_{11} - \Delta) + (P_{12} - \Delta)\ln(P_{12} - \Delta) \\ &\qquad + (P_{21} - \Delta)\ln(P_{21} - \Delta) + (P_{22} - \Delta)\ln(P_{22} - \Delta) \end{aligned}$$

It can be shown using Lagrange multipliers that the value of Δ that provides the greatest loss in entropy is either $\min\{P_{11}, P_{22}\}$ or $-\min\{P_{12}, P_{21}\}$. Let $\Delta_1 = \min\{P_{11}, P_{22}\}$ and $\Delta_2 = -\min\{P_{12}, P_{21}\}$. The loss function is calculated for both Δ_1 and Δ_2 and the value of Δ_i that provides the greatest loss is used as Δ [3].

The algorithm proceeds as follows:

- Select an initial TFD $P(n,\omega)$ that is manifestly positive and has the desired marginal properties.
- Choose four points in the TFD matrix that create a rectangle: $P(n_1,\omega_1)$, $P(n_1,\omega_2)$, $P(n_2,\omega_1)$, and $P(n_2,\omega_2)$.
- Find the value of Δ that provides the greatest loss in entropy.

- Update the four points using Δ: $P(n_1,\omega_1) = P(n_1,\omega_1) - \Delta$, $P(n_1,\omega_2) = P(n_1,\omega_2) + \Delta$, $P(n_2,\omega_1) = P(n_2,\omega_1) + \Delta$, and $P(n_2,\omega_2) = P(n_2,\omega_2) - \Delta$.
- If the entropy cannot be reduced for any rectangle in the TFD, stop. Otherwise, continue with step 2 [3].

It is evident that the resulting distribution will be highly localized and will exhibit a large number of zero-points. In each iteration, at least one point will be set to zero. It is also possible for one or more zero-points to be set to non-zero values. Thus, the number of total zero-points is not guaranteed to increase in each iteration, but will generally increase as the algorithm progresses.

The algorithm must maintain the time and frequency marginals for each value of time and frequency in the initial distribution. For a distribution that has n time values and w frequency values, the minimum entropy algorithm has $n+w$ marginal constraints that must be satisfied with $n*w$ degrees of freedom. For example, for a distribution with 32 time values and 32 frequency values, the algorithm seeks to find the minimum entropy solution while satisfying the 64 total time and frequency marginal constraints by manipulating the 1024 points in the initial distribution.

The two most significant variables in the minimum entropy algorithm are the initial TFD that is chosen and the rectangle selection method. An infinite number of initial TFDs, or priors, exist for a given signal. To achieve a proper TFD, the only two requirements of the prior are that it must be positive at every point and it must satisfy the time and frequency marginals. Different priors will yield different minimum entropy TFDs. In addition, an infinite number of rectangle selection methods exist with each one yielding a different final distribution. Thus, the order in which the rectangles are selected has a significant effect on the outcome of the algorithm. Both of these areas will be explored in this thesis.

2.1 EXAMPLE

To demonstrate the minimum entropy algorithm, we will first consider a simple chirp signal. The equation for the simple chirp signal is as follows:

$$x(n) = c*e^{j\pi n\text{^}2/(2N)}$$

where c is the normalization constant and N is the length of the signal which will be 32 for this example. Figure 2.1 shows a plot of the real part of the signal. Figure 2.2 shows the spectrogram of the signal.

The minimum entropy algorithm was performed on this signal using $P(n,\omega) = |x(n)|^2 |X(\omega)|^2$ as the initial TFD and selecting the rectangle that results in the greatest loss in entropy for each iteration. The initial TFD had a total entropy of 6.3725 and had 0 zero-points. The final minimum entropy TFD exhibited a total entropy of 3.7022 and contained 961 zero-points. Figure 2.3 shows a plot of the initial TFD, $P(n,\omega) = |x(n)|^2 |X(\omega)|^2$. Figure 2.4 shows a plot of the final minimum entropy TFD. Figure 2.5 and Figure 2.6 show the time and frequency marginals for the signal x(n), respectively. Figure 2.7 and Figure 2.8 show the time and frequency marginals for the final minimum entropy TFD, respectively.

These results indicate that the minimum entropy algorithm yields a valid time-frequency distribution that is manifestly positive, satisfies the marginals, and is highly localized with an increased number of zero-points. Although the algorithm creates a highly localized TFD for this example, the final plot does not appear to be a meaningful representation of the chirp signal. Some modifications to the algorithm such as selecting a different initial TFD or using a different rectangle selection method may provide better, more meaningful results.

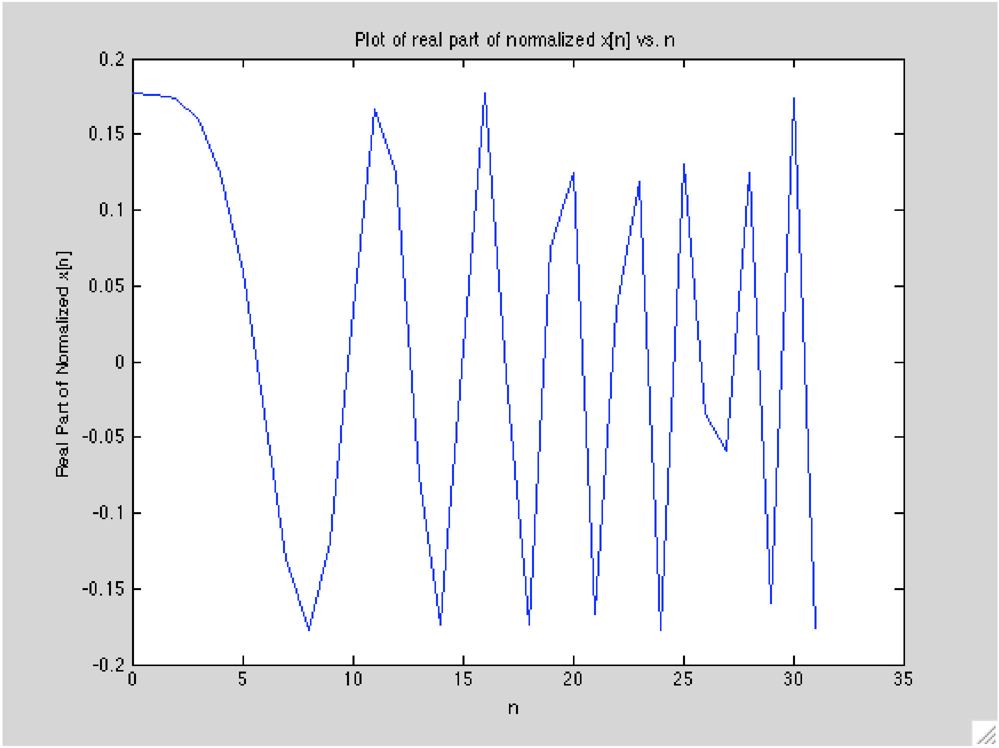

Figure 2.1: Plot of the real part of the simple chirp signal

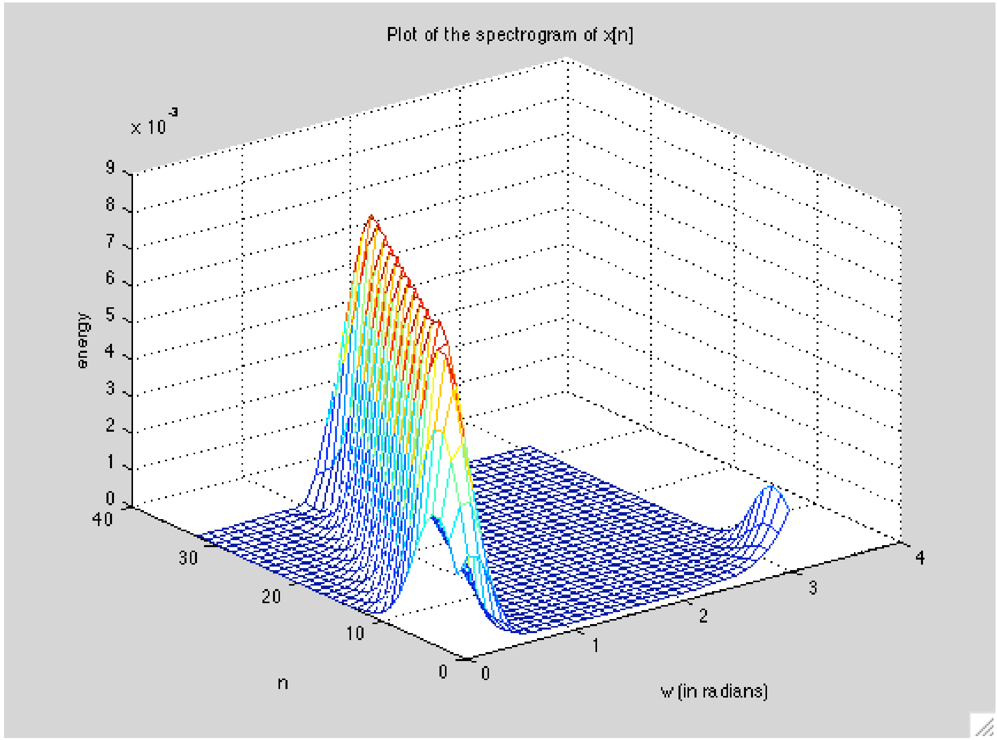

Figure 2.2: Plot of the spectrogram of the simple chirp signal

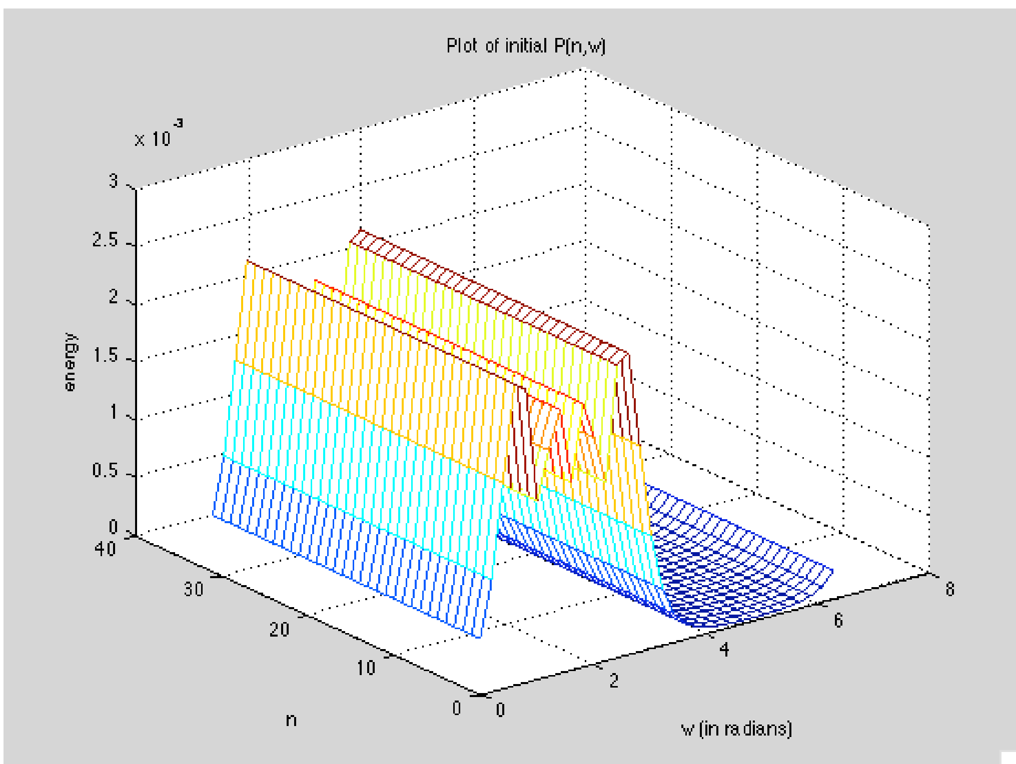

Figure 2.3: Plot of the initial TFD, $P(n,\omega) = |x(n)|^2\ |X(\omega)|^2$, for the simple chirp signal

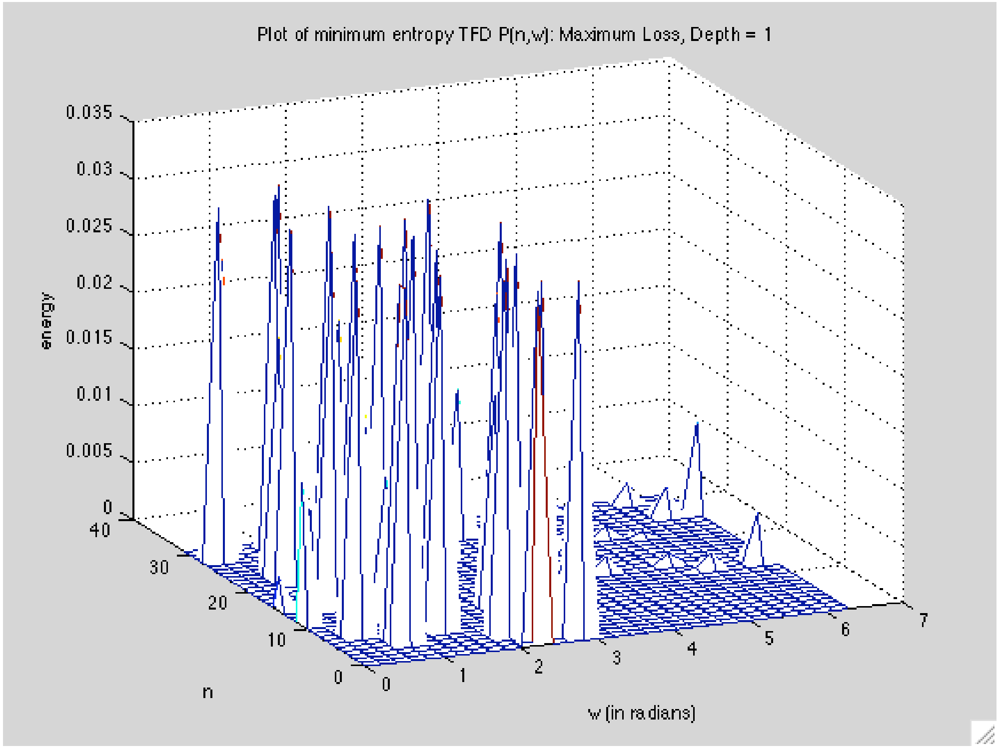

Figure 2.4: Plot of the final minimum entropy TFD for the simple chirp signal

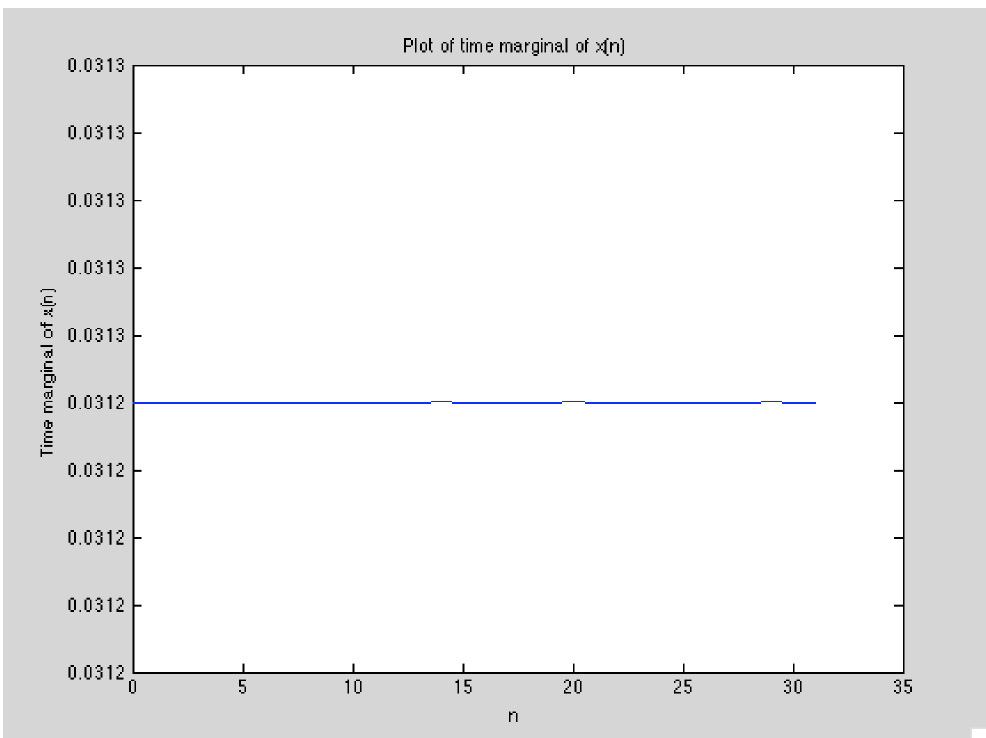

Figure 2.5: Plot of the time marginal for the simple chirp signal

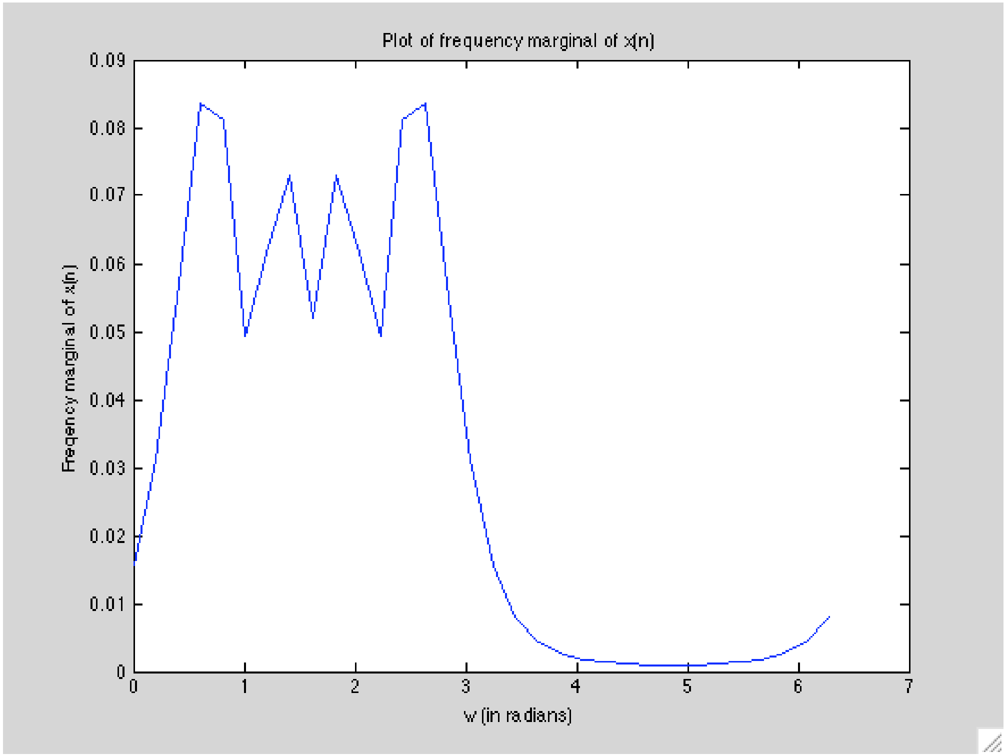

Figure 2.6: Plot of the frequency marginal for the simple chirp signal

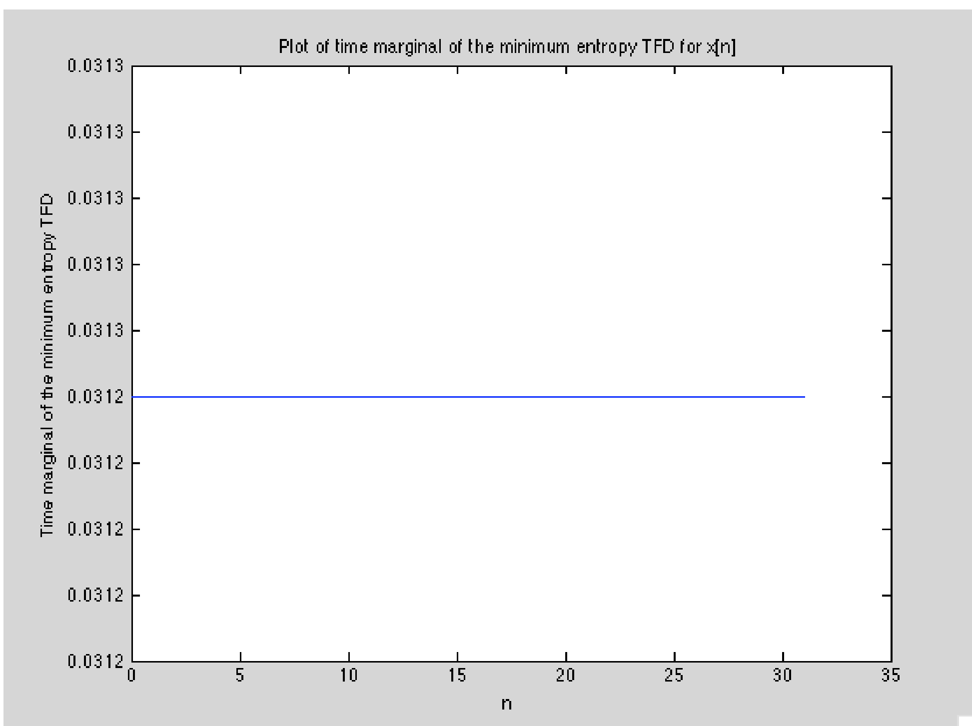

Figure 2.7: Plot of the time marginal for the minimum entropy TFD of the simple chirp signal

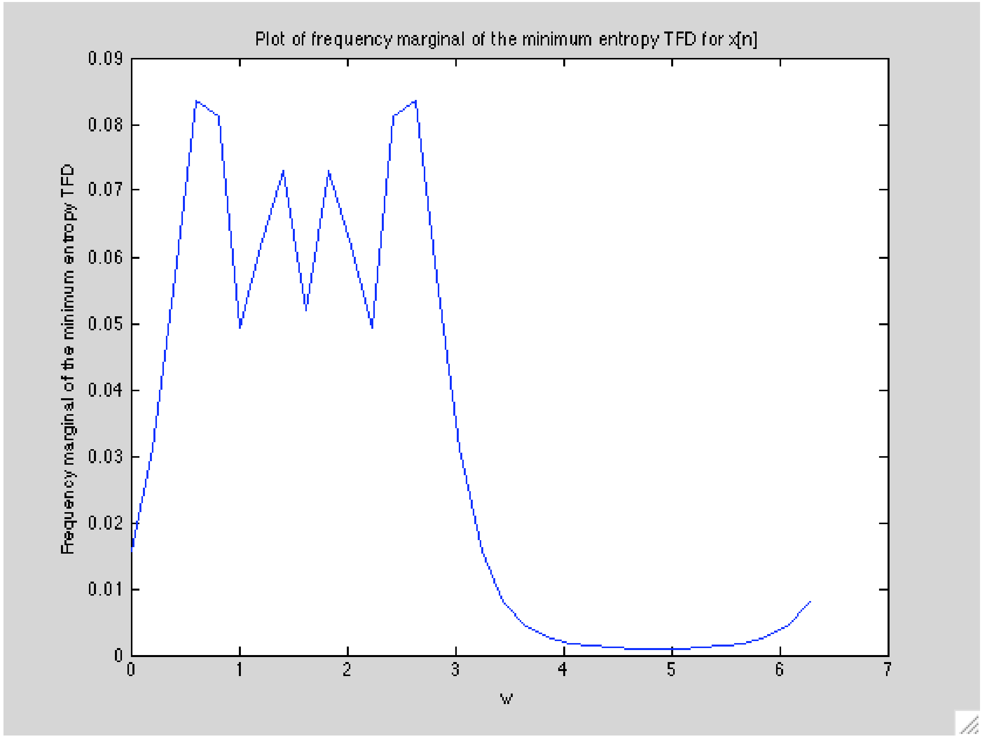

Figure 2.8: Plot of the frequency marginal for the minimum entropy TFD of the simple chirp signal

3.0 MINIMUM ENTROPY TFDS: RECTANGLE SELECTION METHODS

With the El-Jaroudi minimum entropy algorithm, there are an infinite number of possible minimum entropy TFDs depending on how the minimization rectangles are selected. For example, randomly selecting rectangles will produce a different result than selecting the rectangle that produces the greatest loss in entropy. This characteristic brings up two questions. First, which rectangle selection method produces the "best" (i.e. most meaningful) TFD? Secondly, is it possible/feasible to find the absolute minimum entropy TFD and does this TFD give a meaningful result?

To explore these questions, we will start with an initial TFD, $P(n,\omega)$, that is manifestly positive and satisfies the marginals. For example, the initial TFD could be the maximum entropy TFD of the signal, $P(n,\omega) = |x(n)|^2\ |X(\omega)|^2$. The resulting $P(n,\omega)$ matrix will be defined as follows:

	ω_1	ω_2	…	ω_M
n_1	$P(n_1,\omega_1)$	$P(n_1,\omega_2)$	…	$P(n_1,\omega_M)$
n_2	$P(n_2,\omega_1)$	$P(n_2,\omega_2)$	…	$P(n_2,\omega_M)$
…	…	…	…	…
n_N	$P(n_N,\omega_1)$	$P(n_N,\omega_2)$	…	$P(n_N,\omega_M)$

With this definition, $P(n_1,\omega_1)$ is considered to be the top-left corner, $P(n_1,\omega_M)$ is the top-right corner, $P(n_N,\omega_1)$ is the bottom-left corner, and $P(n_N,\omega_M)$ is the bottom-right corner.

There are two different types of rectangle selection methods that can be utilized to determine the minimum entropy TFD: the brute-force selection method and the intelligent selection method. In the brute-force selection method, the minimization rectangles are chosen using a pattern without regard to the values in the rectangles. For example, the algorithm could start at one corner of the TFD matrix and work its way to the opposite corner, taking every possible rectangle. Once the opposite corner is reached, it will begin the process again at the initial corner. This continues until none of the rectangles in the matrix can be reduced. Note that the algorithm does not necessarily have to start at a corner. It simply has to loop through all of the possible rectangles until none of them can be reduced any further.

This section will explore four different brute-force selection methods: top-left corner, top-right corner, bottom-left corner, and bottom-right corner. In the top-left corner method, the algorithm will begin with the top-left most rectangle and work its way to the bottom-right most rectangle. It will work its way to the right before going down. For example, for a 3x3 matrix, the rectangle selection order will be as follows:

	1	2	3
1	P(1,1)	P(1,2)	P(1,3)
2	P(2,1)	P(2,2)	P(2,3)
3	P(3,1)	P(3,2)	P(3,3)

1. [P(1,1), P(2,2)]
2. [P(1,1), P(2,3)]
3. [P(1,1), P(3,2)]
4. [P(1,1), P(3,3)]
5. [P(1,2), P(2,3)]
6. [P(1,2), P(3,3)]
7. [P(2,1), P(3,2)]
8. [P(2,1), P(3,2)]
9. [P(2,2), P(3,3)]

In the top-right method, the algorithm will start with the top-right most rectangle and work its way to the bottom-left most rectangle. It will work its way left before going down. In the bottom-left method, the algorithm will start with the bottom-left most rectangle and work its way to the top-right most rectangle. It will work its way right before going up. In the bottom-right method, the algorithm will start with the bottom-right most rectangle and work its way to the top-left most rectangle. It will work its way left before going up.

In the intelligent selection method, the algorithm searches through all of the possible rectangles and selects the rectangle that matches a given criteria. Some examples of this method include maximum entropy loss, n^{th} maximum entropy loss, and minimum entropy loss. In the maximum entropy loss method, the algorithm searches through all of the possible rectangles and selects the rectangle that gives the greatest loss in entropy. The n^{th} maximum entropy loss method searches for the rectangle that provides the n^{th} greatest loss in entropy. The minimum entropy loss method searches for the rectangle that provides the smallest non-zero loss in entropy. Unfortunately, this method takes a very long time to complete and is not very useful. This paper will explore the maximum entropy loss method and the n^{th} maximum entropy loss method for several different values of n.

3.1 SIMPLE CHIRP EXAMPLE

To evaluate each rectangle selection method, we will first look at the same simple chirp signal that was used in the initial example:

$$x(n) = c*e^{j\pi n^2/(2N)}$$

where c is the normalization constant and N is the length of the signal which is 32 in this example. As before, the initial TFD will be $P(n,\omega) = |x(n)|^2 \, |X(\omega)|^2$ where $X(\omega)$ is the 32 point DFT of x(n). The initial TFD has a total entropy of 6.3725 and 0 zero-points. Figures 2.1 through 2.3 show plots of the signal, the spectrogram of the signal, and the initial TFD, respectively.

3.1.1 Brute Force Methods

First, we will consider the four brute force rectangle selection methods: top-left method, top-right method, bottom-left method, and bottom-right method. The minimum entropy algorithm was performed on the simple chirp signal for each of the four brute force rectangle selection methods. Table 3.1 summarizes the total entropy and number of zero-points for the final minimum entropy TFD for each brute force selection method for the simple chirp signal.

In the top-left rectangle selection method, the algorithm starts with the top-left most rectangle and progresses to the bottom-right rectangle, moving to the right before going down. The final TFD exhibits a total entropy of 3.6885 and contains 961 zero-points. Figure 3.1 shows the minimum entropy TFD obtained using the top-left method for the simple chirp signal.

In the top-right rectangle selection method, the algorithm starts with the top-right most rectangle and progresses to the bottom-left rectangle moving to the left before going down. The final TFD exhibits a total entropy of 3.6983 and contains 961 zero-points. Figure 3.2 shows the minimum entropy TFD obtained using the top-right method for the simple chirp signal.

In the bottom-left rectangle selection method, the algorithm starts with the bottom-left most rectangle and progresses to the top-right rectangle moving to the right before going up. The final TFD exhibits a total entropy of 3.7058 and contains 961 zero-points. Figure 3.3 shows the minimum entropy TFD obtained using the bottom-left method for the simple chirp signal.

In the bottom-right rectangle selection method, the algorithm starts with the bottom-right most rectangle and progresses to the top-left rectangle, moving to the left before going up. The final TFD exhibits a total entropy of 3.7026 and contains 961 zero-points. Figure 3.4 shows the minimum entropy TFD obtained using the bottom-right method for the simple chirp signal.

Table 3.1: Total entropy and number of zero-points for the minimum entropy TFDs of the simple chirp signal obtained using the different brute force rectangle selection methods.

Method	Total Entropy	Zero-Points
Top-Left	3.6885	961
Top-Right	3.6983	961
Bottom-Left	3.7058	961
Bottom-Right	3.7026	961

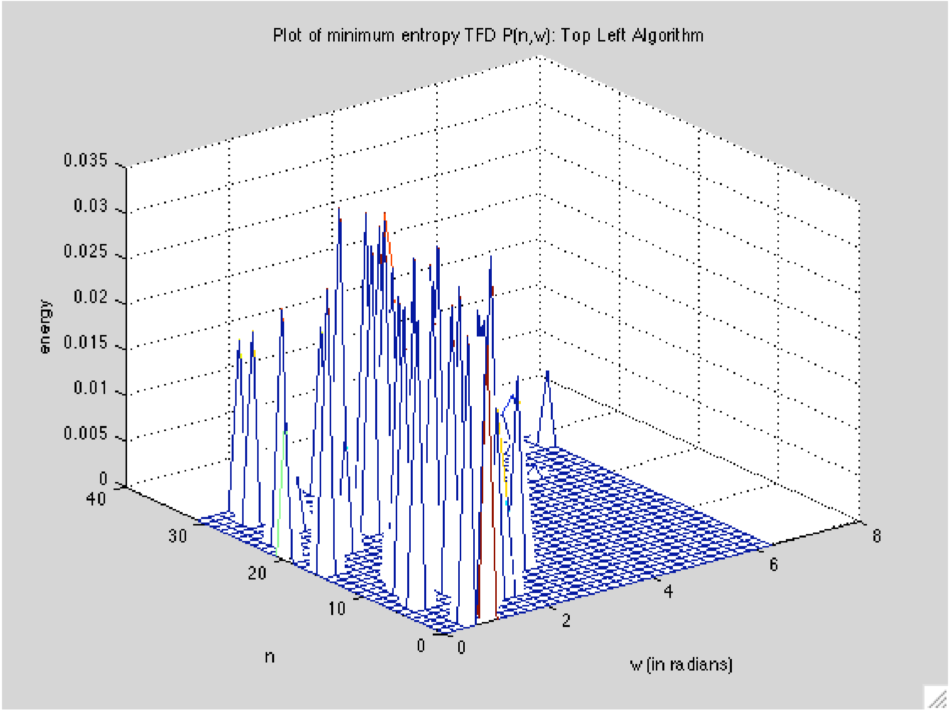

Figure 3.1: Plot of minimum entropy TFD obtained using the top-left method for the simple chirp signal

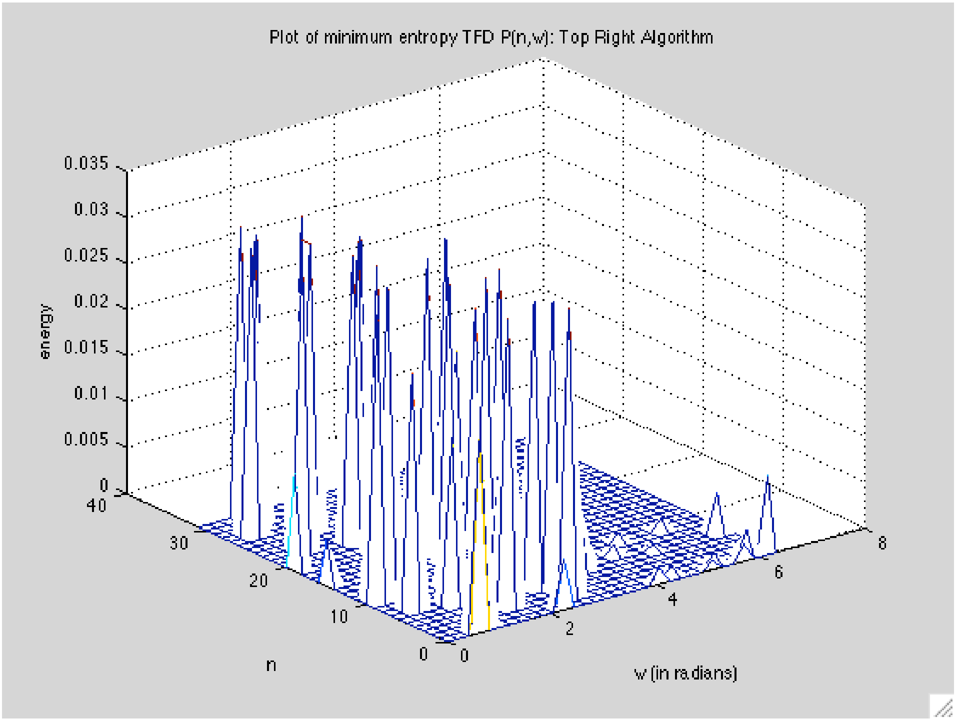

Figure 3.2: Plot of minimum entropy TFD obtained using the top-right method for the simple chirp signal

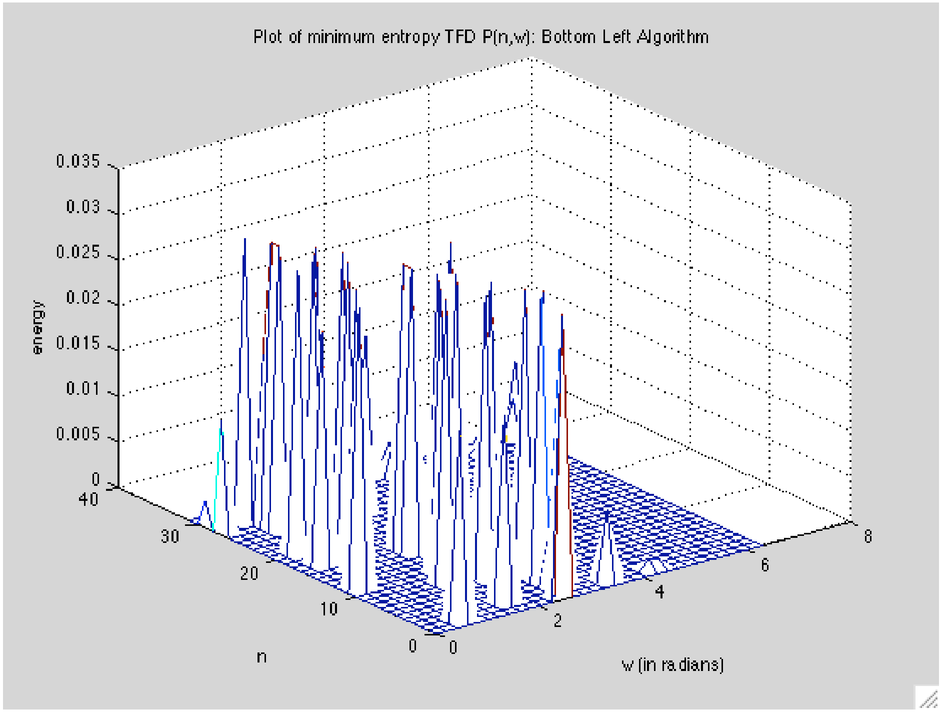

Figure 3.3: Plot of minimum entropy TFD obtained using the bottom-left method for the simple chirp signal

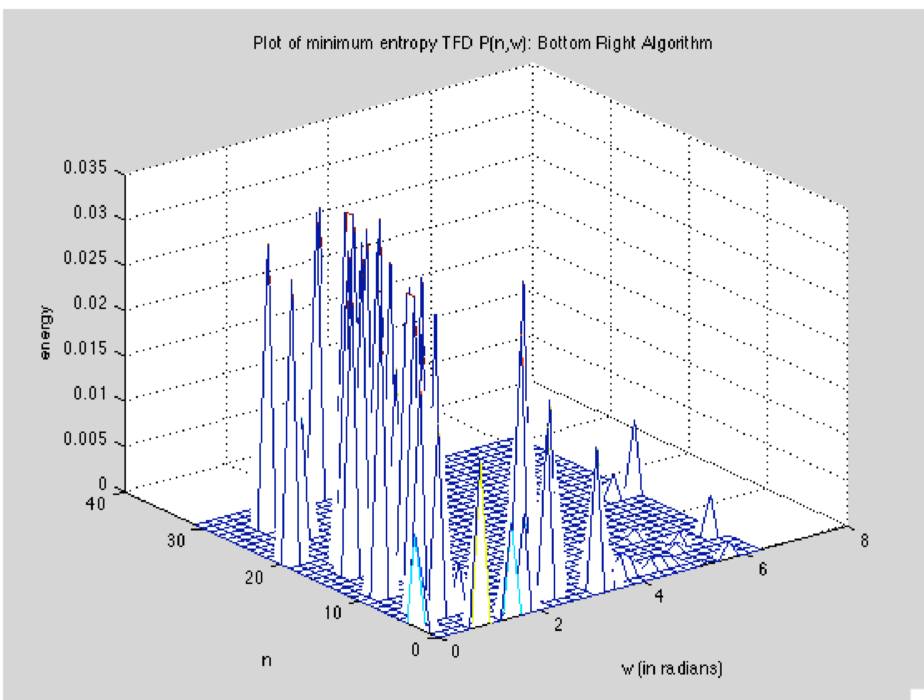

Figure 3.4: Plot of minimum entropy TFD obtained using the bottom-right method for the simple chirp signal

3.1.2 Intelligent Selection Methods

Next, we will consider the following intelligent selection methods: maximum entropy loss and n^{th} maximum entropy loss. The minimum entropy algorithm was performed on the simple chirp signal using the maximum entropy loss rectangle selection method and for the n^{th} maximum entropy loss rectangle selection method for n = 1 to 20, 50, and 100. Table 3.2 summarizes the total entropy and number of zero-points for the final minimum entropy TFD for each intelligent selection method for the simple chirp signal.

In the maximum entropy loss method, the algorithm searches through all of the possible rectangles and selects the rectangle that yields the greatest loss in entropy. Figure 3.5 shows the minimum entropy TFD obtained using the maximum entropy loss method. The final TFD exhibits a total entropy of 3.7022 and contains 961 zero-points.

In the n^{th} maximum entropy loss method, the algorithm searches through all of the possible rectangles and selects the rectangle that yields the n^{th} greatest loss in entropy. Figure 3.6 – 3.10 show the minimum entropy TFD obtained using the n^{th} maximum entropy loss method for n = 5, 10, 15, 20, and 50. The plots for the other values of n are not included because they do not provide any additional insight. Again, Table 3.2 shows the total entropy and number of zero-points for different values of n. Note that the n^{th} maximum entropy loss method is identical to the maximum entropy loss method for n = 1.

Table 3.2: Total entropy and number of zero-points for the maximum entropy loss and n^{th} maximum loss methods for different values of n for the simple chirp signal.

n	Total Entropy	Zero-Points
1	3.7022	961
2	3.7014	961
3	3.6981	961
4	3.6979	961
5	3.6907	961
6	3.6953	961
7	3.6932	961
8	3.6936	961
9	3.7018	961
10	3.6942	961
11	3.6953	961
12	3.6990	961
13	3.6940	961
14	3.6943	961
15	3.6948	961
16	3.6914	961
17	3.6952	961
18	3.6916	961
19	3.6932	961
20	3.6940	961
50	3.7005	961
100	3.6979	962

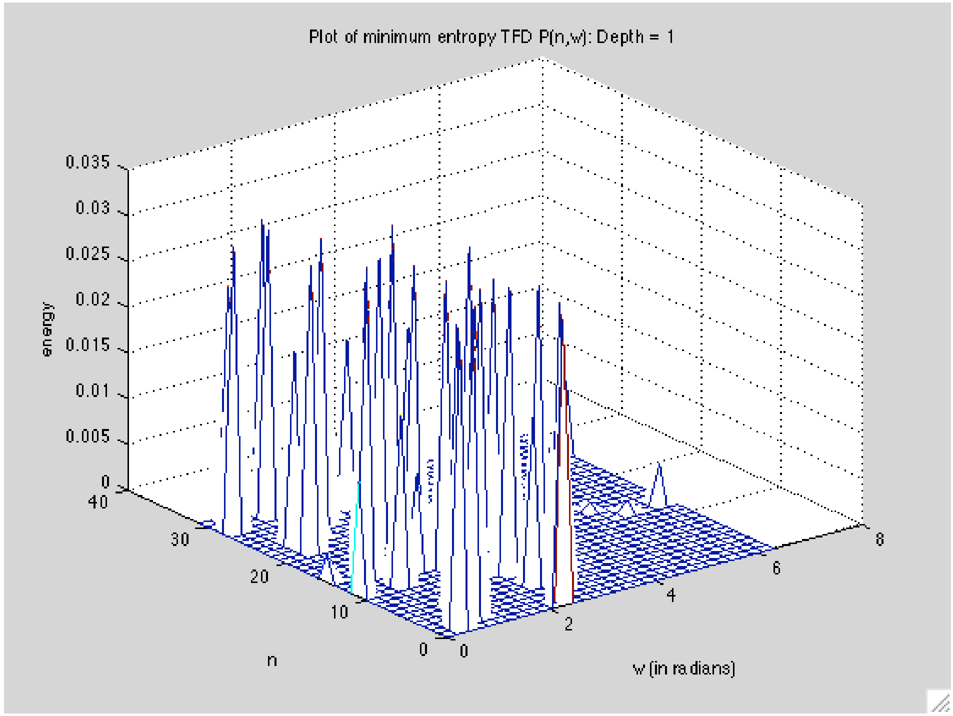

Figure 3.5: Plot of minimum entropy TFD obtained using the maximum entropy loss method for the simple chirp signal

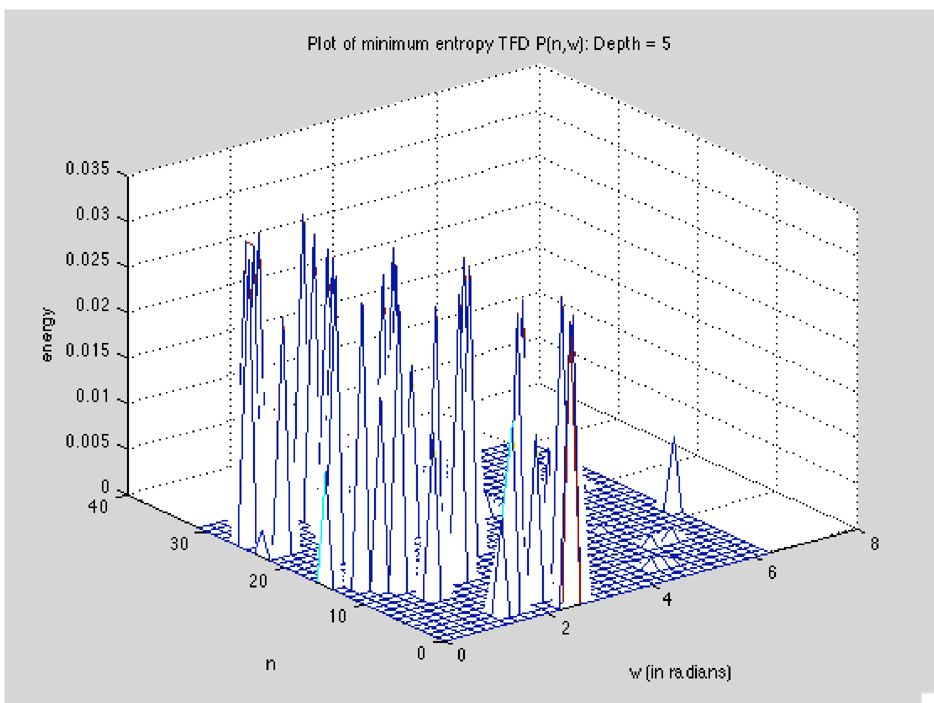

Figure 3.6: Plot of minimum entropy TFD obtained using the 5th maximum entropy loss method for the simple chirp signal

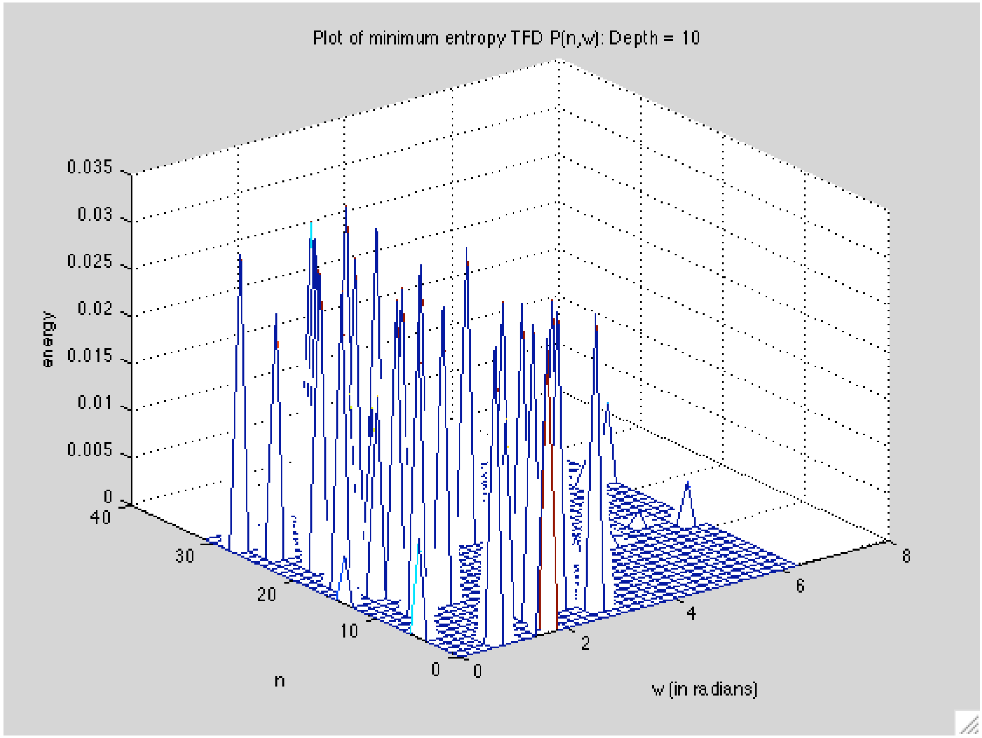

Figure 3.7: Plot of minimum entropy TFD obtained using the 10th maximum entropy loss method for the simple chirp signal

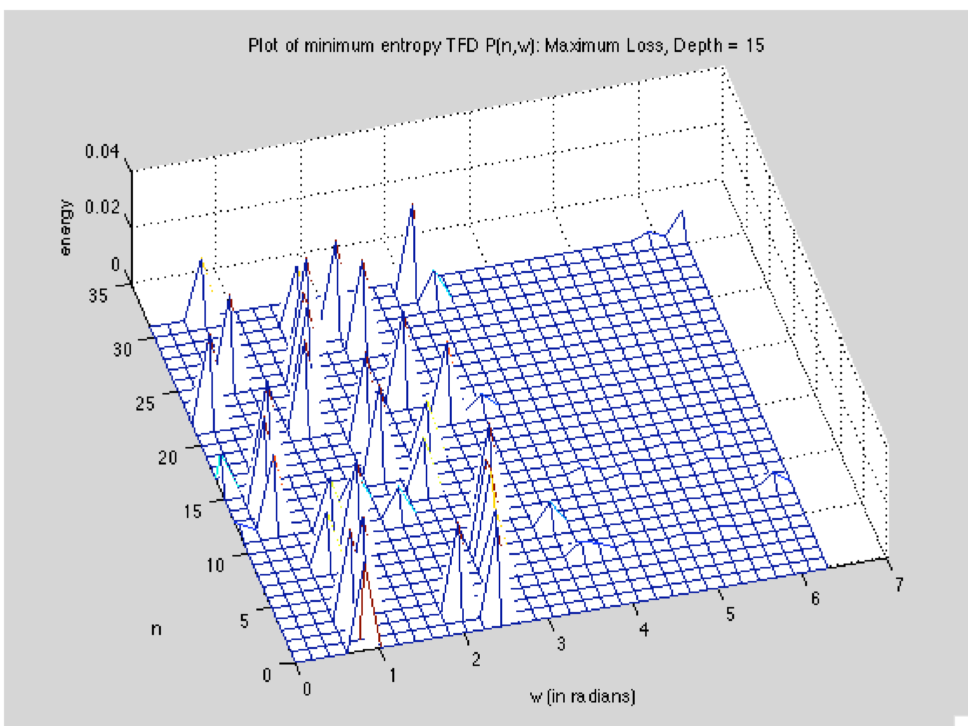

Figure 3.8: Plot of minimum entropy TFD obtained using the 15th maximum entropy loss method for the simple chirp signal

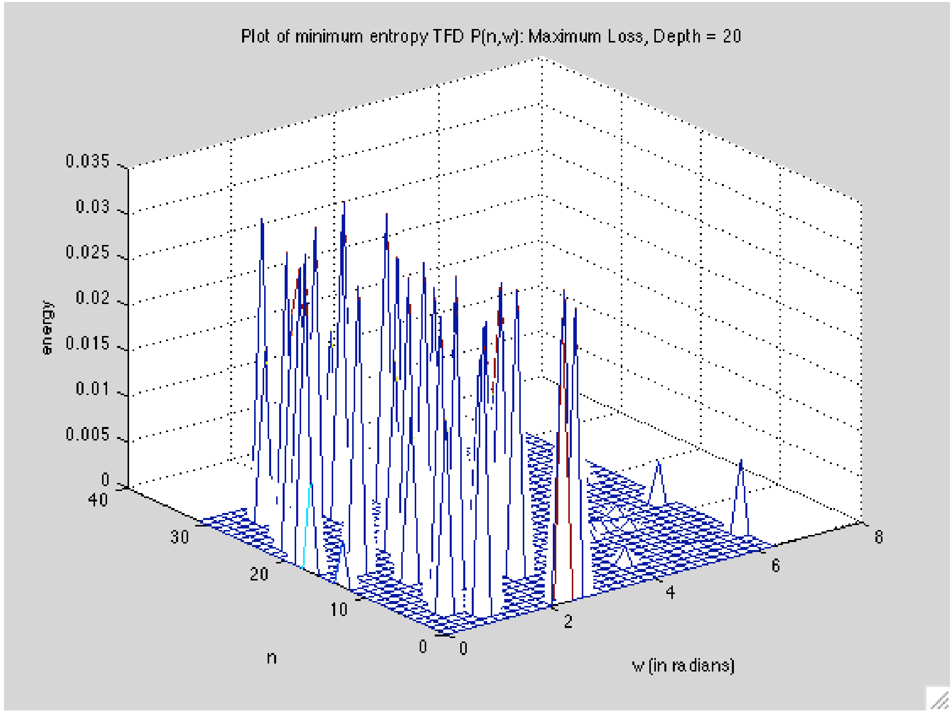

Figure 3.9: Plot of minimum entropy TFD obtained using the 20th maximum entropy loss method for the simple chirp signal

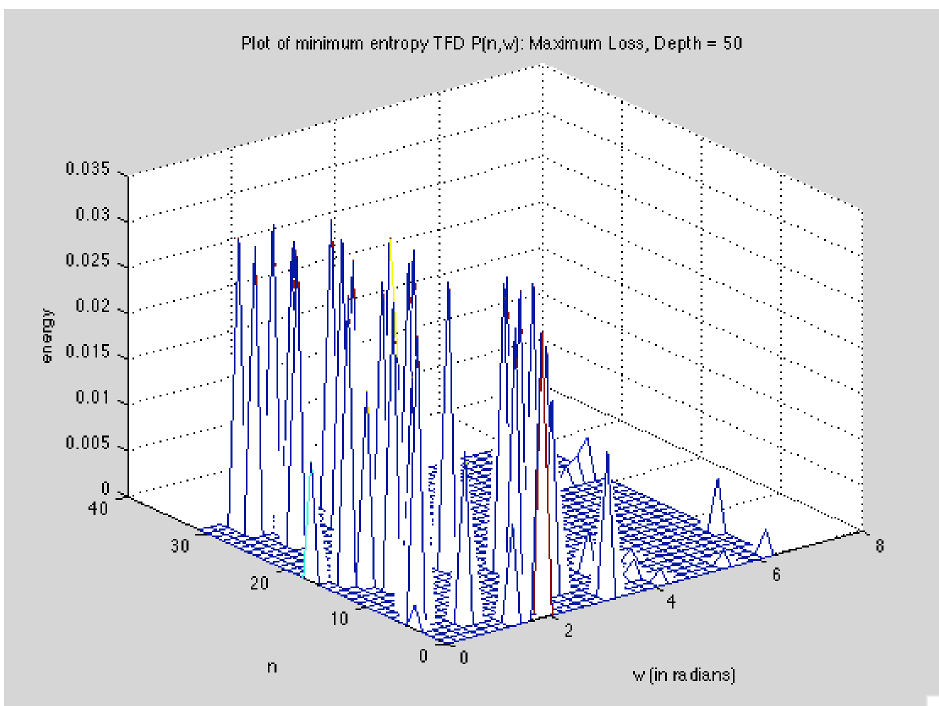

Figure 3.10: Plot of minimum entropy TFD obtained using the 50th maximum entropy loss method for the simple chirp signal

3.1.3 Summary

Each TFD in this section is a valid, proper TFD. They are all manifestly positive and satisfy the time and frequency marginals. In addition, each TFD is highly localized with the energy concentrated at certain locations. Each method provides a different final minimum TFD that has a different total entropy. The methods have final entropies ranging from 3.68 to 3.71 and zero-points ranging from 961 to 962.

None of the resulting minimum entropy TFDs appears to be a meaningful representation of the simple chirp signal. For the brute force methods, the only distribution that is remotely close to the desired result is that of the bottom-right method. Unfortunately, this distribution exhibits a significant amount of energy at time-frequency points that should not have any. The rest of the results from the brute force methods are not close to being accurate. The results from the intelligent selection methods are also not meaningful representations. None of the plots have energy concentrated in the expected areas.

Also note that the initial TFD, $P(n,\omega)$, that is used in this example is not a very good representation of the chirp signal. Since the minimum entropy algorithm modifies points of the initial TFD, it is highly unlikely that the final TFD will be a meaningful representation if the initial TFD is not meaningful. Thus, using the maximum entropy TFD as the prior could be one of the causes for the less than desirable results. Different priors will be explored in later sections of this thesis.

3.2 CHIRP WITH SINUSOID EXAMPLE

To further explore the different rectangle selection methods, we will next look at a chirp signal with the addition of a constant sinusoid at $\omega = 5\pi/4$ rad. The equation under consideration is:

$$x(n) = c*(e^{j\pi n^2/(2N)} + e^{j5\pi n/4})$$

where c is the normalization constant and N is the length of the signal which is 32 in this example. As before, the initial TFD will be $P(n,\omega) = |x(n)|^2 \, |X(\omega)|^2$ where $X(\omega)$ is the 32 point DFT of x(n). The initial TFD has a total entropy of 5.2592 and 64 zero-points. Figure 3.11 shows a plot of the real part of x(n). Figure 3.12 shows a plot of the spectrograms for x(n), respectively. Figure 3.13 shows a plot of the initial TFD $P(n,\omega)$.

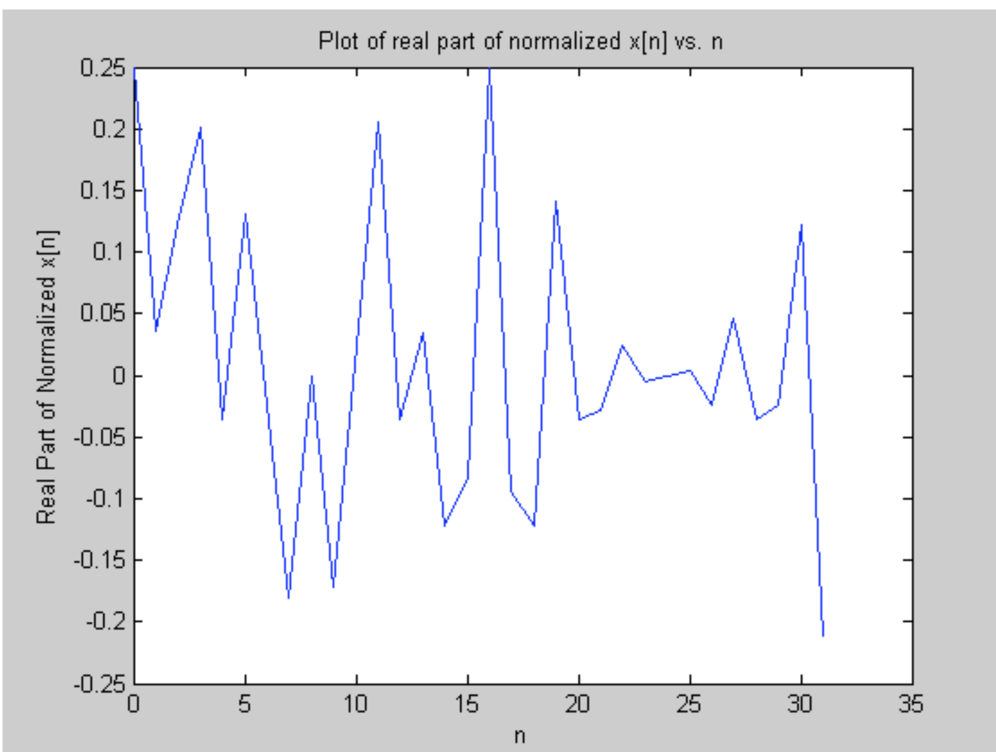

Figure 3.11: Plot of the real part of the chirp with sinusoid signal

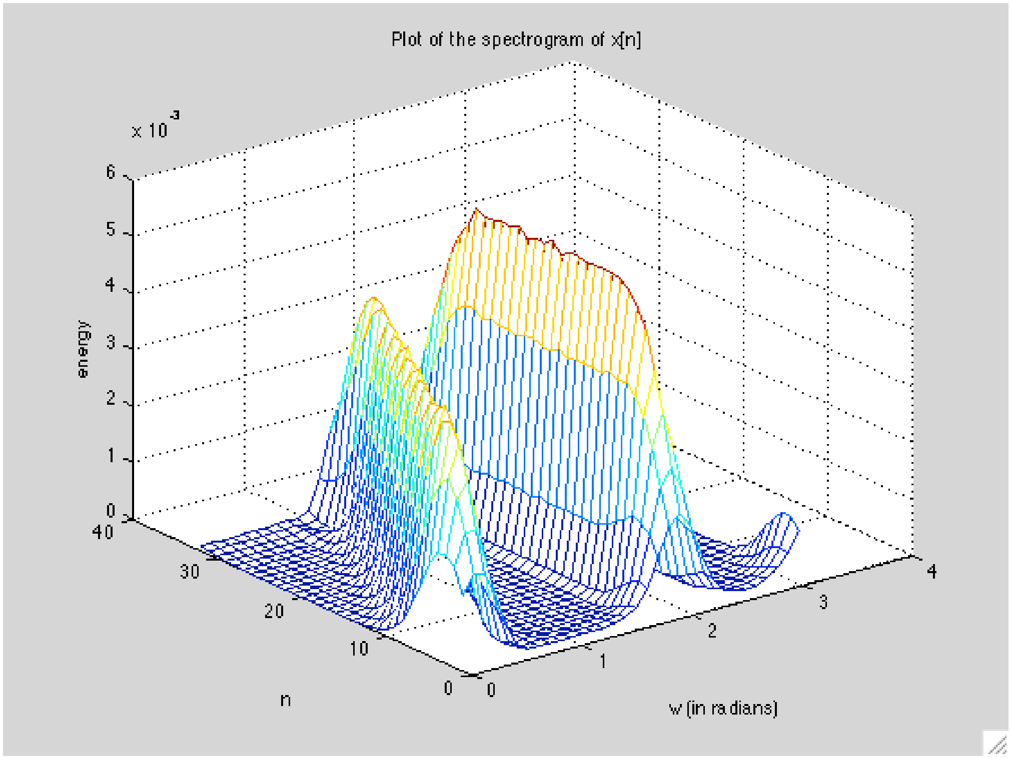

Figure 3.12: Plot of the spectrogram of the chirp with sinusoid signal

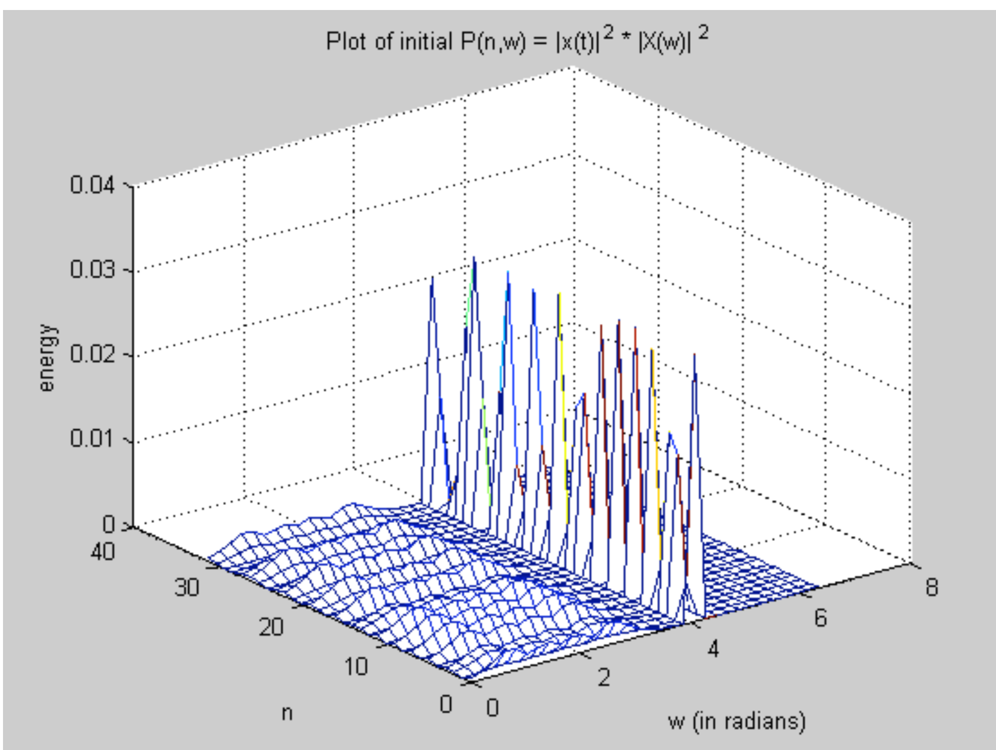

Figure 3.13: Plot of the initial TFD, $P(n,\omega) = |x(n)|^2 \, |X(\omega)|^2$, for the chirp with sinusoid signal

3.2.1 Brute Force Methods

First, we will consider the four brute force rectangle selection methods: top-left method, top-right method, bottom-left method, and bottom-right method. Table 3.3 shows the total entropy and number of zero-points for the minimum entropy TFDs for the chirp with sinusoid signal obtained using the different brute force rectangle selection methods. Figure 3.14 shows the minimum entropy TFD obtained using the top-left method. Figure 3.15 shows the minimum entropy TFD obtained using the top-right method. Figure 3.16 shows the minimum entropy TFD obtained using the bottom-left method. Figure 3.17 shows the minimum entropy TFD obtained using the bottom-right method.

Table 3.3: Total entropy and number of zero-points for the minimum entropy TFDs for the chirp with sinusoid signal obtained using the different brute force methods.

Method	Total Entropy	Zero-Points
Top-Left	3.3244	963
Top-Right	3.3438	963
Bottom-Left	3.2978	963
Bottom-Right	3.3325	963

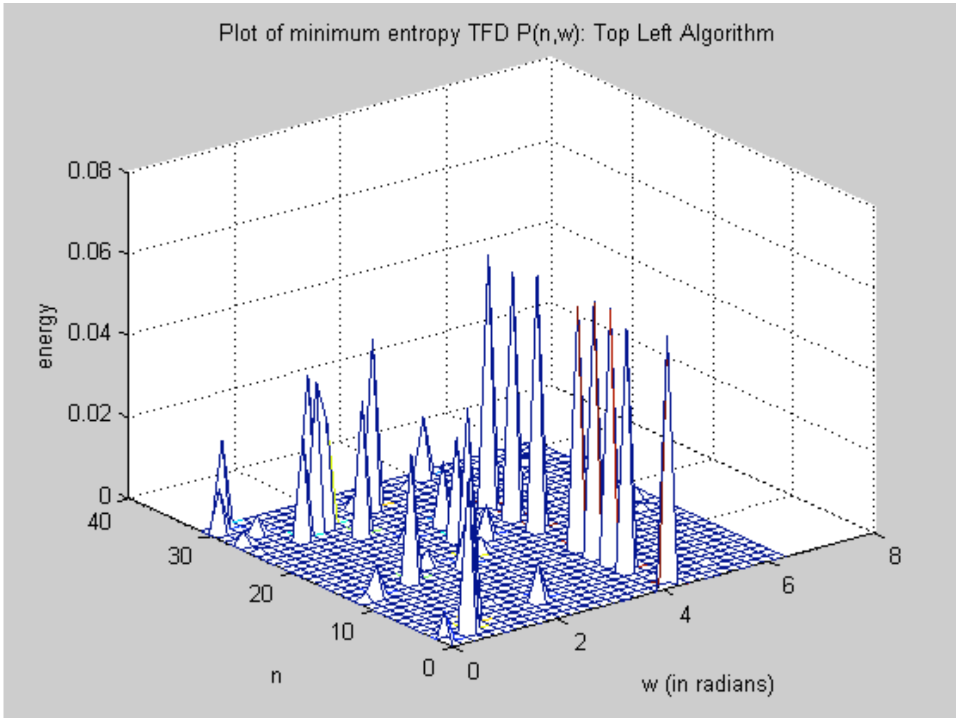

Figure 3.14: Plot of the minimum entropy TFD obtained using the top-left method for the chirp with sinusoid signal

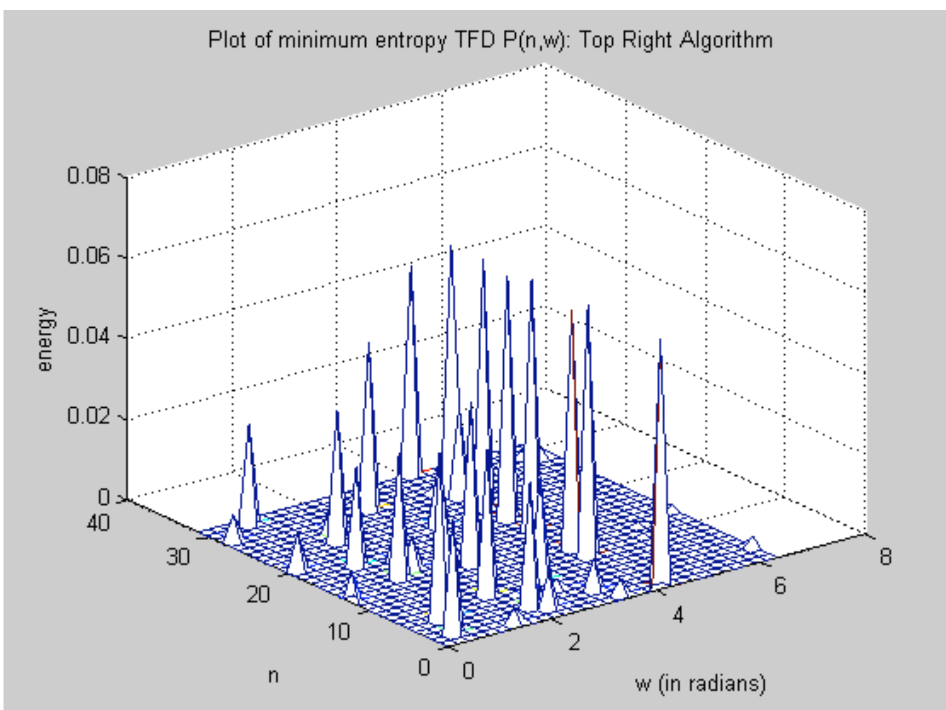

Figure 3.15: Plot of the minimum entropy TFD obtained using the top-right method for the chirp with sinusoid signal

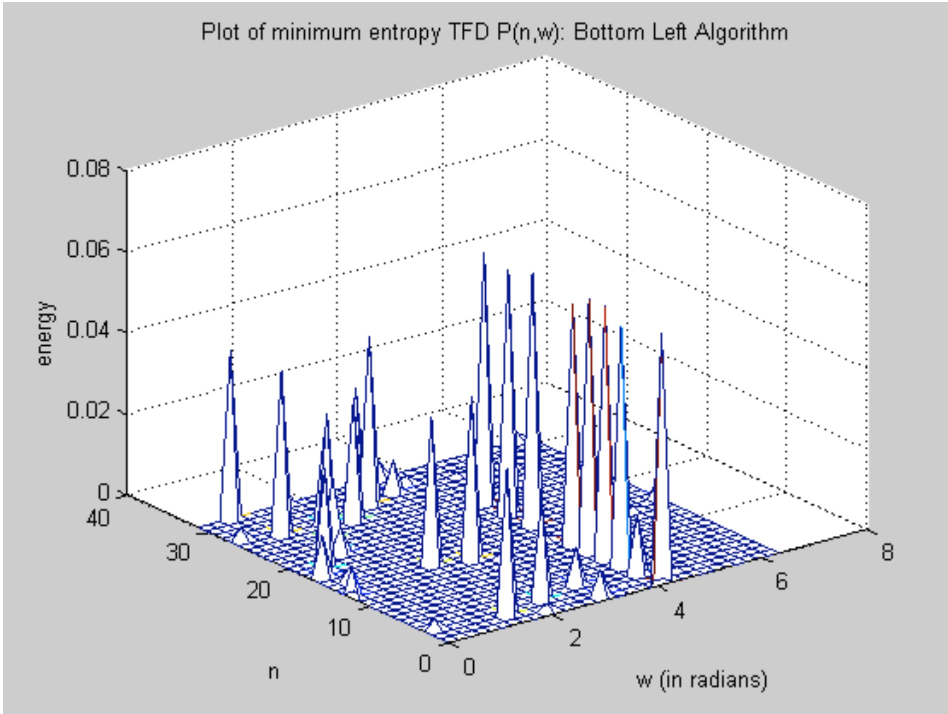

Figure 3.16: Plot of the minimum entropy TFD obtained using the bottom-left method for the chirp with sinusoid signal

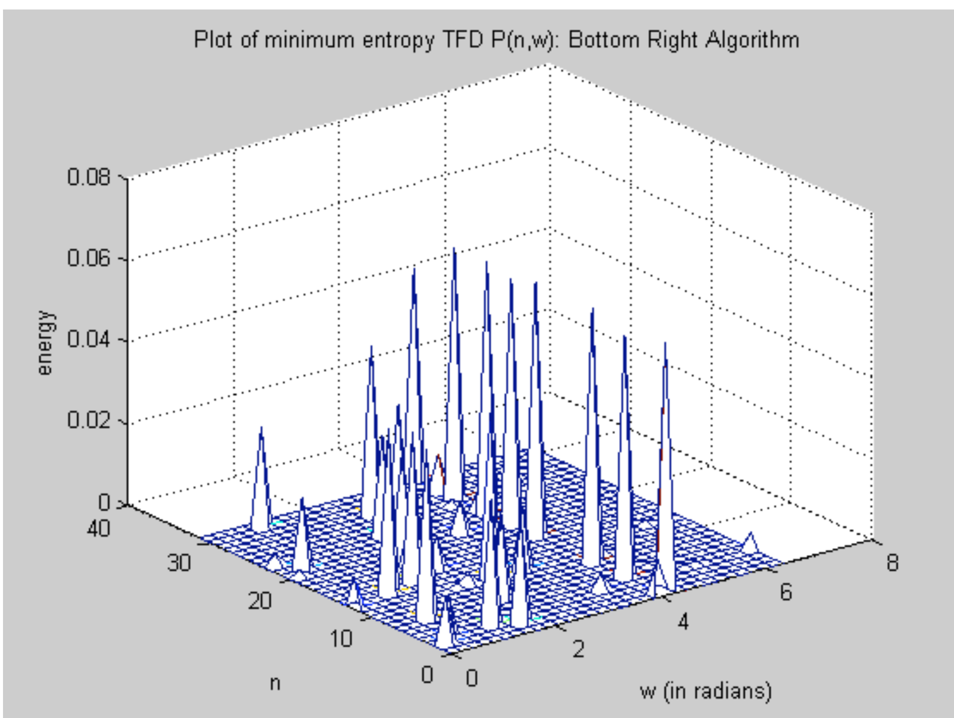

Figure 3.17: Plot of the minimum entropy TFD obtained using the bottom-right method for the chirp with sinusoid signal

3.2.2 Intelligent Selection Methods

Next, we will consider the intelligent selection methods for the chirp with sinusoid signal. The minimum entropy TFD was found for the signal using the maximum entropy loss and n^{th} maximum entropy loss rectangle selection methods for n = 1 to 20, 50, and 100. Table 3.4 shows the total entropy and number of zero-points for the minimum entropy TFDs obtained using the intelligent selection methods. Figures 3.18 through 3.23 show the minimum entropy TFDs obtained using the maximum entropy loss method and the n^{th} maximum entropy loss method where n = 5, 10, 15, 20, and 50. The plots for the other values of n are not included because they do not provide any additional insight.

Table 3.4: Total entropy and number of zero-points for the maximum entropy loss and nth maximum loss methods for different values of n for the chirp with sinusoid signal.

n	Total Entropy	Zero-Points
1	3.3465	963
2	3.3188	963
3	3.3287	963
4	3.3122	963
5	3.3431	962
6	3.3152	963
7	3.3052	963
8	3.2953	963
9	3.3277	963
10	3.2990	963
11	3.2841	963
12	3.3105	963
13	3.3170	963
14	3.3195	963
15	3.3029	962
16	3.3306	962
17	3.3415	963
18	3.3018	963
19	3.3243	963
20	3.3229	963
50	3.3366	963
100	3.3152	963

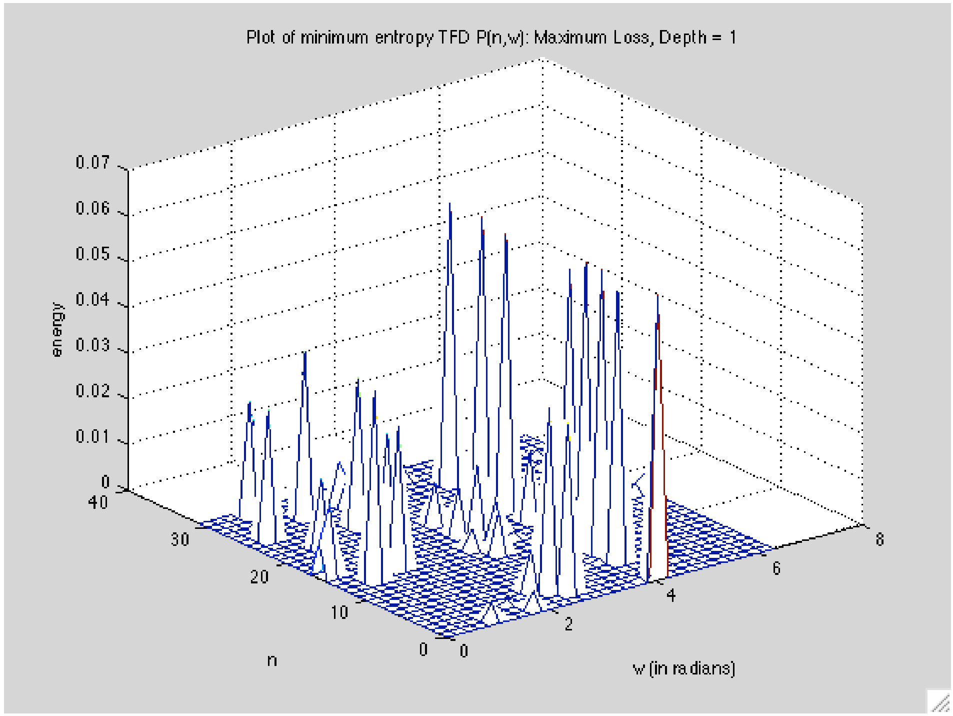

Figure 3.18: Plot of minimum entropy TFD obtained using the maximum entropy loss method for the chirp with sinusoid signal

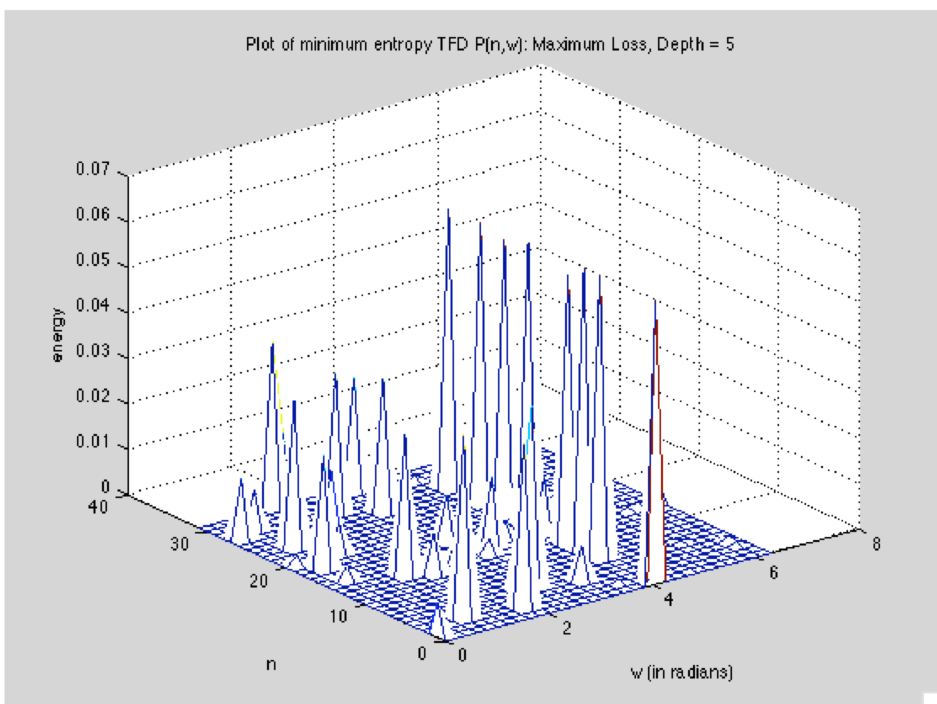

Figure 3.19: Plot of minimum entropy TFD obtained using the 5th maximum entropy loss method for the chirp with sinusoid signal

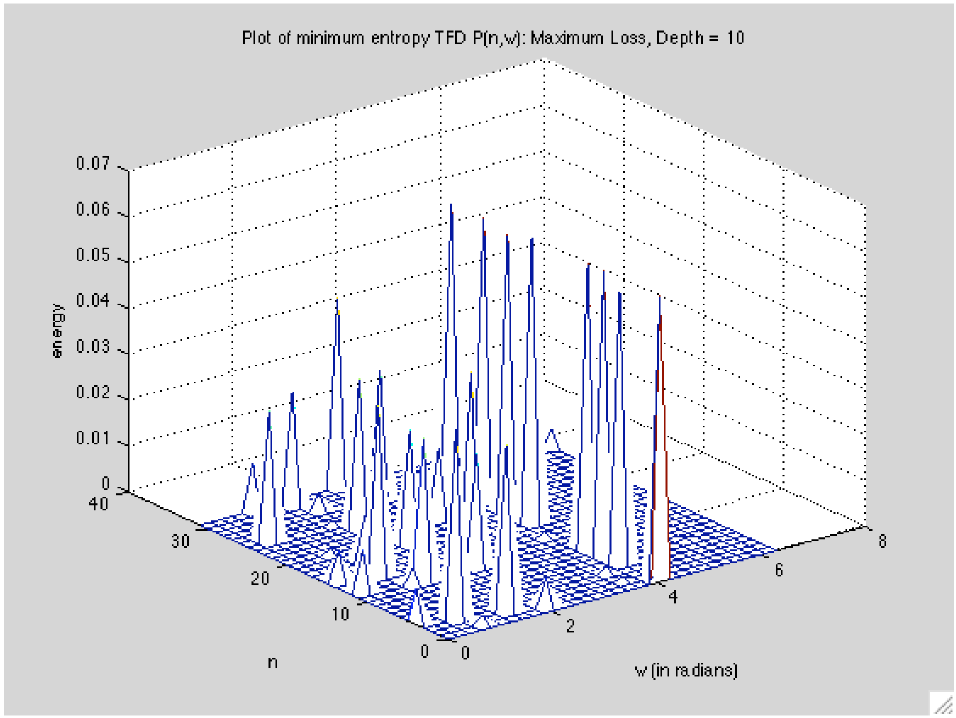

Figure 3.20: Plot of minimum entropy TFD obtained using the 10^{th} maximum entropy loss method for the chirp with sinusoid signal

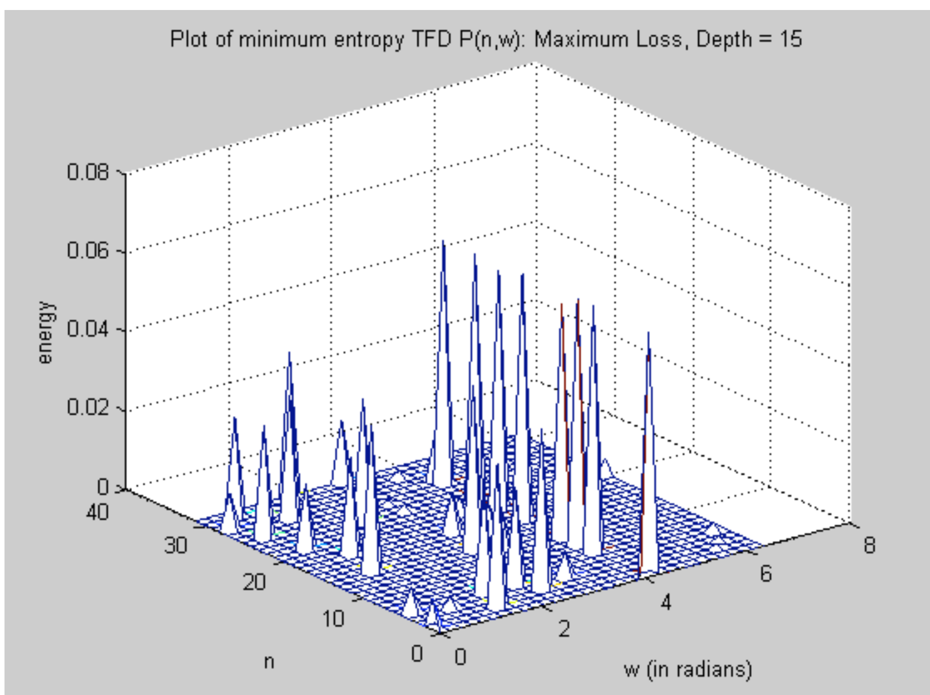

Figure 3.21: Plot of minimum entropy TFD obtained using the 15^{th} maximum entropy loss method for the chirp with sinusoid signal

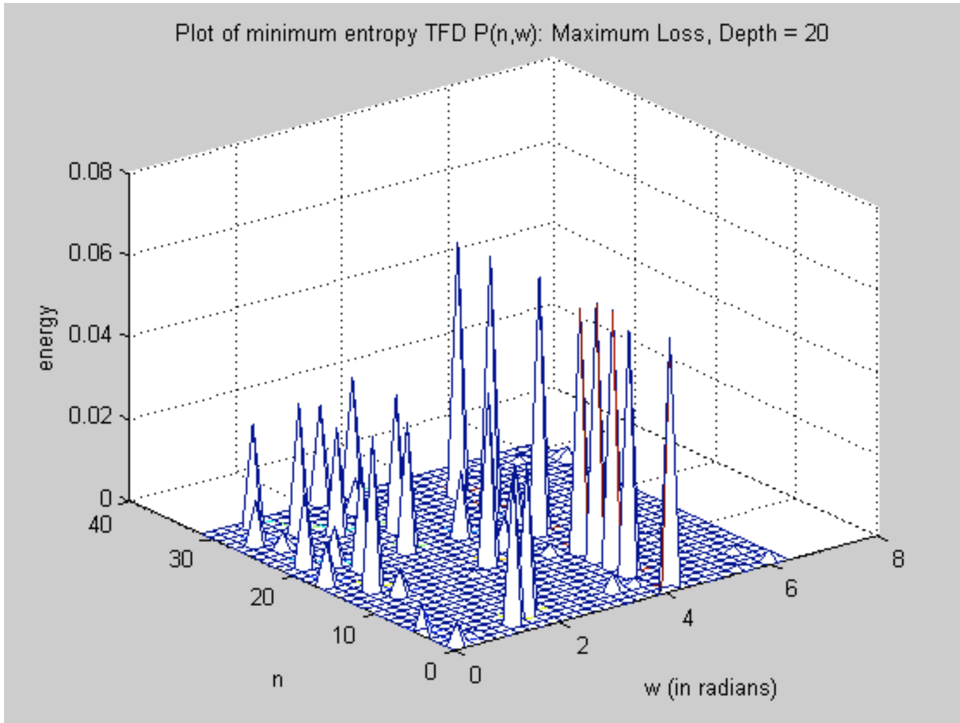

Figure 3.22: Plot of minimum entropy TFD obtained using the 20th maximum entropy loss method for the chirp with sinusoid signal

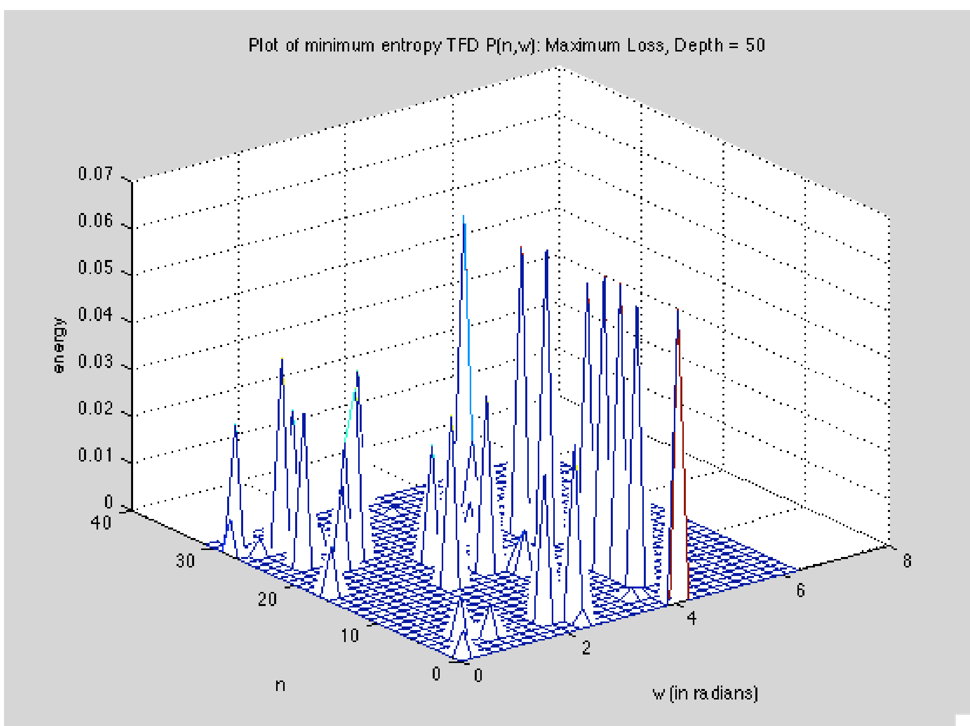

Figure 3.23: Plot of minimum entropy TFD obtained using the 50th maximum entropy loss method for the chirp with sinusoid signal

3.2.3 Summary

All of the final minimum entropy TFDs in this section are valid, highly-localized, proper TFDs. The TFDs ended with total entropies ranging from 3.28 to 3.35 and exhibited 962 or 963 zero-points. Unfortunately, none of the minimum entropy distributions appear to be a meaningful representation of the chirp with sinusoid signal.

Looking at the four brute force minimum entropy TFDs, the distribution found using the bottom-right algorithm appears to be the closest to what is expected or desired (i.e. energy at $\omega = 5\pi/4$ radians and along the chirp diagonal). All of the TFDs have energy along $\omega = 5\pi/4$ radians. The top-left, top-right, and bottom-left methods seem to completely lose the chirp information. The bottom-right method appears to have kept some of this information, but not completely.

As with the brute force methods, none of the intelligent selection methods produced a meaningful representation of the chirp with sinusoid signal. Although the sinusoidal energy along $\omega = 5\pi/4$ appears to remain for the most part, the chirp portion of the signal appears to be completely lost for all of the methods.

Like with the simple chirp signal, the initial TFD, $P(n,\omega)$, is not a very good representation of the signal. Since the minimum entropy algorithm modifies points of the initial TFD, it is highly unlikely that the final TFD will be a meaningful representation if the initial TFD is not meaningful. Thus, using the maximum entropy TFD as the prior could be one of the causes for the less than desirable results found in this section.

3.3 DOUBLE CHIRP EXAMPLE

Finally, we will next look at a double chirp signal that has the chirp signal from the first example plus a chirp signal that is decreasing in frequency. The equation under consideration is:

$$x(n) = c*(e^{j\pi n\text{^}2/(2N)} + e^{j\pi(2\pi - n)\text{^}2/(2N)})$$

where c is the normalization constant and N is the length of the signal which is 32 in this example. As before, the initial TFD will be $P(n,\omega) = |x(n)|^2 \, |X(\omega)|^2$ where $X(\omega)$ is the 32 point DFT of x(n). The initial TFD has a total entropy of 6.5581 and 0 zero-points. Figure 3.24 shows a plot of the real part of x(n). Figure 3.25 shows a plot of the spectrogram for the double chirp signal. Figure 3.26 shows a plot of the initial TFD P(n,ω) for the double chirp signal.

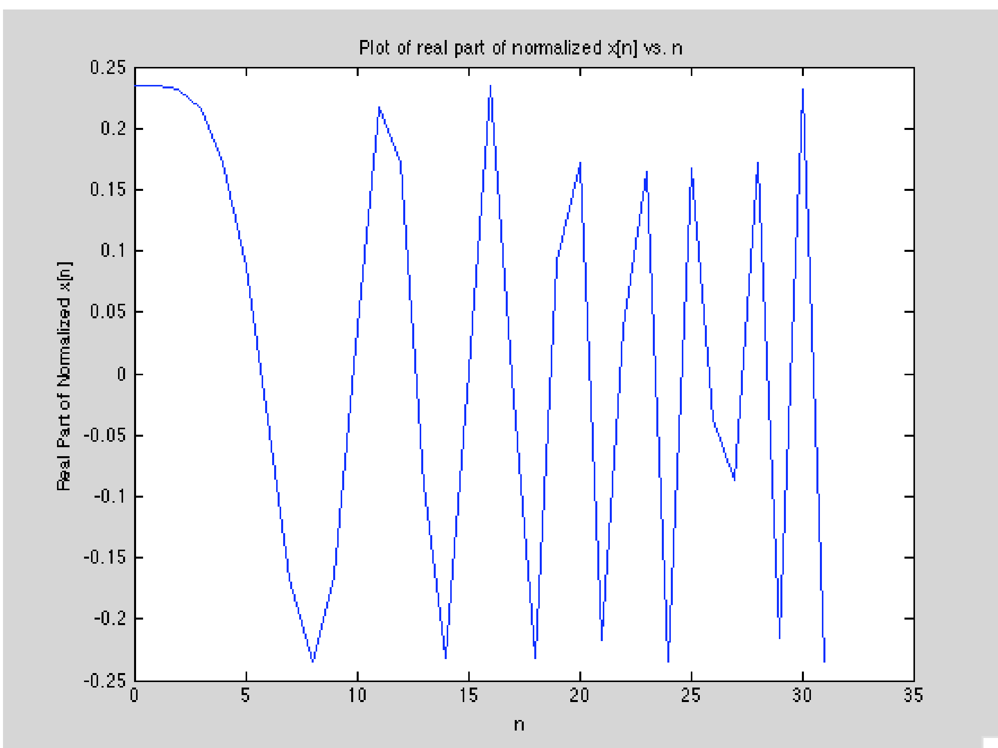

Figure 3.24: Plot of the real part of the double chirp signal

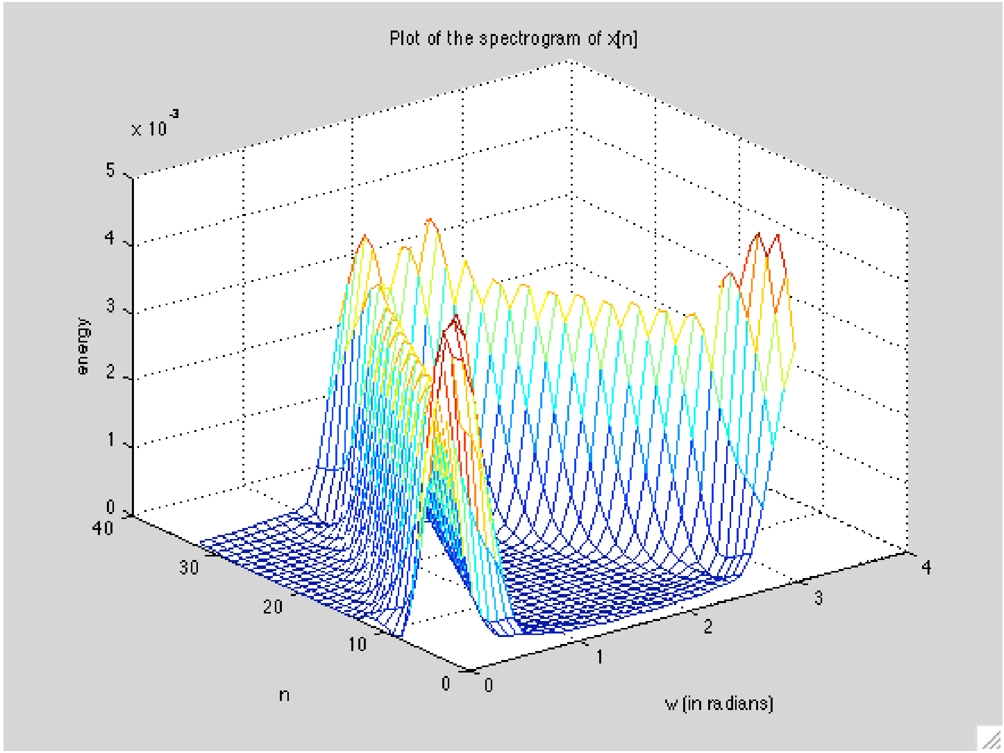

Figure 3.25: Plot of the spectrogram of the double chirp signal

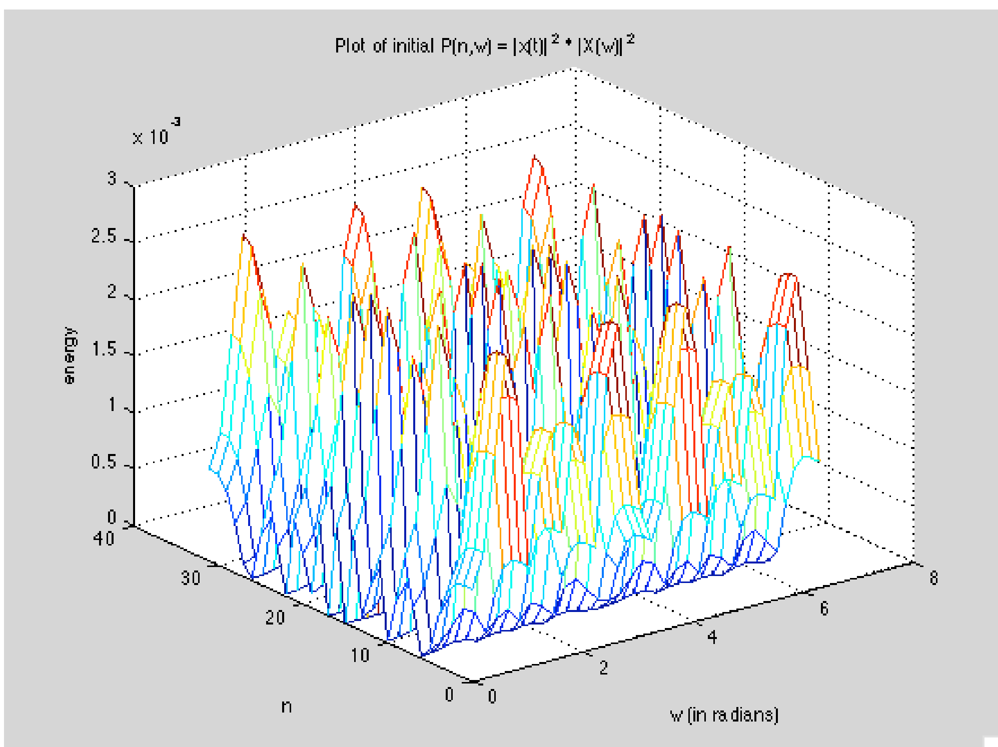

Figure 3.26: Plot of the initial TFD, $P(n,\omega) = |x(n)|^2\ |X(\omega)|^2$, for the double chirp signal

3.3.1 Brute Force Methods

We will first consider the four brute force rectangle selection methods: top-left method, top-right method, bottom-left method, and bottom-right method. Table 3.5 shows the total entropy and number of zero-points for the minimum entropy TFDs for the double chirp signal obtained using the different brute force rectangle selection methods. Figure 3.27 shows the minimum entropy TFD obtained using the top-left method. Figure 3.28 shows the minimum entropy TFD obtained using the top-right method. Figure 3.29 shows the minimum entropy TFD obtained using the bottom-left method. Figure 3.30 shows the minimum entropy TFD obtained using the bottom-right method.

Table 3.5: Total entropy and number of zero-points for the minimum entropy TFDs for the double chirp signal obtained using the different brute force methods.

Method	Total Entropy	Zero-Points
Top-Left	3.5945	961
Top-Right	3.5623	961
Bottom-Left	3.5955	961
Bottom-Right	3.5857	961

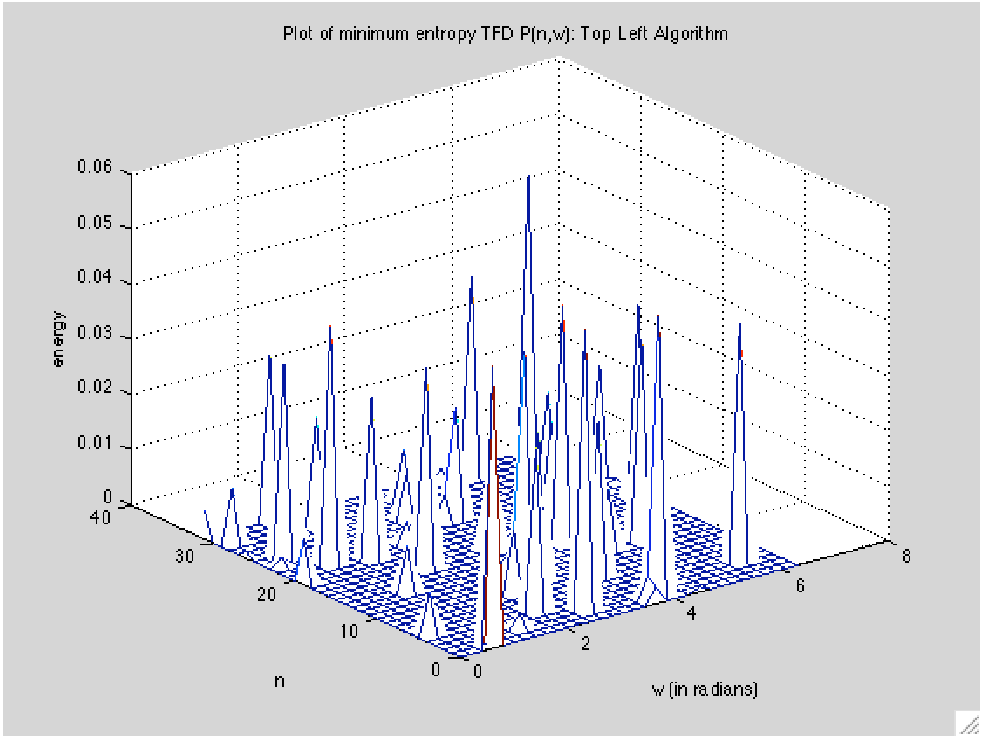

Figure 3.27: Plot of the minimum entropy TFD obtained using the top-left method for the double chirp signal

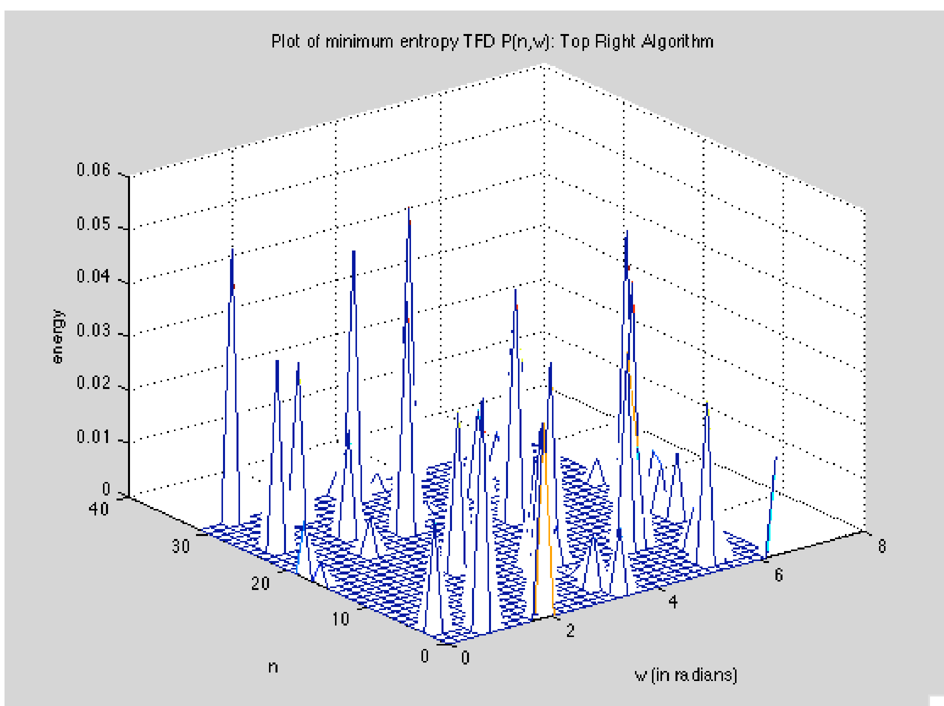

Figure 3.28: Plot of the minimum entropy TFD obtained using the top-right method for the double chirp signal

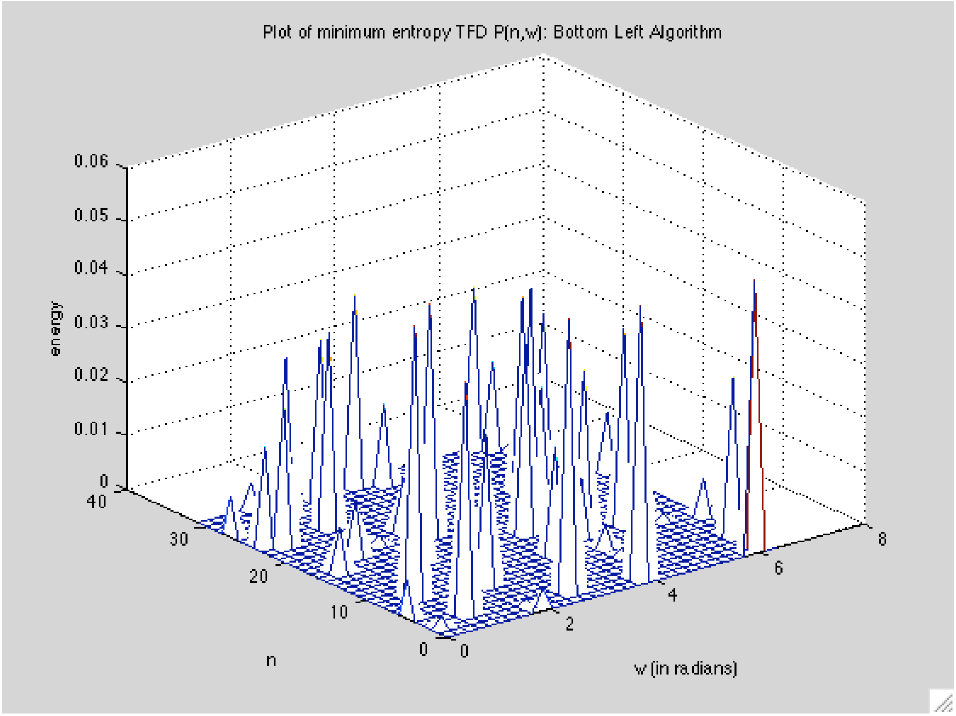

Figure 3.29: Plot of the minimum entropy TFD obtained using the bottom-left method for the double chirp signal

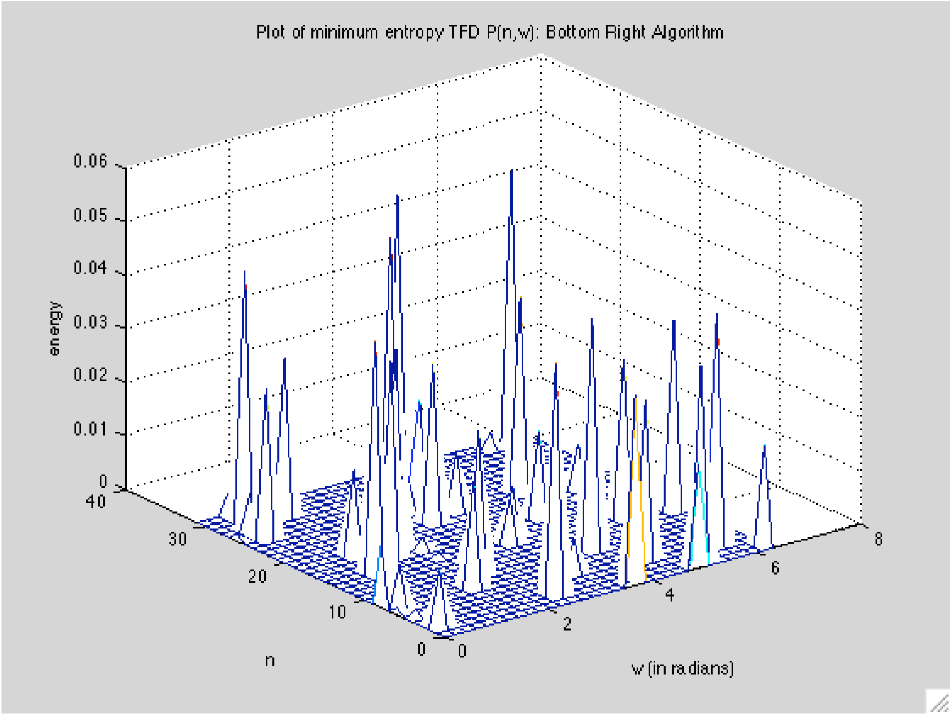

Figure 3.30: Plot of the minimum entropy TFD obtained using the bottom-right method for the double chirp signal

3.3.2 Intelligent Selection Methods

Next, we will consider the intelligent selection methods for the double chirp signal. The minimum entropy TFD was found for the signal using the maximum entropy loss and n^{th} maximum entropy loss rectangle selection methods for n = 1 to 20, 50, and 100. Table 3.6 shows the total entropy and number of zero-points for the minimum entropy TFDs obtained using the intelligent selection methods. Figures 3.31 through 3.36 show the minimum entropy TFDs obtained using the maximum entropy loss method and the n^{th} maximum entropy loss method where n = 5, 10, 15, 20, and 50. The plots for the other values of n are not included because they do not provide any additional insight.

Table 3.6: Total entropy and number of zero-points for the maximum entropy loss and n^{th} maximum loss methods for different values of n for the double chirp signal.

n	Total Entropy	Zero-Points
1	3.5464	961
2	3.5358	961
3	3.5420	961
4	3.5452	961
5	3.5945	961
6	3.5761	961
7	3.5677	961
8	3.5634	961
9	3.5620	961
10	3.5694	961
11	3.5611	961
12	3.5731	961
13	3.5512	961
14	3.5571	961
15	3.5693	961
16	3.5753	961
17	3.5807	961
18	3.5620	961
19	3.5647	961
20	3.5605	961
50	3.5580	961
100	3.5819	961

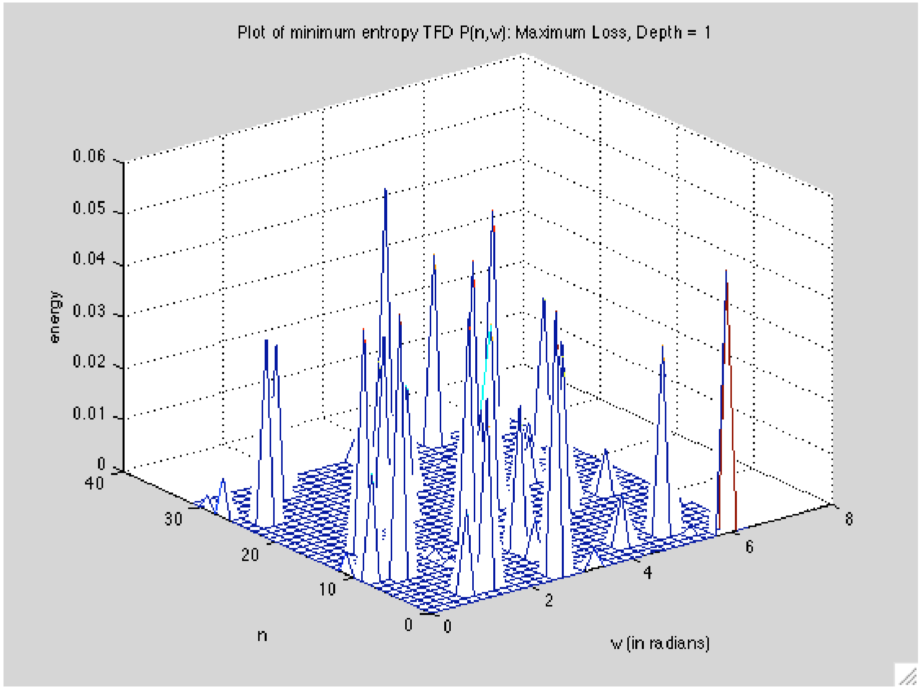

Figure 3.31: Plot of the minimum entropy TFD obtained using the maximum entropy loss method for the double chirp signal

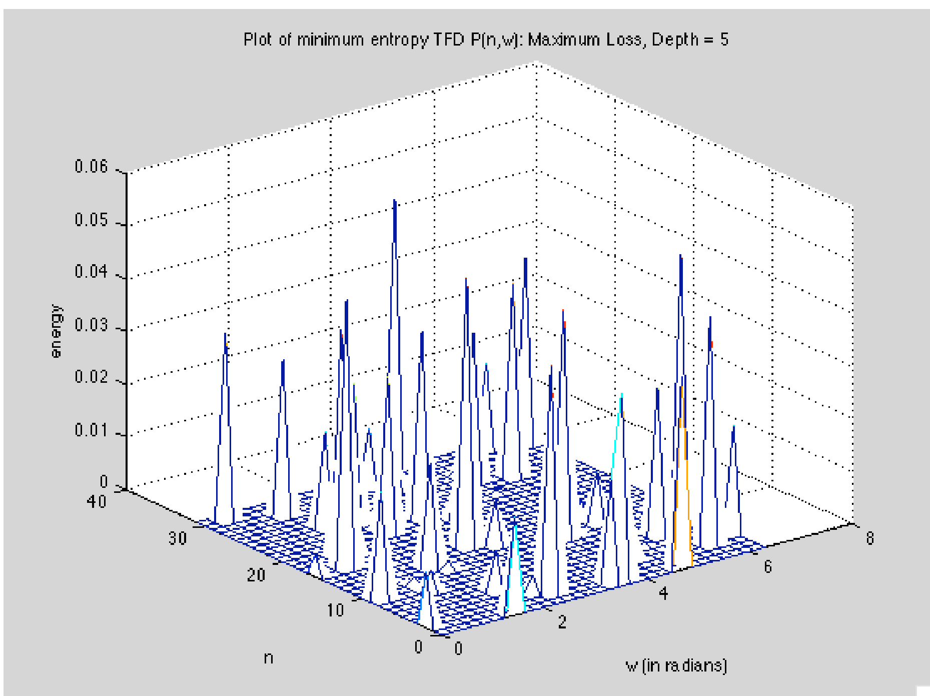

Figure 3.32: Plot of the minimum entropy TFD obtained using the 5^{th} maximum entropy loss method for the double chirp signal

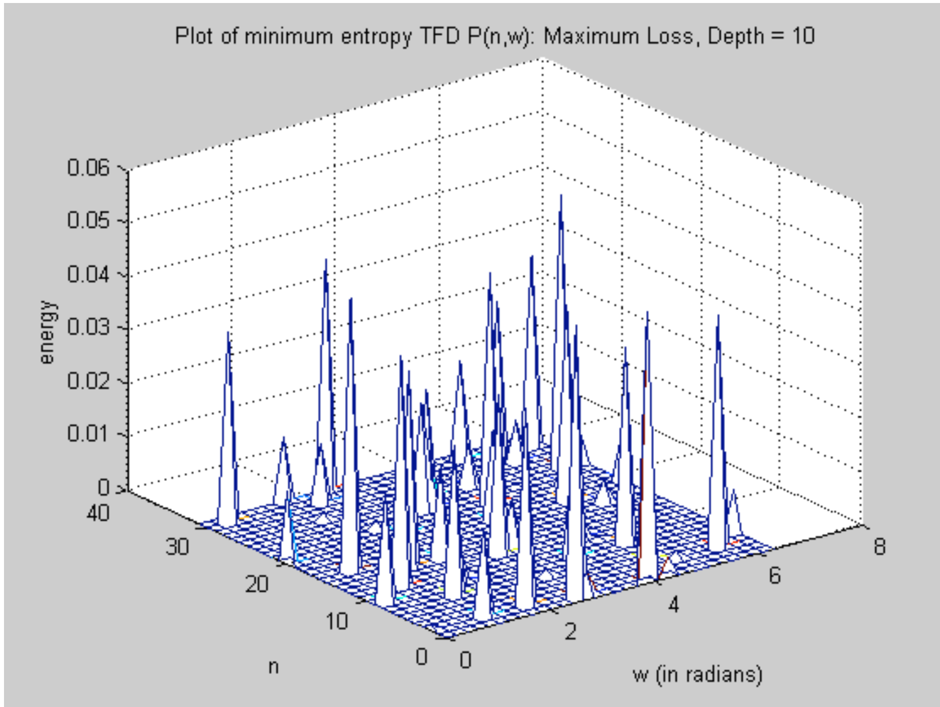

Figure 3.33: Plot of the minimum entropy TFD obtained using the 10th maximum entropy loss method for the double chirp signal

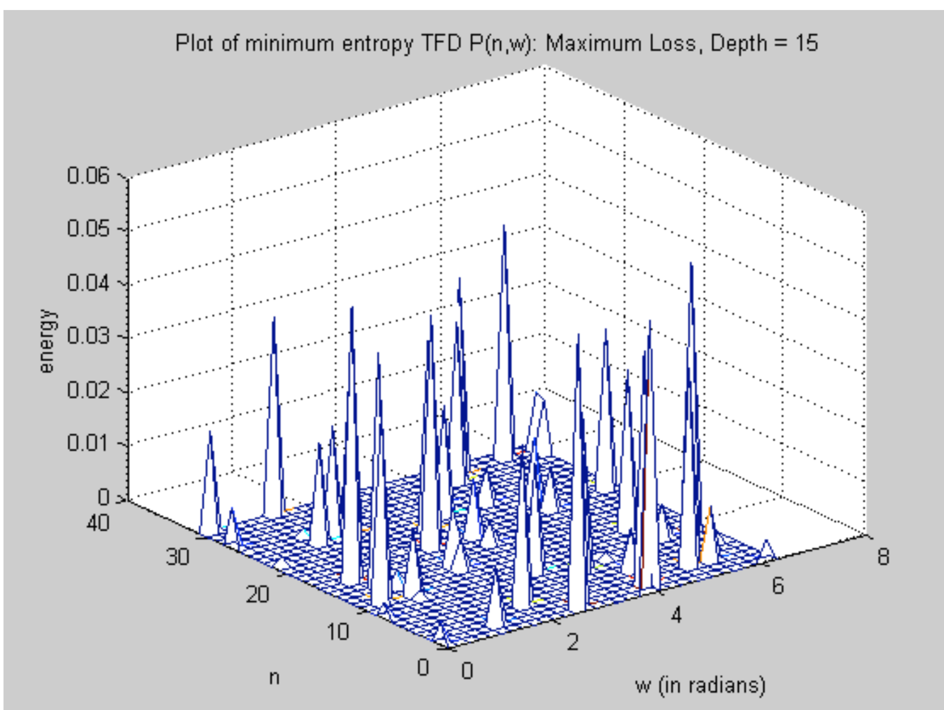

Figure 3.34: Plot of the minimum entropy TFD obtained using the 15th maximum entropy loss method for the double chirp signal

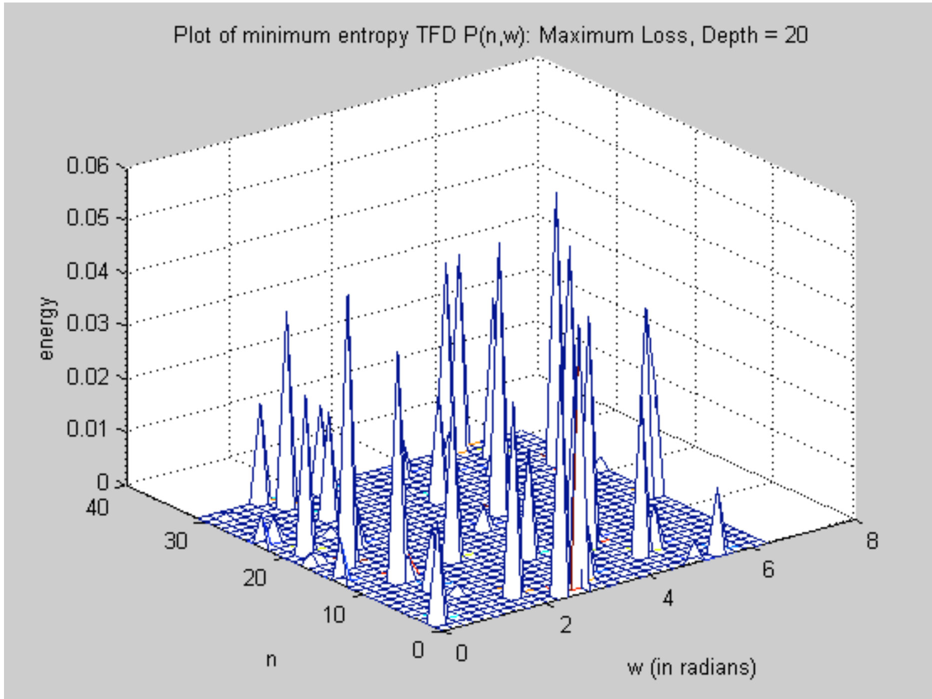

Figure 3.35: Plot of the minimum entropy TFD obtained using the 20th maximum entropy loss method for the double chirp signal

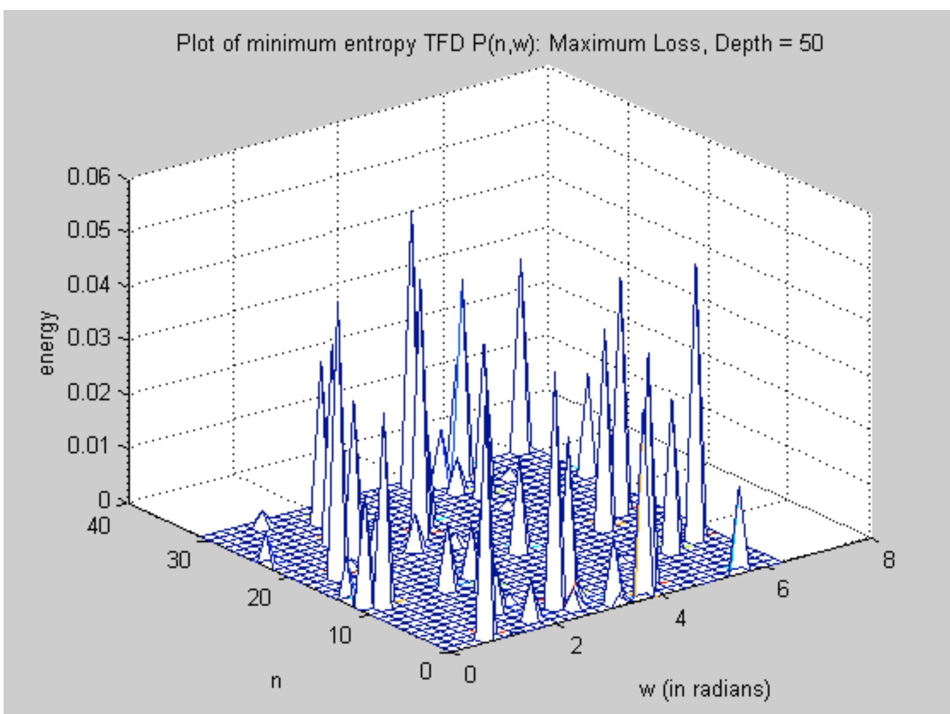

Figure 3.36: Plot of the minimum entropy TFD obtained using the 50th maximum entropy loss method for the double chirp signal

3.3.3 Summary

All of the final minimum entropy TFDs in this section are valid, highly-localized, proper TFDs. The TFDs ended with total entropies ranging from 3.53 to 3.60 and 961 zero-points. Unfortunately, none of the minimum entropy distributions appear to be a meaningful representation of the double chirp signal.

Looking at the four brute force minimum entropy TFDs, none of the final distributions are meaningful or desirable for the original signal. Each one has what appears to be random energy spikes scattered throughout the time-frequency domain.

As with the brute force methods, none of the intelligent selection methods produced a meaningful representation of the double chirp signal. The results of the methods also appear to be random.

Like with the first two signals, the initial TFD, $P(n,\omega)$, is not a very good representation of the signal. Since the minimum entropy algorithm modifies points of the initial TFD, it is highly unlikely that the final TFD will be a meaningful representation if the initial TFD is not meaningful. Thus, using the maximum entropy TFD as the prior could be one of the causes for the less than desirable results found in this section.

3.4 CONCLUSIONS

All of the distributions found using the proposed minimum entropy algorithms are valid, proper TFDs for their respective signal. In other words, they are manifestly positive and satisfy the time and frequency marginals. In addition, each TFD is highly localized and contains a significant number of zero-points.

For each signal and rectangle selection method, the number of zero-points appears to remain consistent. For a 32x32 matrix (i.e. 1024 total points), the number of zero-points always ranges from 960 to 963. Remember that the algorithm in this example has 1024 degrees of freedom (i.e. points) and 64 marginal constraints while minimizing the entropy and setting as many points as possible to zero. The difference between 1024 and 64 is 960. Thus, it is interesting and somewhat logical that the final number of zero-points is always around 960 indicating that the algorithm requires approximately 64 points to satisfy the 64 marginal constraints. In addition, it would be expected that the chirp with sinusoid signal and double chirp signal would require twice as many points to represent than the simple chirp signal since they have two frequency components at each point in time. It appears that the minimum entropy algorithm will always reduce the number of non-zero points to a value close to the total number of points minus the number of marginal constraints or, in mathematical form, $n*m - (n + m)$ where n is the number of time values in the initial TFD and m is the number of frequency values in the initial TFD. This property may not be desirable for many signals.

Unfortunately, none of the rectangle selection methods produced a meaningful representation for any of the signals. The brute force methods produced seemingly random results that differ greatly if a different starting point or selection pattern is chosen. Since these methods simply loop through all of the rectangles without regard to the values in each rectangle, it is difficult to optimize these methods to achieve more desirable results. Thus, minimum entropy algorithms will probably not be able to produce consistently meaningful TFDs using brute force methods.

In the examples, the intelligent selection methods did not fare any better than their brute force counterparts. Neither the maximum entropy loss nor the n^{th} maximum entropy loss methods resulted in meaningful time-frequency representations of the signals. Unlike the brute force methods, the intelligent selection methods are more flexible and may be able to be used to obtain a meaningful TFD. The key is to find the rectangle selection criterion that leads to a better result. From the previous examples, it is evident that the maximum entropy loss criterion may not be the best rectangle selection method.

The brute force algorithms are relatively easy to implement. The algorithm simply begins with a rectangle and loops through all of the possible rectangles. It repeats this process until none of the rectangles can be reduced. Thus, the algorithm does not have to search through every possible rectangle each iteration. Furthermore, these methods lend easily to parallelization since the next iteration does not always depend on the results of the previous iteration. Unfortunately, its ease of implementation is counterbalanced by its lack of flexibility.

The intelligent selection methods are more complicated to implement and require much more time to perform. Each iteration, the algorithm must search each possible rectangle to find the one that matches the stated criteria. This often requires a significant amount of computation and is much less efficient than the brute force methods. Furthermore, it is difficult to parallelize these types of algorithms because each iteration depends on the results of the previous iteration. Unlike the brute force methods, the implementation of the intelligent selection methods leads to increased flexibility.

For the n^{th} maximum entropy loss methods, the results do not appear to improve or worsen as n varies. The final total entropy and number of zero-points do not seem to be correlated to the value of n. Also, the attractiveness of the TFD does not appear to be related to n. Instead, the resulting distributions appear to be somewhat random.

As n increases in the n^{th} maximum entropy loss methods, the time required to perform the algorithm also increases. This correlation occurs for two reasons. First, more iterations are required since less entropy is removed each iteration. In other words, using the 50^{th} largest entropy loss removes less entropy than using the 2^{nd} largest entropy loss. Thus, using the 50^{th} largest entropy loss will require a significantly larger number of iterations to reach the minimum entropy. Secondly, the algorithm must keep track of the n rectangles that provide the greatest loss in entropy. Thus, if n is 50, the algorithm must keep track of the 50 rectangles that have the largest entropy loss. On the other hand, if n is 2, it only needs to keep track of 2 rectangles. The time spent sorting and keeping track of these values becomes significant over thousands of iterations. As a result, using a large value of n may be too inefficient for many applications and is probably not very useful unless the algorithm can be greatly optimized.

4.0 MINIMUM ENTROPY TFDS: DIFFERENT PRIORS

The resulting time-frequency distributions from the previous section do not appear to be very desirable or meaningful. The initial time-frequency distribution that was used in the previous section, $P(n,\omega) = |x(n)|^2 \, |X(\omega)|^2$, is the maximum entropy TFD and makes the fewest assumptions about the signal. As a result, this distribution often loses much of the information from the signal. From the plots of the initial TFDs, it is evident that a significant amount of information is lost before the minimum entropy algorithm begins. If the initial TFD is not a reasonable representation of the signal, it is very unlikely that the final result from the minimum entropy algorithm will be reasonable. In fact, if the final result does appear to represent the signal, it is the result of an extremely lucky rectangle selection pattern.

The prior that is selected for the minimum entropy algorithm affects the properties of the final TFD. First, the final TFD will have the same marginal properties as the initial TFD. Therefore, if it is desired that the final TFD satisfy the time and frequency marginals, the initial TFD must also satisfy the marginals. In addition, the initial TFD must be manifestly positive. The algorithm cannot be performed with negative values because the Shannon entropy does not exist for negative numbers.

This section will explore the effectiveness of the minimum entropy algorithm given different initial TFDs, or priors. The initial TFDs that will be considered are the spectrogram and the Minimum Cross-Entropy Positive TFD (MCE-PTFD) proposed by Loughlin, Pitton, and

Atlas [1]. The results from these priors will be compared to the results found using the maximum entropy prior, $P(n,\omega) = |x(n)|^2\,|X(\omega)|^2$.

4.1 SPECTROGRAM PRIOR

First, we will examine using the spectrogram as the prior for the minimum entropy algorithm. Note that since the spectrogram does not satisfy the time and frequency marginals, the final distribution also will not satisfy the marginals. Also note that using different spectrograms as the initial TFD will produce different results. In the following analysis, a narrowband spectrogram will be used as the initial TFD. As before, we will look at three different signals: a simple chirp signal, a chirp signal with an additional sinusoid, and a double chirp signal. For each signal, the minimum entropy algorithm will be completed using the four previously mentioned brute force rectangle selection methods, plus the maximum entropy loss rectangle selection method. The results will be compared and contrasted with one another and the results from previous sections.

4.1.1 Simple Chirp Signal

To examine using the spectrogram as the prior, we will first look at the simple chirp signal. As before, the simple chirp signal is defined as:

$$x(n) = c*e^{j\pi n^2/(2N)}$$

where c is the normalization constant and N is the length of the signal which is 32 in this example. The initial TFD, $P(n,\omega)$, is defined as the narrowband spectrogram of x(n). The initial spectrogram TFD has a total entropy of 5.1572 and 0 zero-points. Figure 4.1 shows the narrowband spectrogram that will be used as the prior.

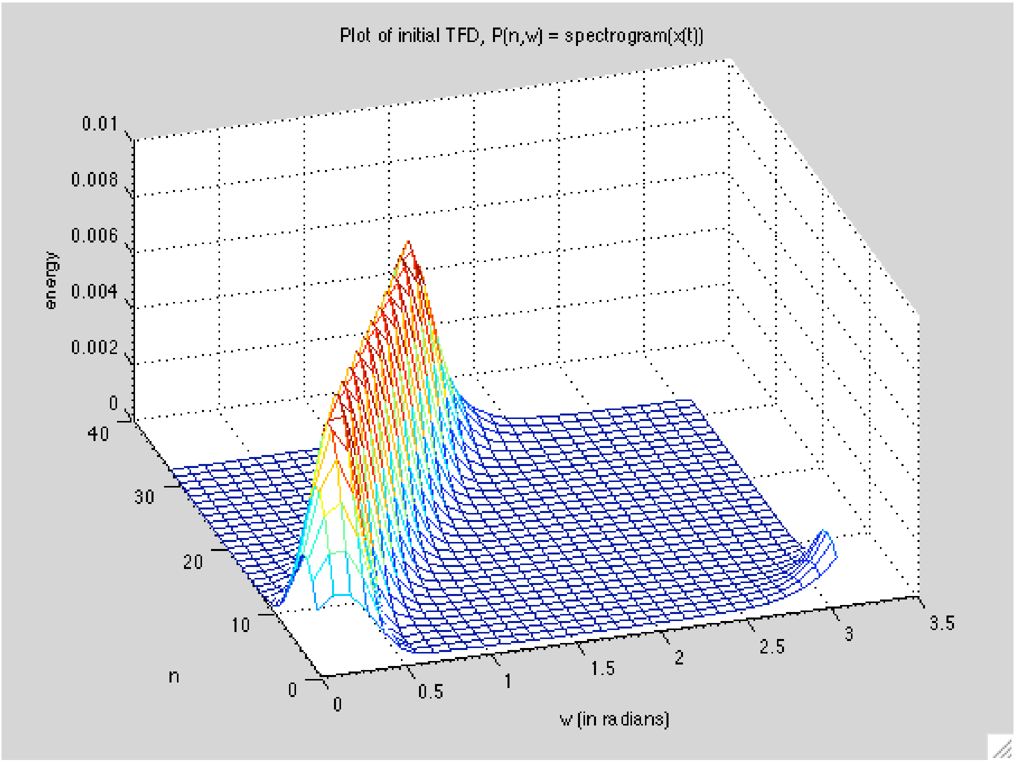

Figure 4.1: Plot of the initial spectrogram TFD for the simple chirp signal

The minimum entropy algorithm was performed using the following rectangle selection methods: top-left method, top-right method, bottom-left method, bottom-right method, and maximum entropy loss method. Table 4.1 shows the final total entropy and number of zero-points for each method. Figure 4.2 shows the minimum entropy TFD obtained using the top-left rectangle selection method. Figure 4.3 shows the minimum entropy TFD obtained using the top-right rectangle selection method. Figure 4.4 shows the minimum entropy TFD obtained using the bottom-left rectangle selection method. Figure 4.5 shows the minimum entropy TFD obtained using the bottom-right rectangle selection method. Figure 4.6 shows the minimum entropy TFD obtained using the maximum entropy loss rectangle selection method.

From the results, we can see that the final TFDs are much better representations of the simple chirp signal than the ones obtained using the maximum entropy TFD as the prior. The bottom-right and maximum entropy loss methods appear the produce the "best" results. It is evident that using a prior that has more of the signal information produces more meaningful results with the minimum entropy algorithm.

Table 4.1: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the narrowband spectrogram as the prior for the simple chirp signal using several different rectangle selection methods.

Method	Total Entropy	Zero-Points
Top-Left	3.4688	961
Top-Right	3.4618	961
Bottom-Left	3.4436	961
Bottom-Right	3.4654	961
Maximum Entropy Loss	3.4393	961

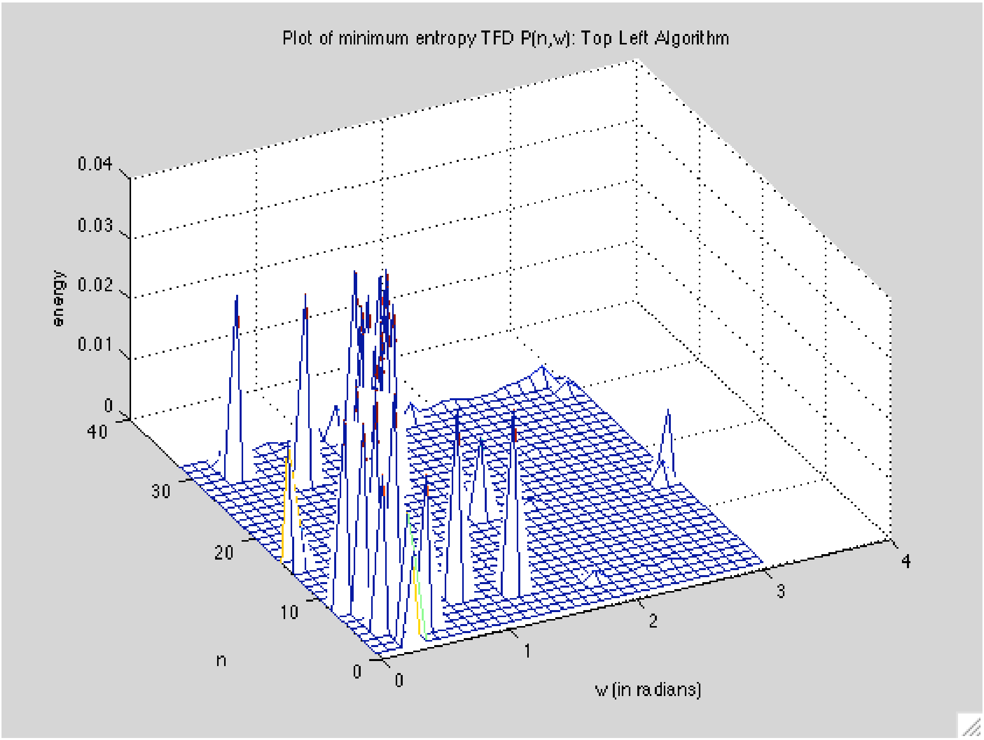

Figure 4.2: Plot of the minimum entropy TFD obtained using the narrowband spectrogram as the prior and the top-left rectangle selection method for the simple chirp signal

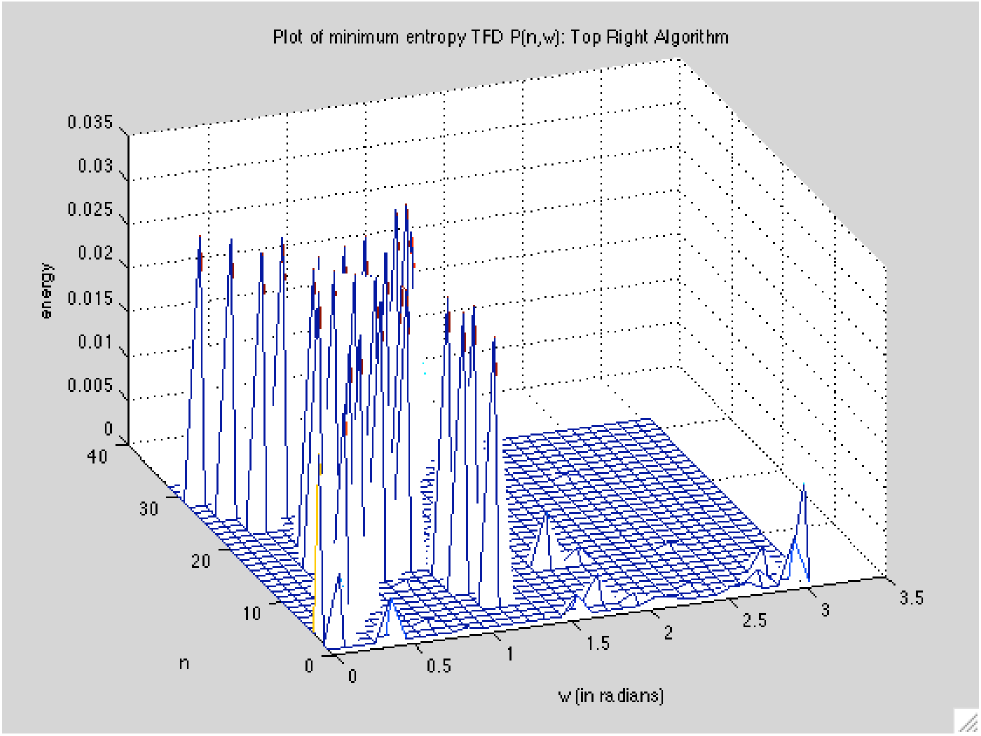

Figure 4.3: Plot of the minimum entropy TFD obtained using the narrowband spectrogram as the prior and the top-right rectangle selection method for the simple chirp signal

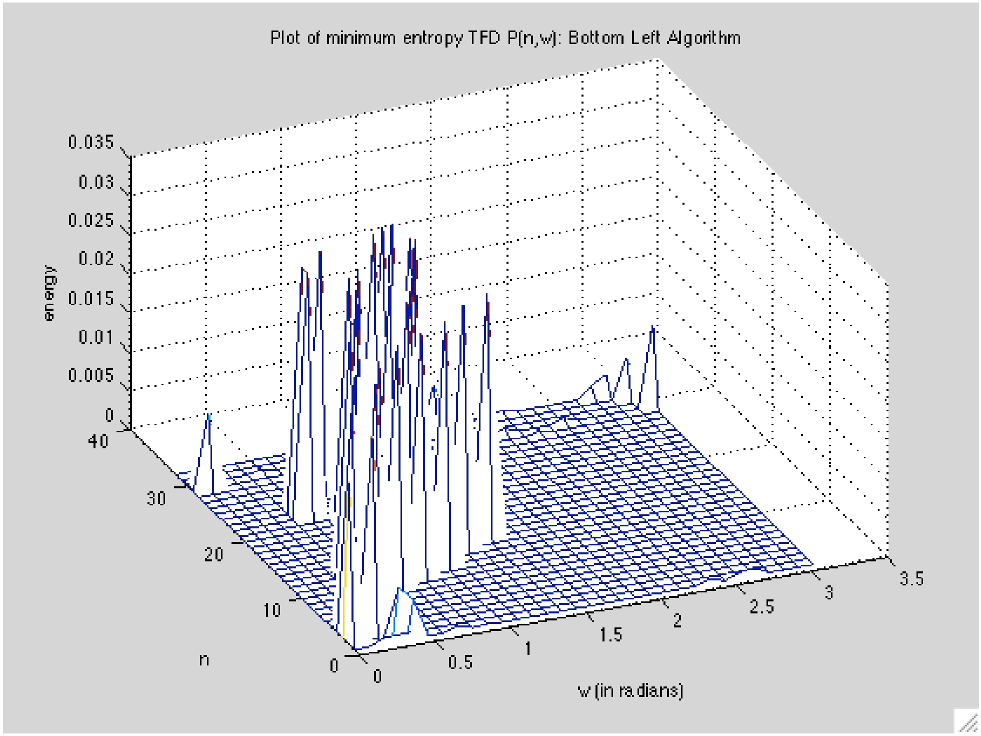

Figure 4.4: Plot of the minimum entropy TFD obtained using the narrowband spectrogram as the prior and the bottom-left rectangle selection method for the simple chirp signal

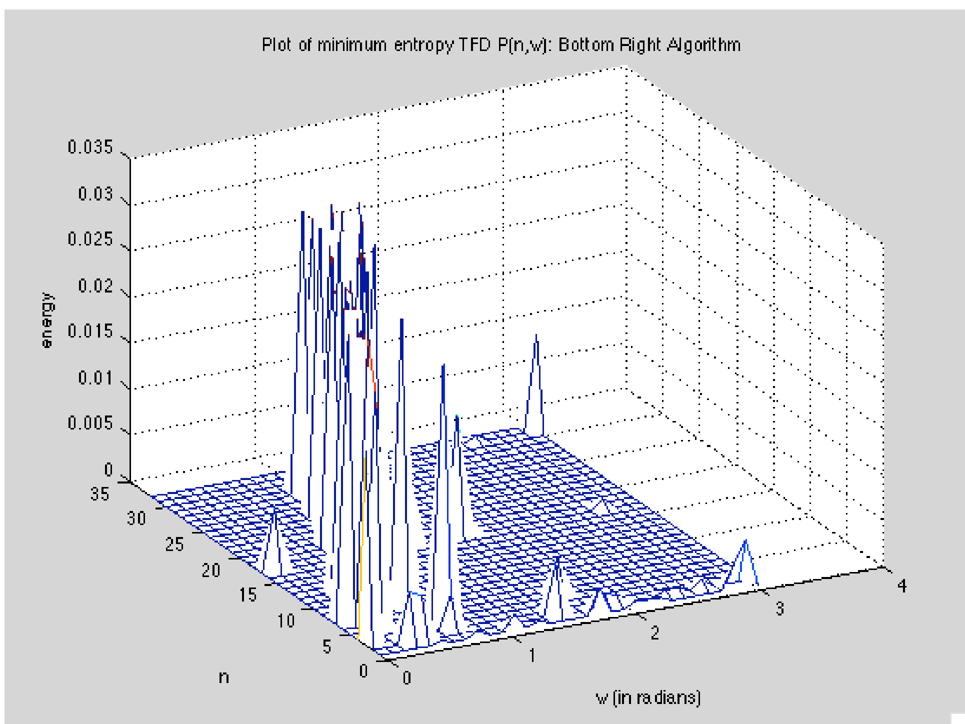

Figure 4.5: Plot of the minimum entropy TFD obtained using the narrowband spectrogram as the prior and the bottom-right rectangle selection method for the simple chirp signal

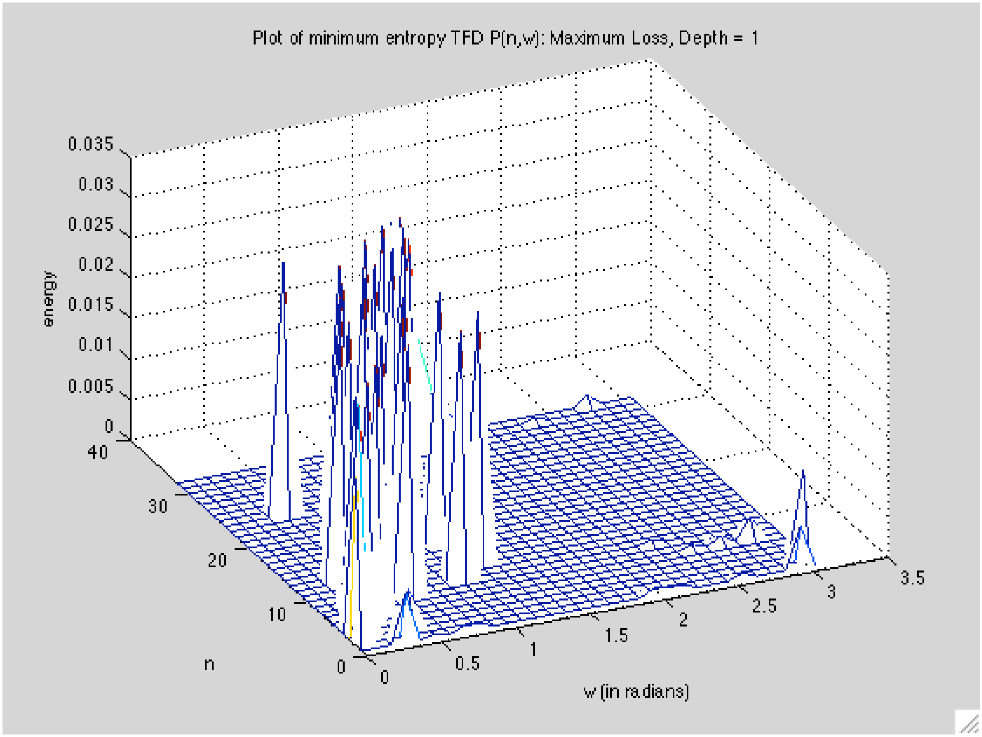

Figure 4.6: Plot of the minimum entropy TFD obtained using the narrowband spectrogram as the prior and the maximum entropy loss rectangle selection method for the simple chirp signal

4.1.2 Simple Chirp with Sinusoid Signal

Next, we will first look at the chirp with sinusoid signal. As before, the chirp with sinusoid signal is defined as:

$$x(n) = c*(e^{j\pi n\wedge 2/(2N)} + e^{j5\pi n/4})$$

where c is the normalization constant and N is the length of the signal which is 32 in this example. The initial TFD, P(n,ω), is defined as the narrowband spectrogram of x(n). The initial spectrogram TFD has a total entropy of 5.5954 and 0 zero-points. Figure 4.7 shows the narrowband spectrogram that will be used as the prior for the chirp with sinusoid signal.

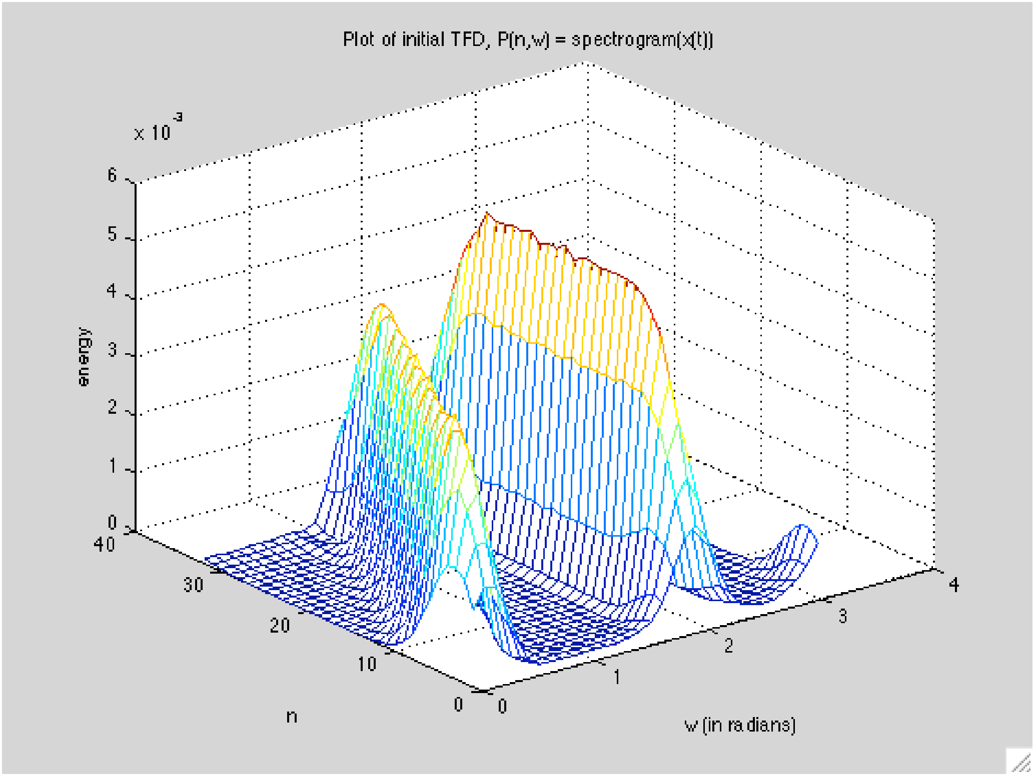

Figure 4.7: Plot of the initial spectrogram TFD for the chirp with sinusoid signal

Again, the minimum entropy algorithm was performed using the following rectangle selection methods: top-left method, top-right method, bottom-left method, bottom-right method, and maximum entropy loss method. Table 4.2 shows the final total entropy and number of zero-points for each method. Figure 4.8 shows the ME-TFD obtained using the top-left rectangle selection method. Figure 4.9 shows the ME-TFD obtained using the top-right rectangle selection method. Figure 4.10 shows the ME-TFD obtained using the bottom-left rectangle selection method. Figure 4.11 shows the ME-TFD obtained using the bottom-right rectangle selection method. Figure 4.12 shows the ME-TFD obtained using the maximum entropy loss rectangle selection method.

The results are once again much better than the ones obtained using the maximum entropy TFD as the prior. The bottom-left and bottom-right rectangle selection methods produced particularly "good" results. The maximum entropy loss method was not as accurate as the brute force methods.

Table 4.2: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the narrowband spectrogram as the prior for the chirp with sinusoid signal using several different rectangle selection methods.

Method	Total Entropy	Zero-Points
Top-Left	3.4482	961
Top-Right	3.4627	961
Bottom-Left	3.4654	961
Bottom-Right	3.4527	961
Maximum Entropy Loss	3.4529	961

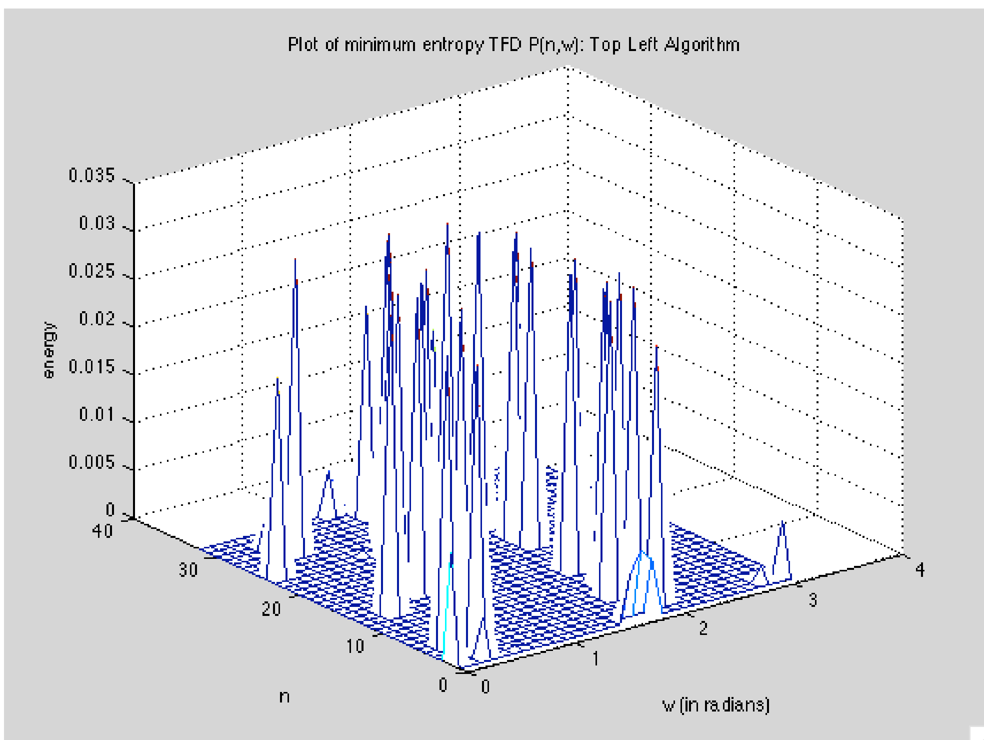

Figure 4.8: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the top-left rectangle selection method for the chirp with sinusoid signal

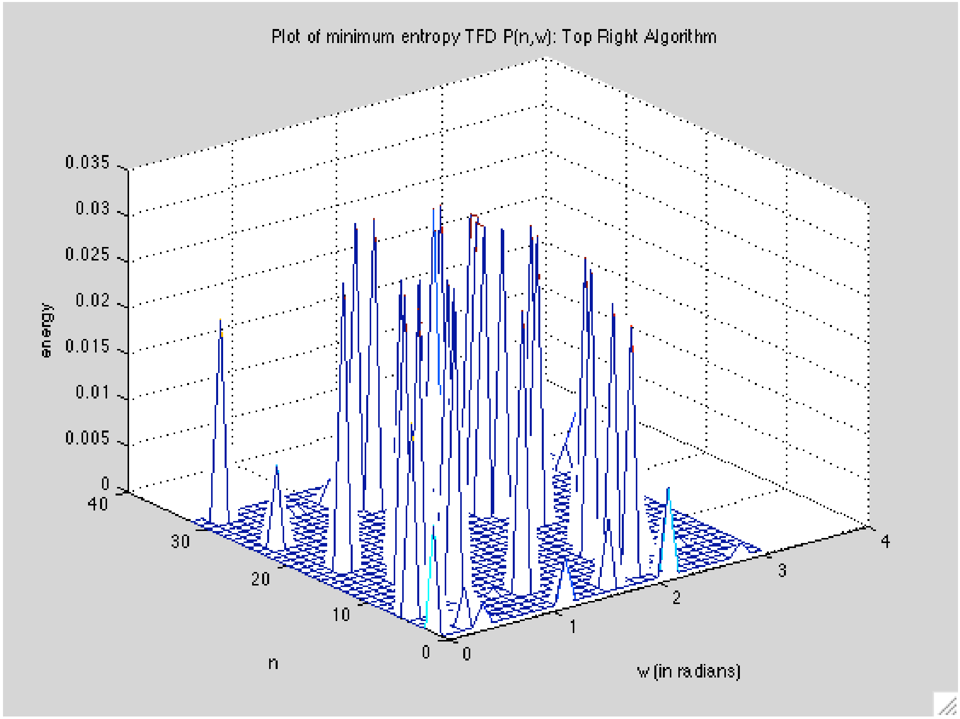

Figure 4.9: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the top-right rectangle selection method for the chirp with sinusoid signal

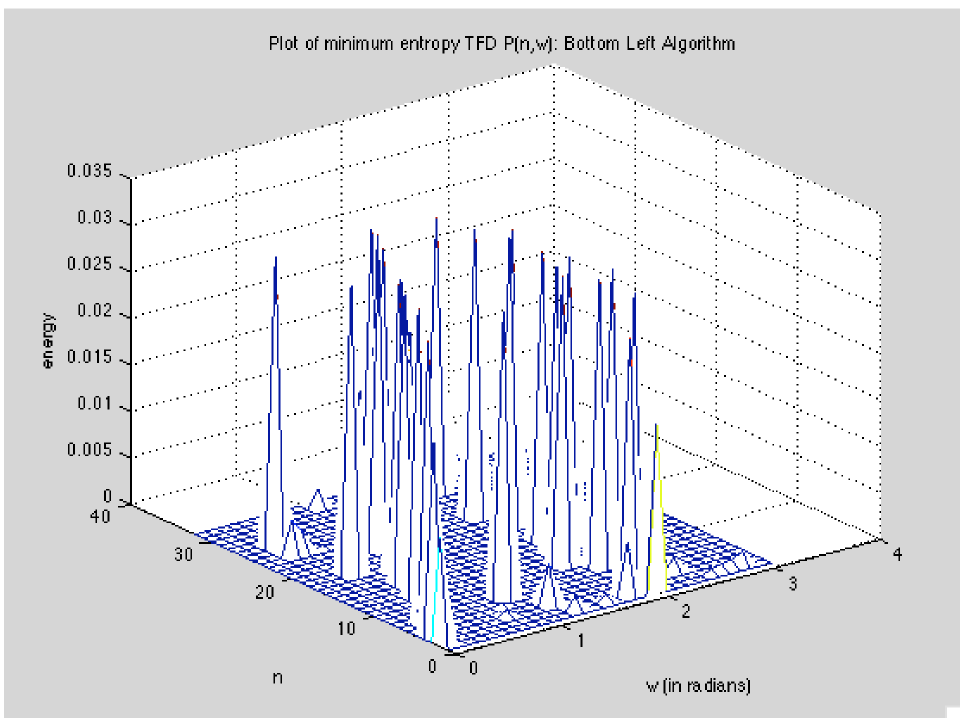

Figure 4.10: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the bottom-left rectangle selection method for the chirp with sinusoid signal

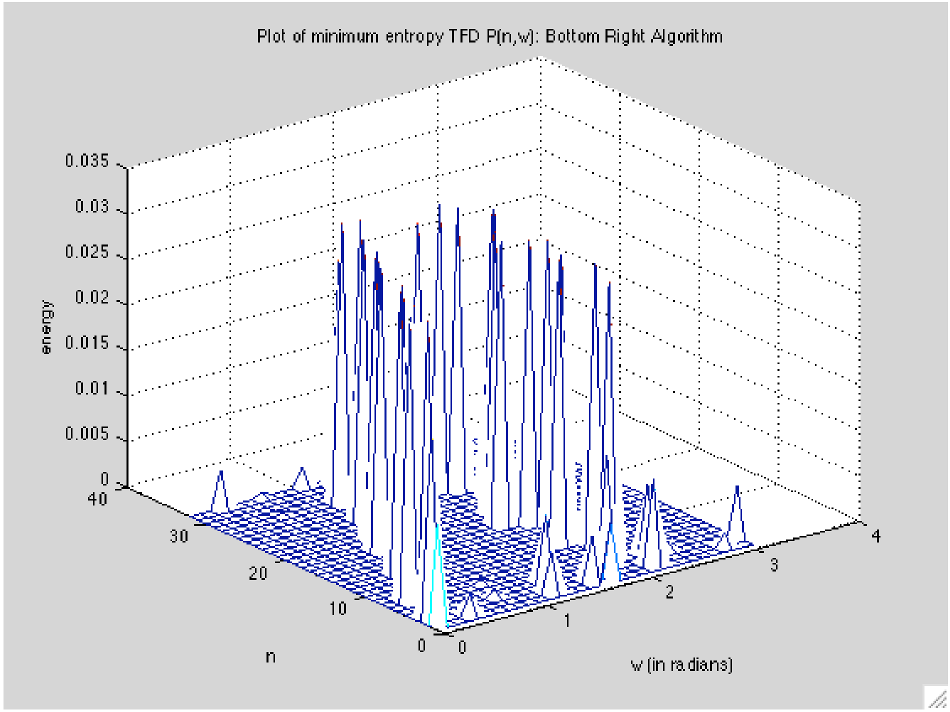

Figure 4.11: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the bottom-right rectangle selection method for the chirp with sinusoid signal

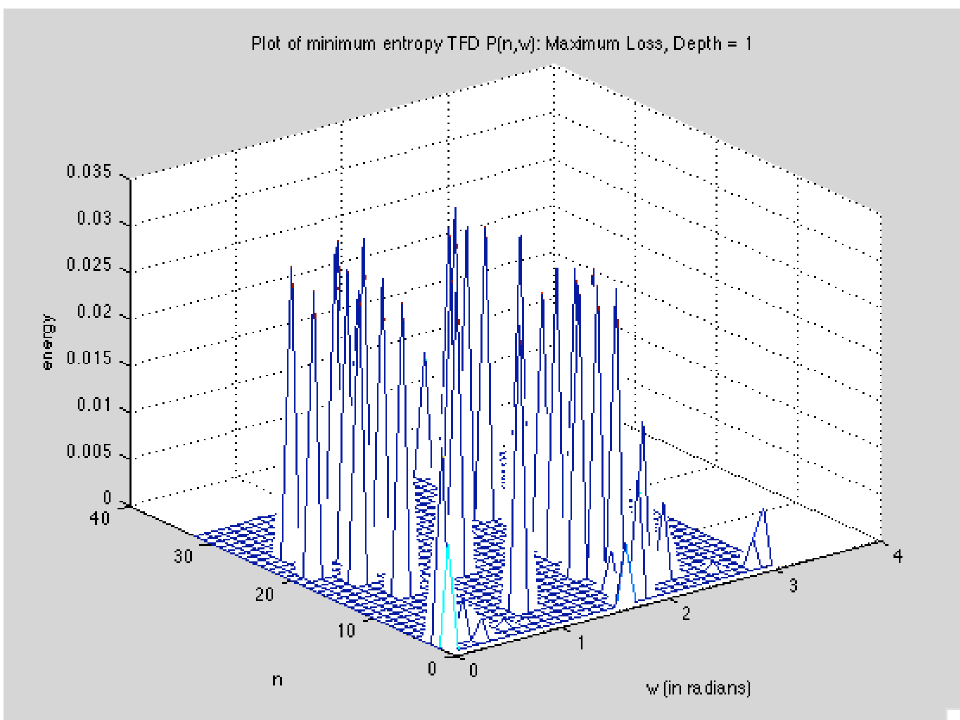

Figure 4.12: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the maximum entropy loss rectangle selection method for the chirp with sinusoid signal

4.1.3 Double Chirp Signal

Finally, we will first look at the double chirp signal. As before, the double chirp signal is defined as:

$$x(n) = c^*(e^{j\pi n^\wedge 2/(2N)} + e^{j\pi(2\pi - n)^\wedge 2/(2N)})$$

where c is the normalization constant and N is the length of the signal which is 32 in this example. The initial TFD, P(n,ω), is defined as the narrowband spectrogram of x(n). The initial spectrogram TFD has a total entropy of 5.4956 and 0 zero-points. Figure 4.13 shows the narrowband spectrogram that will be used as the prior for the double chirp signal.

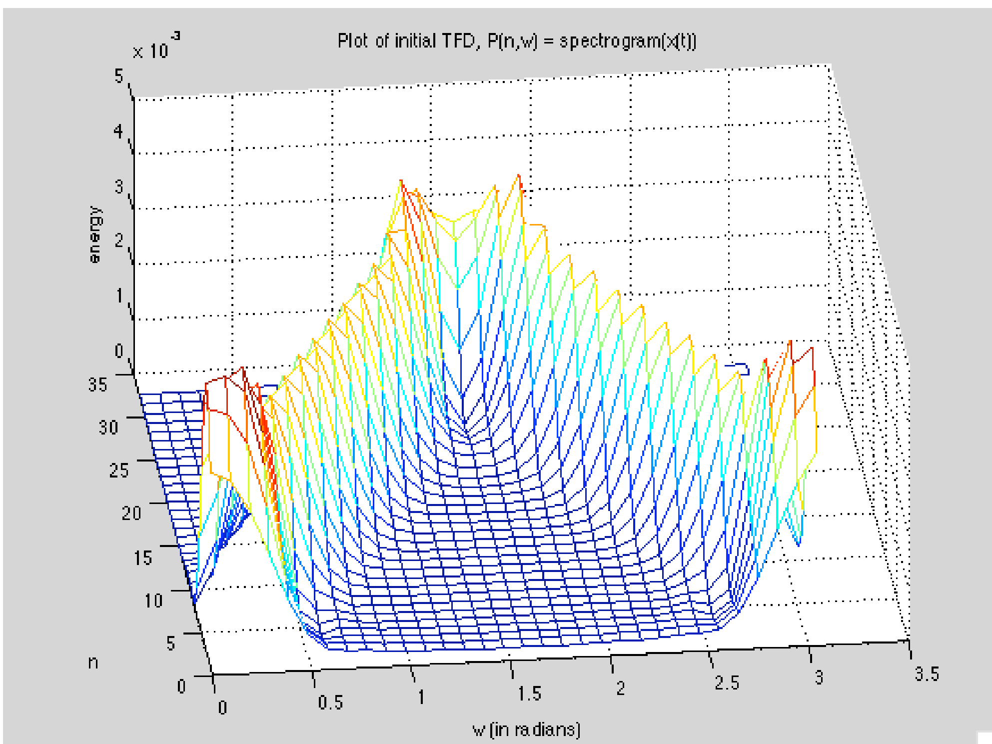

Figure 4.13: Plot of the initial spectrogram TFD for the double chirp signal

Once again, the minimum entropy algorithm was performed using the following rectangle selection methods: top-left method, top-right method, bottom-left method, bottom-right method, and maximum entropy loss method. Table 4.3 shows the final total entropy and number of zero-points for each method. Figure 4.14 shows the ME-TFD obtained using the top-left rectangle selection method. Figure 4.15 shows the ME-TFD obtained using the top-right rectangle selection method. Figure 4.16 shows the ME-TFD obtained using the bottom-left rectangle selection method. Figure 4.17 shows the ME-TFD obtained using the bottom-right rectangle selection method. Figure 4.18 shows the ME-TFD obtained using the maximum entropy loss rectangle selection method.

As with the other two signals, the spectrogram prior produced better results for several of the rectangle selection methods. The bottom-left and bottom-right methods seemed to produce the best results out of all of the different rectangle selection methods.

Table 4.3: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the narrowband spectrogram as the prior for the double chirp signal using several different rectangle selection methods.

Method	Total Entropy	Zero-Points
Top-Left	3.4012	961
Top-Right	3.3752	961
Bottom-Left	3.3858	961
Bottom-Right	3.3808	961
Maximum Entropy Loss	3.3673	961

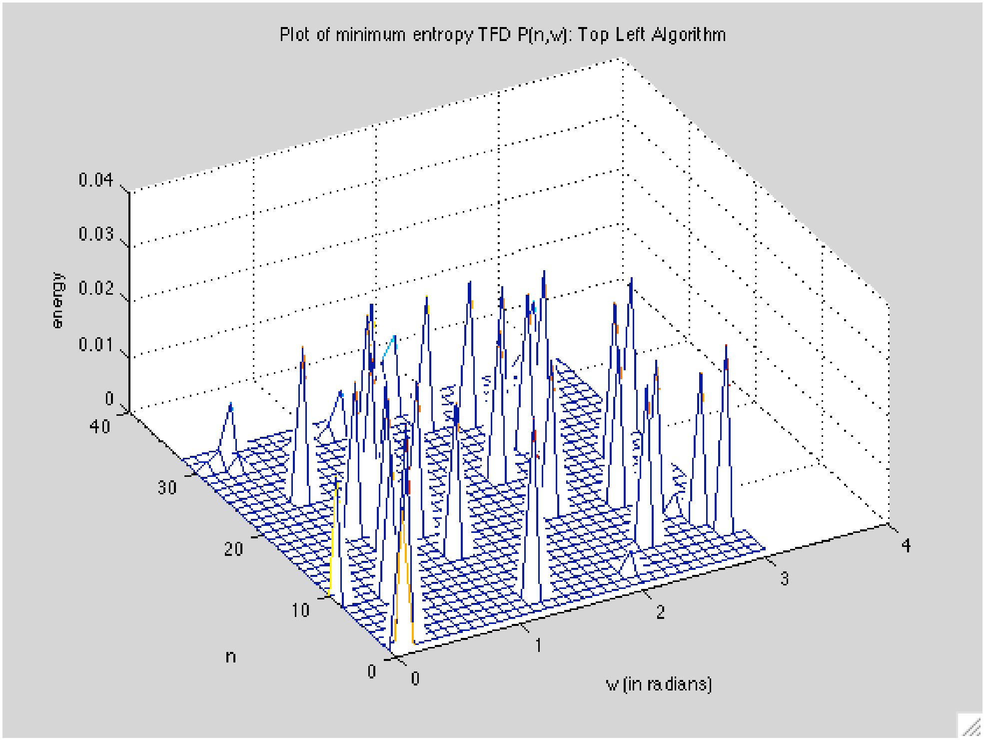

Figure 4.14: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the top-left rectangle selection method for the double chirp signal

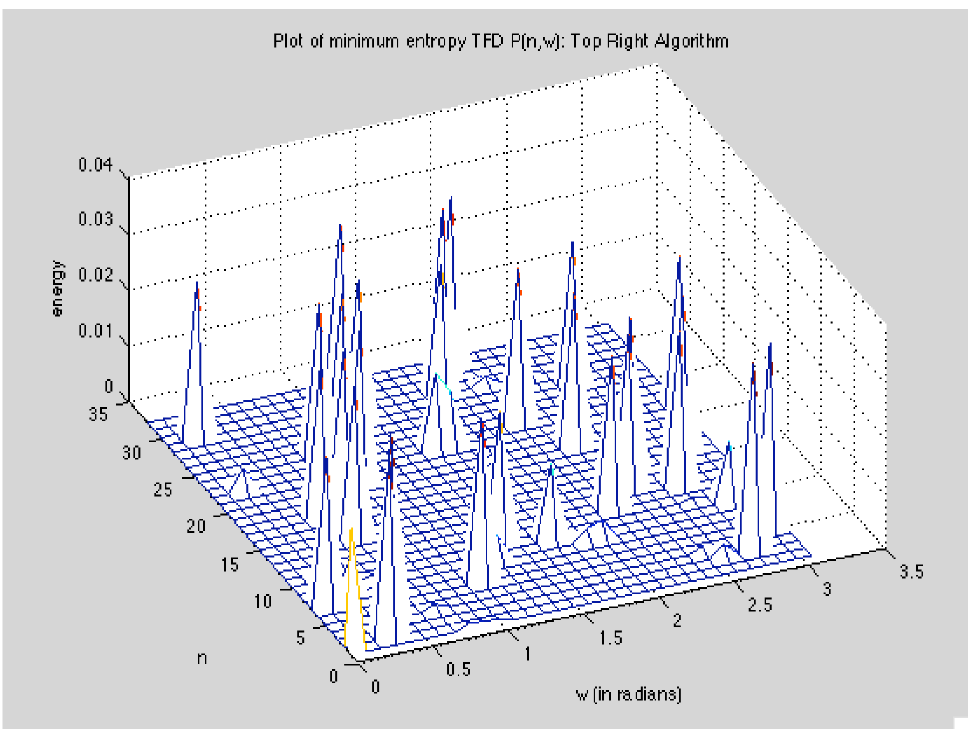

Figure 4.15: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the top-right rectangle selection method for the double chirp signal

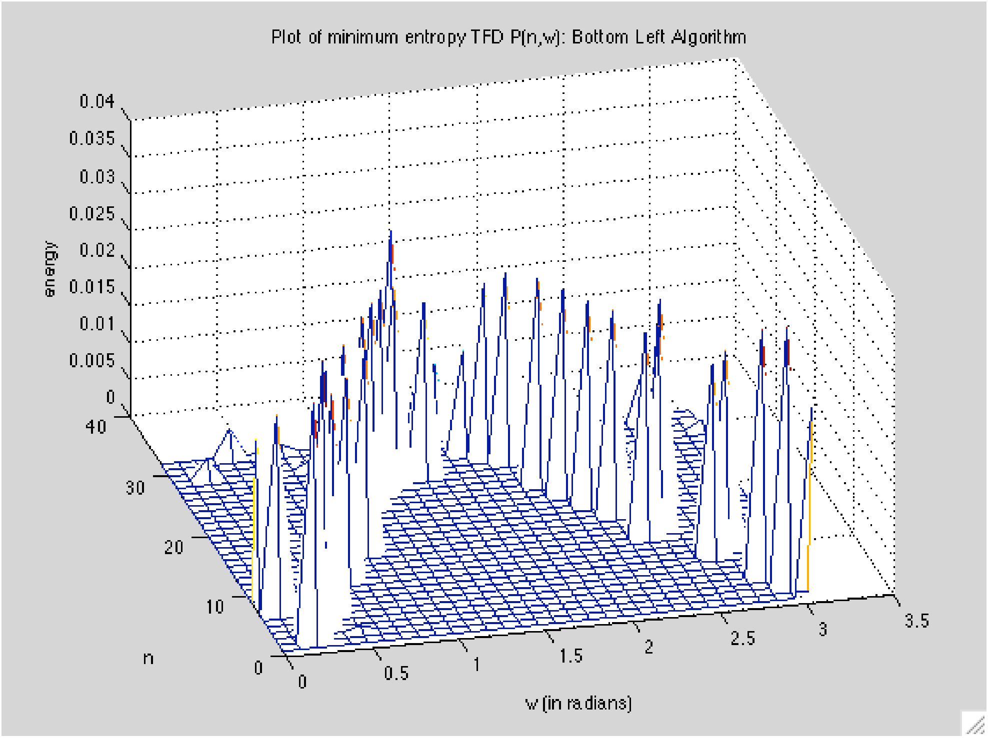

Figure 4.16: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the bottom-left rectangle selection method for the double chirp signal

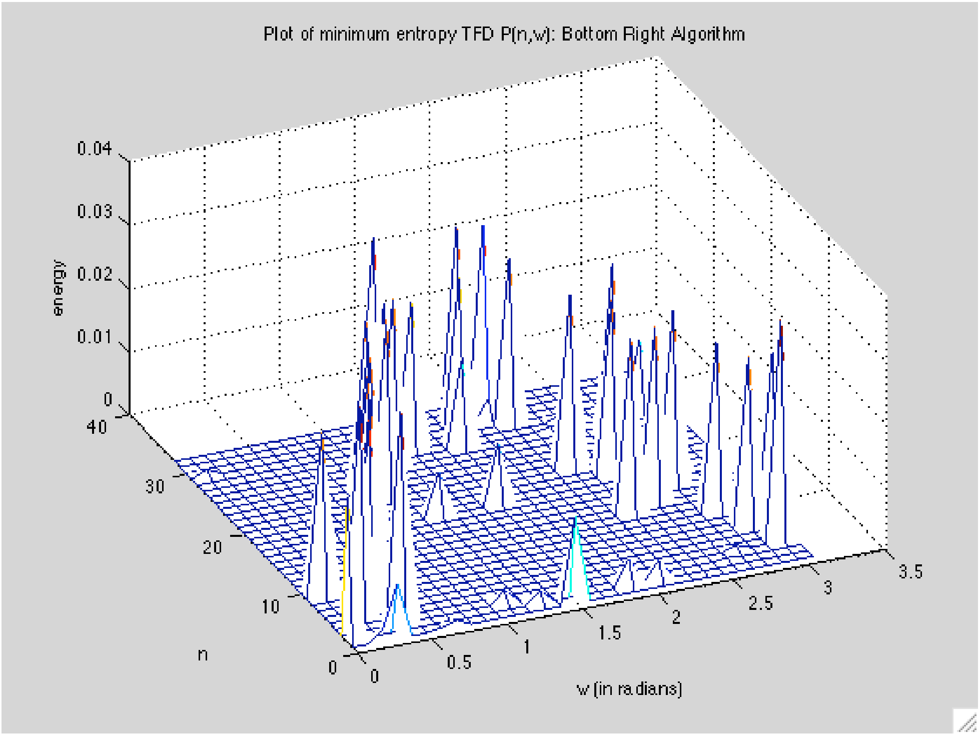

Figure 4.17: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the bottom-right rectangle selection method for the double chirp signal

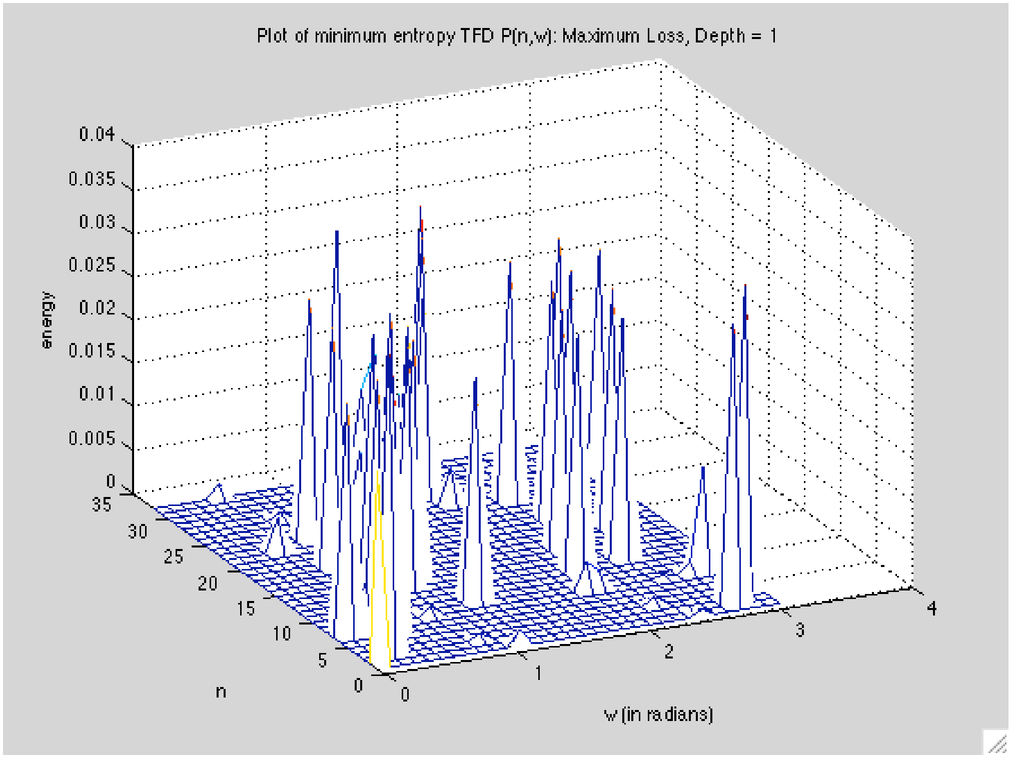

Figure 4.18: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the maximum entropy loss rectangle selection method for the double chirp signal

4.1.4 Summary

Overall, the results from the maximum entropy algorithm with spectrogram prior appear to be more meaningful than the results obtained using the maximum entropy prior for several of the different rectangle selection methods. For each signal, the bottom-left and bottom-right methods appear to give reasonable results. Conversely, the top-left and top-right methods do not appear to give desirable results. At this point, the reason for the bottom-left and bottom-right methods producing more desirable results is unknown. It could either be due to chance or some unknown underlying factor. The maximum entropy loss and n^{th} maximum entropy loss (which are not included in this paper) methods do not appear to provide meaningful results.

As previously mentioned, the resulting minimum entropy TFDs obtained using the spectrogram prior do not satisfy the time or frequency marginals. If satisfying the marginals is necessary, a different prior must be used.

4.2 MCE-PTFD PRIOR

Next, we will examine using the Minimum Cross-Entropy Positive TFD (MCE-PTFD) proposed by Loughlin, Atlas, and Pitton [1] as the prior for the Minimum Entropy algorithm. Unlike the spectrogram, the MCE-PTFD satisfies the time and frequency marginals. As a result, the final distribution will also satisfy the marginals. In the following analysis, a MCE-PTFD will be used as the initial TFD. As before, we will look at three different signals: a simple chirp signal, a chirp signal with an additional sinusoid, and a double chirp signal. For each signal, the minimum entropy algorithm will be completed using the four previously mentioned brute force rectangle selection methods, plus the maximum entropy loss rectangle selection method. The results will be compared and contrasted with one another and the results from previous sections.

4.2.1 Simple Chirp Signal

To examine using the MCE-PTFD as the prior, we will first look at the simple chirp signal. As before, the simple chirp signal is defined as:

$$x(n) = c*e^{j\pi n^2/(2N)}$$

where c is the normalization constant and N is the length of the signal which is 32 for this example. The initial TFD, $P(n,\omega)$, is defined as the MCE-PTFD of x(n). The initial TFD has a total entropy of 5.4648 and 0 zero-points. Figure 4.19 shows the MCE-PTFD that will be used as the prior.

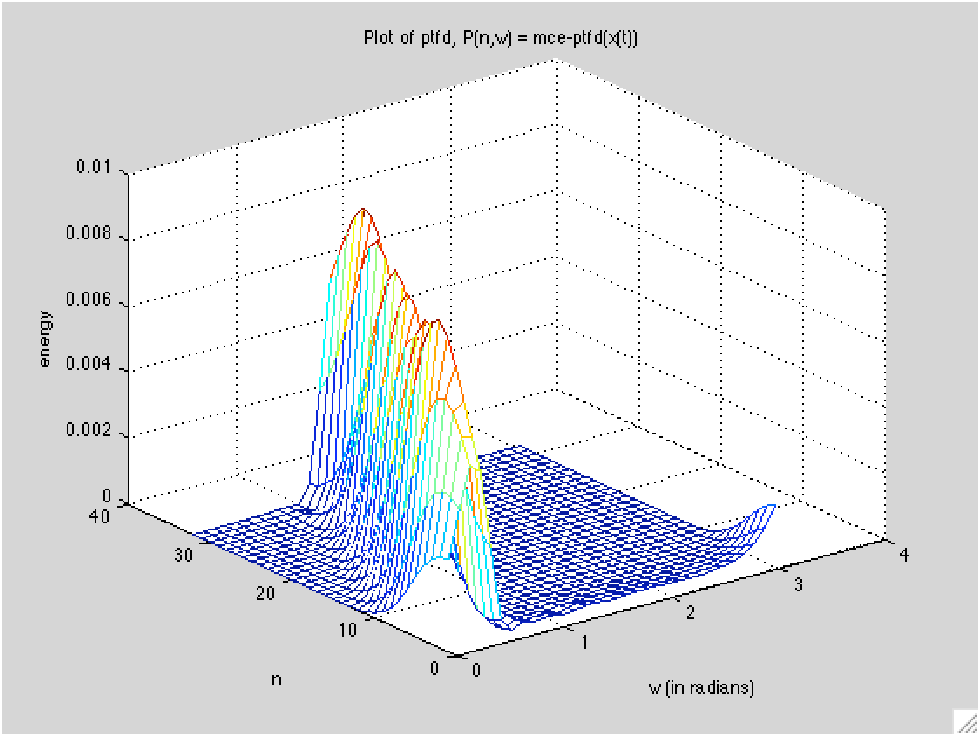

Figure 4.19: Plot of the initial MCE-PTFD prior for the simple chirp signal

The Minimum Entropy algorithm was performed using the following rectangle selection methods: top-left method, top-right method, bottom-left method, bottom-right method, and maximum entropy loss method. Table 4.4 shows the final total entropy and number of zero-points for each method. Figure 4.20 shows the ME-TFD obtained using the top-left rectangle selection method. Figure 4.21 shows the ME-TFD obtained using the top-right rectangle selection method. Figure 4.22 shows the ME-TFD obtained using the bottom-left rectangle selection method. Figure 4.23 shows the ME-TFD obtained using the bottom-right rectangle selection method. Figure 4.24 shows the ME-TFD obtained using the maximum entropy loss rectangle selection method.

From the figures below, it appears that the results are better than the results obtained using the maximum entropy prior, but do not appear to be as good as the results found using the spectrogram prior. One significant advantage for the results obtained using the MCE-PTFD prior is the final TFD satisfies the marginals where the spectrogram results do not.

Table 4.4: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the simple chirp signal using several different rectangle selection methods.

Method	Total Entropy	Zero-Points
Top-Left	3.6977	961
Top-Right	3.6990	962
Bottom-Left	3.7005	961
Bottom-Right	3.6934	961
Maximum Entropy Loss	3.6993	961

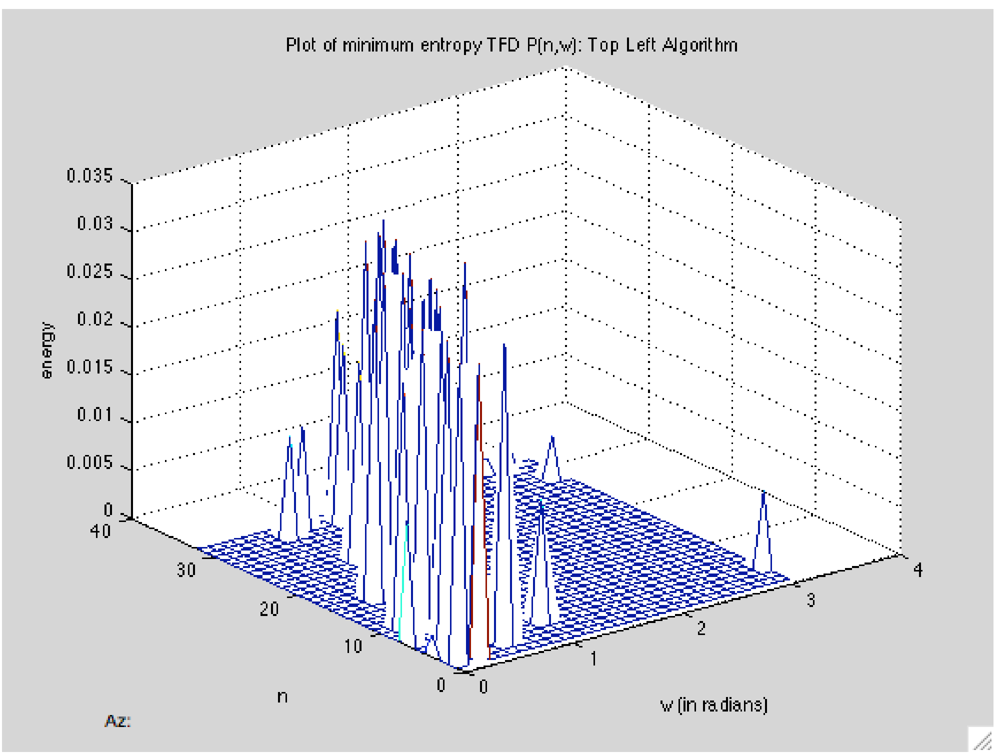

Figure 4.20: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the top-left rectangle selection method for the simple chirp signal

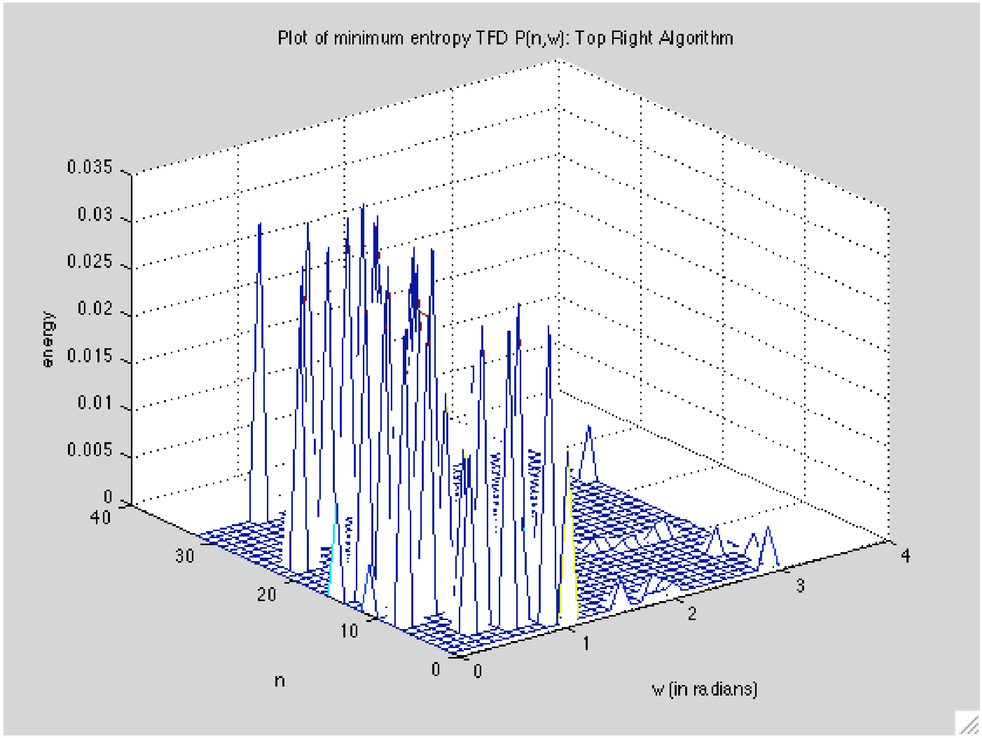

Figure 4.21: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the top-right rectangle selection method for the simple chirp signal

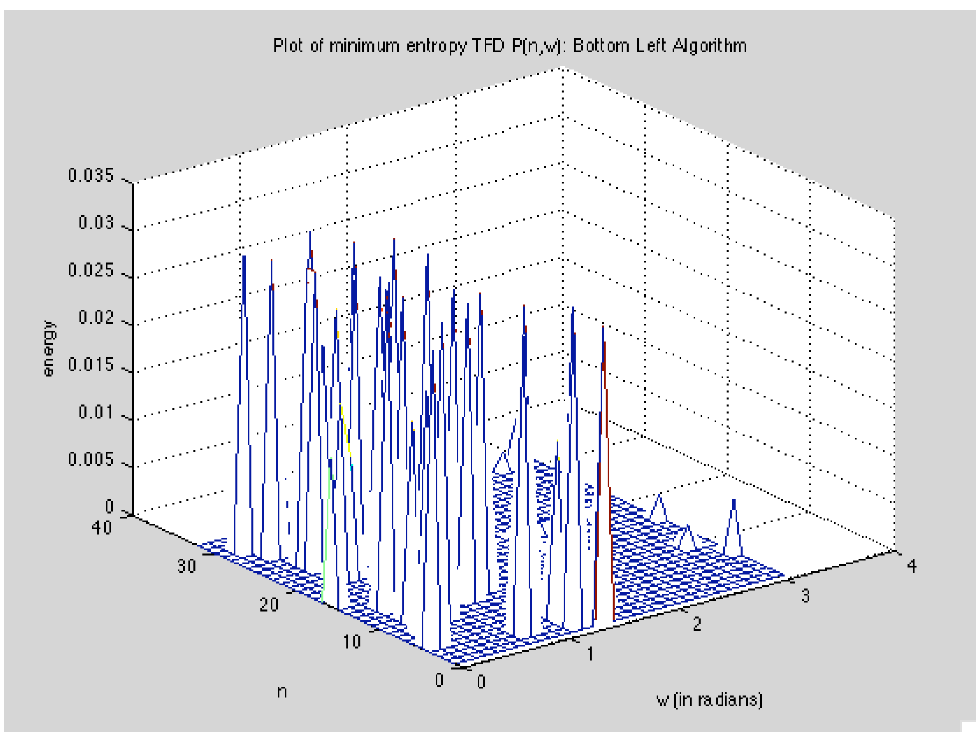

Figure 4.22: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the bottom-left rectangle selection method for the simple chirp signal

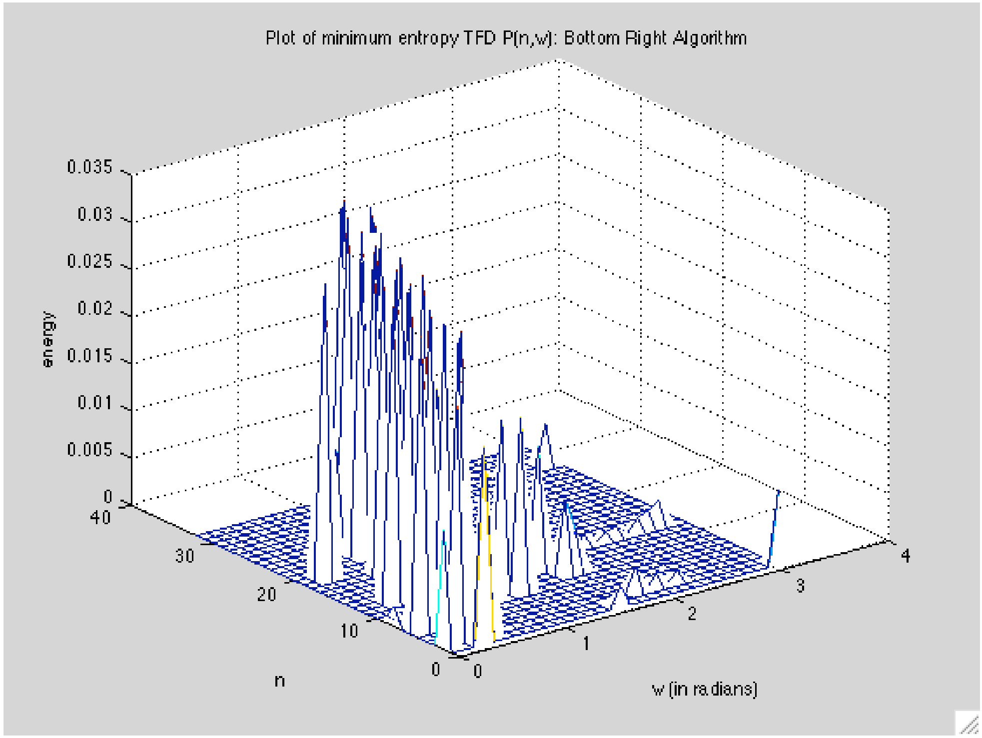

Figure 4.23: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the bottom-right rectangle selection method for the simple chirp signal

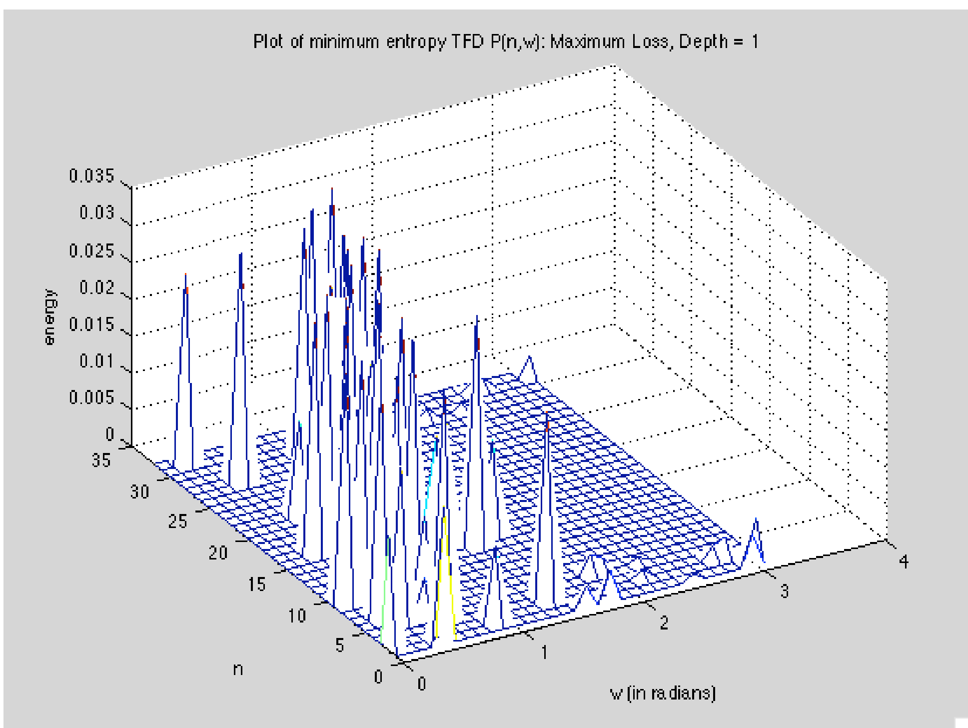

Figure 4.24: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the maximum entropy loss rectangle selection method for the simple chirp signal

4.2.2 Simple Chirp with Sinusoid Signal

To further examine using the MCE-PTFD as the prior, we will look at the simple chirp with sinusoid signal. As before, the simple chirp with sinusoid signal is defined as:

$$x(n) = c^*(e^{j\pi n^2/(2N)} + e^{j5\pi n/4})$$

where c is the normalization constant and N is the length of the signal which is 32 for this example. The initial TFD, $P(n,\omega)$, is defined as the MCE-PTFD of x(n). The initial TFD has a total entropy of 4.8347 and 64 zero-points. Figure 4.25 shows the MCE-PTFD that will be used as the prior.

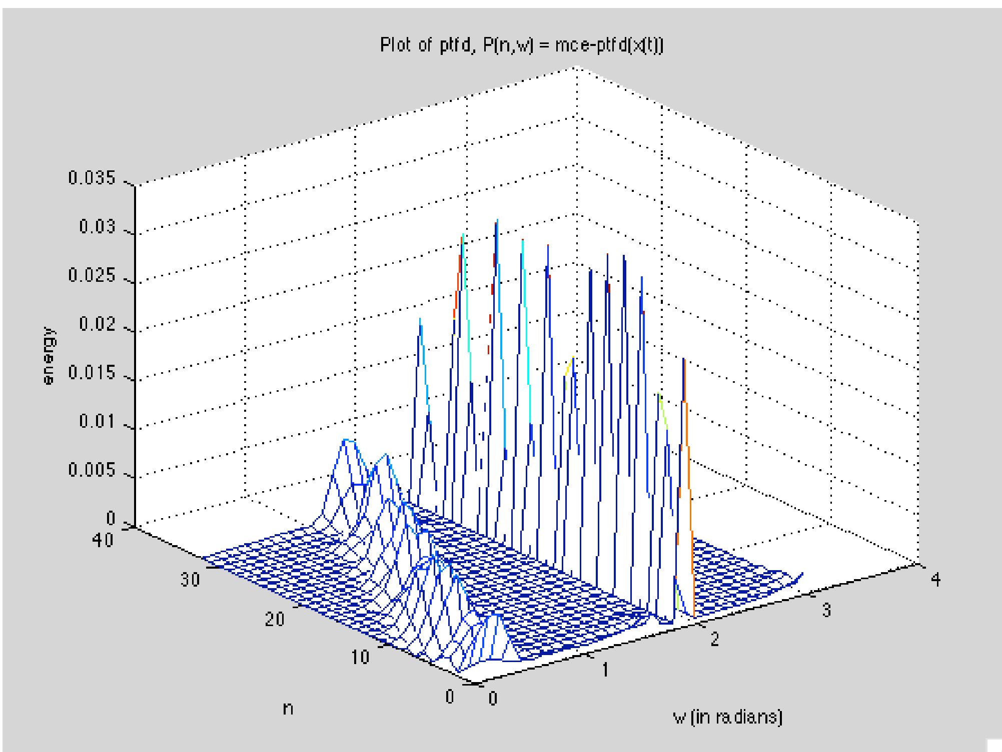

Figure 4.25: Plot of the initial MCE-PTFD prior for the simple chirp with sinusoid signal

The Minimum Entropy algorithm was performed using the following rectangle selection methods: top-left method, top-right method, bottom-left method, bottom-right method, and maximum entropy loss method. Table 4.5 shows the final total entropy and number of zero-points for each method. Figure 4.26 shows the ME-TFD obtained using the top-left rectangle selection method. Figure 4.27 shows the ME-TFD obtained using the top-right rectangle selection method. Figure 4.28 shows the ME-TFD obtained using the bottom-left rectangle selection method. Figure 4.29 shows the ME-TFD obtained using the bottom-right rectangle selection method. Figure 4.30 shows the ME-TFD obtained using the maximum entropy loss rectangle selection method.

From the figures below, it appears that the results are slightly better than the results obtained using the maximum entropy prior, but do not appear to be nearly as good as the results found using the spectrogram prior although they satisfy the marginals.

Table 4.5: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the simple chirp with sinusoid signal using several different rectangle selection methods.

Method	Total Entropy	Zero-Points
Top-Left	3.3006	963
Top-Right	3.3395	963
Bottom-Left	3.3052	963
Bottom-Right	3.3422	963
Maximum Entropy Loss	3.3148	963

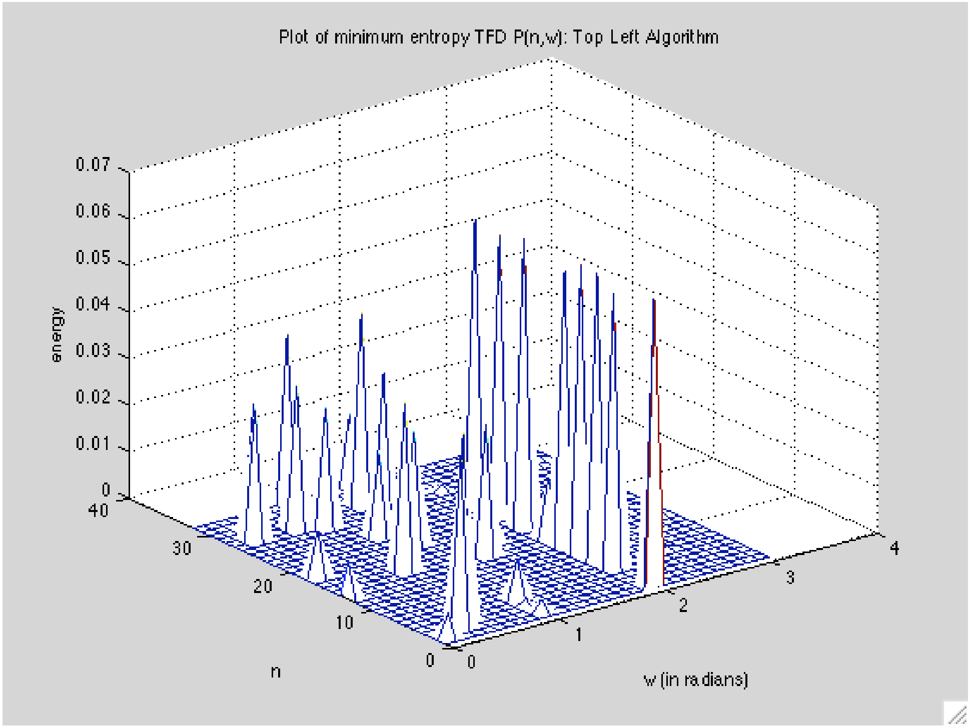

Figure 4.26: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the top-left rectangle selection method for the simple chirp with sinusoid signal

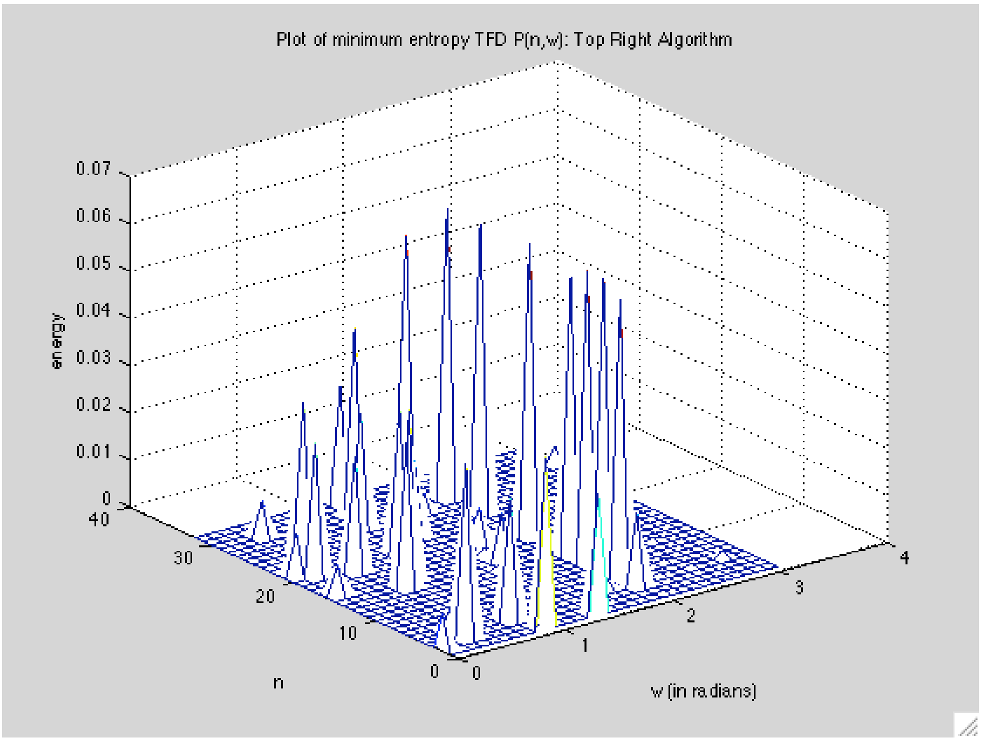

Figure 4.27: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the top-right rectangle selection method for the simple chirp with sinusoid signal

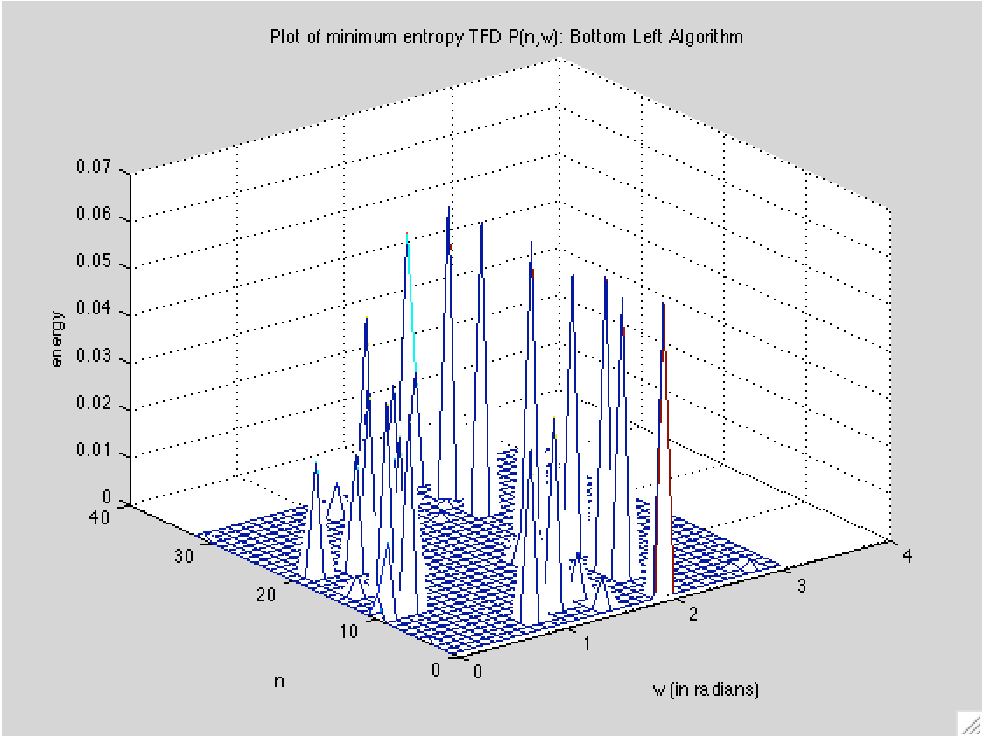

Figure 4.28: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the bottom-left rectangle selection method for the simple chirp with sinusoid signal

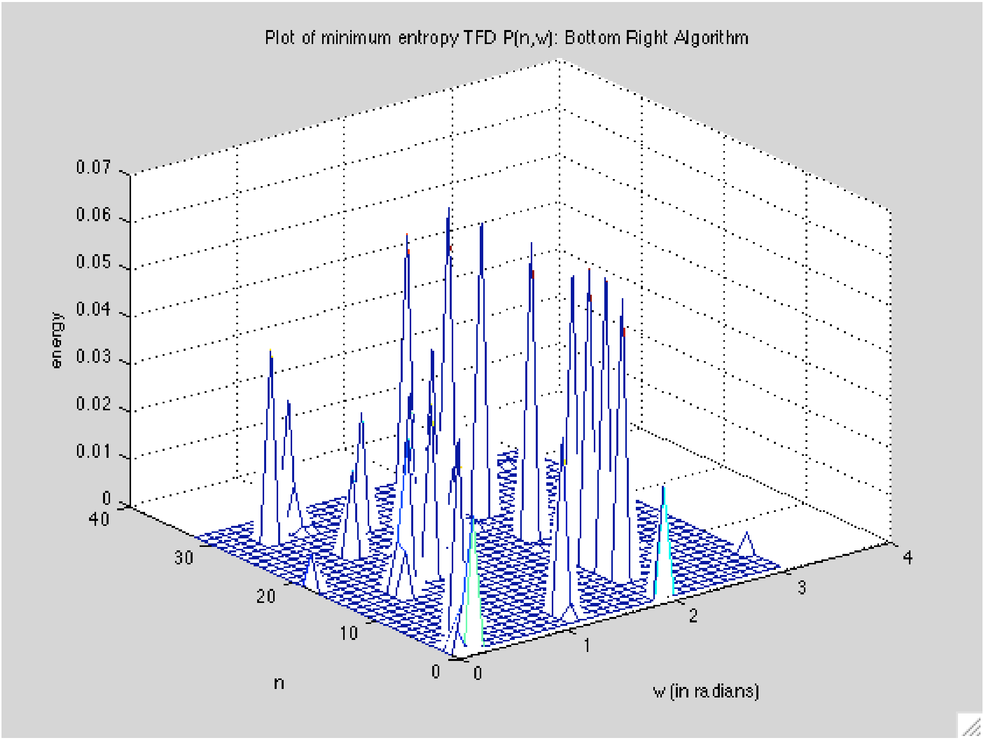

Figure 4.29: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the bottom-right rectangle selection method for the simple chirp with sinusoid signal

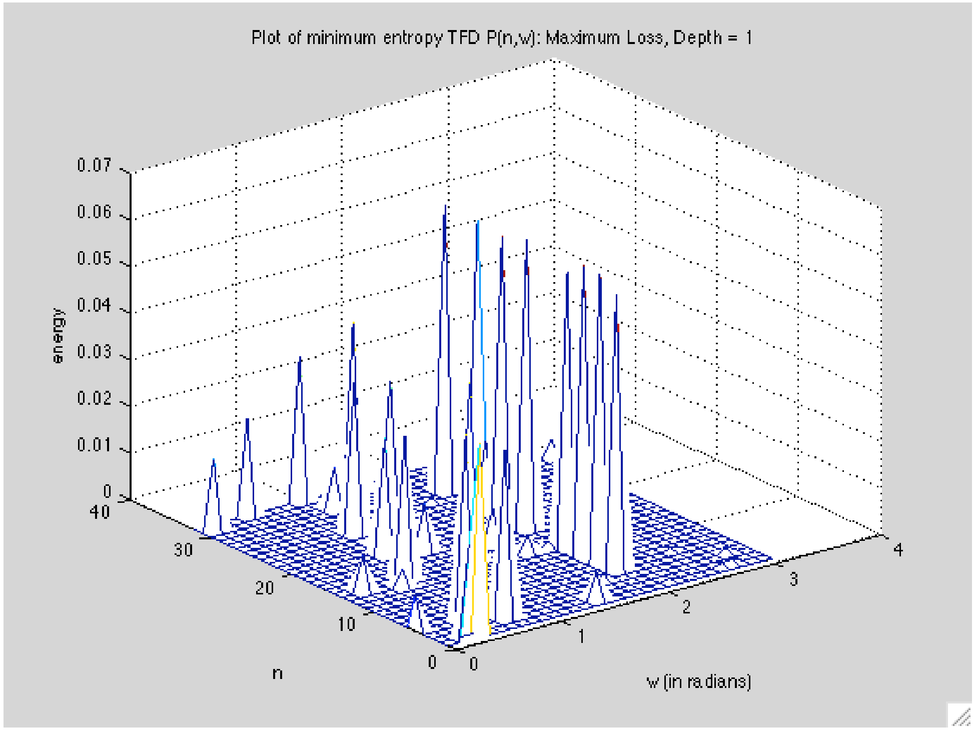

Figure 4.30: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the maximum entropy loss rectangle selection method for the simple chirp with sinusoid signal

4.2.3 Double Chirp Signal

Finally, we will look at the double chirp signal. As before, the double chirp signal is defined as:

$$x(n) = c*(e^{j\pi n^{\wedge}2/(2N)} + e^{j\pi(2\pi - n)^{\wedge}2/(2N)})$$

where c is the normalization constant and N is the length of the signal which is 32 for this example. The initial TFD, P(n,ω), is defined as the MCE-PTFD of x(n). The initial TFD has a total entropy of 5.7037 and 0 zero-points. Figure 4.31 shows the MCE-PTFD that will be used as the prior.

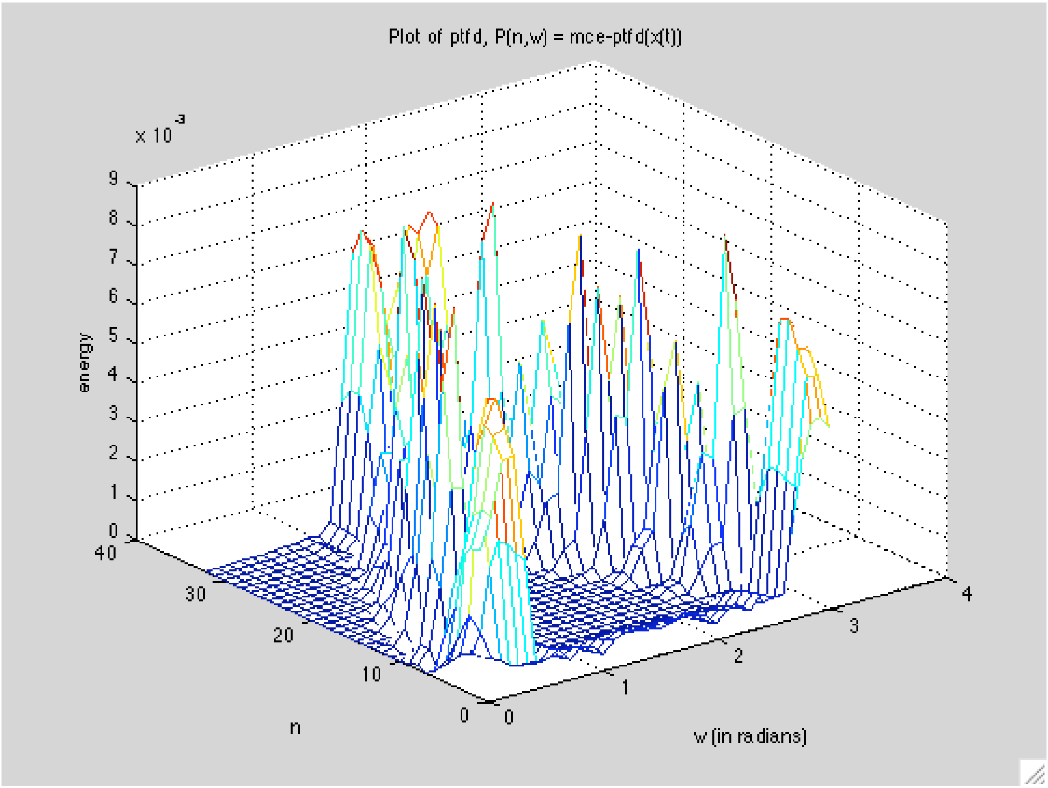

Figure 4.31: Plot of the initial MCE-PTFD prior for the double chirp signal

The Minimum Entropy algorithm was performed using the following rectangle selection methods: top-left method, top-right method, bottom-left method, bottom-right method, and maximum entropy loss method. Table 4.6 shows the final total entropy and number of zero-points for each method. Figure 4.32 shows the ME-TFD obtained using the top-left rectangle selection method. Figure 4.33 shows the ME-TFD obtained using the top-right rectangle selection method. Figure 4.34 shows the ME-TFD obtained using the bottom-left rectangle selection method. Figure 4.35 shows the ME-TFD obtained using the bottom-right rectangle selection method. Figure 4.36 shows the ME-TFD obtained using the maximum entropy loss rectangle selection method.

The resulting minimum entropy TFDs for this signal do not appear to be very meaningful for any of the rectangle selection methods. They do not appear to be nearly as good as the spectrogram prior results and are about even with the maximum entropy prior results.

Table 4.6: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the double chirp signal using several different rectangle selection methods.

Method	Total Entropy	Zero-Points
Top-Left	3.5865	961
Top-Right	3.6234	961
Bottom-Left	3.5621	961
Bottom-Right	3.5765	961
Maximum Entropy Loss	3.5491	961

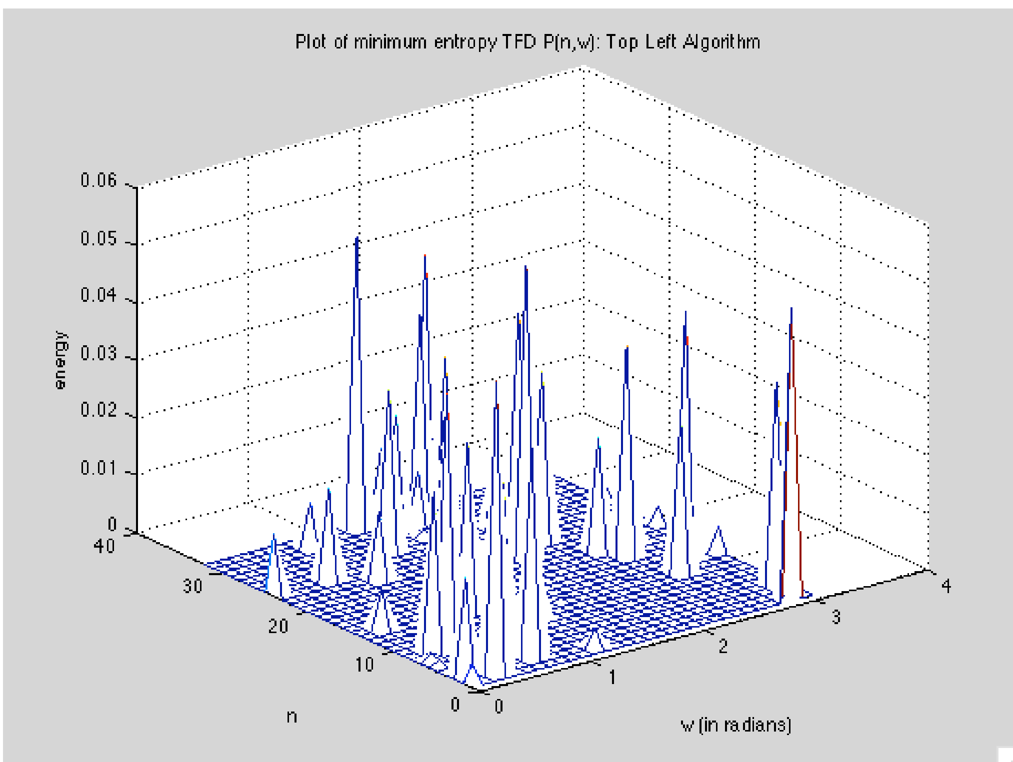

Figure 4.32: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the top-left rectangle selection method for the double chirp signal

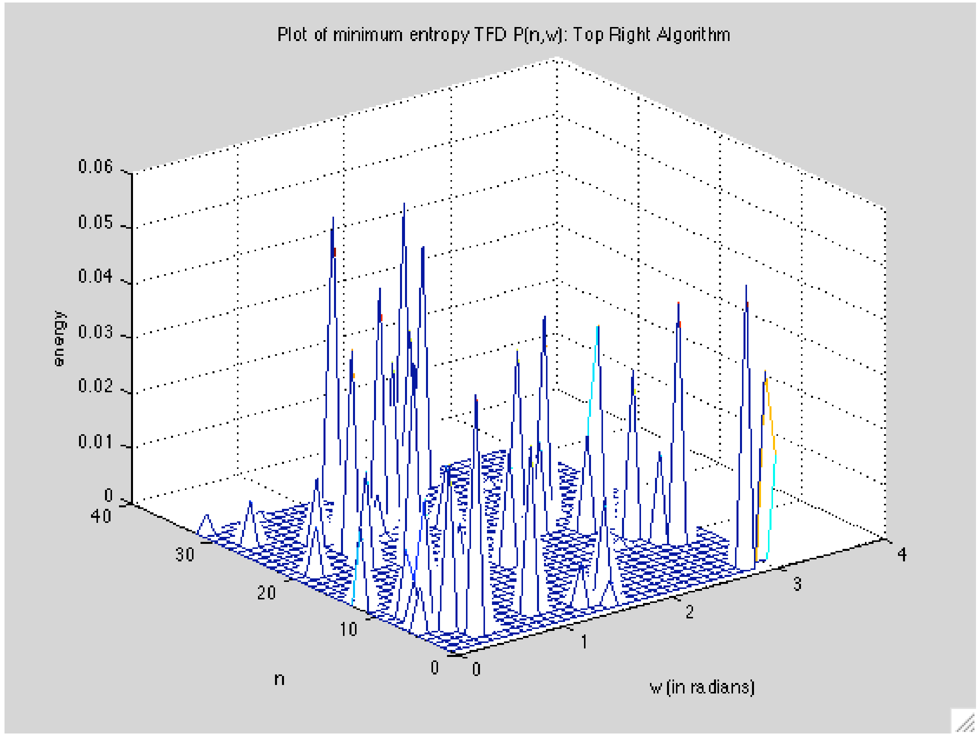

Figure 4.33: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the top-right rectangle selection method for the double chirp signal

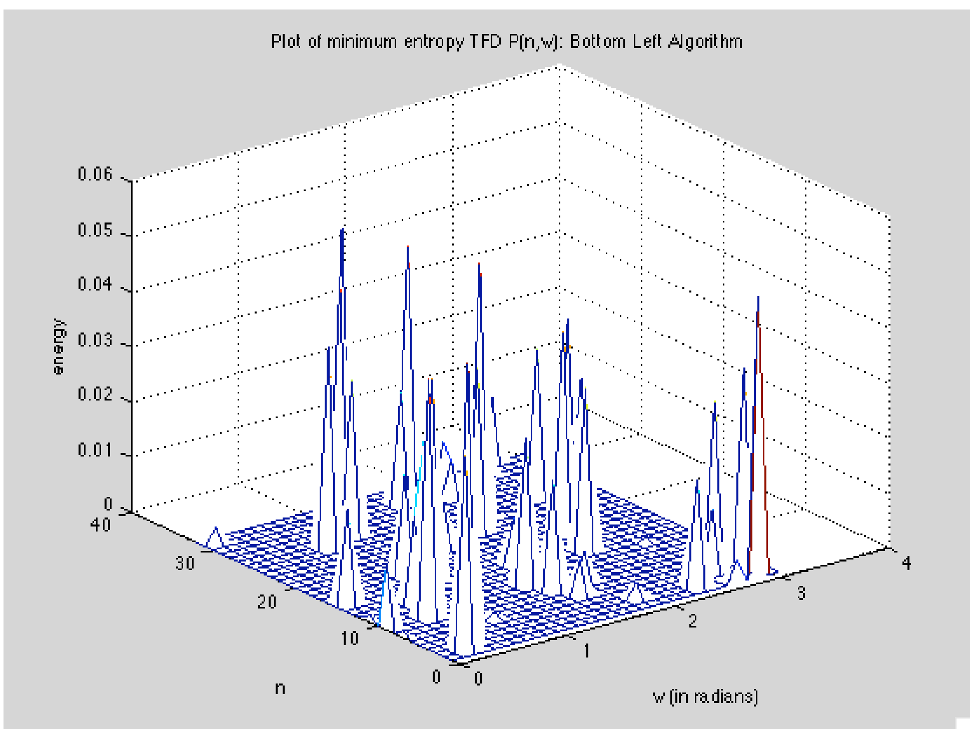

Figure 4.34: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the bottom-left rectangle selection method for the double chirp signal

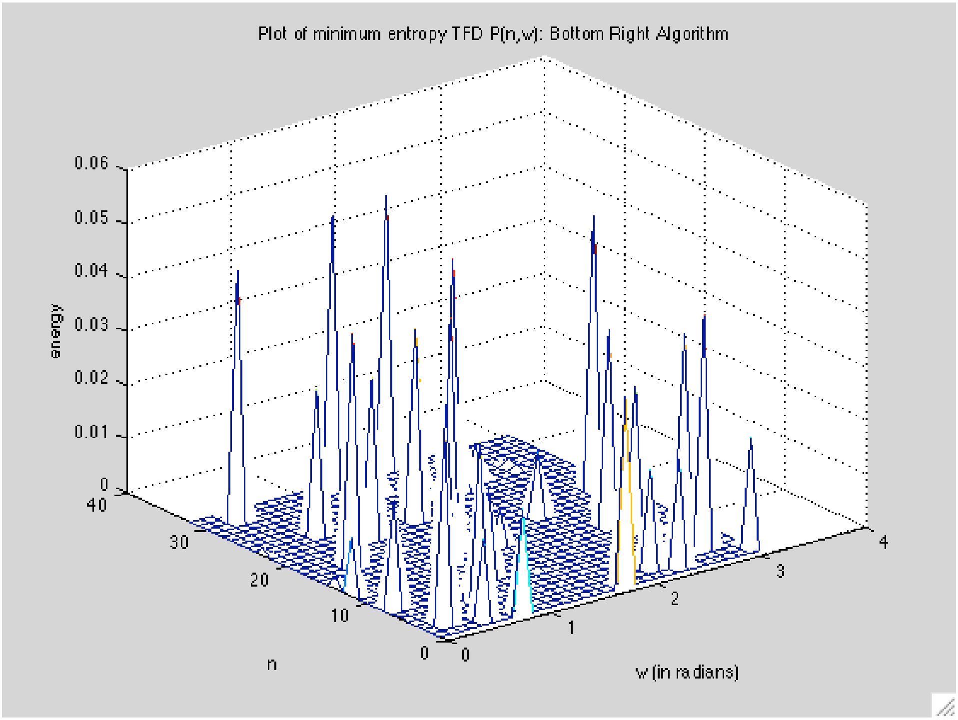

Figure 4.35: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the bottom-right rectangle selection method for the double chirp signal

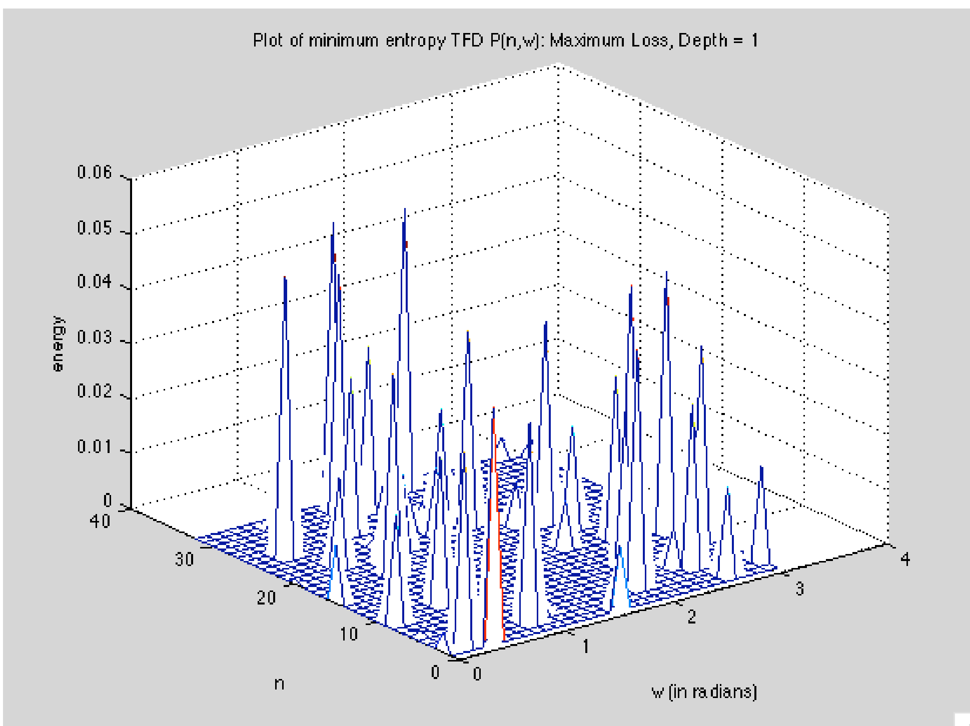

Figure 4.36: Plot of the Minimum Entropy TFD obtained using the MCE-PTFD as the prior and the maximum entropy loss rectangle selection method for the double chirp signal

4.2.4 Summary

Overall, the results from the maximum entropy algorithm using the MCE-PTFD prior appear to be better than the results from the maximum entropy prior but not as meaningful as the results from the spectrogram prior. For the chirp signal, the initial TFD is a good representation of the signal. As a result, some of the final TFDs are decent, especially the result from the bottom-right method. For the chirp with sinusoid signal, the chirp information is not as prominent as the sinusoidal portion of the signal. This characteristic seems to be responsible for much of the chirp information being lost in the final TFD. The chirp portion of the signal seems to have been moved to the sinusoidal portion. None of the representations are particularly good for this signal. For the double chirp signal, the initial representation is not smooth along the two chirp diagonals. Thus, the final TFDs also do not have the energy localized along these lines. Again, none of the final distributions are very meaningful for this signal.

Although these distributions do not appear to be as meaningful as the ones obtained using the spectrogram prior, a significant advantage of these distributions is they satisfy the time and frequency marginals. If a better initial representation that satisfies the marginals is employed, the minimum entropy algorithm may obtain better results while satisfying the marginals.

4.3 CONCLUSIONS

From the results, we can see that the prior that is selected has a significant influence on the final minimum entropy TFD. The first prior, the maximum entropy TFD, did not yield meaningful results for any of the sample signals and rectangle selection methods that were considered. From the plots of the initial maximum entropy TFDs, it appears that most of the signal information was lost before the minimum entropy algorithm was performed. The second prior, the spectrogram prior, produced significantly better results than the maximum entropy prior for each of the signals. With the spectrogram prior, some of the rectangle selection methods produced reasonable representations of the signals. Unfortunately, the final minimum entropy TFDs do not satisfy the time and frequency marginals when this prior is used. The third prior, the MCE-PTFD prior, seemed to produce results somewhere in between the results from the maximum entropy and spectrogram priors. Although the results for the simple chirp signal were reasonable, the results were not very good for the two more complex signals.

The question arises of why the spectrogram produces better results than the other two priors. First, unlike the maximum entropy prior, the spectrogram prior represents the signal and maintains the signal information reasonably well. The minimum entropy algorithm is more likely to produce a meaningful representation of the signal if the prior is a meaningful representation of the signal. Second, for the signals under consideration, the spectrogram is smooth and has maximum values at the appropriate time and frequency values. The MCE-PTFD is relatively choppy for each signal and does not always have its maximum values at the expected places. Also, although the maximum entropy TFD is smooth, it does not have its maximum values at the correct time and frequency values. Thus, the smoothness and positions of the maximum values may have an influence over the effectiveness of the prior. It would be ideal to find a prior that retains the signal information, is smooth, has the maximum values at the appropriate locations, and satisfies the marginals.

The rectangle selection method has a significant effect on the final TFD. For the spectrogram prior, the bottom-left and bottom-right rectangle selection methods produced decent results. On the other hand, the results from the top-left, top-right, and maximum entropy loss rectangle selection methods did not produce results that were very meaningful. The exact reason of why the bottom-left and bottom-right methods produced the best results is unknown. A method that may be good to try is one that accentuates the maximum values in the prior and diminishes the lesser values. This idea will be discussed in following sections.

Each of the final minimum entropy TFDs appears to have a number of outlier points which are defined as spikes of energy at locations where energy is not expected or desired. These spikes could be caused by a number of reasons. First, most of the priors have very few zero-points. This means that there are a number of points that have very small values. As the algorithm runs, it attempts to move energy from lower values to higher values setting the lower values to zero. Thus, some of the spikes could be the sum of all of the near-zero values consolidated at one point. If this is the case, we might be able to either ignore these spikes when considering the final representation or find a way to modify the algorithm to systematically eliminate the outliers. The second possible cause of the outliers is energy being moved to undesired locations. This cause would be the direct result of the rectangle selection method. This issue will be discussed in detail in later sections.

5.0 MINIMUM ENTROPY TFDS: ADVANCED CONCEPTS

So far, we have examined five relatively simple rectangle selection methods: the four brute force methods and the n^{th} maximum entropy loss method. None of these methodologies resulted in consistently meaningful representations. The two biggest issues with the previous methods are outlier spikes and high-energy points being moved to low-energy locations. The rectangle selection methods need to be modified to address these issues.

The outlier spike issue is mainly caused by summing the energies of a large number of near-zero points at a random location. For example, consider the following rectangle containing four near-zero points:

1	2
4	3

The minimum entropy algorithm will modify the rectangle to be as follows:

3	0
2	5

If the high-energy points are at 100 and a number of low-energy rectangles are summed, some of the low-energy points will be equivalent to the high-energy points. To resolve this issue, the rectangle selection method should avoid taking rectangles that have four low-energy points and focus on moving the low-energy points to the high-energy points. Each rectangle should always have at least one high-energy value.

The issue of moving high-energy points to low-energy locations is generally caused by choosing rectangles that have high-energy points on opposite corners of the selected rectangle. For example, consider the following rectangle containing two high-energy points on opposite corners:

100	2
4	95

The minimum entropy algorithm will modify the rectangle to be as follows:

5	97
99	0

The result of the algorithm is to move the high-energy values to low-energy locations. To resolve this issue, the rectangle selection method needs to avoid rectangles that have high-energy points on opposite corners.

This section will focus on more advanced rectangle selection methods that attempt to resolve these two issues.

5.1 SORTED RECTANGLE SELECTION METHOD

The first method that will be examined is the sorted rectangle selection method. The sorted rectangle selection method first sorts all of the points in the initial distribution from largest energy to smallest energy. It then takes the largest point and the smallest point and uses them as opposite corners of the rectangle if the two points create a valid rectangle. Next, the algorithm takes the largest point and the second smallest point. It continues taking the largest point with the rest of the points in order of increasing energy. After looping through all of the points with the largest point, it takes the second largest point and repeats the process of looping through all of the points from lowest energy to highest energy. The method continues in this fashion until all of the points have been used as the initial point. Once all of the points have been used as the initial point, the algorithm starts over again with the largest point. This process continues until none of the rectangles can be reduce any further.

The idea behind this method is the energy of the near-zero points will be moved to the high-energy points. This method should reduce the number of outlier spikes that are the result of adding a large number of near-zero points. Consequently, the sorted method does not prevent the selection of rectangles with high-energy values on opposite corners.

5.1.1 Examples

To evaluate the sorted rectangle selection method, the minimum entropy algorithm was performed with this method for the simple chirp signal, the chirp with sinusoid signal, and the double chirp signal using spectrogram and MCE-PTFD priors.

For the simple chirp signal with the spectrogram prior, the initial TFD has a total entropy of 5.1572 and 0 zero-points. The final minimum entropy TFD has a total entropy of 3.4562 and 961 zero-points. Figure 5.1 shows the final distribution for the simple chirp signal using the sorted rectangle selection method and spectrogram prior.

For the chirp with sinusoid signal with the spectrogram prior, the initial TFD has a total entropy of 5.5954 and 0 zero-points. The final minimum entropy TFD has a total entropy of 3.4567 and 961 zero-points. Figure 5.2 shows the final distribution for the chirp with sinusoid signal using the sorted rectangle selection method and spectrogram prior.

For the double chirp signal with the spectrogram prior, the initial TFD has a total entropy of 5.4956 and 0 zero-points. The final minimum entropy TFD has a total entropy of 3.3850 and 961 zero-points. Figure 5.3 shows the final distribution for the double chirp signal using the sorted rectangle selection method and spectrogram prior.

For the simple chirp signal with the MCE-PTFD prior, the initial TFD has a total entropy of 5.2584 and 0 zero-points. The final minimum entropy TFD has a total entropy of 3.6916 and 961 zero-points. Figure 5.4 shows the final distribution for the simple chirp signal using the sorted rectangle selection method and MCE-PTFD prior.

For the chirp with sinusoid signal with the MCE-PTFD prior, the initial TFD has a total entropy of 4.7352 and 64 zero-points. The final minimum entropy TFD has a total entropy of

3.3004 and 963 zero-points. Figure 5.5 shows the final distribution for the chirp with sinusoid signal using the sorted rectangle selection method and MCE-PTFD prior.

For the double chirp signal with the MCE-PTFD prior, the initial TFD has a total entropy of 5.5150 and 0 zero-points. The final minimum entropy TFD has a total entropy of 3.5974 and 961 zero-points. Figure 5.6 shows the final distribution for the double chirp signal using the sorted rectangle selection method and MCE-PTFD prior.

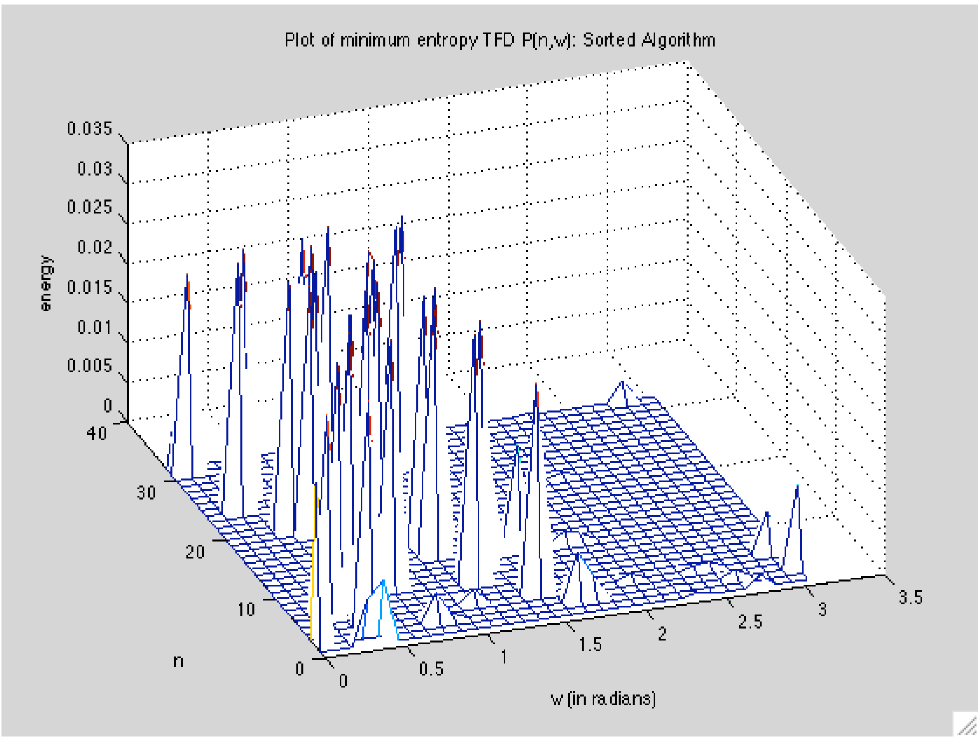

Figure 5.1: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the sorted rectangle selection method for the simple chirp signal

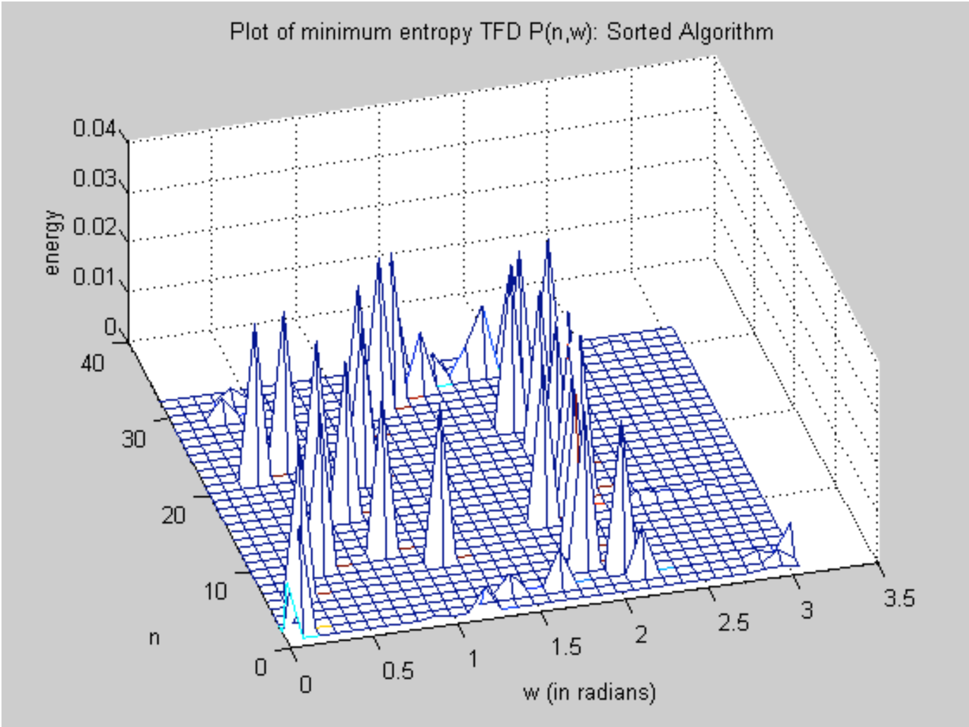

Figure 5.2: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the sorted rectangle selection method for the chirp with sinusoid signal

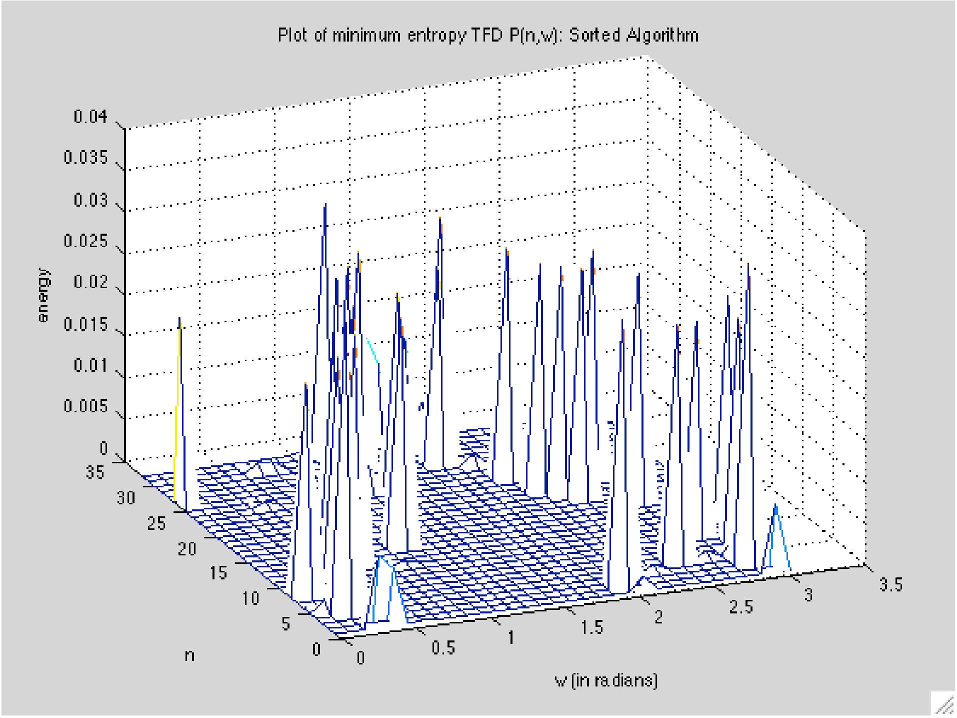

Figure 5.3: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the sorted rectangle selection method for the double chirp signal

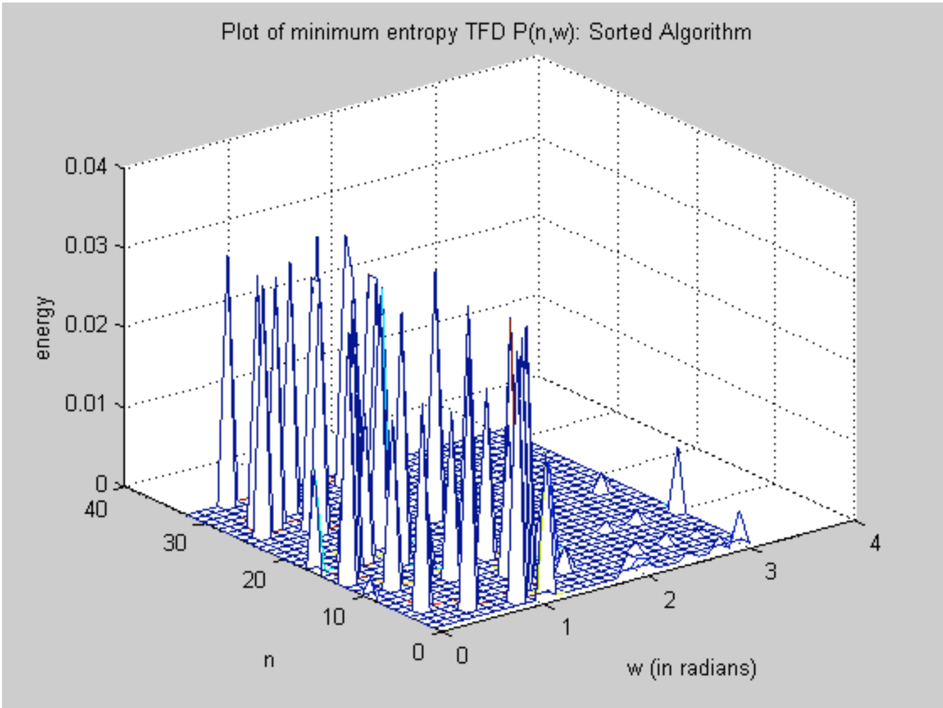

Figure 5.4: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the sorted rectangle selection method for the simple chirp signal

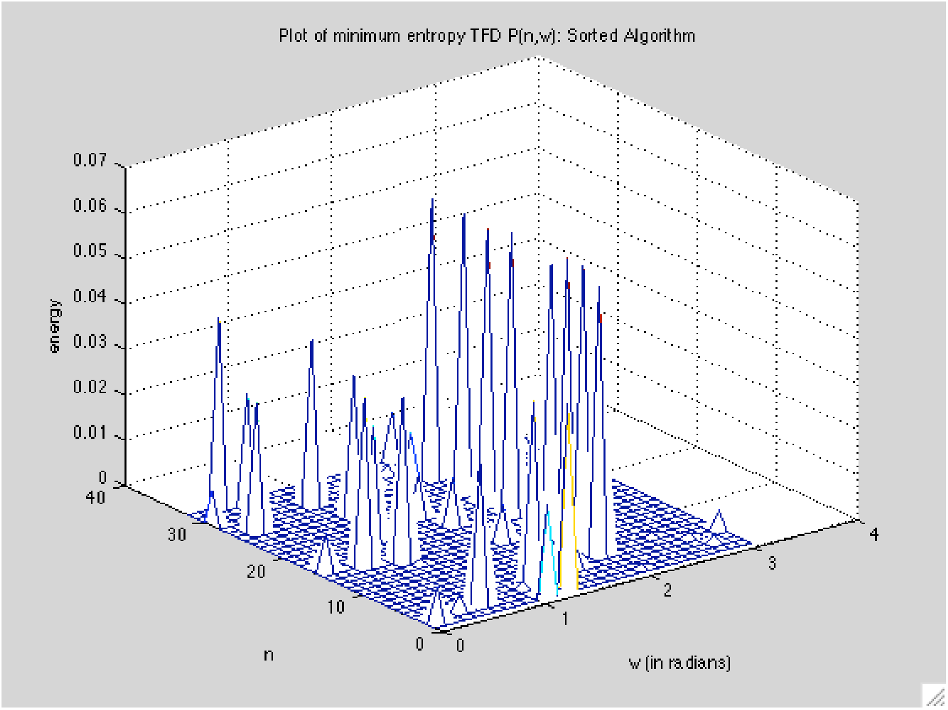

Figure 5.5: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the sorted rectangle selection method for the chirp with sinusoid signal

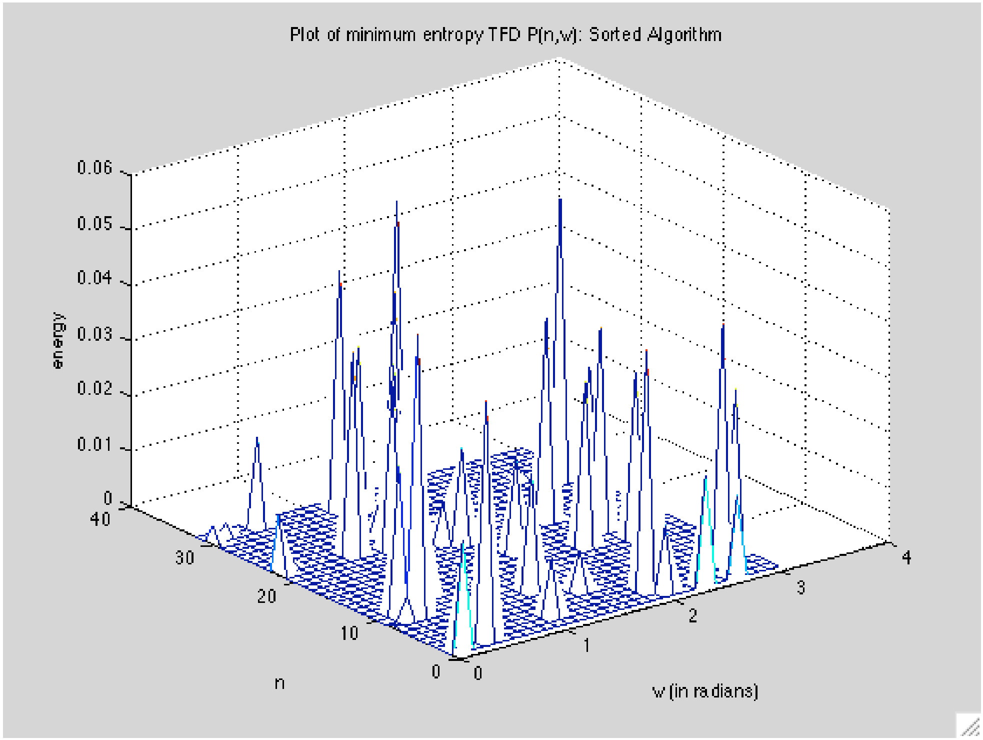

Figure 5.6: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the sorted rectangle selection method for the double chirp signal

5.1.2 Conclusions

Although the sorted rectangle selection method eliminates some of the outlier spikes that result from summing near-zero points, it does not remove all of these spikes and does nothing to prevent high-energy points from being moved to low-energy locations. As a result, the final minimum TFDs still contain a large number of energy spikes at points that should not have much energy.

Also note hat the spectrogram prior appears to produce better results than the MCE-PTFD prior. This trend could be due to the fact that the spectrogram has the highest energy values at the correct time-frequency locations for the signals under consideration where the MCE-PTFD prior does not. It may be important for the initial TFD to have the property of having the maximum energy values at the correct time-frequency points for the sorted rectangle selection method to be effective.

5.2 MODIFIED SORTED RECTANGLE SELECTION METHOD

The modified sorted rectangle selection method is similar to the sorted method except it does not attempt to find the absolute minimum entropy distribution. First, the algorithm sorts all of the points in the initial distribution from the largest energy value to the smallest energy value. It then takes the n highest energy points and the p-n lowest energy points where p is the total number of points in the TFD and places them in separate lists. Next, the algorithm takes the largest point in the first list and the smallest point in the second list and uses them as opposing corners in the rectangle. It continues with the largest point in the first list taking all of the points in the second list in increasing order. Afterwards, the method repeats the process that it completed for the highest value in the first list for the rest of the values in the first list in decreasing order. Once all of the points in the first list have been considered, the process repeats starting with the largest point in the first list. This continues until none of the rectangles can be reduced any further.

Note that this method will not result in a true minimum entropy time-frequency distribution since all of the possible rectangles will not be considered. This property may be advantageous for signals that require more non-zero points than the minimum entropy TFD will contain to accurately represent. Remember that a true minimum entropy distribution will always contain approximately the same number of non-zero points as there are marginal constraints.

Also note that the value of n that is selected will have a profound effect on the resulting distribution. Assuming that the value of n is less than $p/2$ where p is the total number of points in the distribution, smaller values of n will generally produce a final distribution that has fewer zero-points and a higher total entropy while larger values of n will generally produce a final distribution that has more zero-points and a lower total entropy.

It seems reasonable that the value of n should be close to the number of maximum energy points for the ideal distribution. For example, the ideal representation of the 32-point simple chirp signal that has been previously considered should have around 32 maximum values. Thus, a reasonable guess for the best value of n might be somewhere around 32. Conversely, the ideal representation of the 32-point chirp with sinusoid and double chirp signals should have around 64 maximum values. For these signals, a reasonable guess for the best value of n might be somewhere around 64.

To evaluate the modified sorted rectangle selection method, we will again consider three signals: the simple chirp signal, the chirp with sinusoid signal, and the double chirp signal. The modified sorted rectangle selection method will be employed on all three signals for different values of n using both spectrogram and MCE-PTFD priors.

5.2.1 Simple Chirp Example

First, the minimum entropy algorithm was performed using the modified sorted rectangle selection method with several different values of n for the 32-point simple chirp signal using the spectrogram as the prior. The spectrogram prior has a total entropy of 5.1572 and 0 zero-points. Table 5.1 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the modified sorted rectangle selection method for different values of n.

Figures 5.7 – 5.12 show the final minimum entropy TFDs for the modified sorted rectangle method for n = 25, 30, 35, 40, 45, and 50.

Table 5.1: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the spectrogram as the prior for the simple chirp signal using the modified sorted rectangle selection method for different values of n.

n	**Total Entropy**	**Zero-Points**
5	4.8411	227
10	4.5373	443
15	4.2879	612
20	3.9590	765
25	3.7145	873
30	3.7455	830
35	3.7286	838
40	3.6994	844
45	3.7414	825
50	3.6415	893

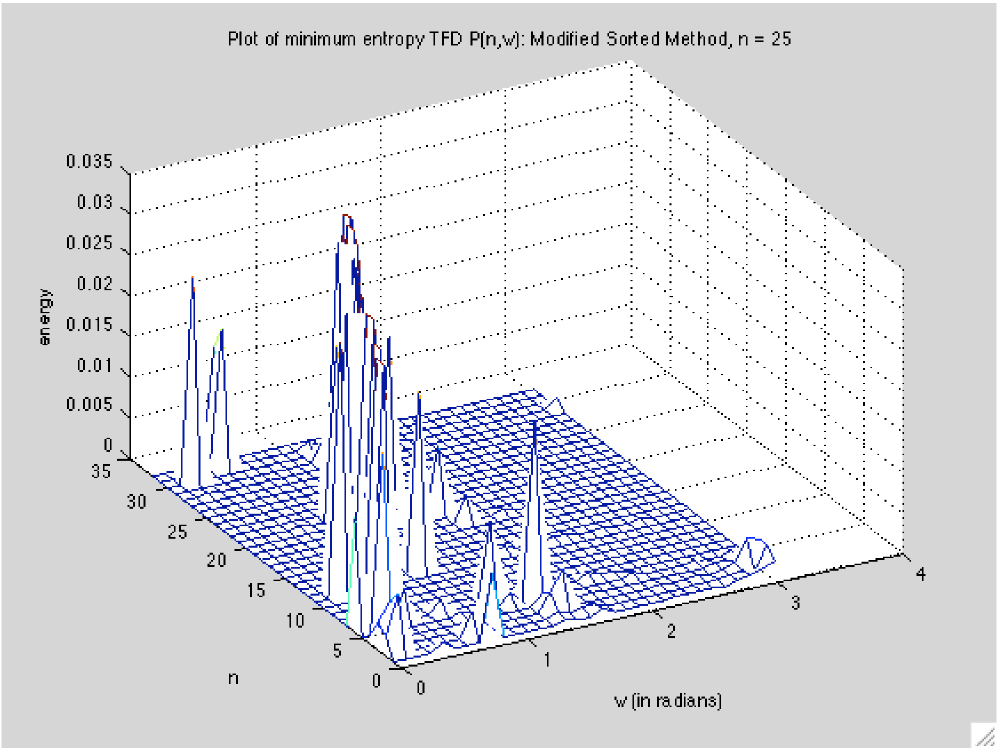

Figure 5.7: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 25 for the simple chirp signal

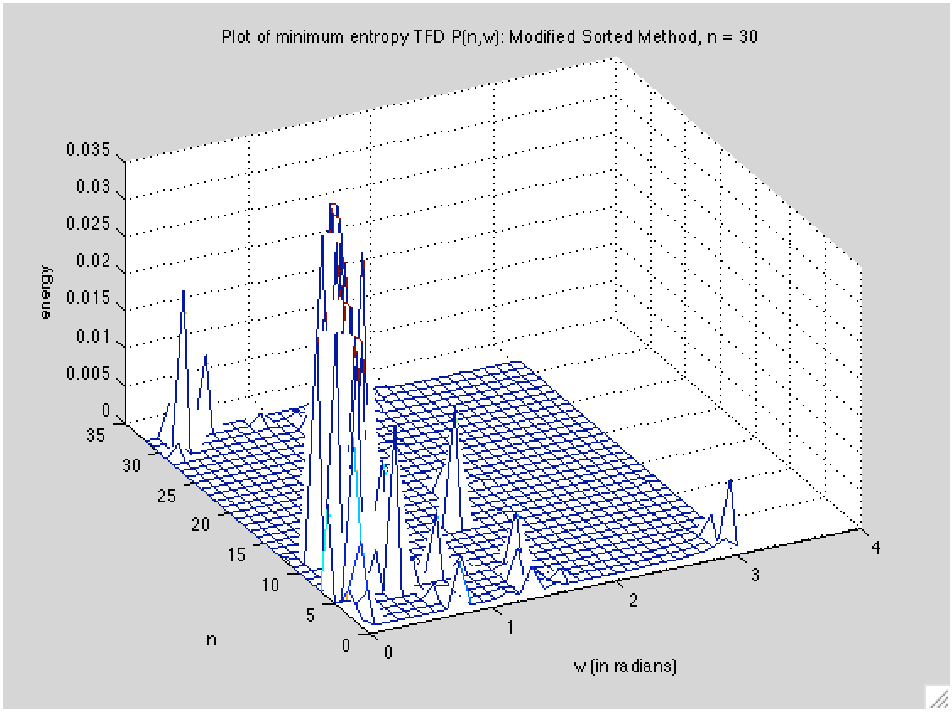

Figure 5.8: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with *n* = 30 for the simple chirp signal

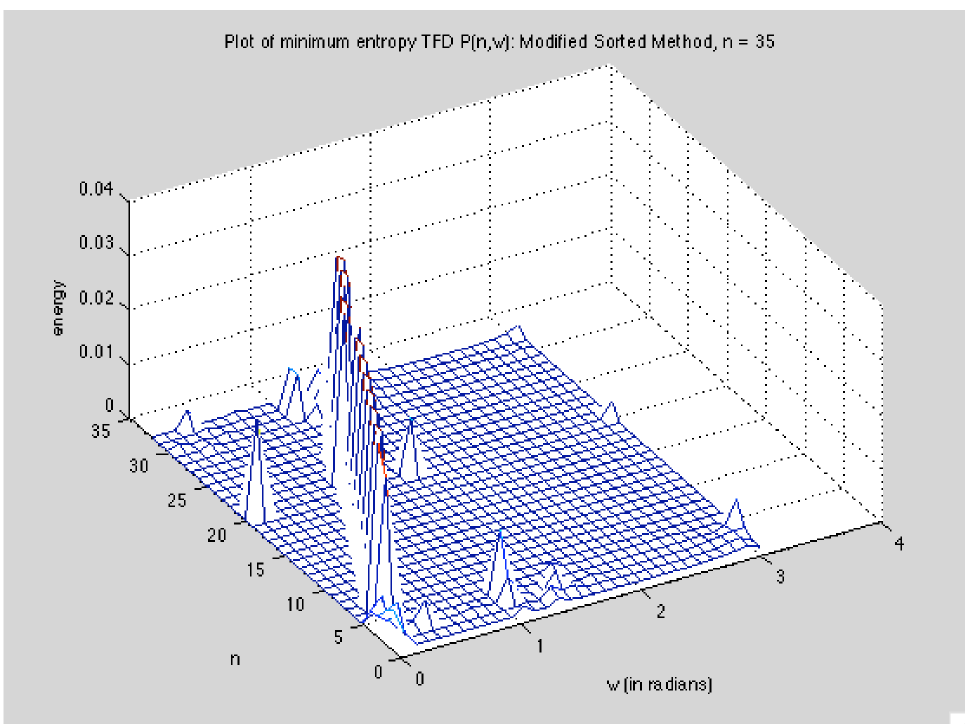

Figure 5.9: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with *n* = 35 for the simple chirp signal

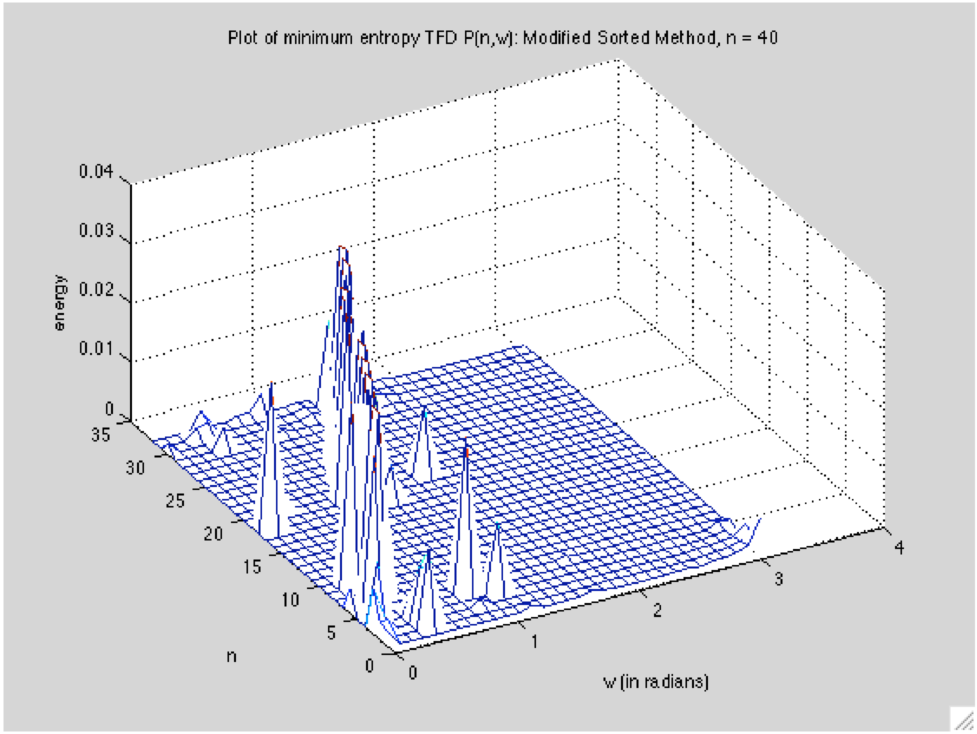

Figure 5.10: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with $n = 40$ for the simple chirp signal

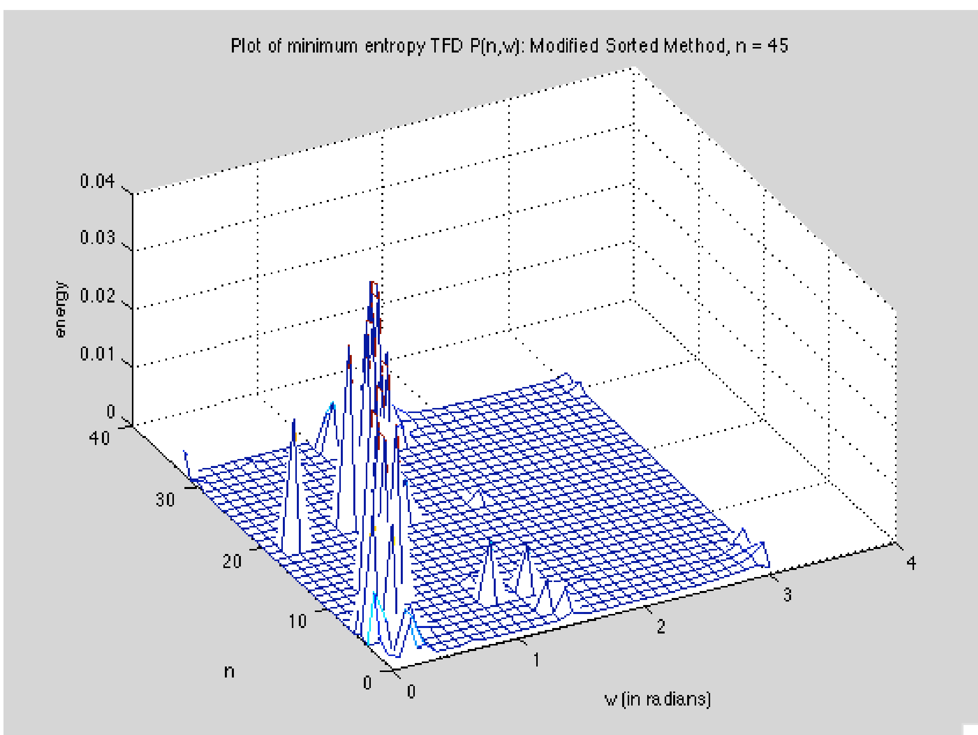

Figure 5.11: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with $n = 45$ for the simple chirp signal

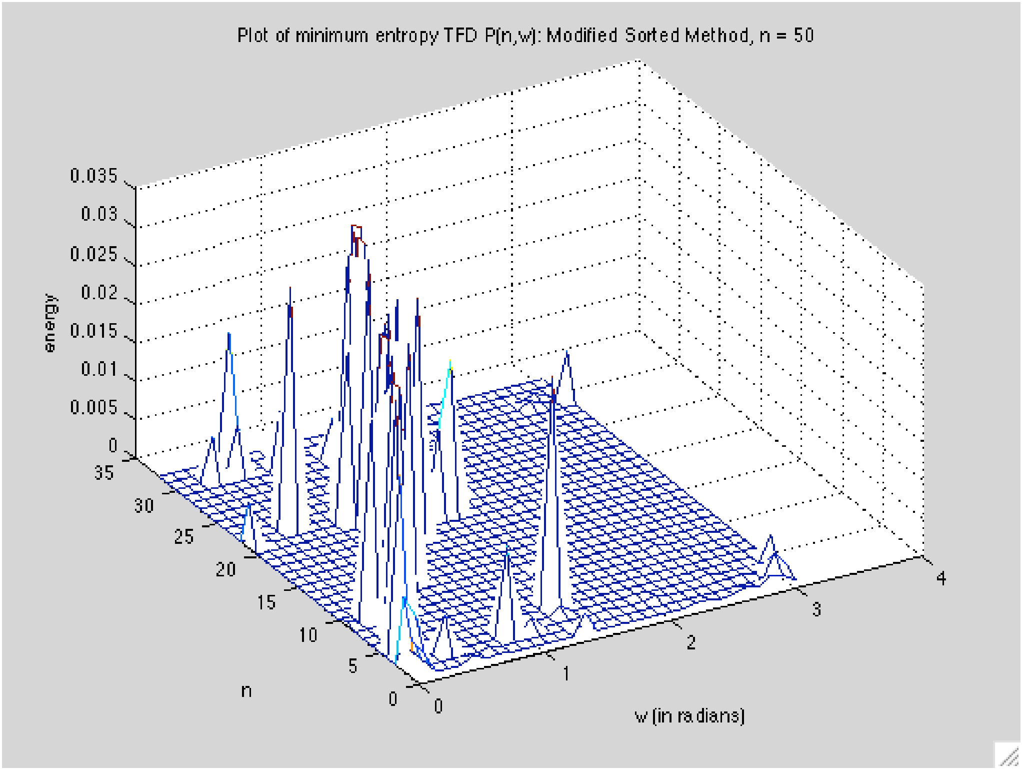

Figure 5.12: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 50 for the simple chirp signal

Next, the minimum entropy algorithm was performed using the modified sorted rectangle selection method with several different values of n for the 32-point simple chirp signal using the MCE-PTFD as the prior. The MCE-PTFD prior has a total entropy of 5.2584 and 0 zero-points. Table 5.2 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the modified sorted rectangle selection method for different values of n. Figures 5.13 – 5.18 show the final minimum entropy TFDs for the modified sorted rectangle method for n = 25, 30, 35, 40, 45, and 50.

Table 5.2: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the simple chirp signal using the modified sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
5	5.0093	250
10	4.7545	421
15	4.5461	573
20	4.3145	726
25	4.1884	810
30	4.1137	845
35	4.0240	843
40	4.0358	839
45	4.0228	829
50	3.8886	906

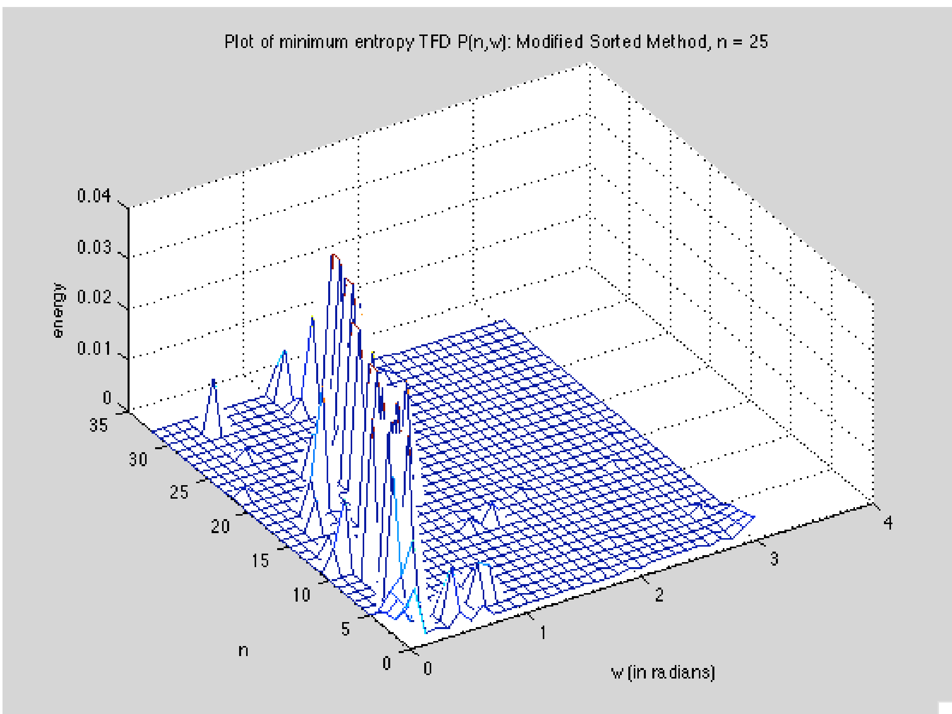

Figure 5.13: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with *n* = 25 for the simple chirp signal

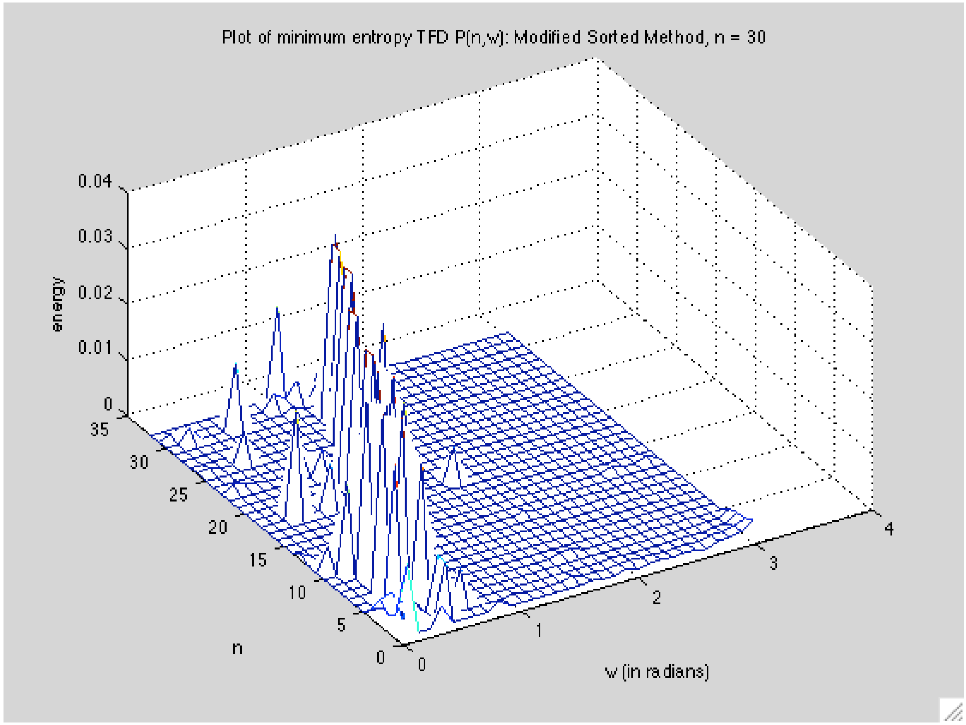

Figure 5.14: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with *n* = 30 for the simple chirp signal

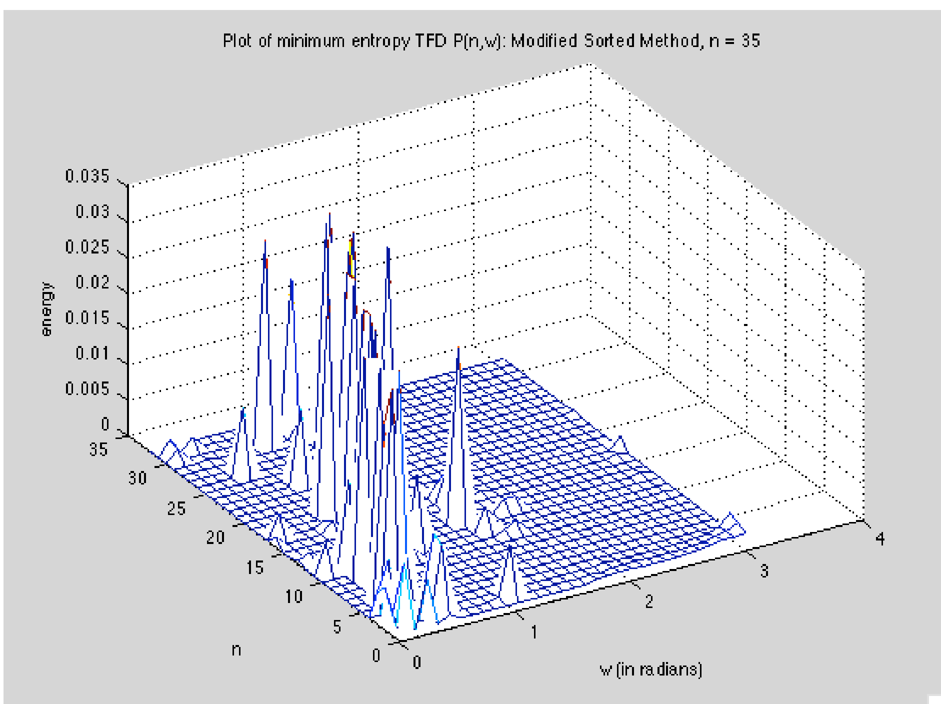

Figure 5.15: : Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with *n* = 35 for the simple chirp signal

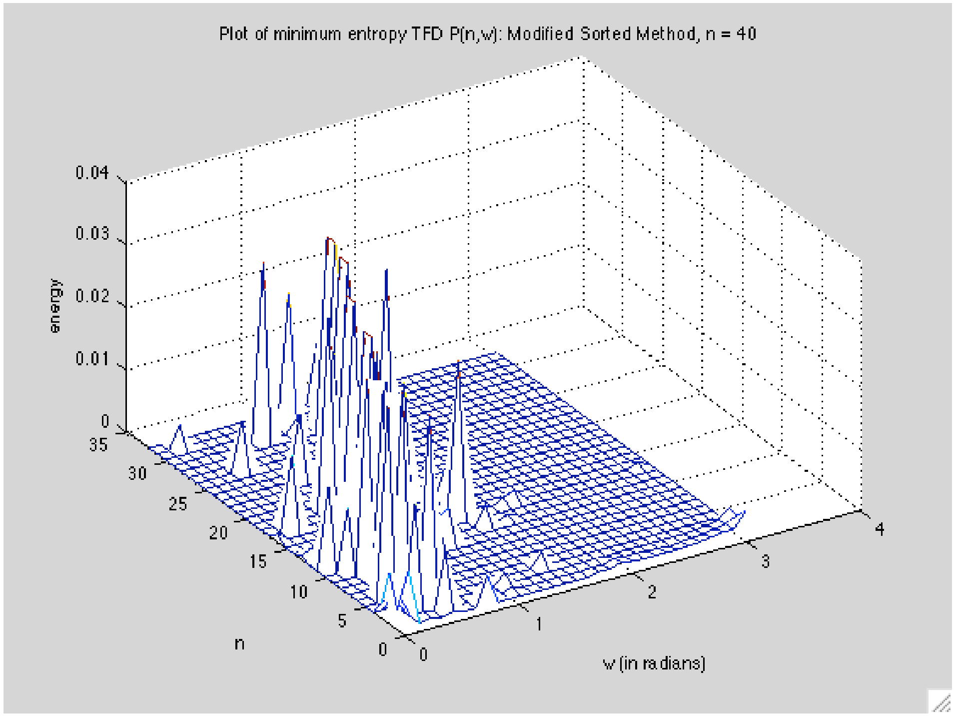

Figure 5.16: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 40 for the simple chirp signal

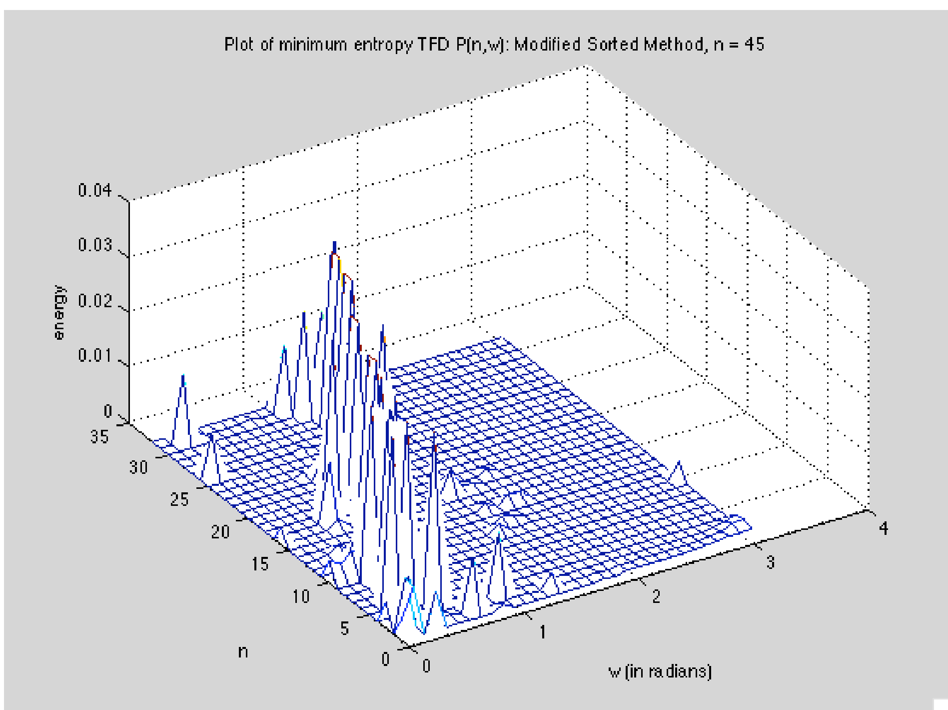

Figure 5.17: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 45 for the simple chirp signal

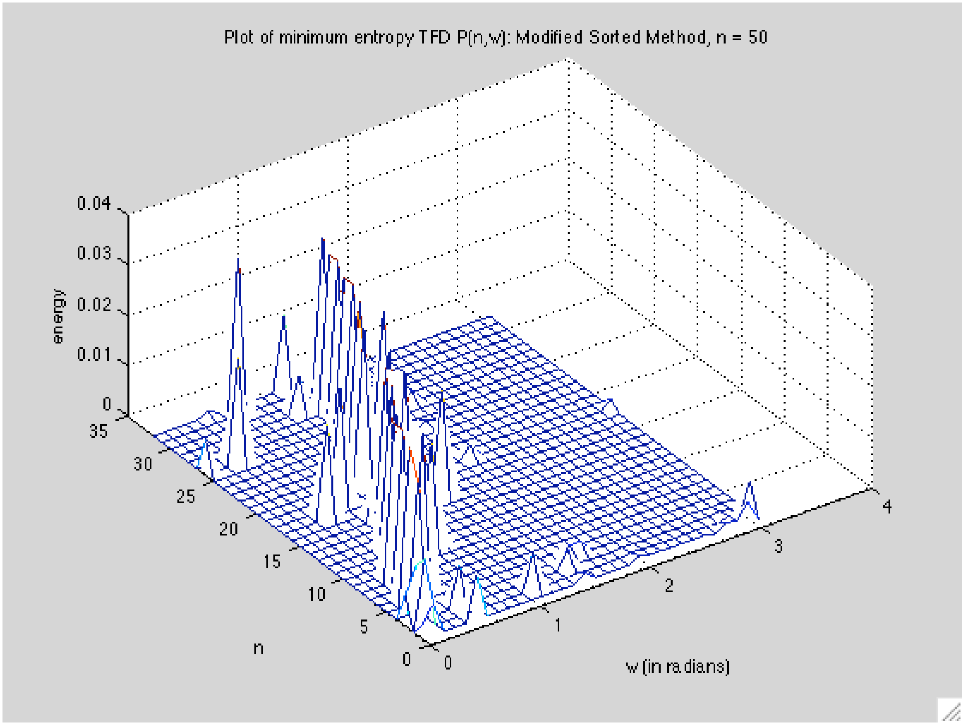

Figure 5.18: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with *n* = 50 for the simple chirp signal

5.2.2 Chirp with Sinusoid Example

The minimum entropy algorithm was performed using the modified sorted rectangle selection method with several different values of *n* for the 32-point chirp with sinusoid signal using the spectrogram as the prior. The spectrogram prior has a total entropy of 5.5954 and 0 zero-points. Table 5.3 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the modified sorted rectangle selection method for different values of *n*. Figures 5.19 – 5.23 show the final minimum entropy TFDs for the modified sorted rectangle method for *n* = 40, 50, 60, 70, and 80.

Table 5.3: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the spectrogram as the prior for the chirp with sinusoid signal using the modified sorted rectangle selection method for different values of *n*.

n	Total Entropy	Zero-Points
35	4.1381	780
40	4.1941	783
45	3.9712	838
50	3.8885	834
55	3.9345	833
60	3.8014	861
65	3.8807	846
70	3.7856	864
75	3.8926	857
80	3.6707	876

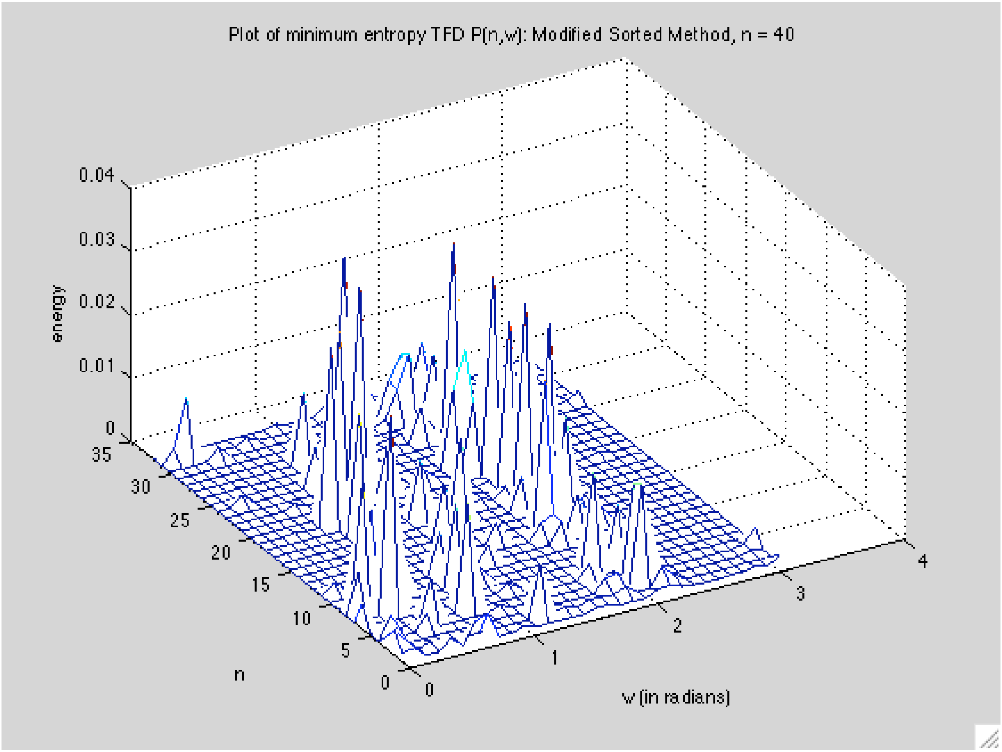

Figure 5.19: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with *n* = 40 for the chirp with sinusoid signal

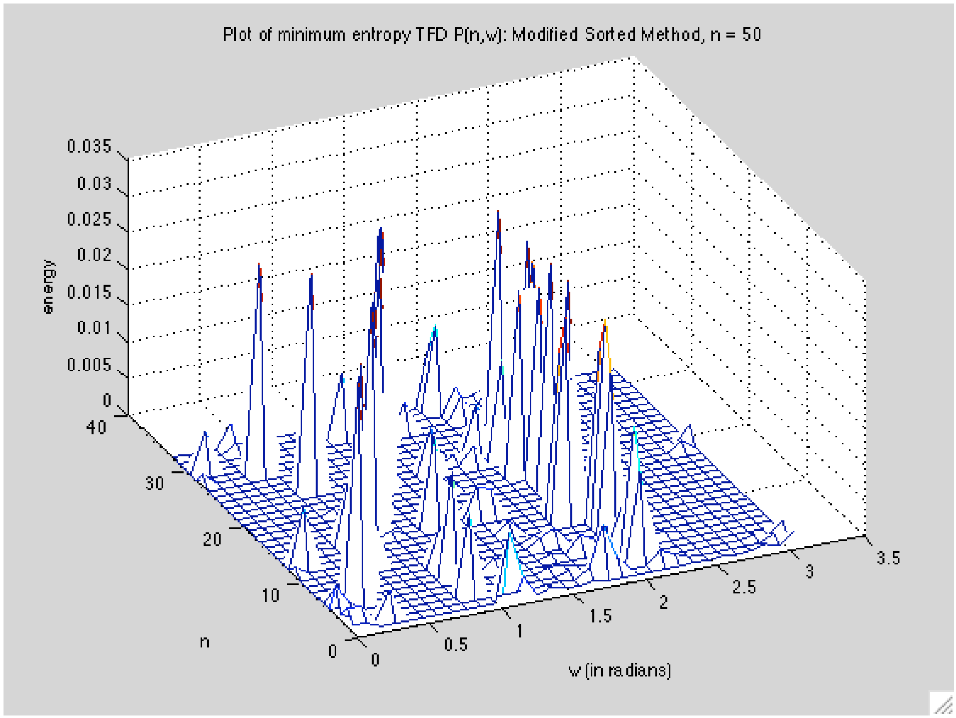

Figure 5.20: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 50 for the chirp with sinusoid signal

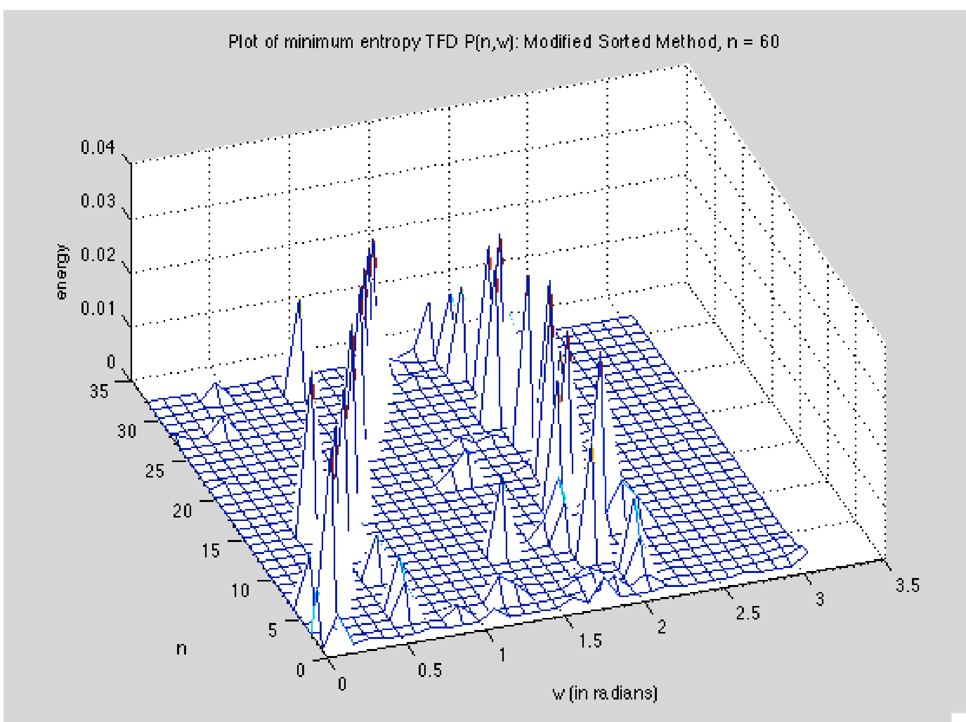

Figure 5.21: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 60 for the chirp with sinusoid signal

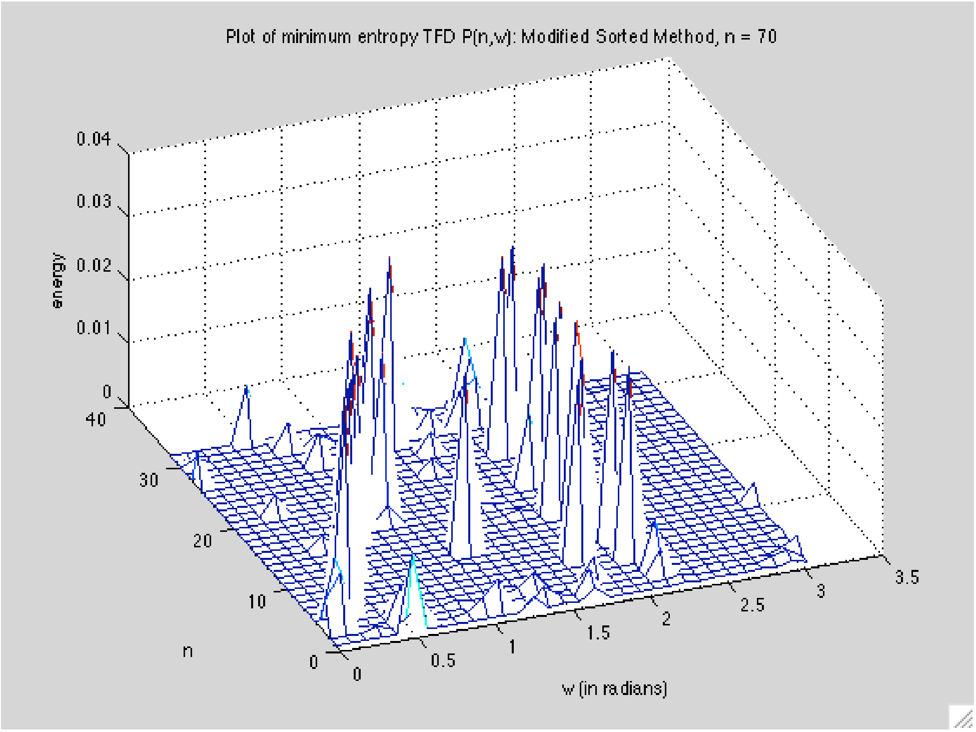

Figure 5.22: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 70 for the chirp with sinusoid signal

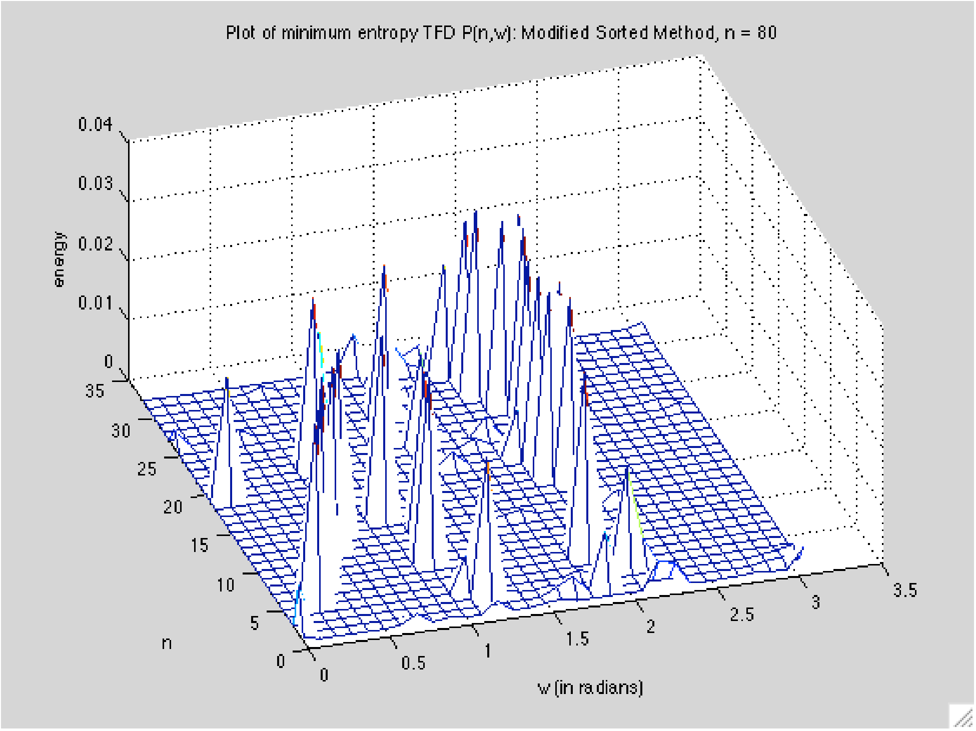

Figure 5.23: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 80 for the chirp with sinusoid signal

Next, the minimum entropy algorithm was performed using the modified sorted rectangle selection method with several different values of *n* for the 32-point chirp with sinusoid signal using the MCE-PTFD as the prior. The MCE-PTFD prior has a total entropy of 4.7352 and 64 zero-points. Table 5.4 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the modified sorted rectangle selection method for different values of *n*. Figures 5.24 – 5.28 show the final minimum entropy TFDs for the modified sorted rectangle method for *n* = 40, 50, 60, 70, and 80.

Table 5.4: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the chirp with sinusoid signal using the modified sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
35	744	3.7982
40	850	3.5082
45	866	3.5629
50	907	3.4276
55	853	3.5470
60	807	3.6384
65	864	3.4873
70	849	3.5298
75	872	3.4694
80	864	3.4640

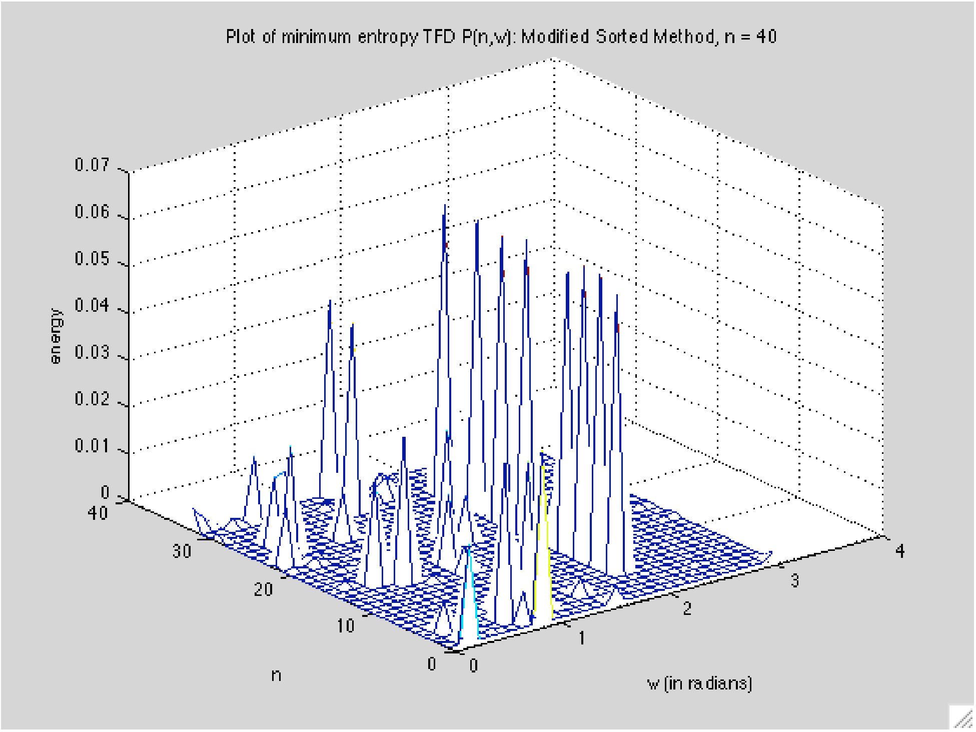

Figure 5.24: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with *n* = 40 for the chirp with sinusoid signal

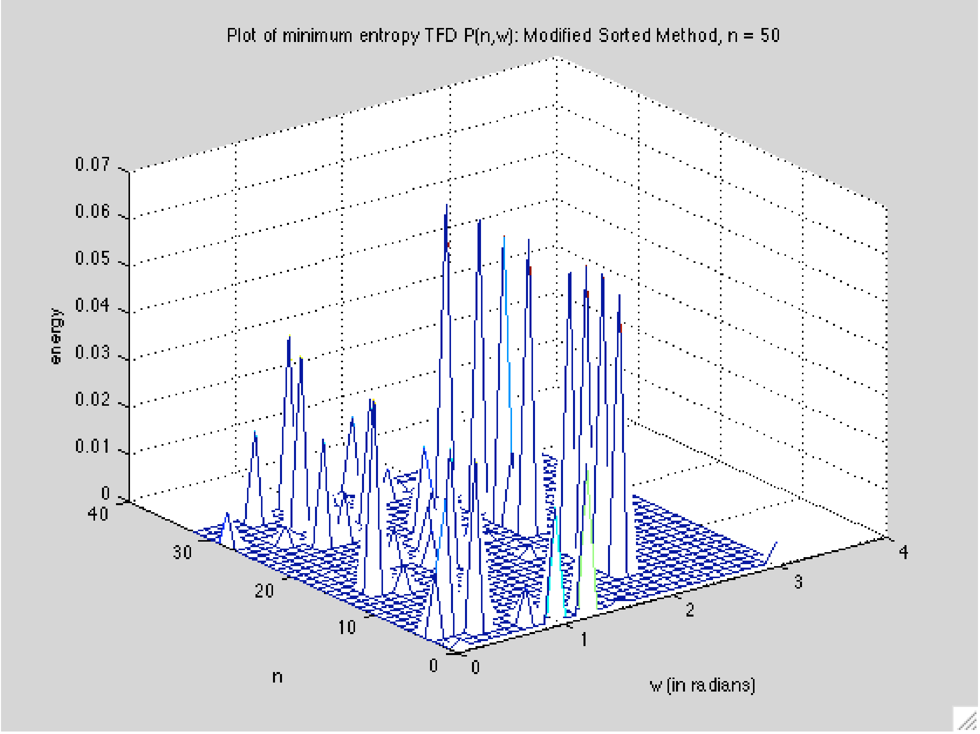

Figure 5.25: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with *n* = 50 for the chirp with sinusoid signal

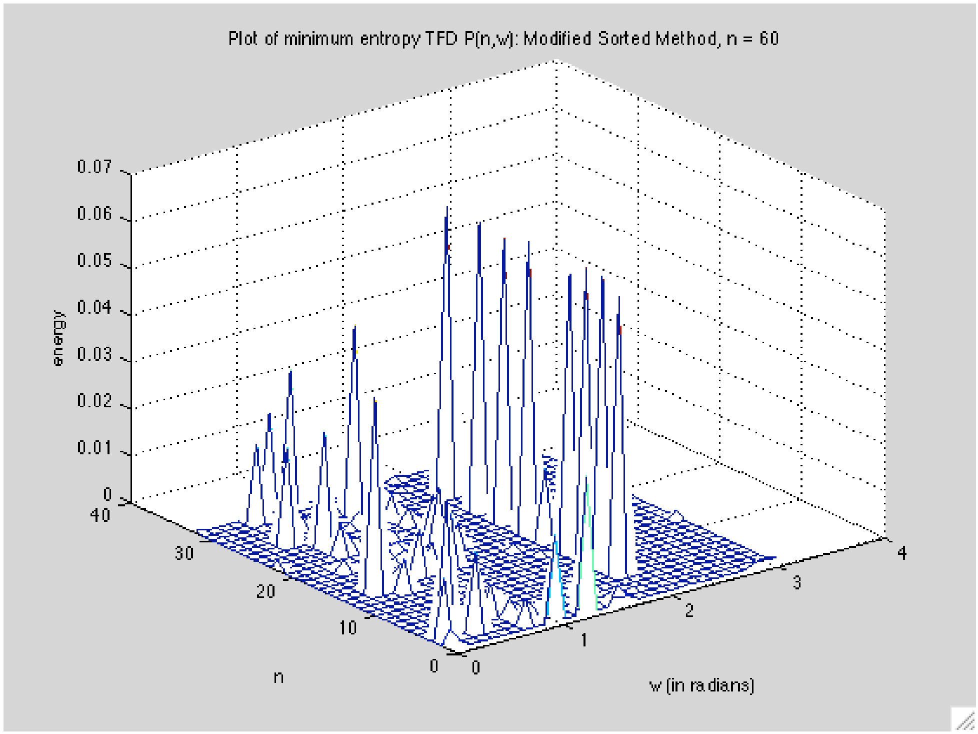

Figure 5.26: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 60 for the chirp with sinusoid signal

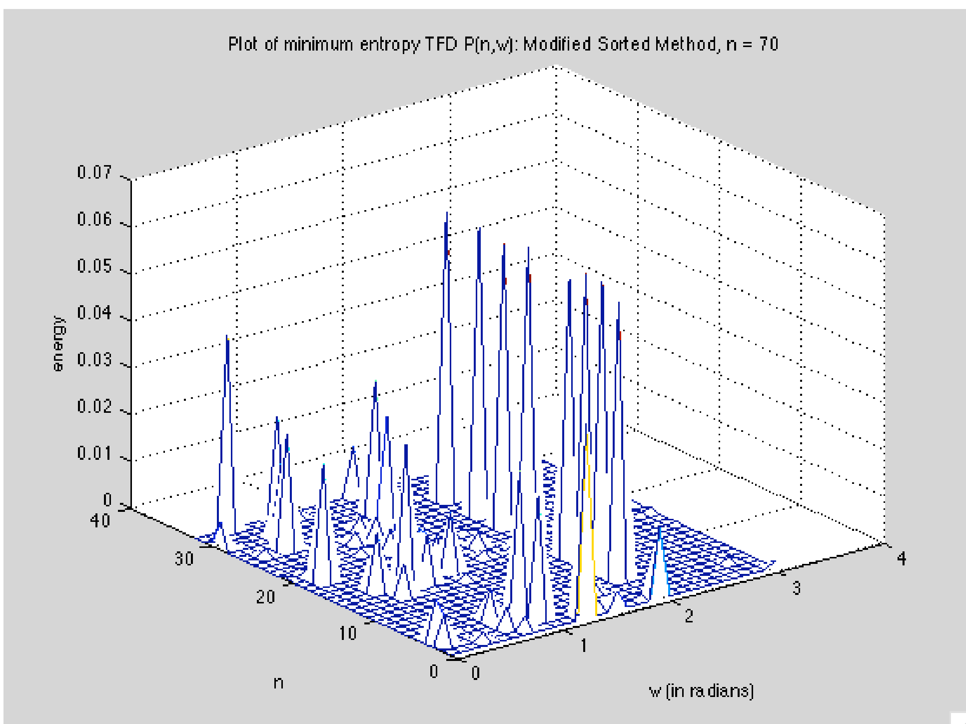

Figure 5.27: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 70 for the chirp with sinusoid signal

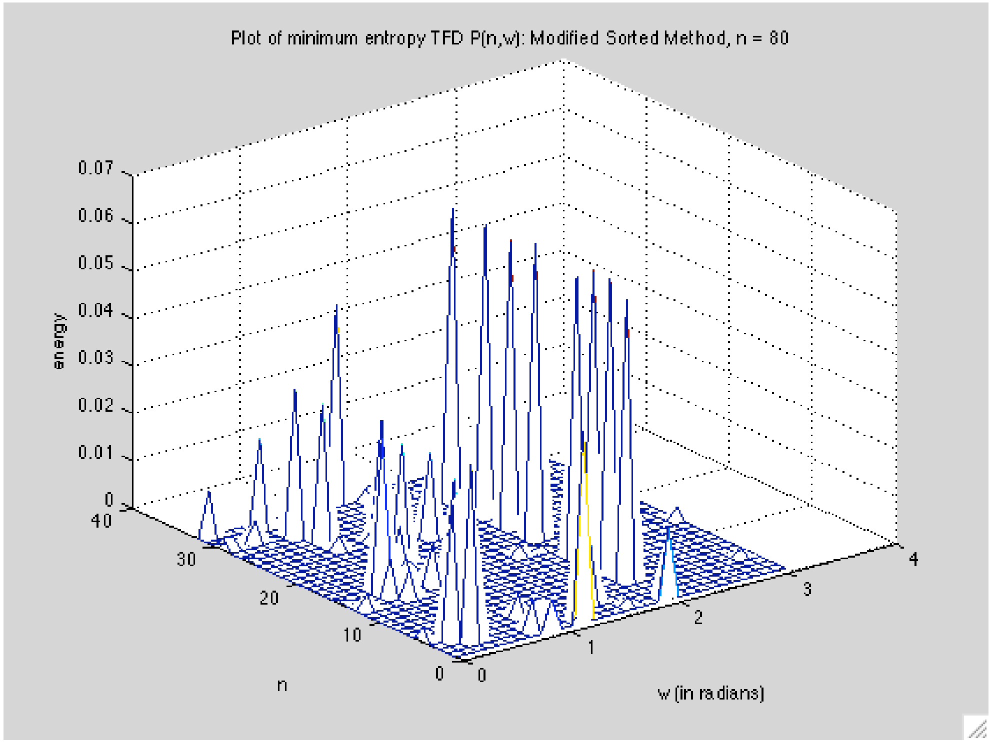

Figure 5.28: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 80 for the chirp with sinusoid signal

5.2.3 Double Chirp Example

The minimum entropy algorithm was performed using the modified sorted rectangle selection method with several different values of n for the 32-point double chirp signal using the spectrogram as the prior. The spectrogram prior has a total entropy of 5.4956 and 0 zero-points. Table 5.5 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the modified sorted rectangle selection method for different values of n. Figures 5.29 – 5.33 show the final minimum entropy TFDs for the modified sorted rectangle method for n = 40, 50, 60, 70, and 80.

Table 5.5: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the spectrogram as the prior for the double chirp signal using the modified sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
35	710	4.4933
40	717	4.5079
45	699	4.5111
50	807	4.2252
55	875	3.9780
60	859	3.9911
65	876	3.8353
70	907	3.7232
75	919	3.6239
80	936	3.4591

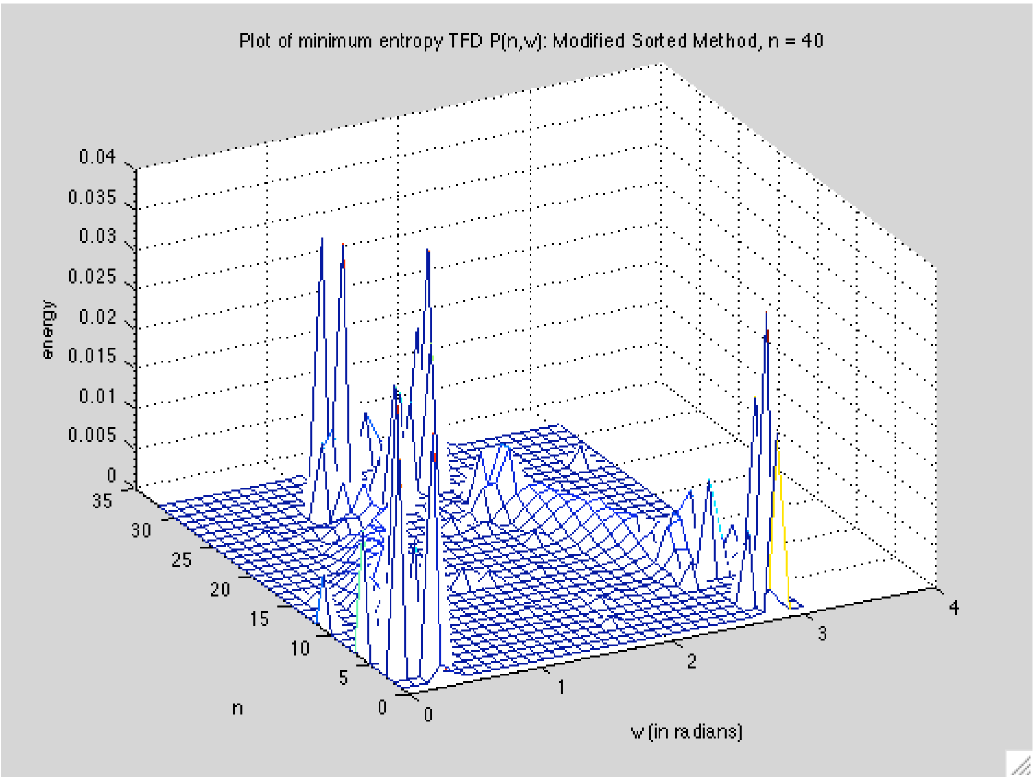

Figure 5.29: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 40 for the double chirp signal

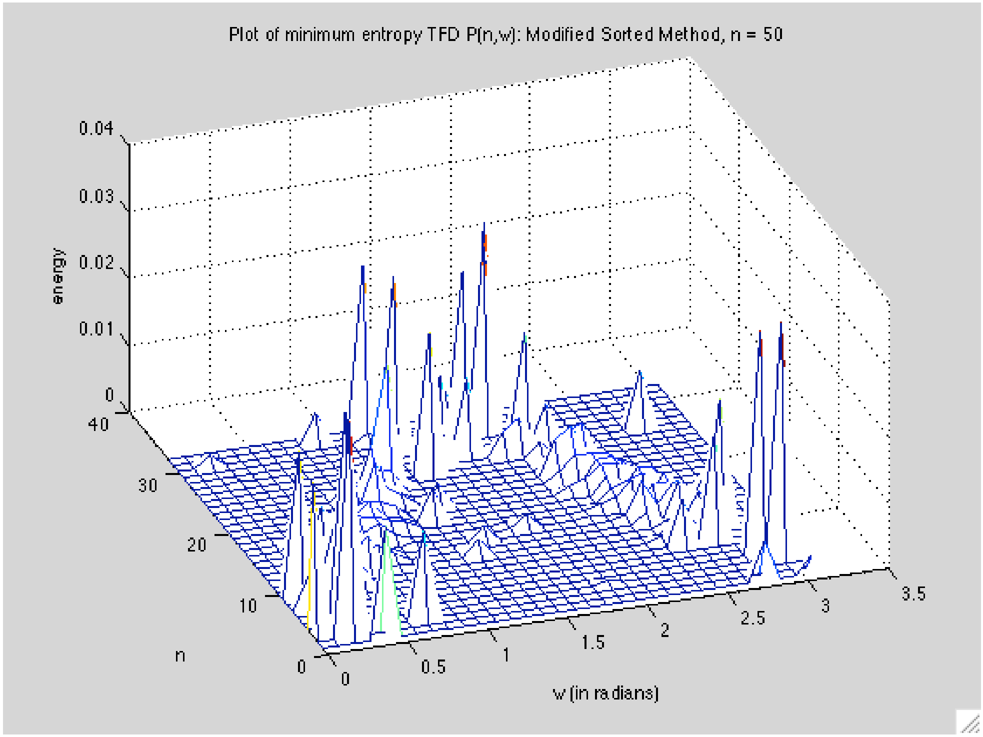

Figure 5.30: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 50 for the double chirp signal

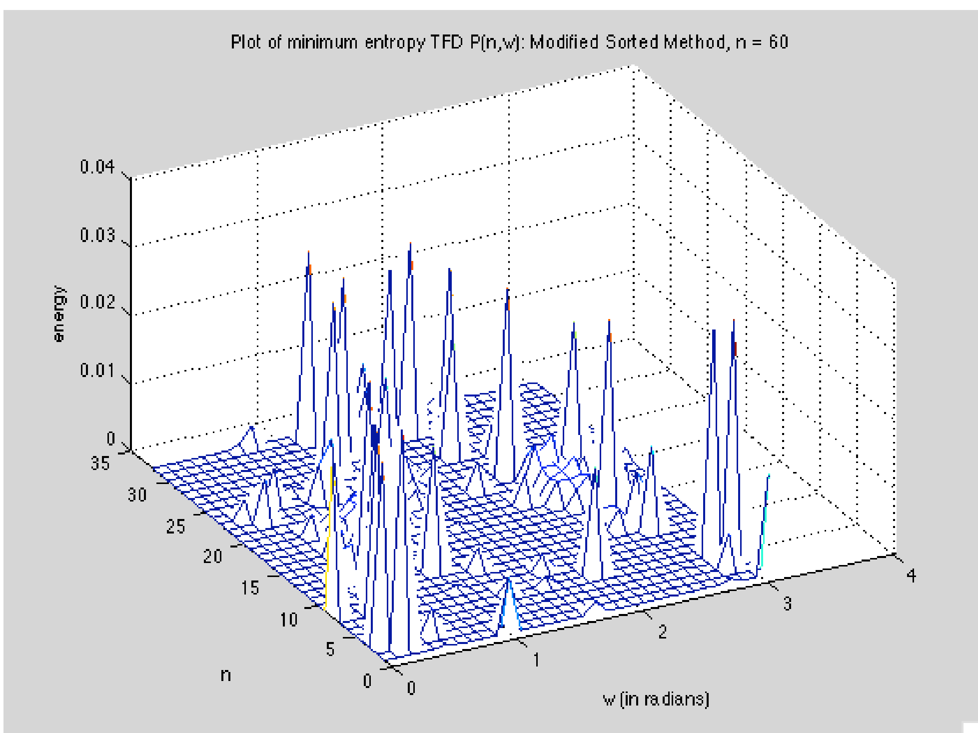

Figure 5.31: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with n = 60 for the double chirp signal

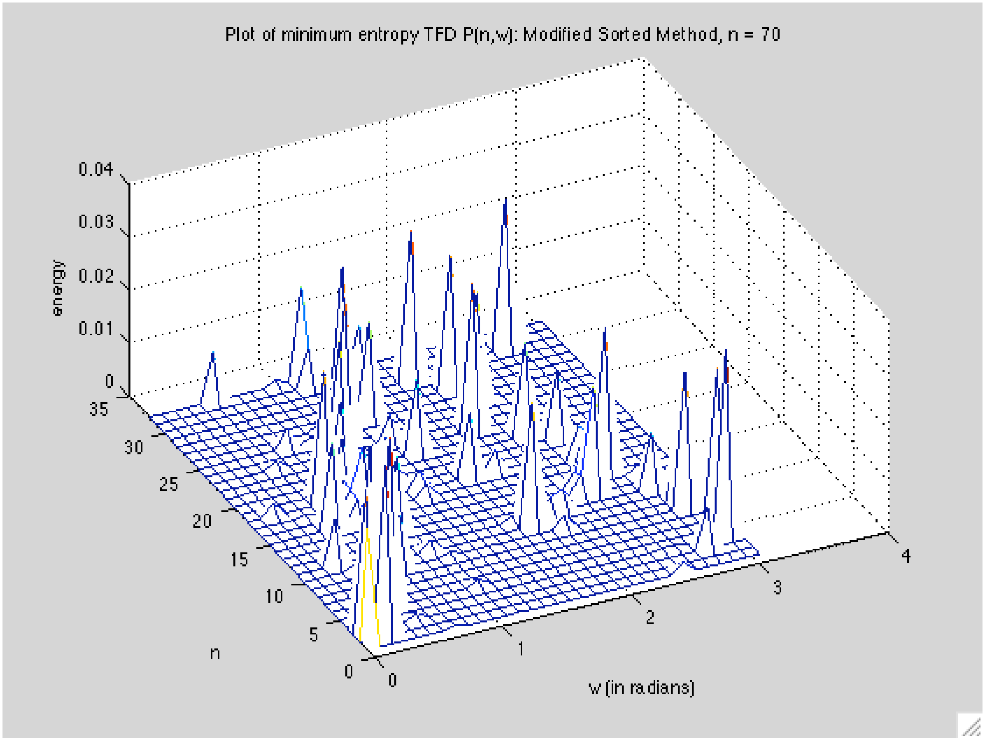

Figure 5.32: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with $n = 70$ for the double chirp signal

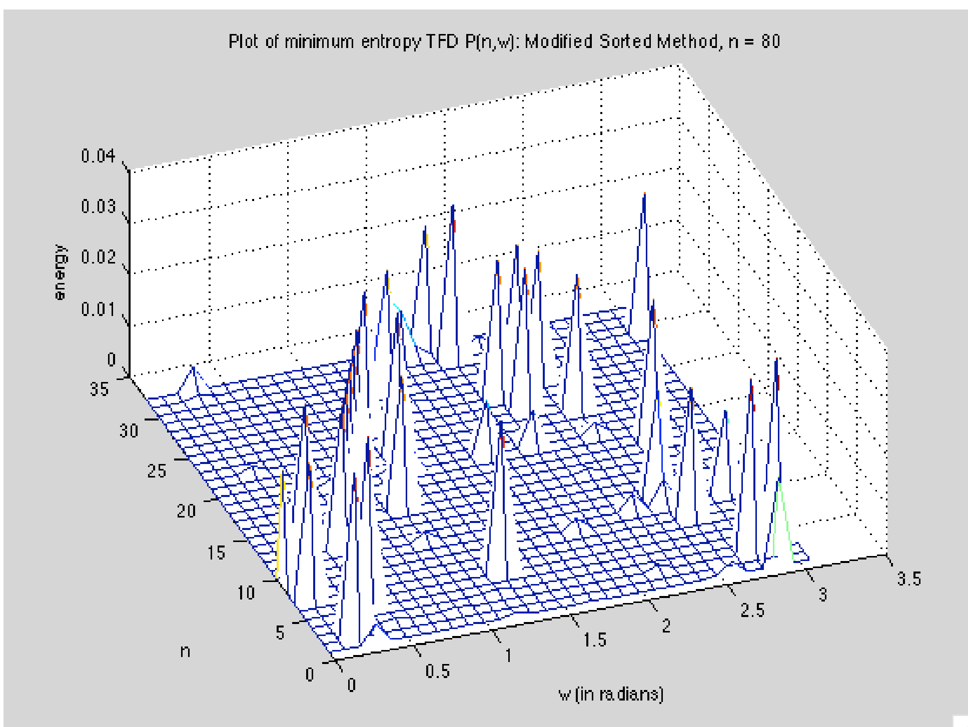

Figure 5.33: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the modified sorted rectangle selection method with $n = 80$ for the double chirp signal

Finally, the minimum entropy algorithm was performed using the modified sorted rectangle selection method with several different values of *n* for the 32-point double chirp signal using the MCE-PTFD as the prior. The MCE-PTFD prior has a total entropy of 5.5150 and 0 zero-points. Table 5.6 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the modified sorted rectangle selection method for different values of *n*. Figures 5.34 – 5.38 show the final minimum entropy TFDs for the modified sorted rectangle method for *n* = 40, 50, 60, 70, and 80.

Table 5.6: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the double chirp signal using the modified sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
35	844	3.9945
40	846	3.9997
45	845	4.0093
50	876	3.8542
55	884	3.8791
60	876	3.8326
65	889	3.8102
70	902	3.7613
75	925	3.7114
80	929	3.7201

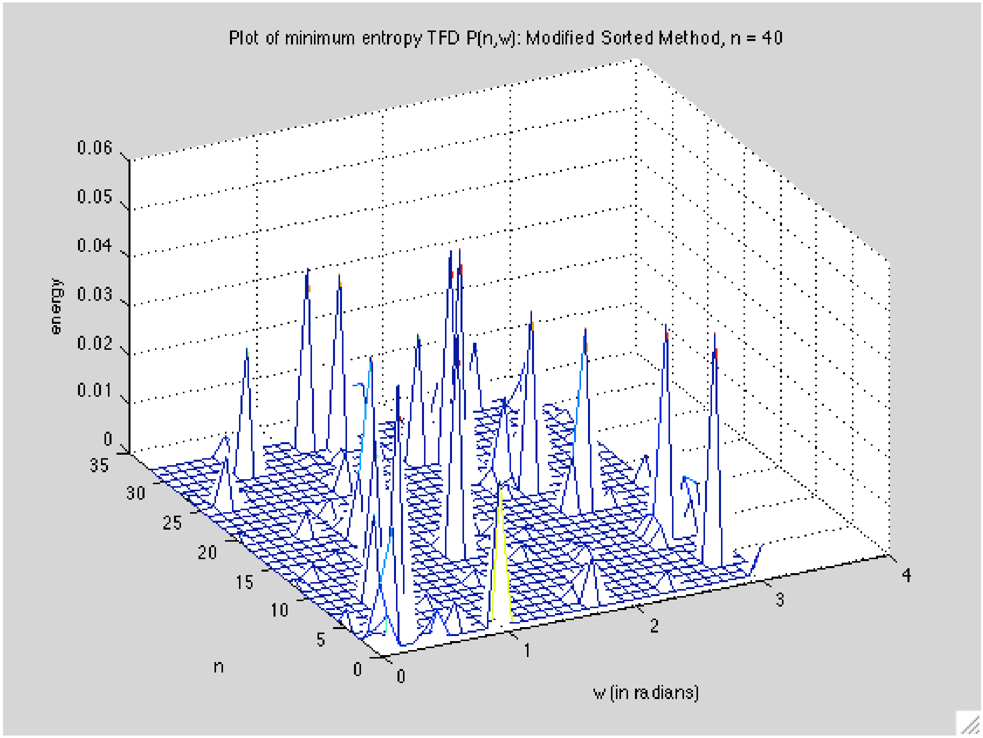

Figure 5.34: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 40 for the double chirp signal

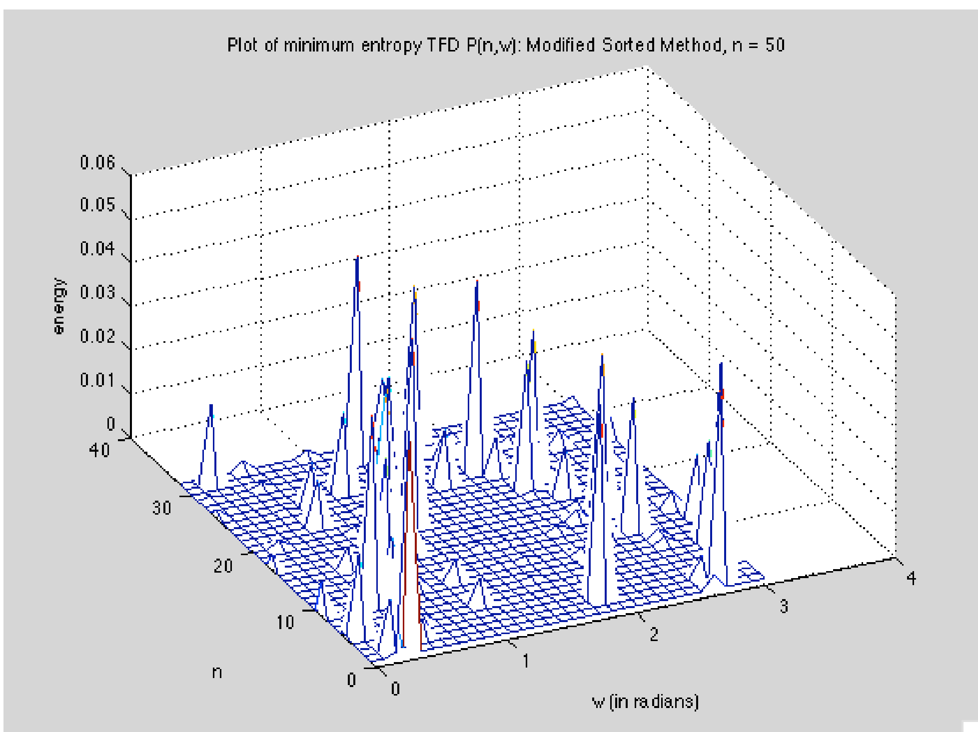

Figure 5.35: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 50 for the double chirp signal

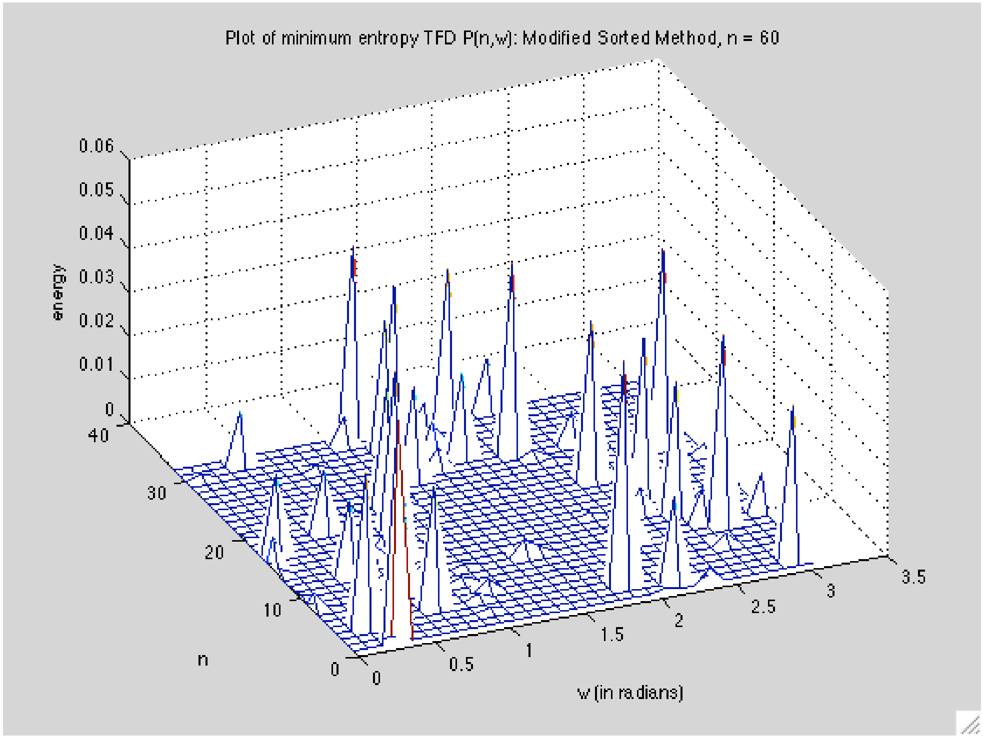

Figure 5.36: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 60 for the double chirp signal

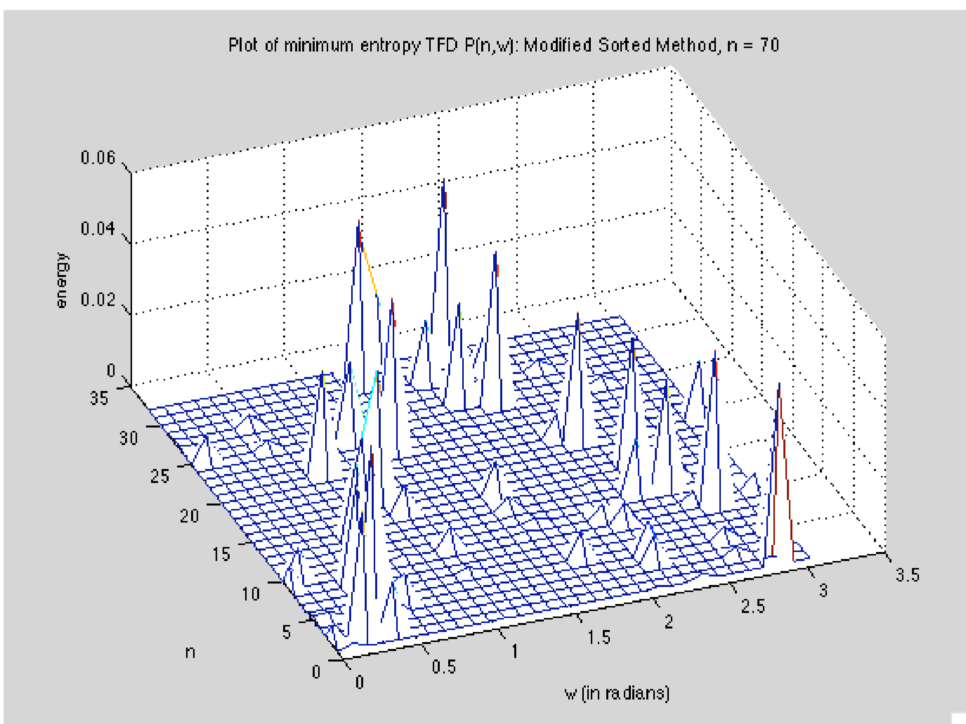

Figure 5.37: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with n = 70 for the double chirp signal

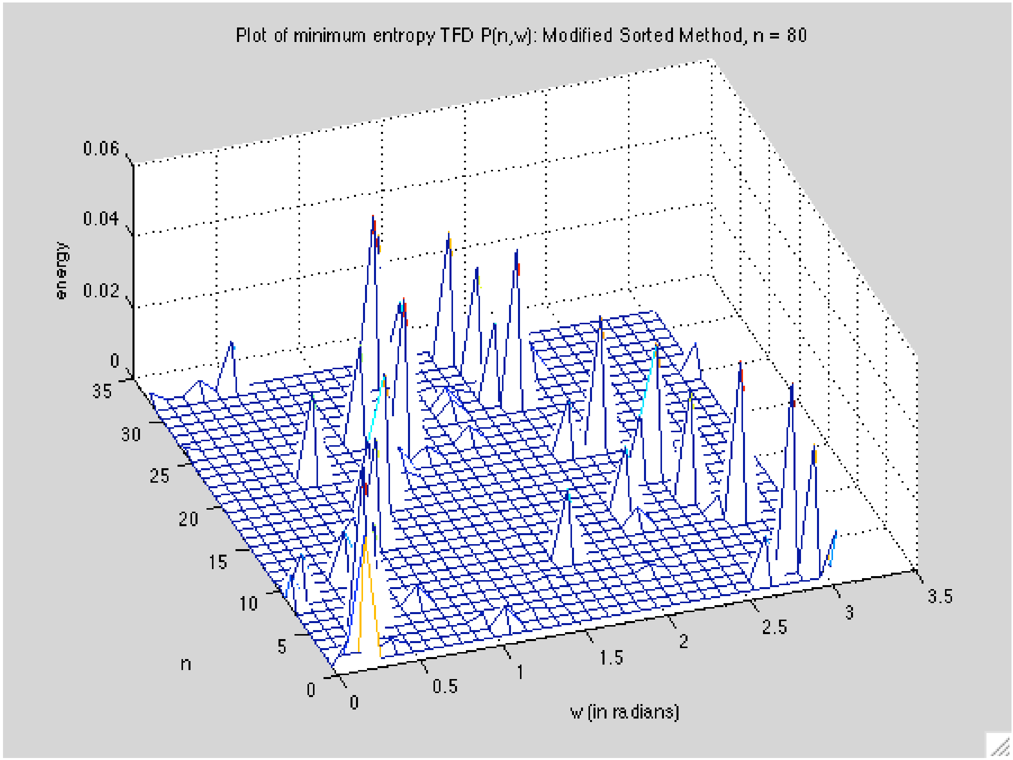

Figure 5.38: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the modified sorted rectangle selection method with *n* = 80 for the double chirp signal

5.2.4 Conclusions

From the results, it is evident that the modified sorted rectangle method reduces the number of outlier spikes, but does not completely eliminate them. Several of the final minimum entropy distributions appear to be quite meaningful for their particular signals.

The spikes are not completely eliminated for several reasons. In each case, the value of *n* is 80 or less. In the initial distributions, there are more than 80 points that are not near-zero points. Although it is impossible to have near-zero points for all four corners of the rectangle, it is still possible for opposing corners of the rectangle to have a significant amount of energy. As a result, higher energy values are still moved to low energy locations. A possible solution for this issue is to add a check that determines if the opposing corners are greater than a certain value. If so, the rectangle should not be minimized. The additional check should significantly reduce the number of outlier spikes.

The value of n has a significant impact on the final distribution. As expected, the number of zero-points generally increases and the total entropy generally decreases as the value of n increases. In the examples, the best values of n tend to be near the expected number of maximum energy points for the TFD of the signal. For the simple chirp signal, this value is around 32. For the chirp with sinusoid and double chirp signals, the value is around 64 since both signals have two frequency components at each time value. In addition, as n increases past the number of expected maximum points, the number of outlier spikes increases. For very small value of n (i.e. between 1-15 for the example signals), the final distributions tend to have n isolated spikes and are not significantly different from the initial TFD.

For the simple chirp signal, the final distributions appear to be decent representations of the signal. The TFD obtained using the spectrogram prior and n = 35 looks especially good. The spectrogram prior generally results in better minimum entropy distributions than the MCE-PTFD prior. This trend could be due to the fact that the maximum values in the spectrogram prior all lay along the chirp time-frequency diagonal. In the MCE-PTFD, the maximum values are not always along this diagonal.

For the simple chirp signal, the final distributions obtained using the spectrogram prior appear to be meaningful while the TFDs obtained using the MCE-PTFD prior appear to lose the chirp portion of the signal. The TFDs found using the spectrogram prior and n = 60 and 70 appear to be especially meaningful. Again, the spectrogram representations are probably better due to the fact that the maximum energy of the initial TFD is concentrated along the chirp diagonal and the sinusoidal frequency. In the MCE-PTFD prior, the energy along the sinusoidal frequency is much greater than the energy along the chirp diagonal. As a result, the chirp diagonal is generally not in the top 100 points of the initial distribution and is moved to the sinusoid. A possible solution to this issue is to split the initial TFD into two separate distributions, one containing the chirp portion of the signal and one containing the sinusoid portion, and perform the minimum entropy algorithm on each TFD individually.

For the double chirp signal, the representations appear to be decent, but still contain a large number of outlier spikes. Neither the spectrogram prior nor the MCE-PTFD appears to have a strong advantage over the other.

The modified sorted rectangle selection method does not work well for signals that have multiple distinct portions in the frequency domain that are at different energy levels. As in the case for the chirp with sinusoid signal using the MCE-PTFD, the information for the lower energy portion of the signal will be lost as it is moved to the higher energy points. The solution for this issue may be to break the initial TFD into different parts, one for each portion of the signal. The algorithm could then be carried out on each TFD and the resulting distributions could be combined.

5.3 SUPER SORTED RECTANGLE SELECTION METHOD

The super sorted rectangle selection method is a variation of the modified sorted rectangle selection method. As in the modified sorted method, the algorithm sorts all of the points in the initial distribution from the largest energy value to the smallest energy value. Next, it takes the n highest energy points and the p-n lowest energy points where p is the total number of points in the TFD and places them in separate lists. The method then selects rectangles using the same pattern as the modified sorted method. For each rectangle, it performs the following two checks. First, if all four points in the rectangle are less than the n^{th} value, the rectangle will be skipped and will not be minimized. This check should prevent the near-zero points from being summed and creating outlier spikes at low-energy points. Secondly, if either of the two opposing corners in the rectangle have both points greater than or equal to the n^{th} energy value, the rectangle will be skipped and will not be minimized. This check should prevent energy from being moved from high-energy locations to low-energy locations. Again, the process continues until none of the rectangles that pass the two checks can be minimized any further.

Like the modified sorted method, the super sorted rectangle selection method will not result in a true minimum entropy time-frequency distribution since all of the possible rectangles are not considered. This property may be advantageous for signals that require more non-zero points than the minimum entropy TFD will contain to accurately represent. Remember that a true minimum entropy distribution will always contain approximately the same number of non-zero points as there are marginal constraints.

To evaluate the super sorted rectangle selection method, we will again consider three signals: the simple chirp signal, the chirp with sinusoid signal, and the double chirp signal. The super sorted rectangle selection method will be employed on all three signals for different values of n using both spectrogram and MCE-PTFD priors.

5.3.1 Simple Chirp Example

First, the minimum entropy algorithm was performed using the super sorted rectangle selection method with several different values of n for the 32-point simple chirp signal using the spectrogram as the prior. The spectrogram prior has a total entropy of 5.1572 and 0 zero-points. Table 5.7 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the super sorted rectangle selection method for different values of *n*. Figures 5.39 – 5.44 show the final minimum entropy TFDs for the super sorted rectangle method for *n* = 350, 360, 370, 380, 390, and 400.

The best results were obtained using large values of *n* (between 200 and 400). The minimum entropy distributions that resulted from small values of *n* contained a large number of outlier spikes. Larger values of *n* resulted in a very small number of outlier spikes and produced the most meaningful representations. This trend occurred because the medium-energy points were not included in the high-energy list for small values of *n*. Thus, the algorithm treated these points as low-energy points and moved them to low-energy locations.

Table 5.7: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the spectrogram as the prior for the simple chirp signal using the super sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
350	3.8301	941
360	3.7184	954
370	3.6735	950
380	3.8160	943
390	3.7361	948
400	3.8200	944

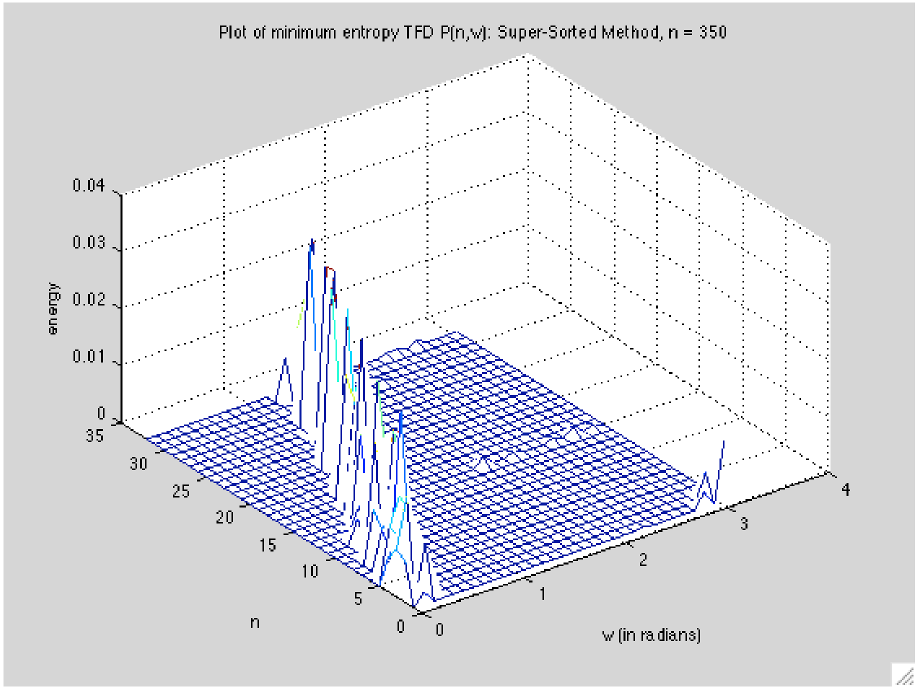

Figure 5.39: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with $n = 350$ for the simple chirp signal

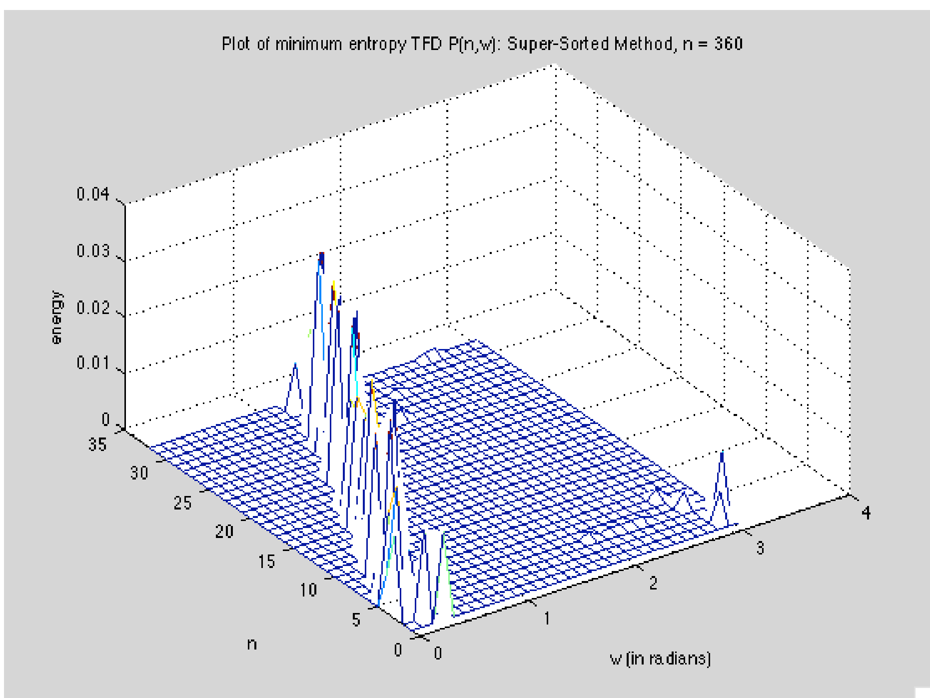

Figure 5.40: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with $n = 360$ for the simple chirp signal

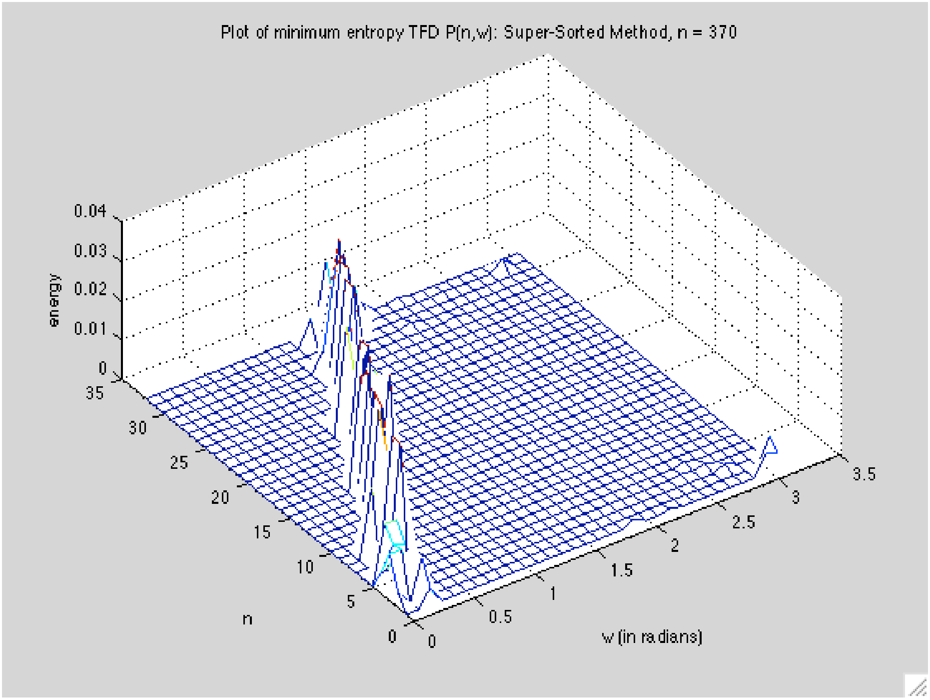

Figure 5.41: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 370 for the simple chirp signal

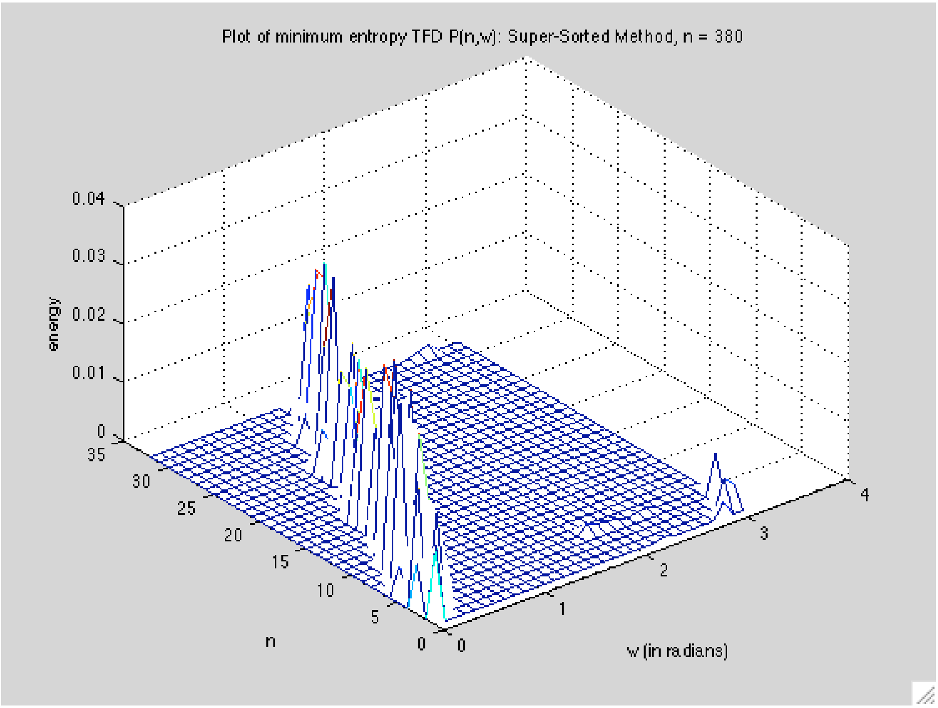

Figure 5.42: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 380 for the simple chirp signal

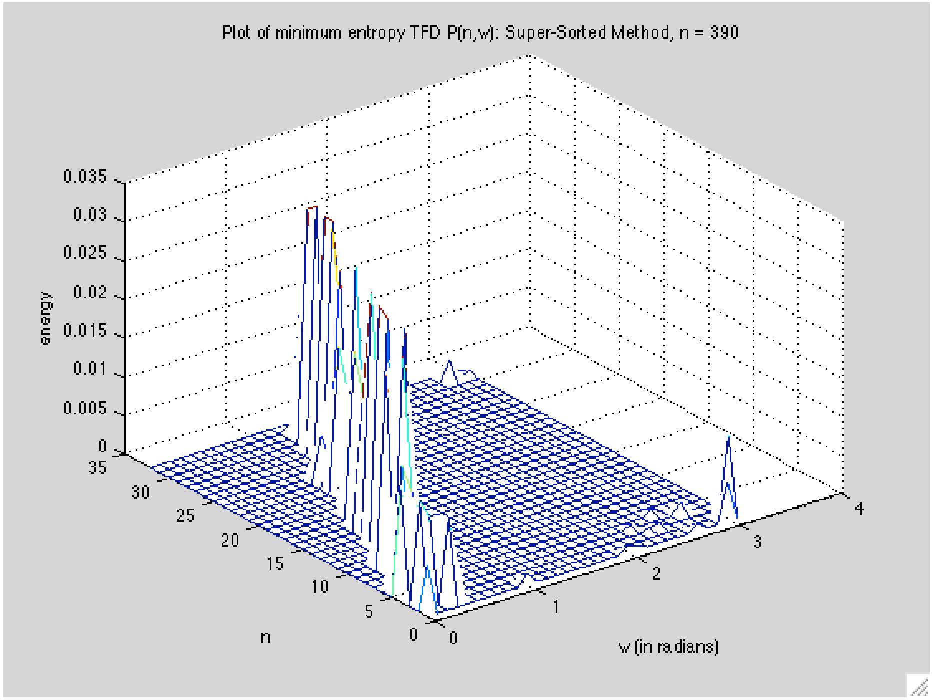

Figure 5.43: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 390 for the simple chirp signal

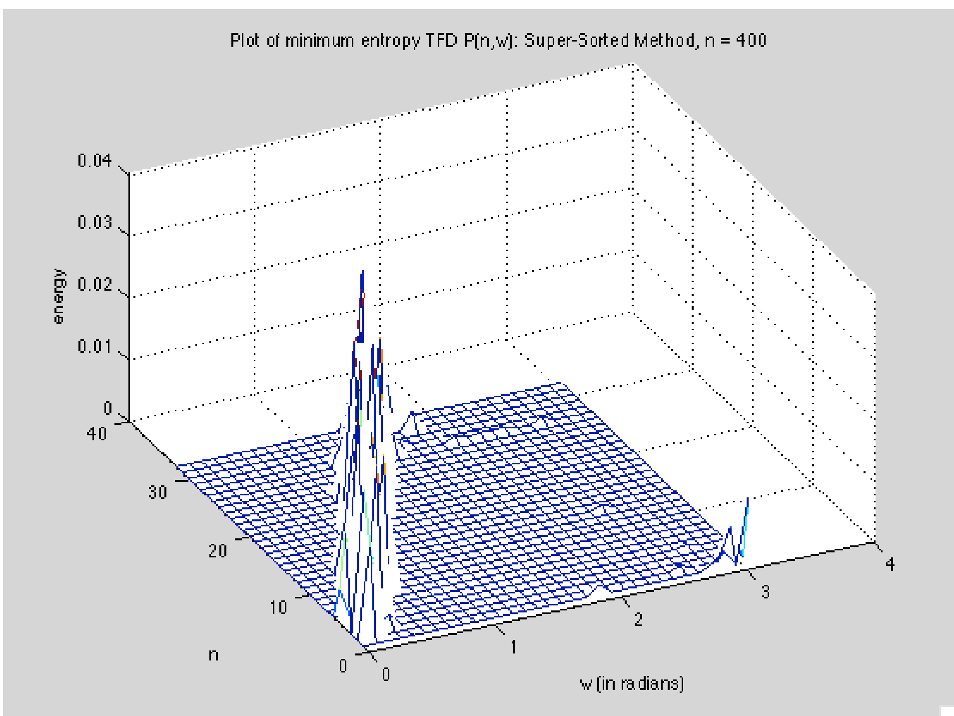

Figure 5.44: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 400 for the simple chirp signal

Next, the minimum entropy algorithm was performed using the super sorted rectangle selection method with several different values of *n* for the 32-point simple chirp signal using the MCE-PTFD as the prior. The MCE-PTFD prior has a total entropy of 5.2584 and 0 zero-points. Table 5.8 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the super sorted rectangle selection method for different values of *n*. Figures 5.45 – 5.50 show the final minimum entropy TFDs for the super sorted rectangle method for *n* = 200, 210, 220, 230, 240, and 250.

As with the spectrogram prior, large values of *n* (between 150 and 350) produced the best distributions. Using a smaller value of *n* resulted in a distribution that contained a large number of outlier spikes.

Table 5.8: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the simple chirp signal using the super sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
200	3.7876	953
210	3.9128	951
220	3.8570	950
230	3.8012	958
240	3.9067	949
250	3.8242	953

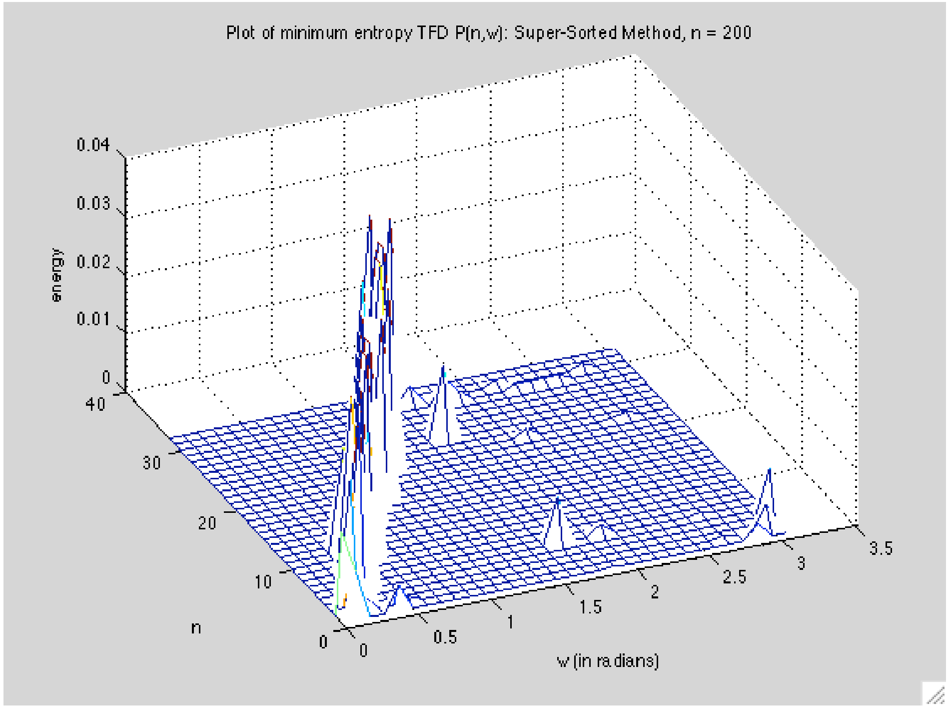

Figure 5.45: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with *n* = 200 for the simple chirp signal

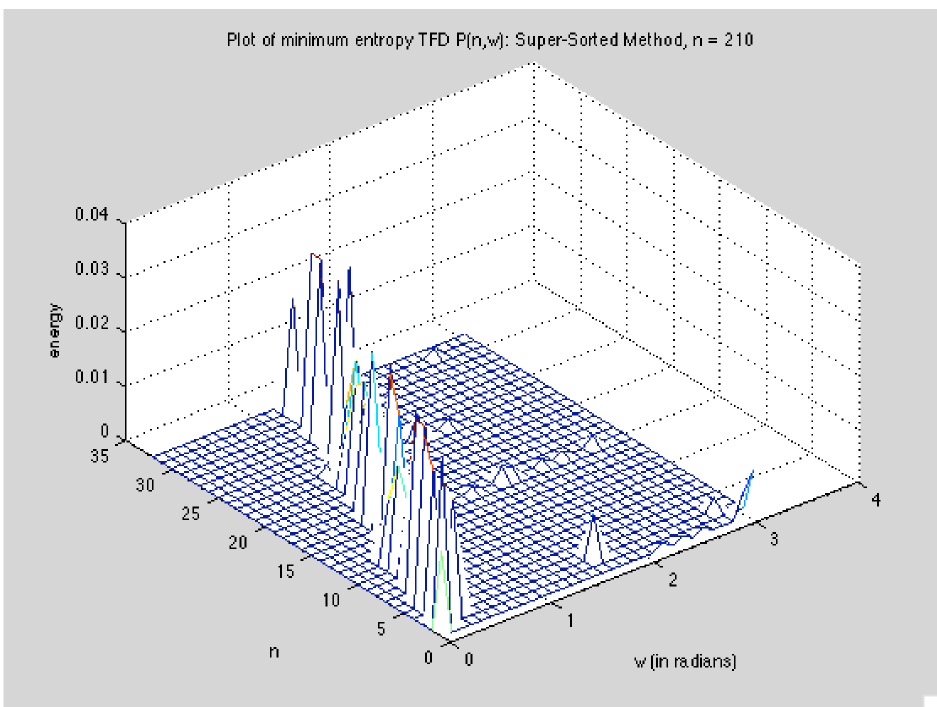

Figure 5.46: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with *n* = 210 for the simple chirp signal

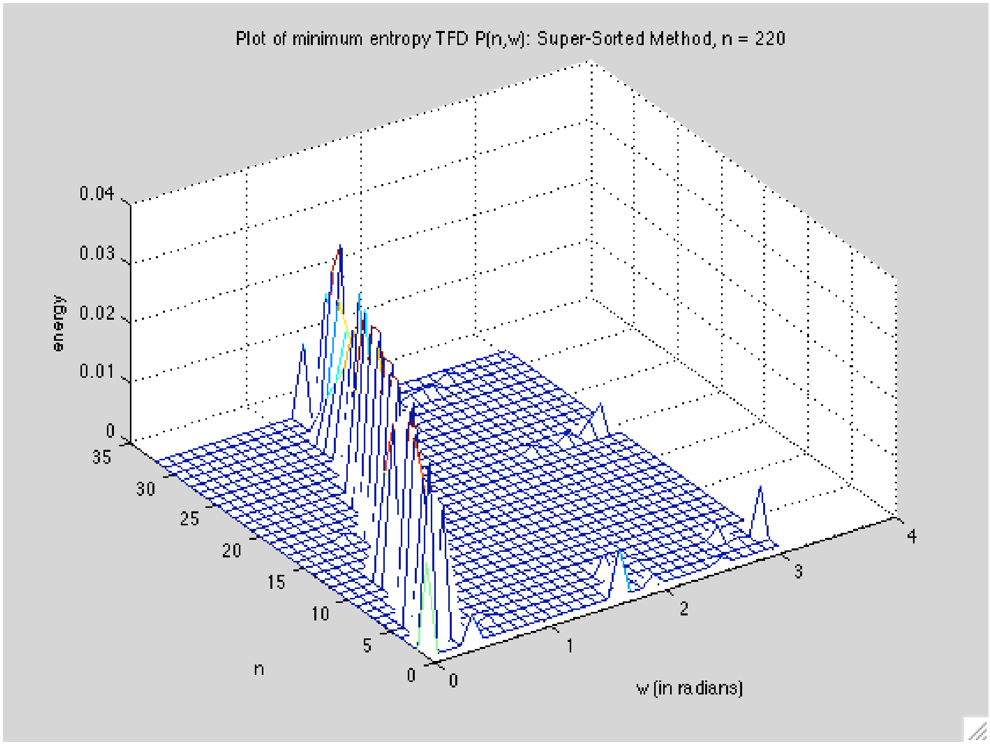

Figure 5.47: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with $n = 220$ for the simple chirp signal

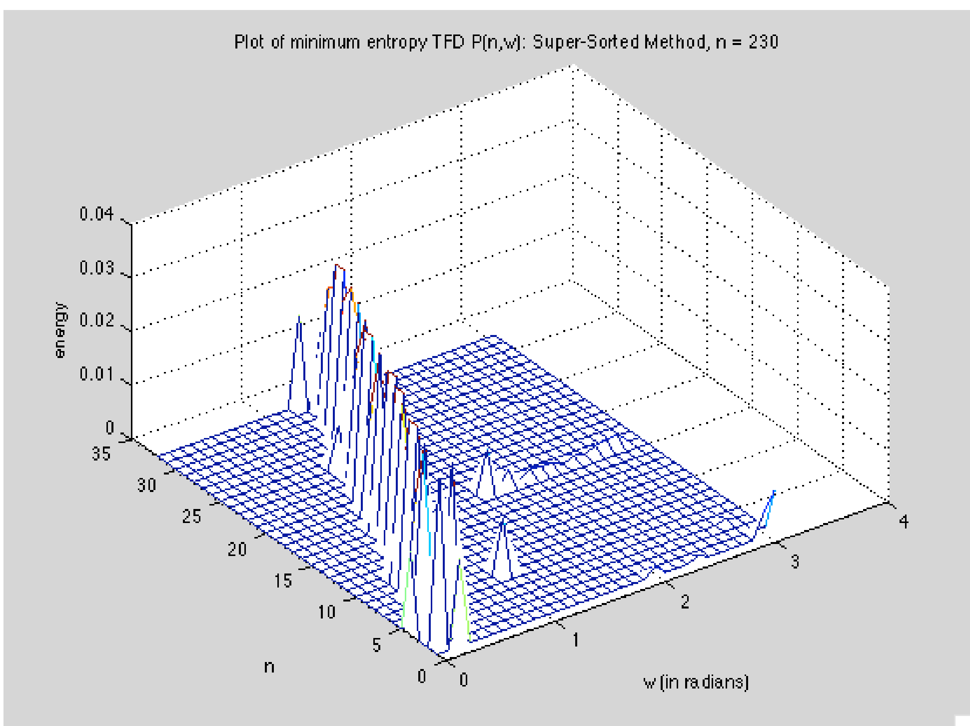

Figure 5.48: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with $n = 230$ for the simple chirp signal

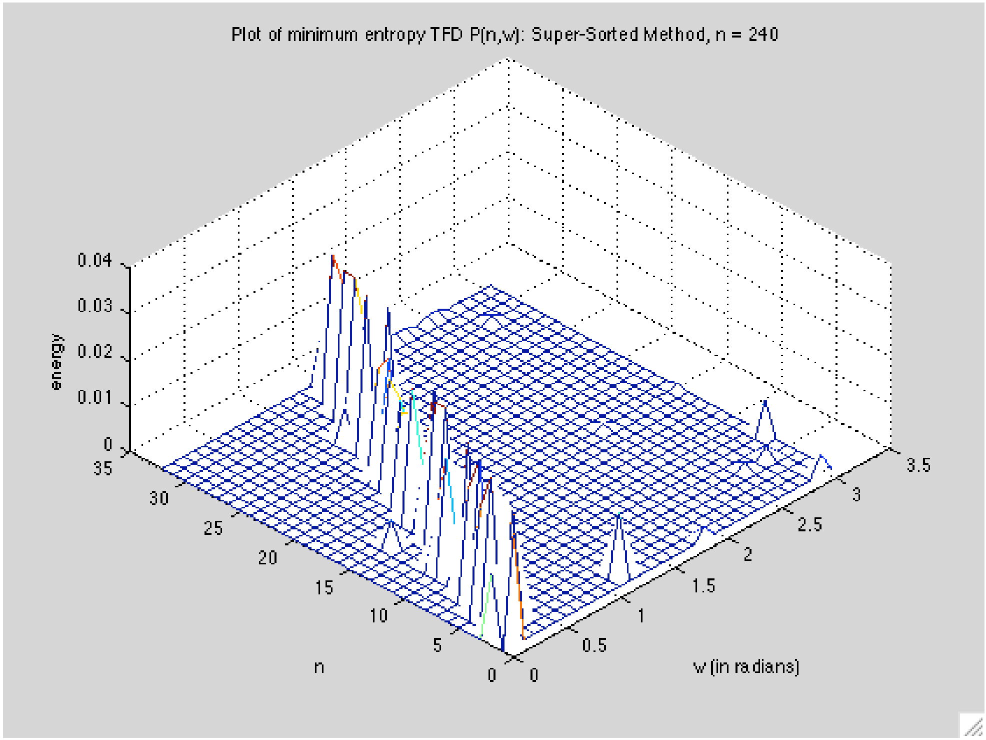

Figure 5.49: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with n = 240 for the simple chirp signal

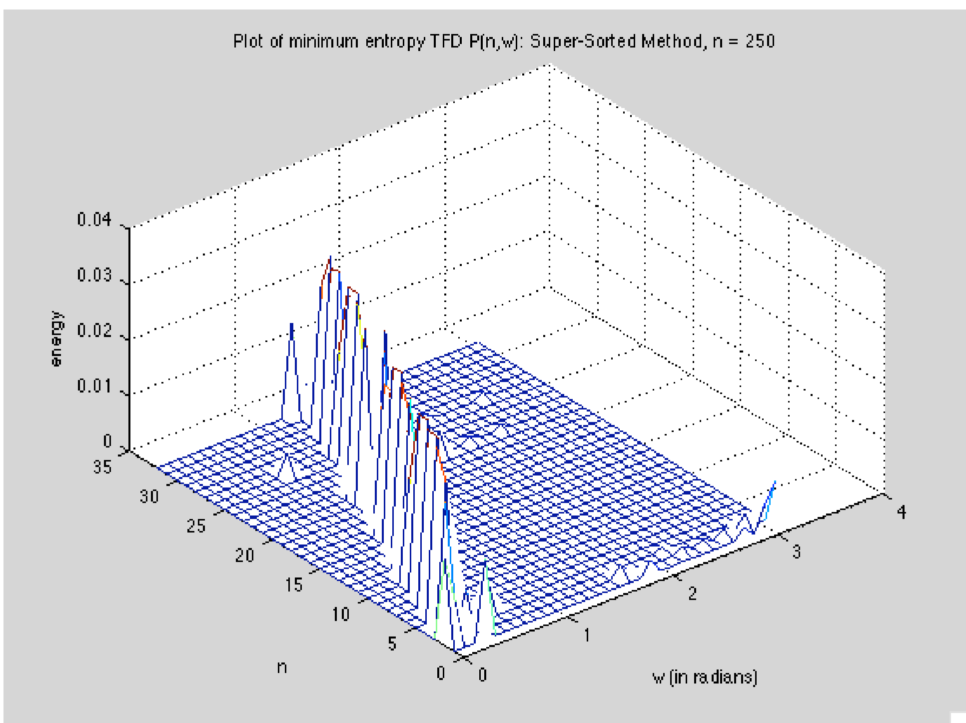

Figure 5.50: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with n = 250 for the simple chirp signal

5.3.2 Chirp with Sinusoid Example

Next, the minimum entropy algorithm was performed using the super sorted rectangle selection method with several different values of *n* for the 32-point chirp with sinusoid signal using the spectrogram as the prior. The spectrogram prior has a total entropy of 5.5954 and 0 zero-points. Table 5.9 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the super sorted rectangle selection method for different values of *n*. Figures 5.51 – 5.56 show the final minimum entropy TFDs for the super sorted rectangle method for *n* = 450, 460, 470, 480, 490, and 500.

Like before, the best results were obtained using large values of *n* (between 300 and 500). Using a smaller value of *n* resulted in a large number of outlier spikes.

Table 5.9: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the spectrogram as the prior for the chirp with sinusoid signal using the super sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
450	3.8477	947
460	3.5948	953
470	3.7236	947
480	3.7637	952
490	3.9022	939
500	3.9319	939

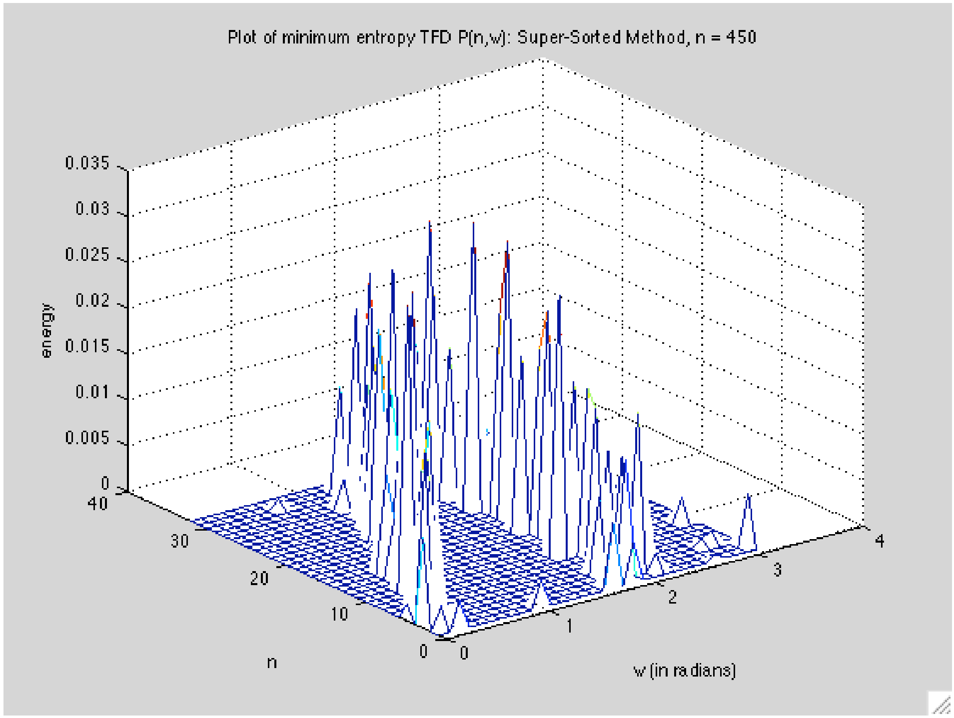

Figure 5.51: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with *n* = 450 for the chirp with sinusoid signal

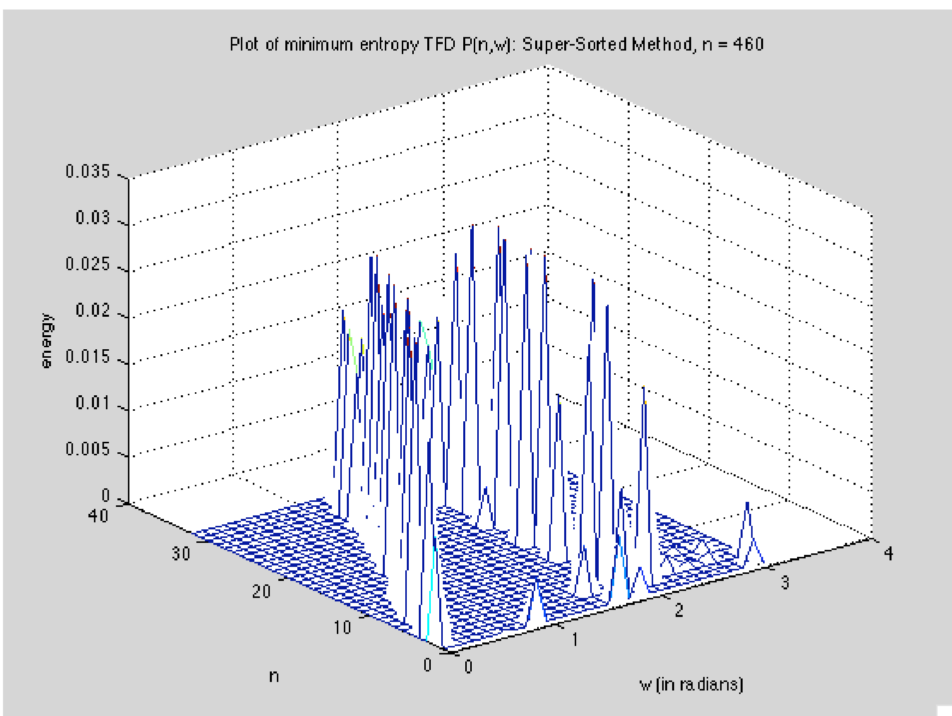

Figure 5.52: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with *n* = 460 for the chirp with sinusoid signal

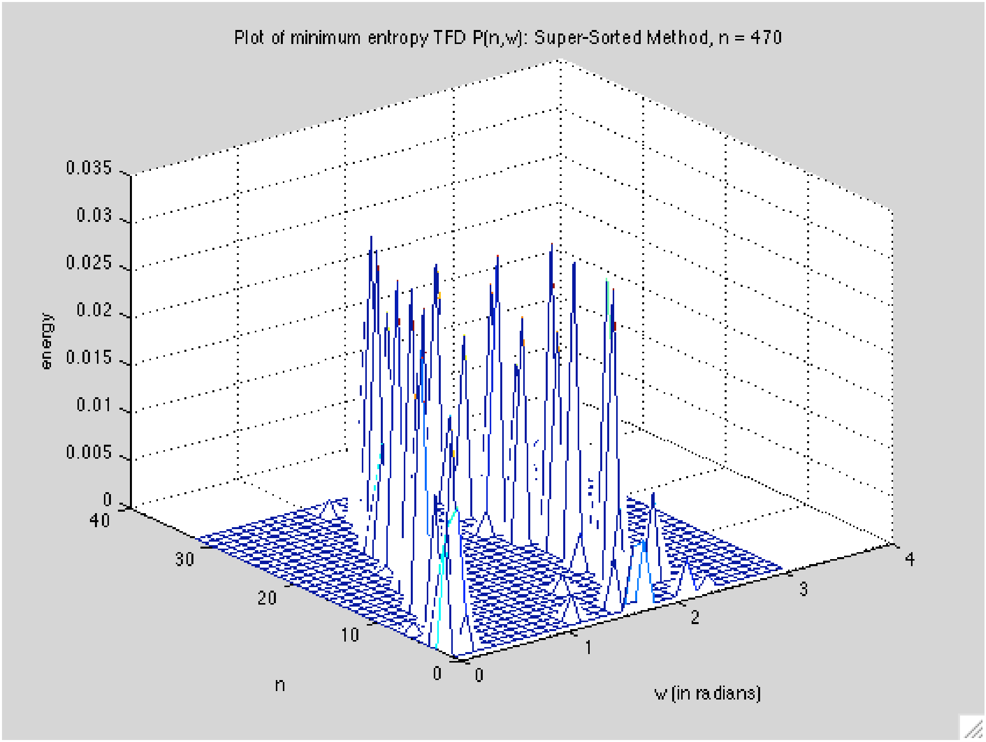

Figure 5.53: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 470 for the chirp with sinusoid signal

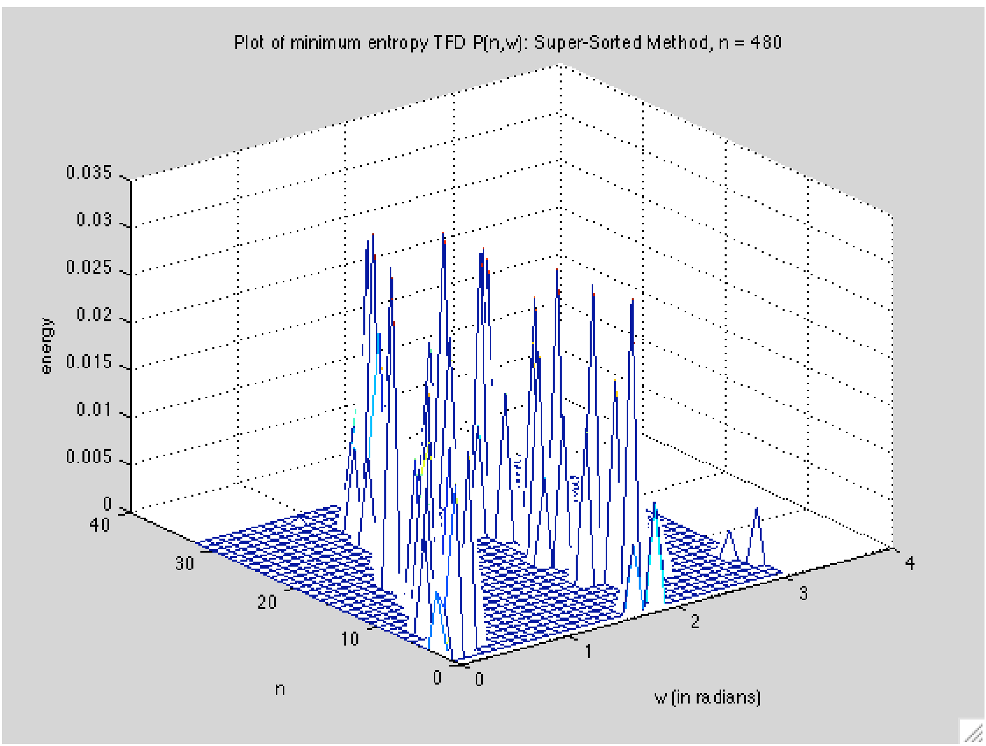

Figure 5.54: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 480 for the chirp with sinusoid signal

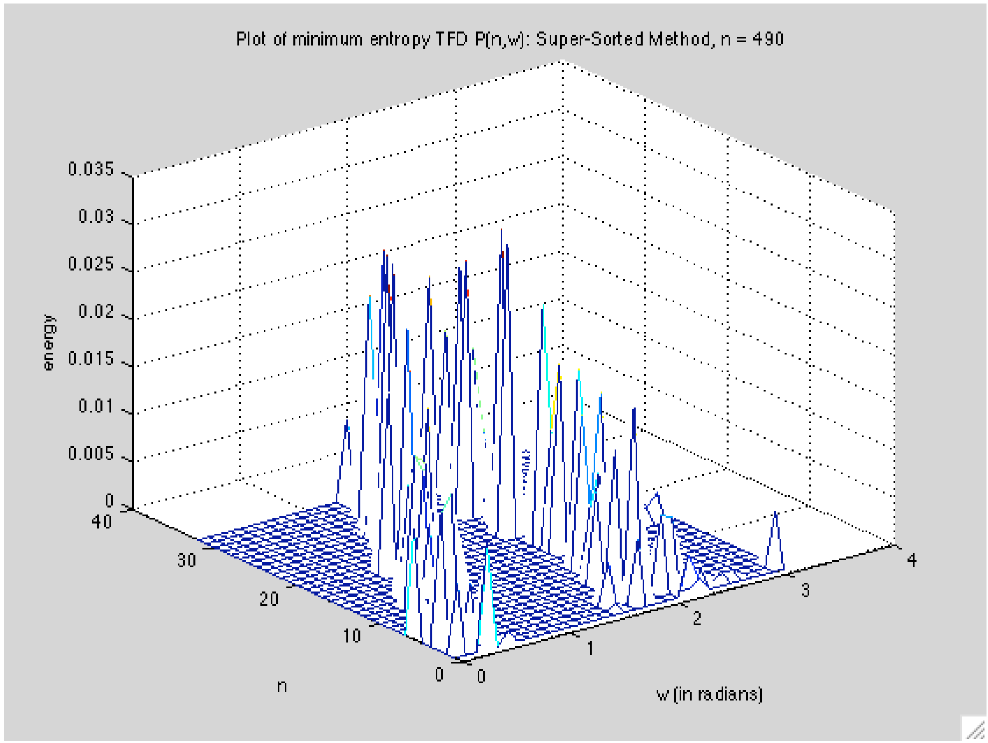

Figure 5.55: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with *n* = 490 for the chirp with sinusoid signal

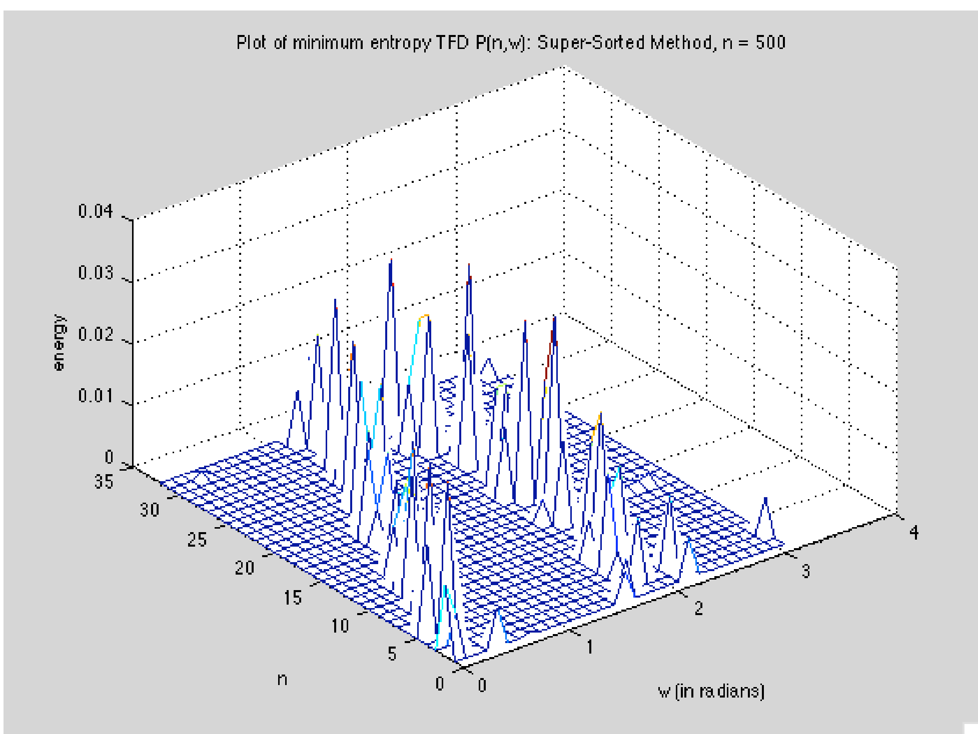

Figure 5.56: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with *n* = 500 for the chirp with sinusoid signal

Next, the minimum entropy algorithm was performed using the super sorted rectangle selection method with several different values of *n* for the 32-point chirp with sinusoid signal using the MCE-PTFD as the prior. The MCE-PTFD prior has a total entropy of 4.7352 and 64 zero-points. Table 5.10 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the super sorted rectangle selection method for different values of *n*. Figures 5.57 – 5.62 show the final minimum entropy TFDs for the super sorted rectangle method for *n* = 250, 260, 270, 280, 290 and 300.

Table 5.10: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the chirp with sinusoid signal using the super sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
250	4.1502	919
260	4.1276	919
270	3.9867	937
280	4.1065	922
290	4.1689	917
300	4.2435	913

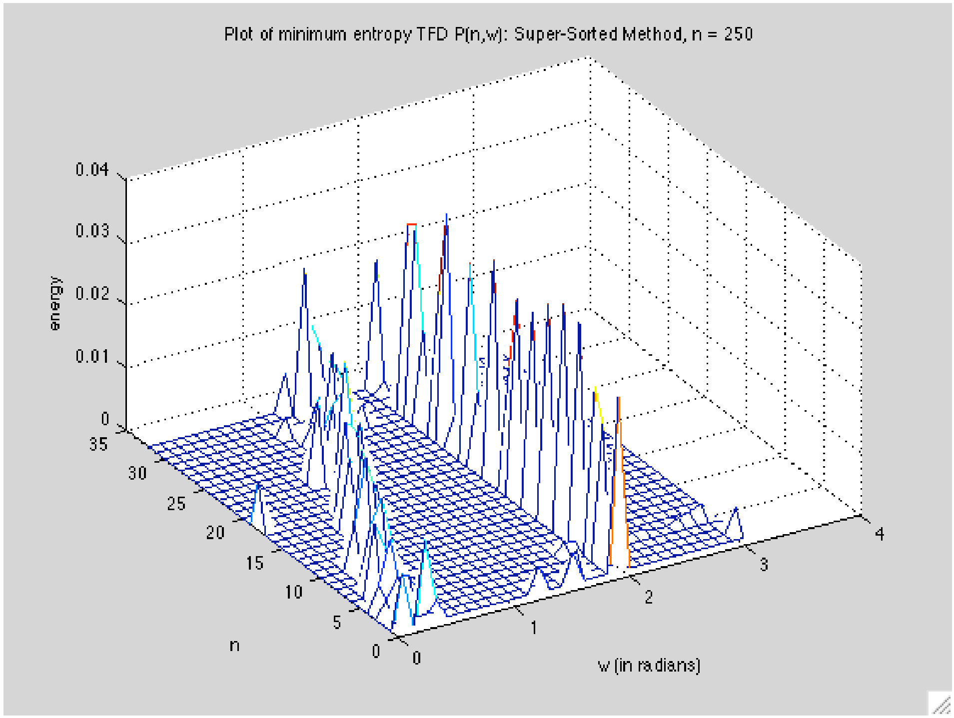

Figure 5.57: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with n = 250 for the chirp with sinusoid signal

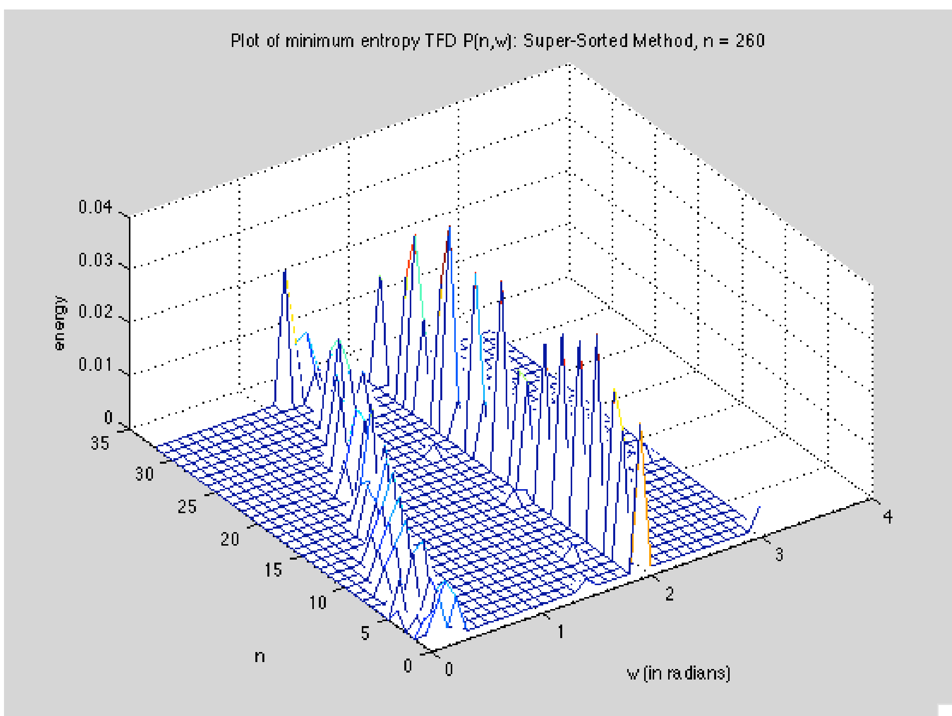

Figure 5.58: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with n = 260 for the chirp with sinusoid signal

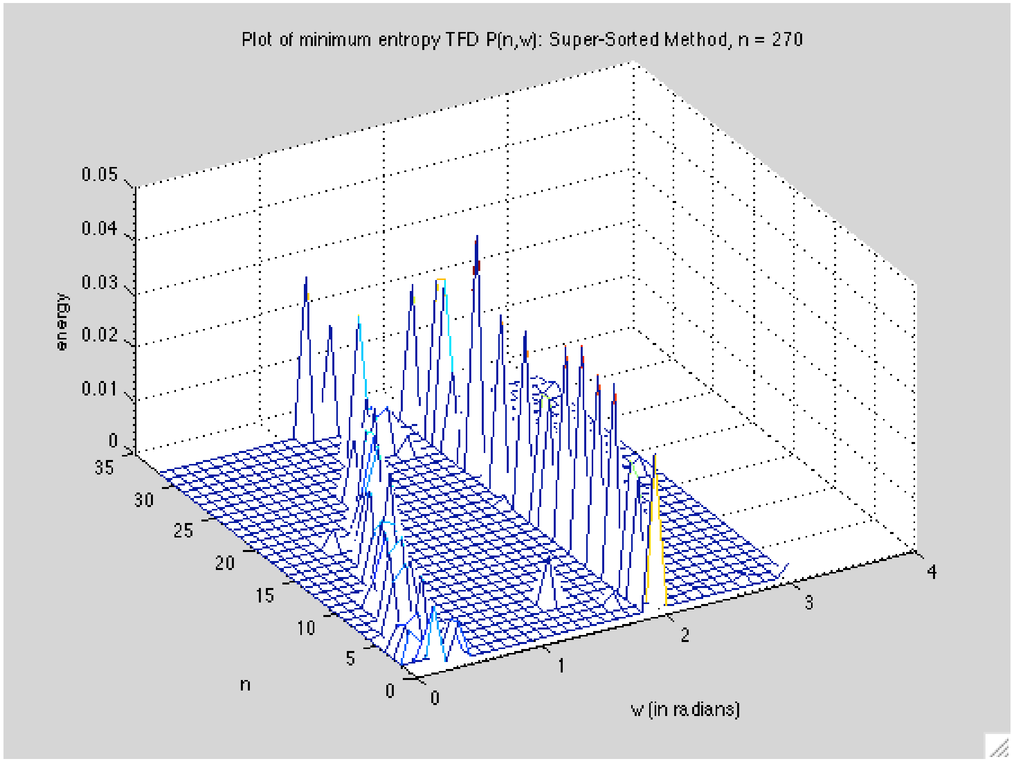

Figure 5.59: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with *n* = 270 for the chirp with sinusoid signal

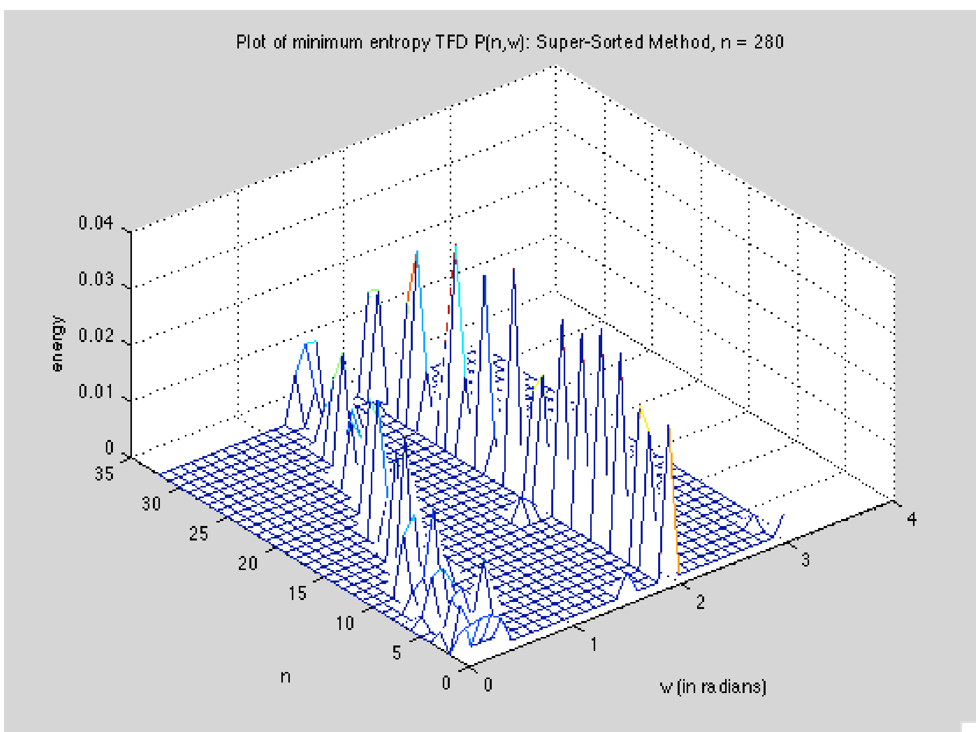

Figure 5.60: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with *n* = 280 for the chirp with sinusoid signal

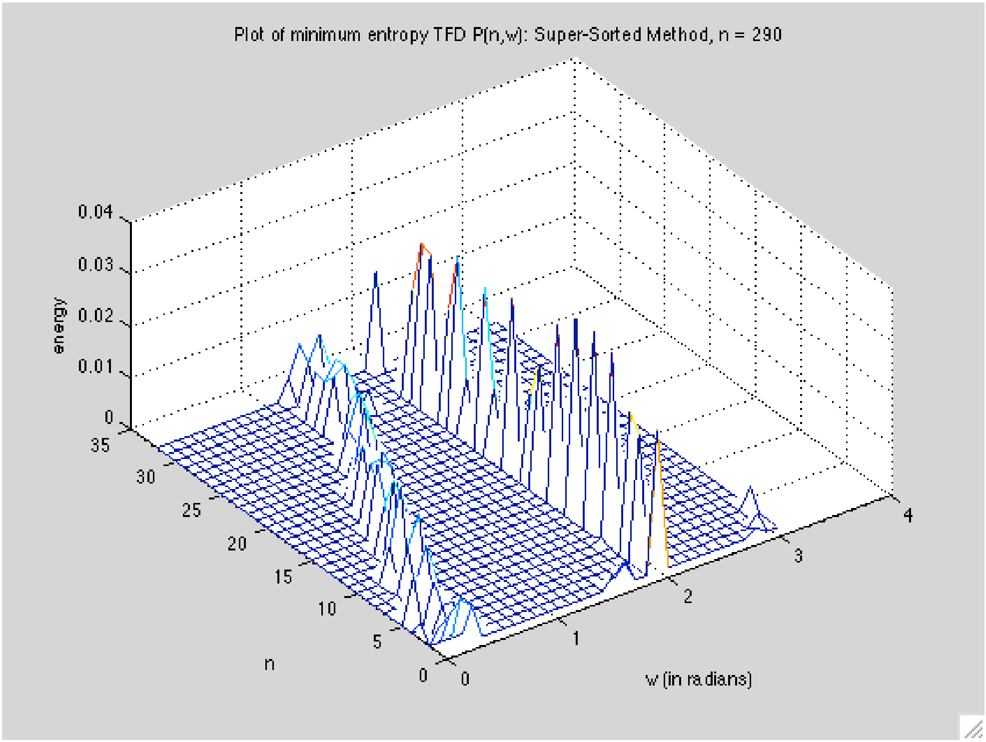

Figure 5.61: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with n = 290 for the chirp with sinusoid signal

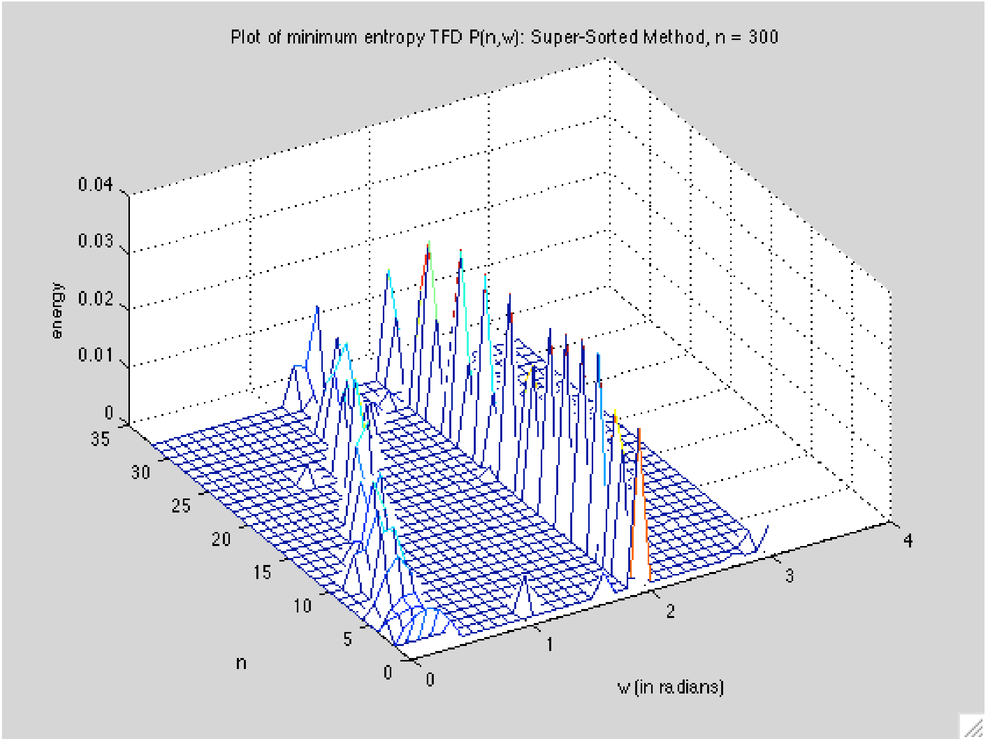

Figure 5.62: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with n = 300 for the chirp with sinusoid signal

5.3.3 Double Chirp Example

Next, the minimum entropy algorithm was performed using the super sorted rectangle selection method with several different values of *n* for the 32-point double chirp signal using the spectrogram as the prior. The spectrogram prior has a total entropy of 5.4956 and 0 zero-points. Table 5.11 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the super sorted rectangle selection method for different values of *n*. Figures 5.63 – 5.68 show the final minimum entropy TFDs for the super sorted rectangle method for *n* = 450, 460, 470, 480, 490, and 500.

Like before, the best results were obtained using large values of *n* (between 300 and 500). Using a smaller value of *n* resulted in a large number of outlier spikes that were caused by medium-energy points being moved to low-energy locations.

Table 5.11: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the spectrogram as the prior for the double chirp signal using the super sorted rectangle selection method for different values of *n*.

n	Total Entropy	Zero-Points
450	3.9177	936
460	3.7772	945
470	3.6137	950
480	3.5687	948
490	3.5969	949
500	3.7854	942

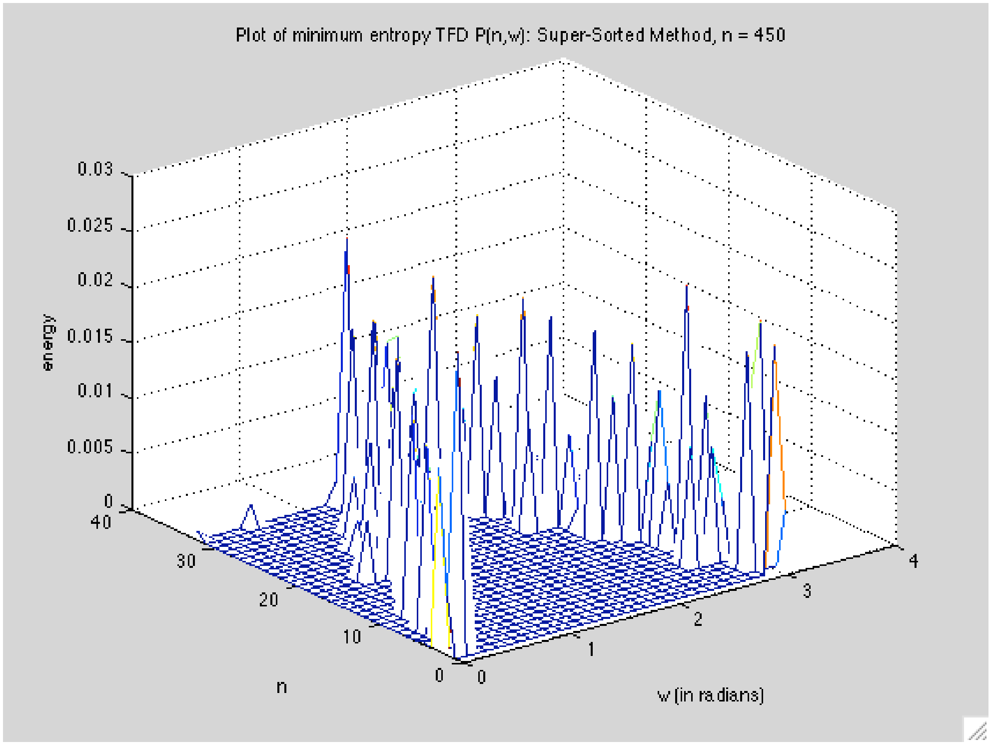

Figure 5.63: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 450 for the double chirp signal

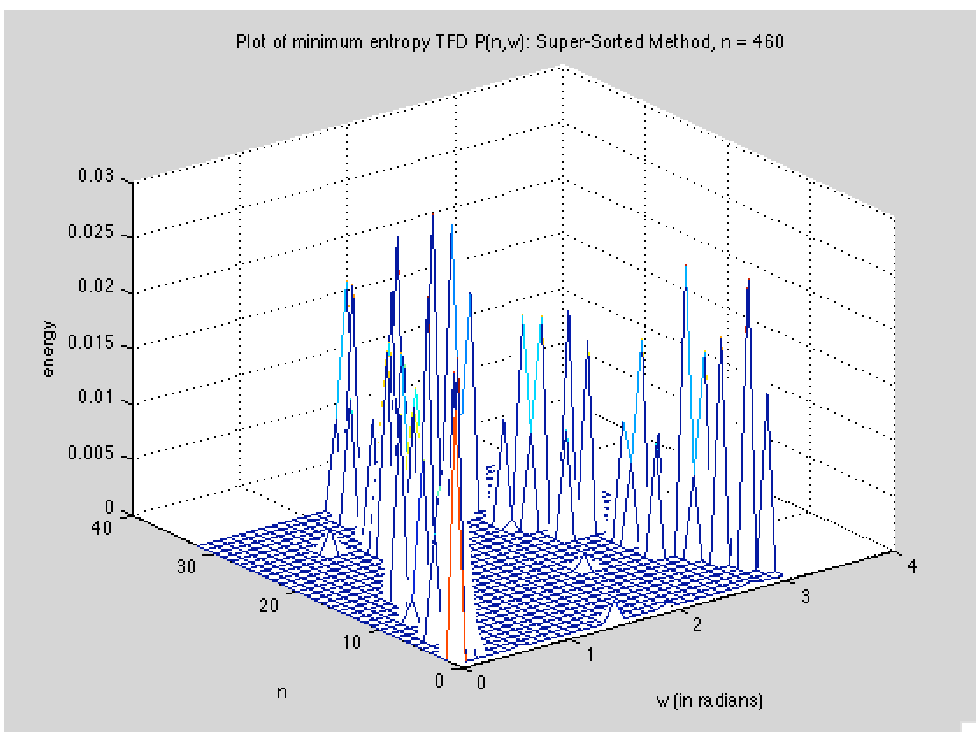

Figure 5.64: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 460 for the double chirp signal

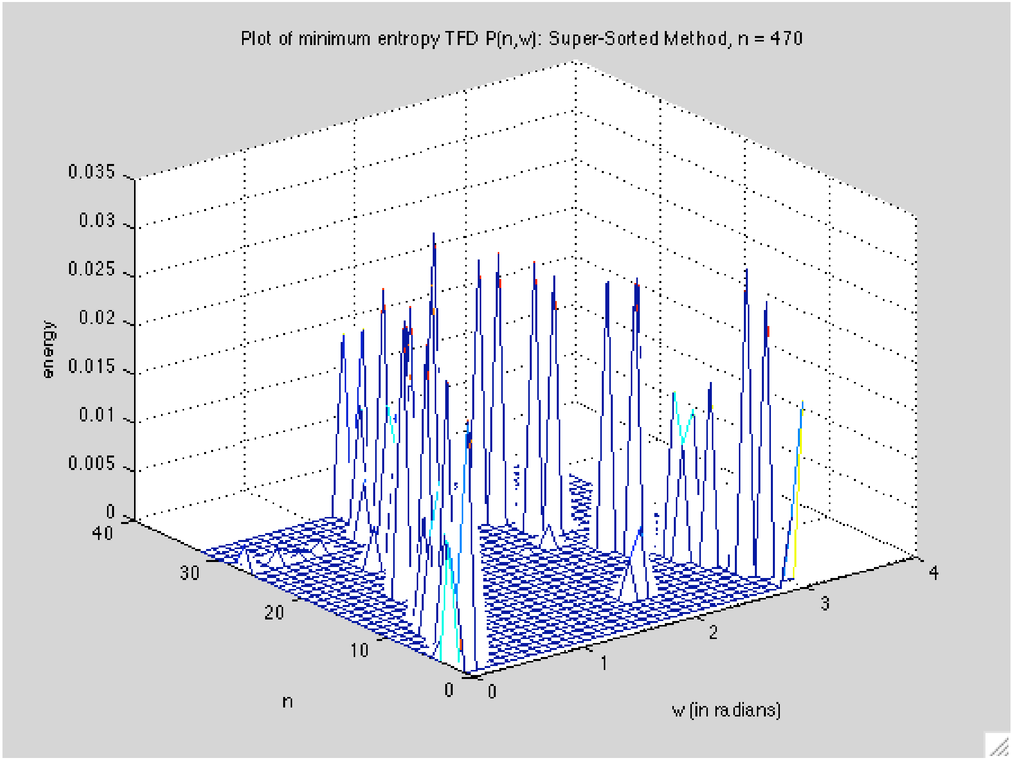

Figure 5.65: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with *n* = 470 for the double chirp signal

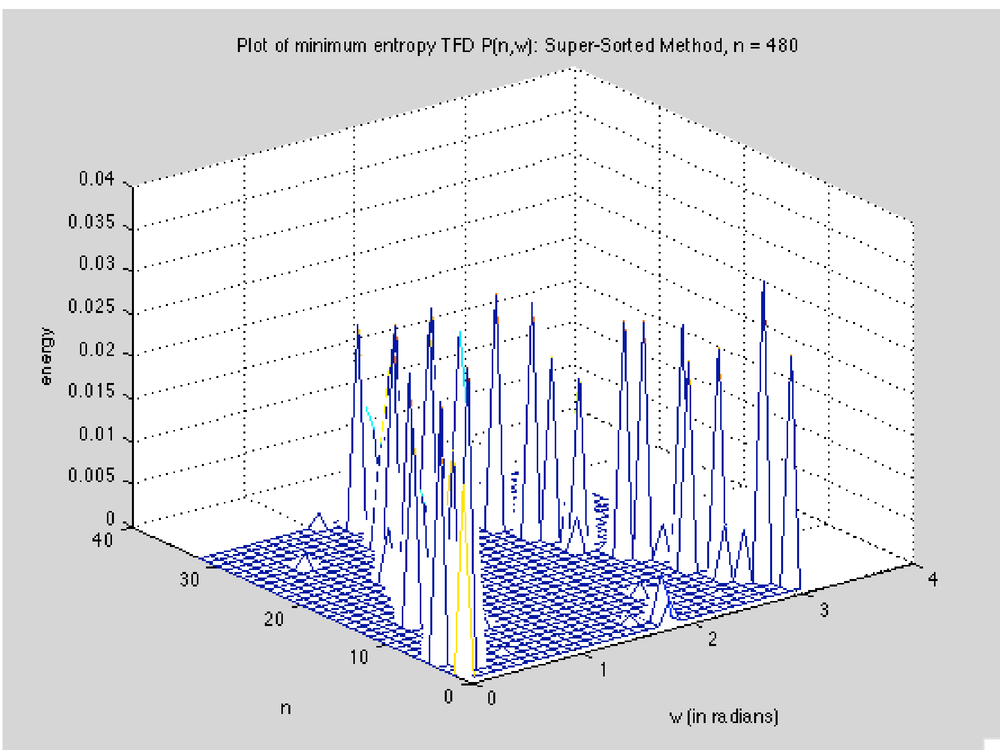

Figure 5.66: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with *n* = 480 for the double chirp signal

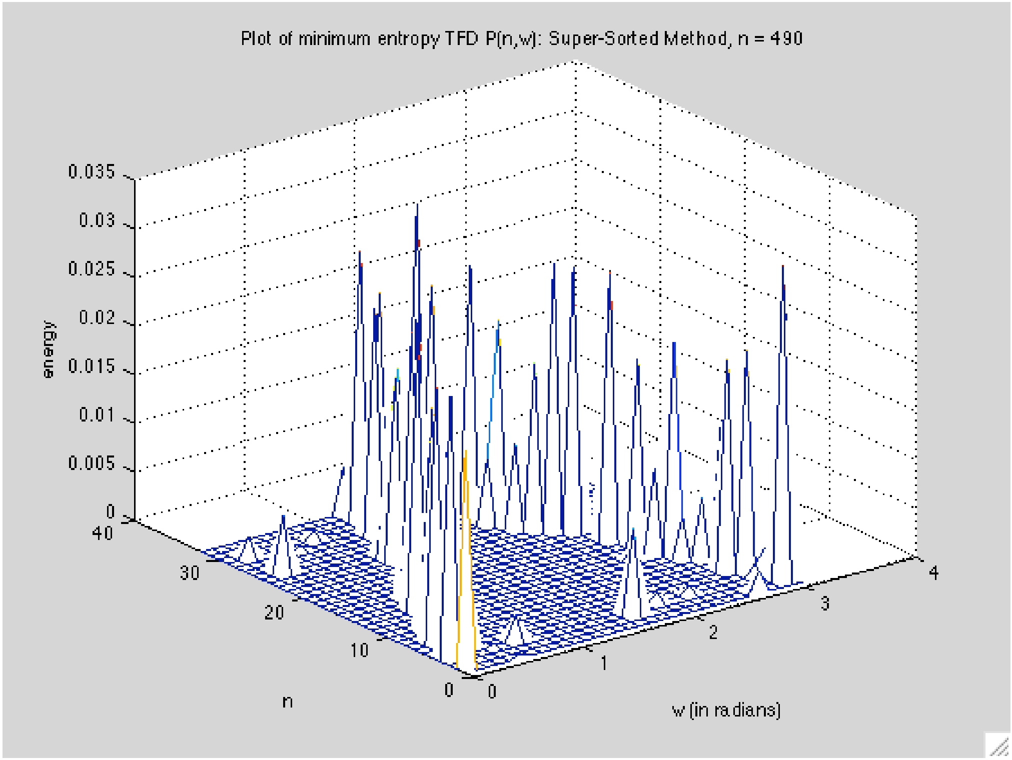

Figure 5.67: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 490 for the double chirp signal

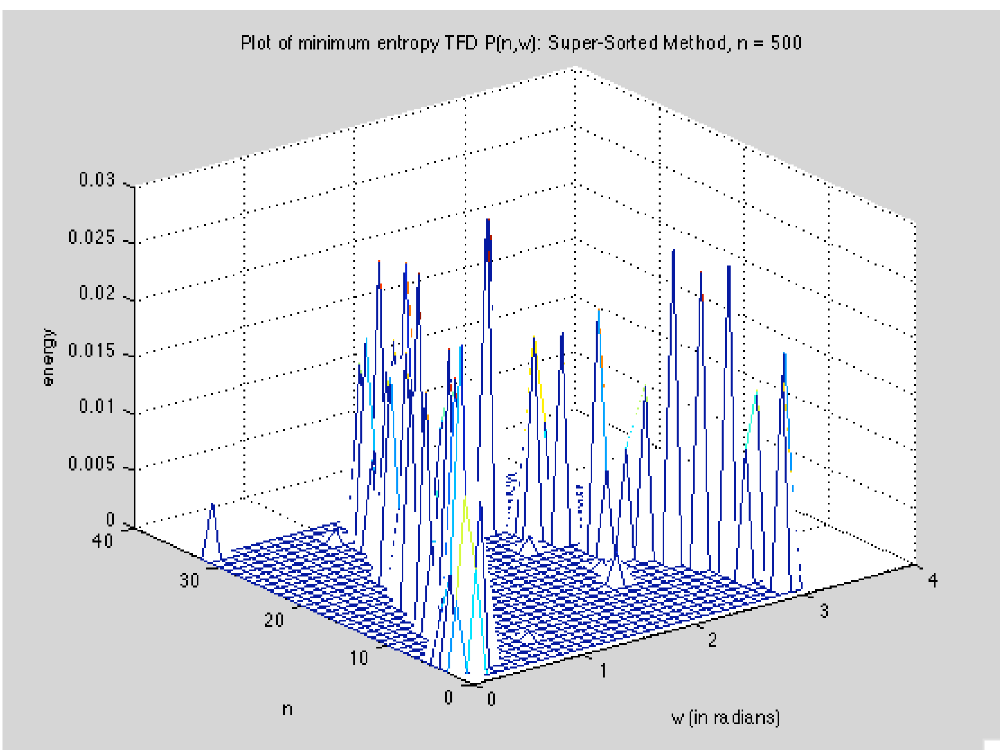

Figure 5.68: Plot of the minimum entropy TFD obtained using the spectrogram as the prior and the super sorted rectangle selection method with n = 500 for the double chirp signal

Next, the minimum entropy algorithm was performed using the super sorted rectangle selection method with several different values of *n* for the 32-point double chirp signal using the MCE-PTFD as the prior. The MCE-PTFD prior has a total entropy of 5.5150 and 0 zero-points. Table 5.12 shows the total entropy and number of zero-points for the final minimum entropy distributions found using the super sorted rectangle selection method for different values of *n*. Figures 5.69 – 5.74 show the final minimum entropy TFDs for the super sorted rectangle method for *n* = 400, 410, 420, 430, 440, and 450.

Table 5.12: Total entropy and number of zero-points for the minimum entropy TFDs obtained using the MCE-PTFD as the prior for the chirp with sinusoid signal using the super sorted rectangle selection method for different values of *n*.

n	**Total Entropy**	**Zero-Points**
400	3.9267	947
410	4.2805	924
420	4.0181	945
430	4.3996	918
440	4.2308	929
450	4.2989	925

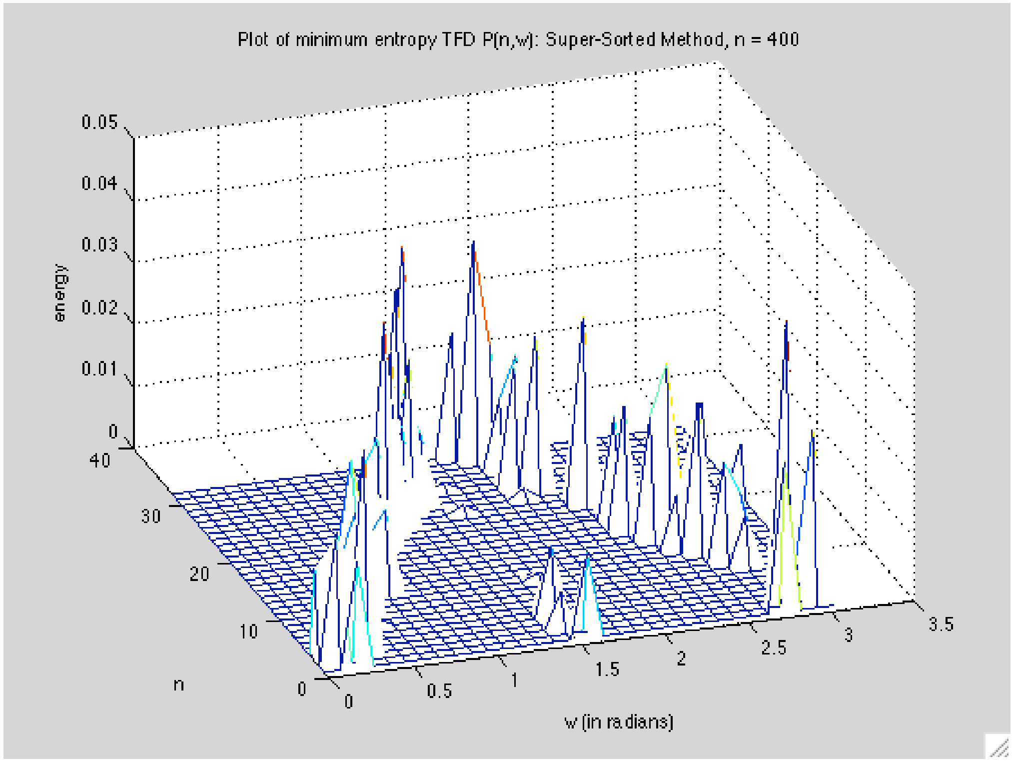

Figure 5.69: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with *n* = 400 for the double chirp signal

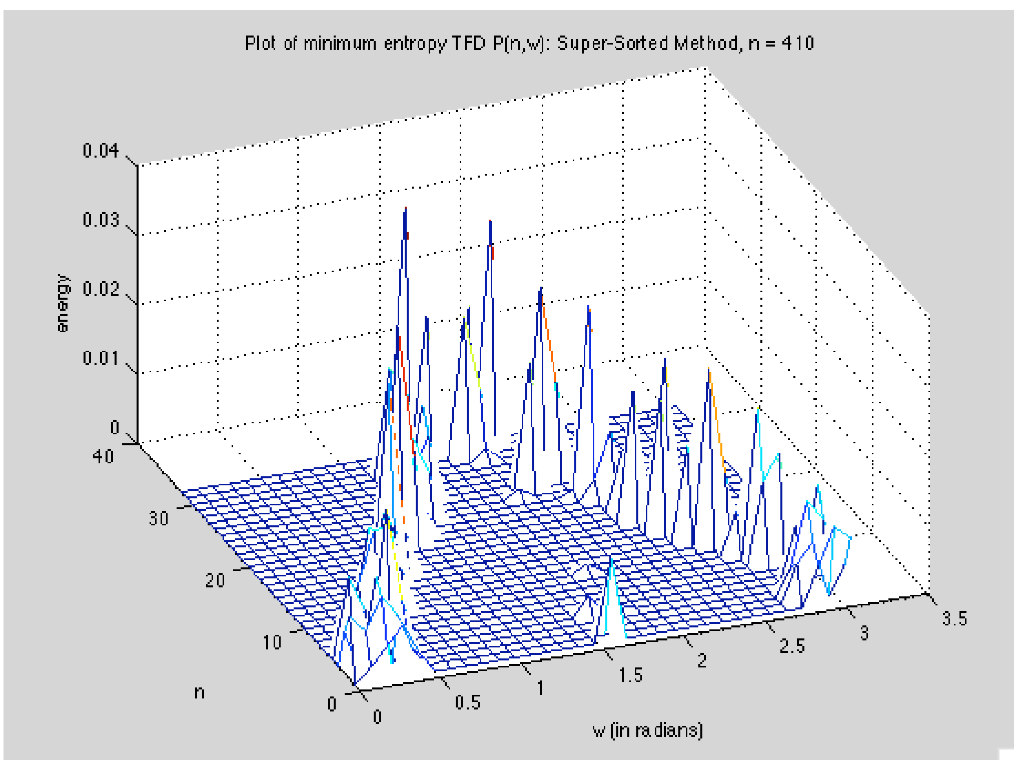

Figure 5.70: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with *n* = 410 for the double chirp signal

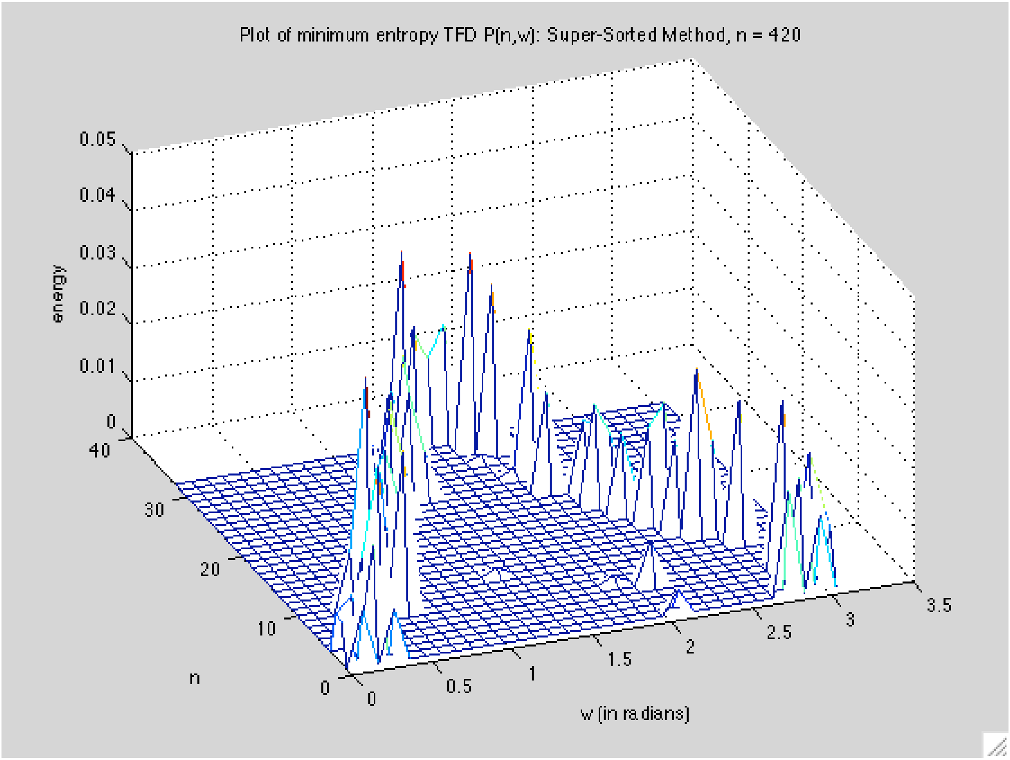

Figure 5.71: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with n = 420 for the double chirp signal

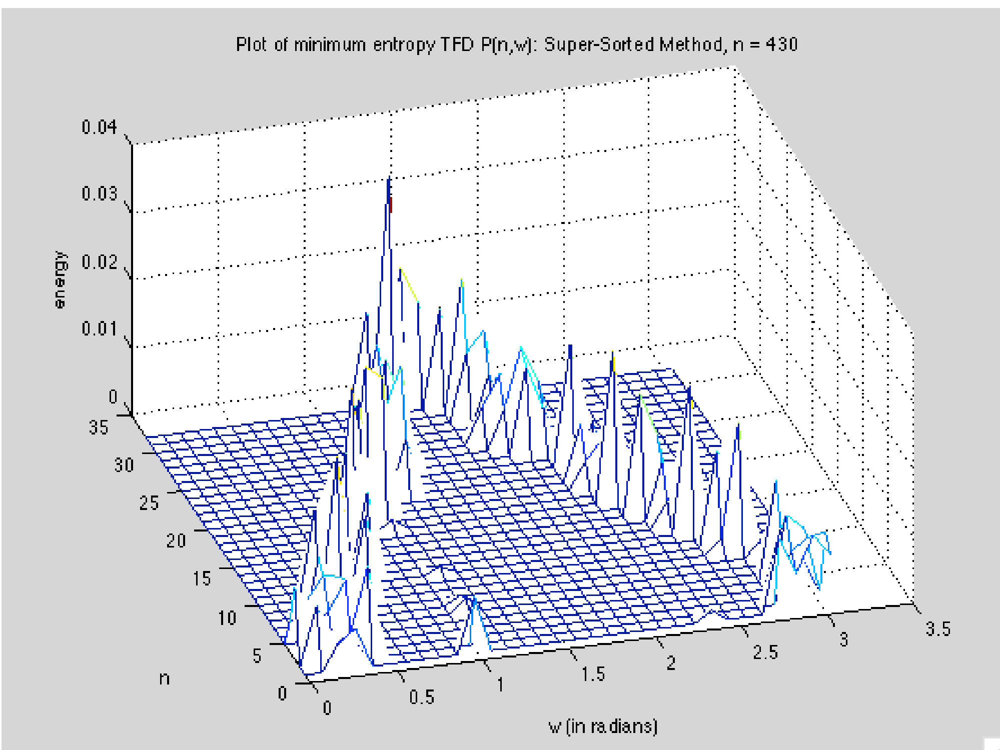

Figure 5.72: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with n = 430 for the double chirp signal

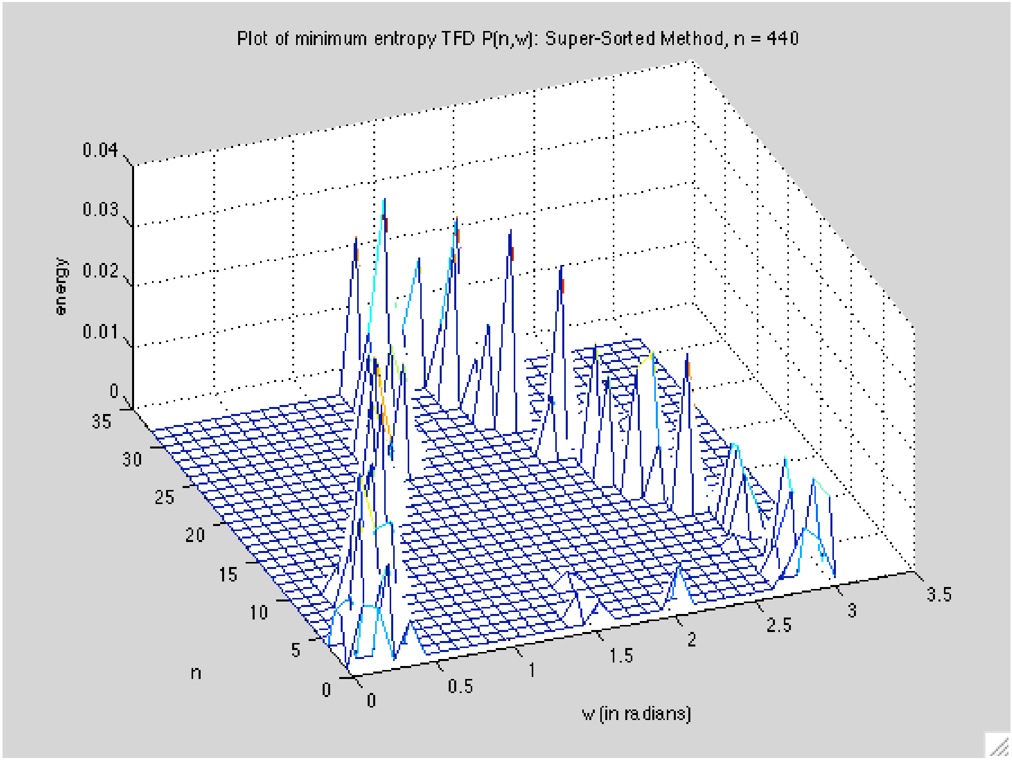

Figure 5.73: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with *n* = 440 for the double chirp signal

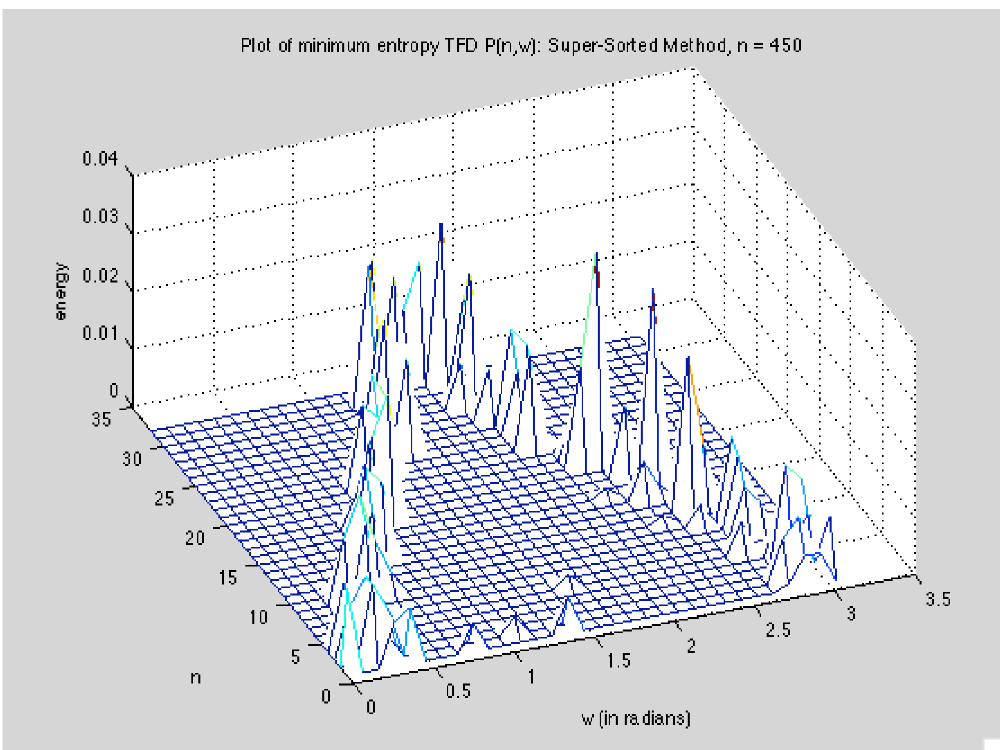

Figure 5.74: Plot of the minimum entropy TFD obtained using the MCE-PTFD as the prior and the super sorted rectangle selection method with *n* = 450 for the double chirp signal

5.3.4 Conclusions

From the resulting distributions, it is evident that the super sorted method can be effectively utilized to create highly localized distributions that are meaningful for the signal under consideration. Out of all of the rectangle selection methods that have been considered, the super sorted method provided the best representations. Most of the outlier spikes that existed for the previous rectangle selection methods have been diminished or eliminated with the super sorted method. In addition, the results appear to be decent for both the spectrogram and MCE-PTFD priors. This indicates that this method will likely work for any prior that is a good representation of the signal.

The results also indicate that the value of n has a profound impact on the effectiveness of the super sorted method. For all of the signals, the most effective values of n tended to be between 200 and 500. The spectrogram prior seemed to work better with higher values of n while the MCE-PTFD prior appeared to work better with slightly lower values of n. For example, the best values of n for the simple chirp signal with the spectrogram prior were between 350 and 400 while the best values of n for the same signal using the MCE-PTFD prior were between 200 and 250. In addition, the simple chirp signal appeared to work better with lower values of n while the simple chirp with sinusoid and double chirp signals worked better with higher values of n. For example, the best values of n for the simple chirp signal with the MCE-PTFD prior were between 200 and 250 while the best values of n for the double chirp signal using the same prior were between 350 and 400. This trend is probably due to the fact that the double chirp signal contained more high-energy points than the simple chirp signal.

As the value of n increased, the number of outlier spikes tended to decrease. For small values of n, the final minimum entropy distributions exhibited a large number of outlier spikes. This occurs because the medium-energy points are included in the low-energy list and, as a result, can be moved to low-energy locations. Higher values of n cause the medium-energy points to be included in the high-energy list which prevents them from being moved to low-energy locations. If n is too large, the initial distribution is not reduced by a significant amount and the final TFD is not highly localized. For example, if the simple chirp signal is reduced using the MCE-PTFD prior and super sorted method with n = 900, the final distribution has a total entropy of 5.2581 and 373 zero-points. Also, the final distribution appears to be nearly identical to the initial distribution. Thus, it is important to select a value of n that is not too low so that the outlier spikes are prevented and not too high so that the final distribution is highly localized.

6.0 CONCLUSIONS

It has been shown that the minimum entropy algorithm can be effectively utilized to create highly localized time-frequency distributions that are manifestly positive and satisfy the time and frequency marginals. We have also observed how different rectangle selection methods and initial time-frequency distributions can affect the final minimum entropy distribution. This section will analyze the results from the previous sections, draw conclusions from these results, and offer additional ideas and questions for future research.

6.1 RECTANGLE SELECTION METHODS

The rectangle selection method greatly impacts the effectiveness of the minimum entropy algorithm. The five rectangle selection methods that were initially considered (i.e. the top-left, top-right, bottom-left, bottom-right, and n^{th} maximum entropy loss rectangle selection methods) produced minimum entropy distributions that contained a large number of outlier spikes and were not meaningful representations for the signals under consideration regardless of the prior that was employed. These methods cannot be used to create correct representations of signals in the time-frequency domain. The sorted method also did not create consistently meaningful representations and also cannot be effectively utilized to create decent minimum entropy TFDs. The final two methods that were considered, the modified sorted and super sorted rectangle selection methods, appeared to create very good representations with very few outlier spikes. These methods can be used to create highly localized time-frequency distributions that reflect the signal characteristics.

Next, we will define a true minimum entropy rectangle selection method as one that attempts to minimize every rectangle in the initial distribution. A minimum entropy rectangle selection method that is not true will be referred to as a partial minimum entropy rectangle selection method. Thus, the top-left, top-right, bottom-left, bottom-right, n^{th} maximum loss, and sorted rectangle selection methods are all true rectangle selection methods. The modified sorted and super sorted rectangle selection methods are both partial rectangle selection methods since they exclude some of the rectangles from the minimization process.

For true minimum entropy rectangle selection methods, the number of non-zero points will generally be reduced to approximately the number of marginal constraints. For all of the signals, true rectangle selection methods, and priors that were examined in this thesis, the final number of non-zero points was always in the vicinity of 64 which was the total number of time and frequency marginal constraints. As previously noted, the signal under consideration may require more or fewer non-zero points for it to be accurately represented. Thus, it may not be possible to obtain true minimum entropy distributions that adequately represent the signal for many signals. The algorithm can be modified to use a partial minimum entropy rectangle selection method so that the final distribution has more non-zero points. The results demonstrated how partial rectangle selection methods such as the modified sorted and super sorted methods can lead to more non-zero points and better overall representations of the signals.

The rectangle selection method also determines the presence of outlier spikes. Remember that the outlier spikes are generally created by two distinct causes: summing many near-zero points and moving high-energy points to low-energy locations by minimizing rectangles that have high-energy values on opposing corners. All of the true minimum entropy rectangle selection methods resulted in a large number of outlier spikes. These methods do not account for the two cases that cause these spikes. As a result, the final minimum entropy TFDs are often not very meaningful. The rectangle selection methods that account for the outlier causes can be used to create consistently meaningful representations of the signal.

Since the true minimum entropy rectangle selection methods do not produce the best results, it is very unlikely that the absolute minimum entropy distribution for the signal would be a strong representation. Thus, it is probably fruitless to attempt to find the absolute minimum entropy TFD for a signal since an infinite number of possible minimum entropy TFDs exist and it would most likely not be a very good result.

6.2 PRIORS

The initial TFD or prior that is selected also has a profound influence on the final minimum entropy time-frequency distribution. In the initial examples, the maximum entropy prior produced nearly meaningless results for all of the signals and rectangle selection methods. Since this distribution makes the fewest assumptions about the signal, much of the signal information is lost. When this prior is used with the minimum entropy algorithm, most of the signal information is lost before the iterative process begins and the algorithm does not have much of a chance of producing a meaningful distribution. The spectrogram and MCE-PTFD priors produced markedly better results than the maximum entropy TFD. For these priors, most of the signal information is prominent in the distributions. Therefore, the initial TFD must be a reasonable representation of the signal for the minimum entropy algorithm to be effective.

In addition, the final minimum entropy TFD exhibits the same marginal and finite support properties as the initial TFD. Since the maximum entropy and MCE-PTFD priors satisfied the time and frequency marginals and contained strong finite support, the final TFDs created using these priors also had these properties. Conversely, the spectrogram prior did not satisfy either marginal and did not have any finite support which resulted in minimum entropy distributions without these properties. Hence, if it is desired that the final distribution have certain marginal and finite support characteristics, the prior that is employed must have the same characteristics.

The smoothness of the initial TFD and the location of the maximum energy values also seem to have an effect of the resulting distribution. For the signals considered in this thesis, smoother initial TFDs appeared to produce better final distributions. For example, the spectrogram tended to give slightly better results than the MCE-PTFD. This conclusion may not be accurate for other signals and rectangle selection methods. Also, the initial TFDs that contained the maximum energy values at the appropriate time-frequency locations produced better results than ones that did not. For example, the maximum entropy prior did not contain the correct maximum values and resulted in minimum entropy distributions that were not accurate. On the other hand, the spectrogram and MCE-PTFD priors did contain the correct maximum energy values and produced much more meaningful representations. It can be concluded that the best prior is one that is relatively smooth and has its maximum energy points at the correct time-frequency values.

6.3 AREAS FOR ADDITIONAL RESEARCH

Thus far, the minimum entropy algorithm is in its nascent stage and has not been fully investigated. This thesis is one of the initial explorations into the topic. As such, a number of questions and research areas exist regarding the minimum entropy algorithm that require further analysis.

First, the algorithm needs to be tested with other signals. So far, the algorithm has only been used on three relatively simple synthetic signals. These signals do not provide a good indication for how the algorithm will work on signals that contain added complexity such as the addition of noise or many frequency components. The algorithm needs to be tested with other, more complex synthetic and real-world signals to fully determine its effectiveness.

Secondly, only a handful of rectangle selection methods have been proposed and analyzed. Other rectangle selection methods could exist that may provide superior results. Research needs to be conducted to determine if other constraints or criteria exist that will lead to better, more effective rectangle selection methods. Also, variations on existing rectangle selection methods should be considered. An example of a variation is splitting an initial distribution into strips or squares and performing the algorithm on each strip or square. This approach may work well for longer signals or signals that have distinct frequency components.

Thirdly, this thesis only considered three initial TFDs for the minimum entropy algorithm: the maximum entropy TFD, the spectrogram, and the MCE-PTFD. Other priors could exist that may result in better, more meaningful distributions. Additional research needs to be completed to find more effective priors for the algorithm.

So far, the only measure of entropy that has been considered is the Shannon entropy. Another measure of entropy, such as the Renyi entropy, may provide better results. Also, using the Shannon entropy limits the distribution to using all positive values. A different definition of entropy may remove this limitation.

In addition, the area of signal noise has not been considered in conjunction with the minimum entropy algorithm. Would signal noise cause major spikes in the final distribution? If the energy of the noise is significantly less than the energy of the signal, could the noise be completely eliminated if the super sorted rectangle selection method is used? These areas require additional research to fully understand.

Algorithm implementation is another area that requires some research. Currently, the algorithm is somewhat inefficient and takes a long time to complete. Research needs to be completed to determine ways to both optimize and parallelize the algorithm.

The energy values of the final minimum entropy TFD do not accurately represent the actual energy values at each time-frequency point. The final values depend strongly on the initial distribution and the minimization order. Additional research needs to be completed to determine if any meaning can be derived from the energy values in the minimum entropy TFDs. The ultimate goal would be to find a prior/rectangle selection method combination that creates an accurate final distribution.

Finally, the application aspect of the minimum entropy algorithm has not yet been examined. Research needs to be completed to determine the applications that can benefit from this algorithm. Due to the length of time required to complete the algorithm, this method is not useful for real-time applications. On the other hand, it may be useful for creating highly localized, proper TFDs for applications that are not real-time.

APPENDIX A

MATLAB CODE

This appendix contains the Matlab scripts used to generate the results in this thesis.

A.1 SCRIPT TO IMPLEMENT MAXIMUM ENTROPY ALGORITHM USING MAXIMUM ENTROPY PRIOR

```
% Paul Bradley
% Thesis Project - Maximum Entropy TFD as prior

clear all;

N = 32;
n = 0:N-1;

W = N;
w = linspace(0,2*pi,W);

% Chirp signal:
x = exp((j*pi*n.^2)/(2*N));

% Chirp with sinusoid signal:
% x = exp((j*pi*n.^2)/(2*N)) + exp(j*5*pi*n/4);

% Double chirp signal:
% x = exp((j*pi*n.^2)/(2*N)) + exp((j*pi*(pi/2 - n.^2))/(2*N));

% Normalize input signal
x = x/sqrt(sum(abs(x).^2));

% Plot normalized input signal
figure(1)
plot(n,real(x));
xlabel('n');
ylabel('Real Part of Normalized x[n]');
title('Plot of real part of normalized x[n] vs. n');
```

```
% Determine spectrum of the signal, normalize spectrum
xw = fft(x,W);
xw = xw/sqrt(sum(abs(xw).^2));

% Plot spectrum
figure(2)
plot(w,real(xw));
xlabel('w');
ylabel('Real Part of X[w]');
title('Plot of real part of X[w] vs. w');

% Calculate spectrogram
[S,F,T] = basic_spectrogram(x,16);
S = abs(S);

figure(3)
mesh(F*pi,T,S);
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of the spectrogram of x[n]');

% Calculate P = |x(t)|^2 * |X(w)|^2
P_init = (abs(x).^2)'*(abs(xw).^2);

% Plot initial TFD
figure(6)
mesh(w,n,P_init)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of initial P(n,w) = |x(t)|^2 * |X(w)|^2');
pause(1);

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'topleft');
figure(7)
mesh(w,n,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Top Left Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'topright');
figure(8)
mesh(w,n,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Top Right Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'bottomleft');
figure(9)
mesh(w,n,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Bottom Left Algorithm');

start_z
start_e
end_z
end_e
```

```
[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'bottomright');
figure(10)
mesh(w,n,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Bottom Right Algorithm');

start_z
start_e
end_z
end_e

for idx = 100:100
    pause(1);
    % Find minimum entropy TFD
    [P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'maxloss',idx);

    idx
    start_z
    start_e
    end_z
    end_e

    figure(10+idx)
    mesh(w,n,P)
    xlabel('w (in radians)');
    ylabel('n');
    zlabel('energy');
    title(sprintf('Plot of minimum entropy TFD P(n,w): Maximum Loss, Depth = %d',idx));
end
```

A.2 SCRIPT TO IMPLEMENT MAXIMUM ENTROPY ALGORITHM USING SPECTROGRAM PRIOR

```
% Paul Bradley
% Thesis Project - Spectrogram as prior

clear all;

N = 32;
n = 0:N-1;

W = N;
w = linspace(0,2*pi,W);

% Chirp signal:
% x = exp((j*pi*n.^2)/(2*N));

% Chirp with sinusoid signal:
x = exp((j*pi*n.^2)/(2*N)) + exp(j*5*pi*n/4);

% Double chirp signal:
% x = exp((j*pi*n.^2)/(2*N)) + exp((j*pi*(pi/2 - n.^2))/(2*N));

% Normalize input signal
x = x/sqrt(sum(abs(x).^2));

% Plot normalized input signal
figure(1)
plot(n,real(x));
xlabel('n');
ylabel('Real Part of Normalized x[n]');
title('Plot of real part of normalized x[n] vs. n');

% Calculate spectrogram
[S,F,T] = basic_spectrogram(x,16);
P_init = abs(S);

figure(3)
mesh(F*pi,T,P_init);
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of initial TFD, P(n,w) = spectrogram(x(t))');

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'topleft');
figure(7)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Top Left Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'topright');
figure(8)
mesh(F*pi,T,P)
xlabel('w (in radians)');
```

```
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Top Right Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'bottomleft');
figure(9)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Bottom Left Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'bottomright');
figure(11)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Bottom Right Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'center');
figure(12)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Center Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'sorted');
figure(10)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Sorted Algorithm');

start_z
start_e
end_z
end_e

for idx = 450:10:500
    [P,start_z,start_e,end_z(idx),end_e(idx)] = min_entropy_tfd(P_init,'supersorted',idx);
    figure(300+idx)
    mesh(F*pi,T,P)
    xlabel('w (in radians)');
    ylabel('n');
    zlabel('energy');
    title(sprintf('Plot of minimum entropy TFD P(n,w): Super-Sorted Method, n = %d',idx));

    idx
    start_z
    start_e
```

```
    end_z
    end_e
end

for idx = 95:5:145
    [P,start_z,start_e,end_z(idx),end_e(idx)] = min_entropy_tfd(P_init,'modifiedsorted',idx);
    figure(200+idx)
    mesh(F*pi,T,P)
    xlabel('w (in radians)');
    ylabel('n');
    zlabel('energy');
    title(sprintf('Plot of minimum entropy TFD P(n,w): Modified Sorted Method, n = %d',idx));

    idx
    start_z
    start_e
    end_z
    end_e
end

for idx = 125:125
    pause(1);
    % Find minimum entropy TFD
    [P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'maxloss',idx);

    idx
    start_z
    start_e
    end_z
    end_e

    figure(10+idx)
    mesh(F*pi,T,P)
    xlabel('w (in radians)');
    ylabel('n');
    zlabel('energy');
    title(sprintf('Plot of minimum entropy TFD P(n,w): Maximum Loss, Depth = %d',idx));
end
```

A.3 SCRIPT TO IMPLEMENT MAXIMUM ENTROPY ALGORITHM USING MCE-PTFD PRIOR

```
% Paul Bradley
% Thesis Project - MCE-PTFD as prior

clear all;

% ***********************************************************
% Section: Original signal

N = 32;
n = 0:N-1;

W = N;
w = linspace(0,2*pi,W);

% Chirp signal:
x = exp((j*pi*n.^2)/(2*N));

% Chirp with sinusoid signal:
% x = exp((j*pi*n.^2)/(2*N)) + exp(j*5*pi*n/4);

% Double chirp signal:
% x = exp((j*pi*n.^2)/(2*N)) + exp((j*pi*(pi/2 - n.^2))/(2*N));

% Normalize input signal
x = x/sqrt(sum(abs(x).^2));

% Plot normalized input signal
figure(1)
plot(n,real(x));
xlabel('n');
ylabel('Real Part of Normalized x[n]');
title('Plot of real part of normalized x[n] vs. n');

% % Calculate signal energy
% x_energy = sum(abs(x).^2);
%
% % Determine spectrum of the signal, normalize spectrum
% xw = fft(x,W);
% xw = xw/sqrt(sum(abs(xw).^2));
%
% % Find time, frequency marginals
% tm = abs(x).^2;
% fm = abs(xw).^2;
%
% % Plot time marginal
% figure(2)
% plot(n,tm);
% xlabel('n');
% ylabel('Time marginal of x(n)');
% title('Plot of time marginal of x(n)');
%
% % Plot frequency marginal
% figure(3)
% plot(w,fm);
% xlabel('w (in radians)');
% ylabel('Freqency marginal of x(n)');
% title('Plot of frequency marginal of x(n)');
% ***********************************************************
```

```
% % ***********************************************************
% % Section: Initial prior
%
% % Calculate spectrogram, nomalize it to signal energy
% [S,F,T] = paulSpectrogram(x,5,0.5);
% prior = abs(S').^2;
% prior = x_energy*prior/sum(sum(prior));
%
% figure(4)
% mesh(F*pi,T,prior);
% xlabel('w (in radians)');
% ylabel('n');
% zlabel('energy');
% title('Plot of prior spectrogram for mce-ptfd');
%
% % Find time, frequency marginals of prior
% prior_tm = sum(abs(prior').^2);
% prior_fm = sum(abs(prior).^2);
%
% % Plot time marginal
% figure(5)
% plot(T,prior_tm);
% xlabel('n');
% ylabel('Time marginal for prior');
% title('Plot of time marginal for prior');
%
% % Plot frequency marginal
% figure(6)
% plot(w,prior_fm);
% xlabel('w');
% ylabel('Frequency marginal for prior');
% title('Plot of frequency marginal for prior');
% % ***********************************************************
%
% % ***********************************************************
% % Section: Initial marginals
%
% % Calculate time marginal, normalize it
% tmi = abs(x).^2;
% tmi = x_energy*tmi/sum(tmi);
%
% % Calculate frequency marginal, normalize it
% fmi = abs(xw).^2;
% fmi = x_energy*fmi/sum(fmi);
%
% % Plot time marginal
% figure(7)
% plot(n,tmi);
% xlabel('n');
% ylabel('Time marginal for mce_ptfd algorithm');
% title('Plot of time marginal for mce_ptfd algorithm');
%
% % Plot frequency marginal
% figure(8)
% plot(w,fmi);
% xlabel('w');
% ylabel('Frequency marginal for mce_ptfd algorithm');
% title('Plot of frequency marginal for mce_ptfd algorithm');
% % ***********************************************************

% ***********************************************************
% Section: MCE-PTFD

% Find Minimum Cross Entropy PTFD
[ptfd,F,T] = basic_mce_ptfd(x);

% Plot Minimum Cross Entropy PTFD
figure(9)
mesh(F*pi,T,ptfd);
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
```

```
title('Plot of ptfd, P(n,w) = mce-ptfd(x(t))');

% % Find time, frequency marginals
% tm = sum(abs(ptfd'));
% fm = sum(abs(ptfd));
%
% % Plot time marginal
% figure(10)
% plot(n,tm);
% xlabel('n');
% ylabel('Time marginal of mce-ptfd');
% title('Plot of time marginal of mce-ptfd');
%
% % Plot frequency marginal
% figure(11)
% plot(w,fm);
% xlabel('w');
% ylabel('Frequency marginal of mce-ptfd');
% title('Plot of frequency marginal of mce-ptfd');
% ***********************************************************

% ***********************************************************

P_init = ptfd;

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'topleft');
figure(12)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Top Left Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'topright');
figure(13)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Top Right Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'bottomleft');
figure(14)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Bottom Left Algorithm');

start_z
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'bottomright');
figure(15)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Bottom Right Algorithm');

start_z
```

```
start_e
end_z
end_e

[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'center');
figure(16)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Center Algorithm');

start_z
start_e
end_z
end_e


[P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'sorted');
figure(17)
mesh(F*pi,T,P)
xlabel('w (in radians)');
ylabel('n');
zlabel('energy');
title('Plot of minimum entropy TFD P(n,w): Sorted Algorithm');

start_z
start_e
end_z
end_e

for idx = 900:900
    [P,start_z,start_e,end_z(idx),end_e(idx)] = min_entropy_tfd(P_init,'supersorted',idx);
    figure(300+idx)
    mesh(F*pi,T,P)
    xlabel('w (in radians)');
    ylabel('n');
    zlabel('energy');
    title(sprintf('Plot of minimum entropy TFD P(n,w): Super-Sorted Method, n = %d',idx));

    idx
    start_z
    start_e
    end_z
    end_e
end

for idx = 35:5:80
    [P,start_z,start_e,end_z(idx),end_e(idx)] = min_entropy_tfd(P_init,'modifiedsorted',idx);
    figure(200+idx)
    mesh(F*pi,T,P)
    xlabel('w (in radians)');
    ylabel('n');
    zlabel('energy');
    title(sprintf('Plot of minimum entropy TFD P(n,w): Modified Sorted Method, n = %d',idx));

    idx
    start_z
    start_e
    end_z
    end_e
end

for idx = 100:100
    pause(1);
    % Find minimum entropy TFD
    [P,start_z,start_e,end_z,end_e] = min_entropy_tfd(P_init,'maxloss',idx);

    idx
    start_z
    start_e
    end_z
    end_e
```

```
    figure(20+idx)
    mesh(F*pi,T,P)
    xlabel('w (in radians)');
    ylabel('n');
    zlabel('energy');
    title(sprintf('Plot of minimum entropy TFD P(n,w): Maximum Loss, Depth = %d',idx));
end
```

A.4 MATLAB FILE: CALC_ZERO_POINTS.M

```
function [z] = calc_zero_points(P)
% Author: Paul Bradley
%
% Finds the number of zeros in the passed matrix
%
% Parameters:
%   z       - returns number of zeros in matrix
%   P       - matrix to check

[rows,cols] = size(P);

z = 0;

for i = 1:rows
    for j = 1:cols
        if P(i,j) < 10^-17
            z = z+1;
        end
    end
end
```

A.5 MATLAB FILE: CALC_TOTAL_ENTROPY.M

```
function [entropy] = calc_total_entropy(P)
% Author: Paul Bradley
%
% Finds the total Shannon entropy of the passed matrix
%
% Parameters:
%   entropy - returns the entropy of the matrix
%   P       - matrix to check

[rows,cols] = size(P);

entropy = 0;

for i = 1:rows
    for j = 1:cols
        entropy = entropy + calc_entropy(P(i,j));
    end
end
```

A.6 MATLAB FILE: CALC_ ENTROPY.M

```
function [entropy] = calc_entropy(P)
% Author: Paul Bradley
%
% Finds the Shannon entropy of the passed value, vector, or matrix
%
% Parameters:
%   entropy - returns the entropy of the matrix (same size as P)
%   P       - value, vector, or matrix for which to calculate the entropy

% Determine number of rows, columns in P
[rows,cols] = size(P);

% Init entropy matrix
entropy = zeros(rows,cols);

% For each entry in P matrix, calculate entropy
for i = 1:rows
    for j = 1:cols
        if P(i,j) > 0
            entropy(i,j) = -P(i,j).*log(P(i,j));
        end
    end
end
```

A.7 MATLAB FILE: MIN_ENTROPY_TFD.M

```
function [P,s_z,s_ent,e_z,e_ent] = min_entropy_tfd(P,type,depth)
% Author: Paul Bradley
%
% Finds the maximum entropy time-frequency distribution for
% the input signal
%
% Inputs:
%   P - initial prior matrix to minimize
%   type - type of minimum entropy TFD
%          maxloss: nth Maximum Loss method
%          topleft: Top-Left method
%          topright: Top-Right method
%          bottomleft: Bottom-Left method
%          bottomright: Bottom-Right method
%          modifiedsorted: Modified Sorted method
%          sorted: Sorted method
%          supersorted: Super Sorted method
%   depth - Used with 'maxloss', 'modifiedsorted', and 'supersorted'
%           methods
%
% Output:
%   P - minimum entropy TFD
%   s_z - start number of zero-points
%   s_ent - start total entropy
%   e_z - end number of zero-points
%   e_ent - end total entropy

% Calculate initial zero-points, entropy
s_z = calc_zero_points(P);
s_ent = calc_total_entropy(P);

if strcmp(type,'maxloss') == true
    [P] = min_entropy_max_loss(P,depth);
elseif strcmp(type,'topleft') == true
    [P] = min_entropy_topleft(P);
elseif strcmp(type,'topright') == true
    [P] = min_entropy_topright(P);
elseif strcmp(type,'bottomleft') == true
    [P] = min_entropy_bottomleft(P);
elseif strcmp(type,'bottomright') == true
    [P] = min_entropy_bottomright(P);
elseif strcmp(type,'modifiedsorted') == true
    [P] = min_entropy_modified_sorted(P,depth);
elseif strcmp(type,'sorted') == true
    [P] = min_entropy_sorted(P);
elseif strcmp(type,'supersorted') == true
    [P] = min_entropy_super_sorted(P,depth);
end

e_z = calc_zero_points(P);
e_ent = calc_total_entropy(P);

end


function [P] = min_entropy_max_loss(P,depth)
    e = calc_entropy(P);
    [L,D] = create_loss_matrix(P,e);

    last_z = 0;
    count = 0;
```

```
    [loss,d,n1,w1,n2,w2] = find_maximum_loss(P,L,D,depth);
    while loss > 0
        P(n1,w1) = P(n1,w1) - d;
        P(n1,w2) = P(n1,w2) + d;
        P(n2,w1) = P(n2,w1) + d;
        P(n2,w2) = P(n2,w2) - d;

        e(n1,w1) = calc_entropy(P(n1,w1));
        e(n1,w2) = calc_entropy(P(n1,w2));
        e(n2,w1) = calc_entropy(P(n2,w1));
        e(n2,w2) = calc_entropy(P(n2,w2));

        [L,D] = update_loss_matrix(P,e,L,D,n1,w1,n2,w2);

        % Prevent infinite loop
        z = calc_zero_points(P);
        if z == last_z
            count = count + 1;
        else
            count = 0;
            last_z = z;
        end

        if count > 5000
            break;
        end

        [loss,d,n1,w1,n2,w2] = find_maximum_loss(P,L,D,depth);
    end
end

function [P] = min_entropy_topleft(P)
    [N,W] = size(P);

    e = calc_entropy(P);

    i = 0;
    change = true;
    while change == true && i <= 500
        change = false;
        for n1 = 1:N-1
            for n2 = n1+1:N
                for w1 = 1:W-1
                    for w2 = w1+1:W
                        [P,e,change] = do_entropy_loss(P,e,change,n1,w1,n2,w2);
                    end
                end
            end
        end

        i = i+1;
    end
end

function [P] = min_entropy_topright(P)
    [N,W] = size(P);

    e = calc_entropy(P);

    i = 0;
    change = true;
    while change == true && i <= 500
        change = false;
        for n1 = N:-1:2
            for n2 = n1-1:-1:1
                for w1 = 1:W-1
                    for w2 = w1+1:W
                        [P,e,change] = do_entropy_loss(P,e,change,n1,w1,n2,w2);
                    end
                end
            end
        end
```

```
        i = i+1;
    end
end

function [P] = min_entropy_bottomleft(P)
    [N,W] = size(P);

    e = calc_entropy(P);

    i = 0;
    change = true;
    while change == true && i <= 500
        change = false;
        for n1 = 1:N-1
            for n2 = n1+1:N
                for w1 = W:-1:2
                    for w2 = w1-1:-1:1
                        [P,e,change] = do_entropy_loss(P,e,change,n1,w1,n2,w2);
                    end
                end
            end
        end

        i = i+1;
    end
end

function [P] = min_entropy_bottomright(P)
    [N,W] = size(P);

    e = calc_entropy(P);

    i = 0;
    change = true;
    while change == true && i <= 500
        change = false;
        for n1 = N:-1:2
            for n2 = n1-1:-1:1
                for w1 = W:-1:2
                    for w2 = w1-1:-1:1
                        [P,e,change] = do_entropy_loss(P,e,change,n1,w1,n2,w2);
                    end
                end
            end
        end

        i = i+1;
    end
end

function [P] = min_entropy_modified_sorted(P,depth)
    [N,W] = size(P);
    num_points = N*W;

    [sort_n,sort_w] = sort_points(P);

    e_line = P(sort_n(depth),sort_w(depth));
    P_init = P;

    e = calc_entropy(P);

    count = 0;
    change = true;
    while change == true && count <= 40
        change = false;
        for idx = 1:depth
            n1 = sort_n(idx);
            w1 = sort_w(idx);

            for idx2 = num_points:-1:depth
                n2 = sort_n(idx2);
                w2 = sort_w(idx2);
```

```
                if n1 == n2 || w1 == w2
                    continue;
                elseif P_init(n1,w1) >= e_line && P_init(n2,w2) >= e_line
                    continue;
                elseif P_init(n1,w2) >= e_line && P_init(n2,w1) >= e_line
                    continue;
                elseif P_init(n1,w1) < e_line && P_init(n2,w2) < e_line && P_init(n1,w2) < e_line
&& P_init(n2,w1) < e_line
                    continue;
                end

                [P,e,change] = do_entropy_loss(P,e,change,n1,w1,n2,w2);
            end
        end

        count = count + 1;
    end
end

function [P] = min_entropy_sorted(P)
    [N,W] = size(P);
    num_points = N*W;

    [sort_n,sort_w] = sort_points(P);

    e = calc_entropy(P);

    count = 0;
    change = true;
    while change == true && count <= 40
        change = false;
        for idx = 1:num_points
            n1 = sort_n(idx);
            w1 = sort_w(idx);

            for idx2 = num_points:-1:1
                n2 = sort_n(idx2);
                w2 = sort_w(idx2);

                if n1 == n2 || w1 == w2
                    continue;
                end

                [P,e,change] = do_entropy_loss(P,e,change,n1,w1,n2,w2);
            end
        end

        count = count + 1;
    end
end

function [P] = min_entropy_super_sorted(P,depth)
    [N,W] = size(P);
    num_points = N*W;

    [sort_n,sort_w] = sort_points(P);

    e_line = P(sort_n(depth),sort_w(depth));
    P_init = P;

    e = calc_entropy(P);

    count = 0;
    change = true;
    while change == true && count <= 30
        for idx = 1:num_points
            n1 = sort_n(idx);
            w1 = sort_w(idx);

            for idx2 = num_points:-1:1
                n2 = sort_n(idx2);
                w2 = sort_w(idx2);
```

```
                if n1 == n2 || w1 == w2
                    continue;
                elseif P_init(n1,w1) >= e_line && P_init(n2,w2) >= e_line
                    continue;
                elseif P_init(n1,w2) >= e_line && P_init(n2,w1) >= e_line
                    continue;
                elseif P_init(n1,w1) < e_line && P_init(n2,w2) < e_line && P_init(n1,w2) < e_line
&& P_init(n2,w1) < e_line
                    continue;
                end

                [P,e,change] = do_entropy_loss(P,e,change,n1,w1,n2,w2);
            end
        end

        count = count + 1;
    end
end

function [sort_n,sort_w] = sort_points(P)
    [N,W] = size(P);
    num_points = N*W;

    sort_p = zeros(1,num_points);
    sort_n = zeros(1,num_points);
    sort_w = zeros(1,num_points);

    for n = 1:N
        for w = 1:W
            value = P(n,w);

            idx = 1;
            while idx <= num_points
                if sort_p(idx) == 0 || value >= sort_p(idx)
                    sort_p(idx:num_points) = [value sort_p(idx:num_points-1)];
                    sort_n(idx:num_points) = [n sort_n(idx:num_points-1)];
                    sort_w(idx:num_points) = [w sort_w(idx:num_points-1)];
                    break;
                end
                idx = idx + 1;
            end
        end
    end
end

function [P,e,change] = do_entropy_loss(P,e,change,n1,w1,n2,w2)
    e_before = e(n1,w1) + e(n1,w2) + e(n2,w1) + e(n2,w2);

    d1 = min([P(n1,w1),P(n2,w2)]);
    if abs(d1) > 10^-17
        ed = calc_entropy([P(n1,w1)-d1 P(n1,w2)+d1 P(n2,w1)+d1 P(n2,w2)-d1]);
        e_after1 = ed(1) + ed(2) + ed(3) + ed(4);
        loss1 = e_before - e_after1;
    else
        loss1 = 0;
        d1 = 0;
    end

    d2 = -min([P(n1,w2), P(n2,w1)]);
    if abs(d2) > 10^-17
        ed = calc_entropy([P(n1,w1)-d2, P(n1,w2)+d2, P(n2,w1)+d2, P(n2,w2)-d2]);
        e_after2 = ed(1) + ed(2) + ed(3) + ed(4);
        loss2 = e_before - e_after2;
    else
        loss2 = 0;
        d2 = 0;
    end

    if loss1 > loss2
        d = d1;
    else
        d = d2;
    end
```

```
    if abs(d) ~= 0
        change = true;

        P(n1,w1) = P(n1,w1) - d;
        P(n1,w2) = P(n1,w2) + d;
        P(n2,w1) = P(n2,w1) + d;
        P(n2,w2) = P(n2,w2) - d;

        e(n1,w1) = calc_entropy(P(n1,w1));
        e(n1,w2) = calc_entropy(P(n1,w2));
        e(n2,w1) = calc_entropy(P(n2,w1));
        e(n2,w2) = calc_entropy(P(n2,w2));
    end
end
```

A.8 MATLAB FILE: FIND_MAXIMUM_ LOSS.M

```
function [max_loss,max_d,max_n1,max_w1,max_n2,max_w2] = find_maximum_loss(P,L,D,depth)
% Author: Paul Bradley
%
% This function finds the depth greatest maximum entropy loss from
% the passed P matrix, loss matrix, and delta matrix.
%
% Inputs:
%   P - matrix to find the greatest entropy loss
%   L - loss matrix of P
%   D - delta matrix of P
%   depth - determines which loss is found; if this is 1, function
%           finds the 1st greatest loss; if this is 2, function
%           finds the 2nd greatest loss, if possible; etc.
%
% Outputs:
%   loss - amount of loss for greatest entropy loss
%   d - delta value for greatest entropy loss
%   max_n1 - n1 value for greatest loss
%   max_w1 - w1 value for greatest loss
%   max_n2 - n2 value for greatest loss
%   max_w2 - w2 value for greatest loss

max_loss = 0;
max_d = 0;
max_n1 = 0;
max_n2 = 0;
max_w1 = 0;
max_w2 = 0;

[N,W] = size(P);

m_loss = zeros(1,depth);
m_d = zeros(1,depth);
m_n1 = zeros(1,depth);
m_w1 = zeros(1,depth);
m_n2 = zeros(1,depth);
m_w2 = zeros(1,depth);
m_idx = 0;

for n1 = 1:N
    for n2 = 1:N
        for w1 = 1:W
            for w2 = 1:W
                if n1 ~= n2 && w1 ~= w2
                    tloss = L(n1,w1,n2,w2);
                    td = D(n1,w1,n2,w2);

                    idx = 1;
                    while idx <= m_idx+1 && idx <= depth
                        if tloss > m_loss(idx)
                            if idx < depth
                                m_loss(idx:depth) = [tloss m_loss(idx:depth-1)];
                                m_d(idx:depth) = [td m_d(idx:depth-1)];
                                m_n1(idx:depth) = [n1 m_n1(idx:depth-1)];
                                m_w1(idx:depth) = [w1 m_w1(idx:depth-1)];
                                m_n2(idx:depth) = [n2 m_n2(idx:depth-1)];
                                m_w2(idx:depth) = [w2 m_w2(idx:depth-1)];
                            else
                                m_loss(idx) = tloss;
                                m_d(idx) = td;
                                m_n1(idx) = n1;
                                m_w1(idx) = w1;
```

```
                                m_n2(idx) = n2;
                                m_w2(idx) = w2;
                            end

                            if m_idx < depth
                                m_idx = m_idx+1;
                            end

                            break;
                        end
                        idx = idx+1;
                    end
                end
            end
        end
    end
end

if m_idx > 0
    max_loss = m_loss(m_idx);
    max_d = m_d(m_idx);
    max_n1 = m_n1(m_idx);
    max_w1 = m_w1(m_idx);
    max_n2 = m_n2(m_idx);
    max_w2 = m_w2(m_idx);
end
```

A.9 MATLAB FILE: CREATE_LOSS_MATRIX.M

```
function [L,D] = create_loss_matrix(P,e)
% Author: Paul Bradley
%
% Creates the loss and delta matrices for the passed P matrix.
%
% Inputs:
%   P - matrix for which to create the loss matrix
%   e - entropy matrix for P
%
% Outputs:
%   L - loss matrix for P
%   D - delta matrix for P

[N,W] = size(P);

L = zeros(N,W,N,W);
D = zeros(N,W,N,W);

for n1 = 1:N-1
    for w1 = 1:W-1
        for n2 = n1+1:N
            for w2 = w1+1:W/2
                [l,d] = calc_loss(P,e,n1,w1,n2,w2);

                L(n1,w1,n2,w2) = l;
                D(n1,w1,n2,w2) = d;

                L(n2,w2,n1,w1) = l;
                D(n2,w2,n1,w1) = d;

                [l,d] = calc_loss(P,e,n1,w2,n2,w1);

                L(n1,w2,n2,w1) = l;
                D(n1,w2,n2,w1) = d;

                L(n2,w1,n1,w2) = l;
                D(n2,w1,n1,w2) = d;
            end
        end
    end
end
```

A.10 MATLAB FILE: CALC_ LOSS.M

```
function [loss,d] = calc_loss(P,e,n1,w1,n2,w2)
% Author: Paul Bradley
%
% Calculates the loss and delta value for the specified P matrix
% and rectangle points.
%
% Inputs:
%   P - TFD matrix
%   e - entropy matrix for P
%   n1, w1 - first rectangle point
%   n2, w2 - second rectangle point
%
% Outputs:
%   loss - loss for specified matrix and points
%   d - delta value for specified matrix and points

% Calculate initial entropy
e_before = e(n1,w1) + e(n1,w2) + e(n2,w1) + e(n2,w2);

% For delta1...
d1 = min([P(n1,w1), P(n2,w2)]);
if abs(d1) > 10^-17
    ed = calc_entropy([P(n1,w1)-d1 P(n1,w2)+d1 P(n2,w1)+d1 P(n2,w2)-d1]);
    e_after1 = ed(1) + ed(2) + ed(3) + ed(4);
    loss1 = e_before - e_after1;
else
    d1 = 0;
    loss1 = 0;
end

% For delta2...
d2 = -min([P(n1,w2), P(n2,w1)]);
if abs(d2) > 10^-17
    ed = calc_entropy([P(n1,w1)-d2 P(n1,w2)+d2 P(n2,w1)+d2 P(n2,w2)-d2]);
    e_after2 = ed(1) + ed(2) + ed(3) + ed(4);
    loss2 = e_before - e_after2;
else
    d2 = 0;
    loss2 = 0;
end

% Find greatest loss
if loss1 > loss2
    loss = loss1;
    d = d1;
else
    loss = loss2;
    d = d2;
end
```

A.11 MATLAB FILE: UPDATE_LOSS_MATRIX.M

```
function [L,D] = update_loss_matrix(P,e,L,D,n1,w1,n2,w2)
% Author: Paul Bradley
%
% Updates the loss and delta matrices for the passed P matrix
% for the specified points.
%
% Inputs:
%   P - matrix for which to create the loss matrix
%   e - entropy matrix for P
%   L - loss matrix for P
%   D - delta matrix for P
%   n1, w1 - first point in rectangle to update
%   n2, w2 - second point in rectangle to update
%
% Outputs:
%   L - loss matrix for P
%   D - delta matrix for P

[N,W] = size(P);

for n = 1:N
    for w = 1:W
        if n ~= n1 && w ~= w1
            [l,d] = calc_loss(P,e,n1,w1,n,w);
            L(n1,w1,n,w) = l;
            D(n1,w1,n,w) = d;
            L(n,w,n1,w1) = l;
            D(n,w,n1,w1) = d;

            [l,d] = calc_loss(P,e,n1,w,n,w1);
            L(n1,w,n,w1) = l;
            D(n1,w,n,w1) = d;
            L(n,w1,n1,w) = l;
            D(n,w1,n1,w) = d;
        end

        if n ~= n2 && w ~= w2
            [l,d] = calc_loss(P,e,n2,w2,n,w);

            L(n2,w2,n,w) = l;
            L(n,w,n2,w2) = l;
            D(n2,w2,n,w) = d;
            D(n,w,n2,w2) = d;

            [l,d] = calc_loss(P,e,n2,w,n,w2);
            L(n2,w,n,w2) = l;
            D(n2,w,n,w2) = d;
            L(n,w2,n2,w) = l;
            D(n,w2,n2,w) = d;
        end

        if n ~= n1 && w ~= w2
            [l,d] = calc_loss(P,e,n1,w2,n,w);

            L(n1,w2,n,w) = l;
            L(n,w,n1,w2) = l;
            D(n1,w2,n,w) = d;
            D(n,w,n1,w2) = d;

            [l,d] = calc_loss(P,e,n1,w,n,w2);
            L(n1,w,n,w2) = l;
            D(n1,w,n,w2) = d;
```

```
            L(n,w2,n1,w) = l;
            D(n,w2,n1,w) = d;
        end

        if n ~= n2 && w ~= w1
            [l,d] = calc_loss(P,e,n2,w1,n,w);

            L(n2,w1,n,w) = l;
            L(n,w,n2,w1) = l;
            D(n2,w1,n,w) = d;
            D(n,w,n2,w1) = d;

            [l,d] = calc_loss(P,e,n2,w,n,w1);
            L(n2,w,n,w1) = l;
            D(n2,w,n,w1) = d;
            L(n,w1,n2,w) = l;
            D(n,w1,n2,w) = d;
        end
    end
end
```

A.12 MATLAB FILE: BASIC_SPECTOGRAM.M

```
function [S,F,T] = basic_spectrogram(x_init,win_length)
% Author: Paul Bradley
%
% Computes spectrogram for given signal.
%
% Inputs:
%   x - input signal
%   Fs - sampling frequency
%   window_time - length of hamming window for spectrogram
%
% Outputs:
%   S - spectrogram matrix
%   F - frequency values for spectrogram
%   T - time values for spectrogram


% Pad signals with zeros
x = [zeros(1,64) x_init zeros(1,64)]';

% Find energy of input signal
Ex = sum(abs(x).^2);

% Make FFT size = 2*largest window length
nfft = 2*win_length;

                                        % zero pad signal by wlen
xx = [zeros(1,fix(win_length/2)) x' zeros(1,fix(win_length/2))];
                                        % compute spectrogram
[S,F,T] = spectrogram(xx,win_length,win_length-1,nfft,1);

S = S';
S = abs(S).^2;
S = Ex*S/sum(sum(S));

S = S(65:65+length(x_init)-1,:);
T = 1:length(x_init);
F = F';
```

A.13 MATLAB FILE: BASIC_MCE_PTFD.M

```
function [ptfd,F,T] = basic_mce_ptfd(x_init)
% Author: Arash Mahboobin and Paul Bradley
%
% Finds a positive TFD for the specified signal, using MCE (minimum
% cross-entropy).  Uses two combined spectrograms as a prior.
%
% Parameters:
%   ptfd    - returns MCE positive TFD matrix
%   x       - input signal

% Pad signals with zeros
x = [zeros(1,64) x_init zeros(1,64)]';

% Find energy of input signal
Ex = sum(abs(x).^2);

% Define window length for different spectrogram
wlen = [12 14 16];
nwins = length(wlen);

% Make FFT size = 2*largest window length
nfft = 2*wlen(nwins);

% Calculate spectrogram for each window length
sg = zeros(nwins,nfft,length(x)+1);     % initialize spectrogram matrix
for kk = 1:nwins                        % for each window length...
                                        % zero pad signal by wlen
    xx = [zeros(1,fix(wlen(kk)/2)) x' zeros(1,fix(wlen(kk)/2))];
                                        % compute spectrogram
    [stft,F,T] = spectrogram(xx,wlen(kk),wlen(kk)-1,nfft,1);
    sg0 = abs(stft).^2;                 % spectrogram = |B|^2
    sg0 = Ex*sg0/sum(sum(sg0));         % normalize to signal energy
    sg(kk,:,:) = sg0;                   % save spectrogram in matrix
end

% Calculate MCE-PTFD prior, use combined spectrograms
c_spec = sg(1,:,:);
for i = 2:nwins
    c_spec = c_spec.*sg(i,:,:);
end
prior = sqrt(squeeze(c_spec));
prior = Ex*prior/sum(sum(prior));       % normalize to signal energy

tm = [abs(x).^2; 0];
tm = tm';

fm = fft(x_init,length(x_init));
fm = abs(fm).^2;

% Calculate MCE-PTFD (10 iterations)
ptfd = mce_ptfd(x,prior',tm,fm,500);

ptfd = ptfd(65:65+length(x_init)-1,:);
T = 1:length(x_init);
F = F';
```

BIBLIOGRAPHY

1. Loughlin, P., Atlas, L., and Pitton, J., "Construction of Positive Time-Frequency Distributions", IEEE Transactions on Signal Processing, vol. 42, no. 10, pp. 2697-2705, 1994.

2. Pitton, J., Atlas, L., and Loughlin, P., "Applications of Positive Time-Frequency Distributions to Speech Processing", IEEE Transactions on Speech and Audio Processing, vol. 2, no. 4, pp. 554-566, 1994.

3. El-Jaroudi, A., "Minimum Entropy Time-Frequency Distributions", Proceedings of the Tenth IEEE Workshop on Statistical Signal and Array Processing, pp. 603-606, 2000.

4. Cohen, L., and Posch, T., "Positive Time-Frequency Distribution Functions", IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. ASSP-33, no.1, pp. 31-38, 1985.

5. Fonollosa, J., "Positive Time-Frequency Distributions Based on Joint Marginal Constraints", IEEE Transactions on Signal Processing, vol. 44, no. 8, August 1996.

6. Cohen, L., *Time-Frequency Analysis*. Upper Saddle River, NJ: Prentice Hall PTR, 1995.