Acoustic Source Localization

by

Nilu Zhao

B.S., Massachusetts Institute of Technology (2014)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

Signature redacted

Author . . . . . .
Department of Mechanical Engineering
May 7, 2016

Signature redacted

Certified by . . . . . .
George Barbastathis
Professor
Thesis Supervisor

Signature redacted

Accepted by . . . . . .
Rohan Abeyaratne
Chairman, Department Committee on Graduate Theses

Acoustic Source Localization

by

Nilu Zhao

Submitted to the Department of Mechanical Engineering
on May 7, 2016, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

Many technologies rely on underwater acoustics. Most of these applications are able to denoise Gaussian noise from the surrounding, but have trouble removing impulse-like noises. One source of impulse-like noise is the snapping shrimps. The acoustic signals they emit from snapping their claws hinder technologies, but can also be used as a source of ambient noise illumination due to the rough uniformity in their spatial distribution. Understanding the spatial distributions of these acoustic signals can be useful in working to mitigate or amplify their effects.

Our collaborators in Singapore use a multi-array sensor to take measurements of the sound pressures of the environment in the ocean. This thesis investigates in solving the signal reconstruction problem – given the measurements, reconstruct the locations of the orignal signal sources. A numerical model for the system consisting of the snapping shrimp signals, the environment, and the sensor is formulated. Three methods of reconstruction - Disciplined Convex Programming, Orthogonal Matching Pursuit, and Compressive Sensing- are explored, and their robustness to noise, and sparsity are examined in simulation. Results show that Two-Step Iterative Shrinkage Threshold (TwIST) is the most robust to noisy and non-sparse signals. The three methods were then tested on real data set, in which OMP and TwIST showed promising consistency in their results, while CVX was unable to converge. Since there is no available information on ground truth, the consistency is a promising result.

Thesis Supervisor: George Barbastathis
Title: Professor

Acknowledgments

First and foremost, I would like to thank my advisor, Professor George Barbastathis. He has guided me on my research since I was an undergraduate at MIT. I am extremely appreciative of his guidance –he has taught me the ways of research and given me much helpful advice.

I would like to thank our collaborators in the National University of Singapore, Professor Mandar Chitre, and Too Yuen Min in their help on the development of the forward model, the acoustic data for the shrimps, and discussions on the project.

I am thankful for Yi Liu, for her mentorship and help on research; Wensheng Chen, for his helpful suggestions and discussions on the project. I am also grateful for the rest of my labmates: Shuai Li, Kelli Xu, Adam Pan, Justin Lee, and past labmates Yunhui Zhu, Max Hsieh, Jeong-gil Kim, for their help and many good memories in the last two years.

Finally, I am thankful for my family, for their love and continuous support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

There are many technologies that rely on underwater acoustics, such as sonar systems. The surrounding environment often plays an important role in these applications in real life. Noises from the surroundings can sometimes hinder the effectiveness of these applications, while at other times these noises can be exploited, for example as sources of illumination [10]. Understanding the spatial distributions of these acoustic signals can be useful in working to mitigate or amplify their effects.

A common source of these noises are marine animals. The snapping shrimps snap and release cavities, by which snapping noises are made. They are present in shallow waters. Snapping shrimps makes impulse-like sounds and are sparse in the environment [10]. The unique shape of their acoustic signal they emit, and their sparsity in the ocean allows the use for certain mathematical approximation in modeling the system of which the wave propagates and the sensor senses, and certain methods to reconstruct their signals.

The contents of this work focus on implementing such a system that simulates the underwater environment with snapping shrimps and reconstructs the original signals. Three methods of reconstruction - Disciplined Convex Programming, Orthogonal Matching Pursuit, and Compressive Sensing- will be explored, and their robustness to noise, and sparsity will be examined. The methods are tested on real data, and the best method is optimized to achieve best reconstruction results.

1.1 Convex Optimization

Convex optimization finds solutions to optimization problems. Generally, optimization problems are difficult to solve. The solutions either take very long time to compute, or are difficult to find. However, certain types of optimization problems are solvable under acceptable computation time. Convex optimization problems are one of such type. [5].

For a problem to be solved by convex optimization, its objective and constraint functions need to be convex. Often times, problems can be transformed such that they have convex objective and constraint functions [5].

Disciplined convex programming (DCP) is a method modeling of convex programming, which is a type of non-linear programming that includes least squares, linear programming, and convex quadratic programming. DCP has specific problem setups for users to follow to solve convex programs [11]. CVX is a modeling language developed for modeling convex optimization systems in Matlab. It uses disciplined convex programming in Matlab [12] and is the software used in the work for this thesis in the convex programming part.

1.2 Orthogonal Matching Pursuit

Matching pursuit is an algorithm that decomposes and describes a signal in its most important signature components. The set of components, or atoms, is called the dictionary. By representing the signal in a way such that it is sparse with respect to the dictionary, matching pursuit is able to represent the original signal as a series of weighted atoms [14]. For example, a typical sinusoidal signal is sparse in the frequency domain. The dictionary for this signal could be composed of Fourier series. Matching pursuit is developed by Zhang and Mallat [14] to compute adaptive signal representations, especially in adaptive time-frequency transformations.

Orthogonal Matching Pursuit (OMP) is a modified matching pursuit algorithm. OMP updates existing coefficients by applying the projection operator to the residual so that the residual is orthogonal to the selected atoms. This allows a improved convergence

rate of OMP, compared to matching pursuit for nonorthogonal dictionaries [16]. Orthongal matching pursuit was developed to provide full backward orthogonality on the residual at every step, resulting in better convergence than Matching Pursuit.

1.3 Compressive Sensing

Compressive sensing is a technique in signal processing that allows for accurate reconstruction of sparse signals given a limited number of measurements and an underdetermined linear system. Traditionally, to recover a signal, enough samples need to be taken to avoid aliasing and reconstruct with good accuracy. However compressive sensing shows that this rule does not need to hold if we know that the signal is sparse, and the system is incoherent, which means that the system should be able to spread out the sparse signal in the measurement. When the system is coherent, or fails to spread out the original signal appropriately, this technique fails [8].

1.4 Thesis Outline

Chapter two discusses the problem this thesis addresses, including the problem's physical and mathematical model. Chapters three, four, and five provide more detailed background on the theory of convex optimization, orthogonal matching pursuit, and compressed sensing, respectively. Chapter six shows the results for these algorithms, and their robustness to noise and sparsity, and their performance on the real data. Chapter seven discusses the optimization of parameters in the applied compressive sensing algoritm. Chapter eight is the conclusion.

Chapter 2

Problem Model and Approach

2.1 Background

There are many snapping shrimps living in Singapore's shallow waters. Typical species include the smooth snapping shrimp, ornamented snapping shrimp, and many-band snapping shrimp [21]. Their body length is 4-6cm [1], and their snapper claw can be as long as 2.8 cm [20]. Figure 2-1 shows the smooth snapping shrimp, and Figure 2-2 shows the the snapping claw pieces. When the claw of the shrimp snaps, a cavitation bubble is created from the action, and the collapse of the bubble makes an impulse-like sound.

Figure 2-1: The Smooth Snapping Shrimp [21].

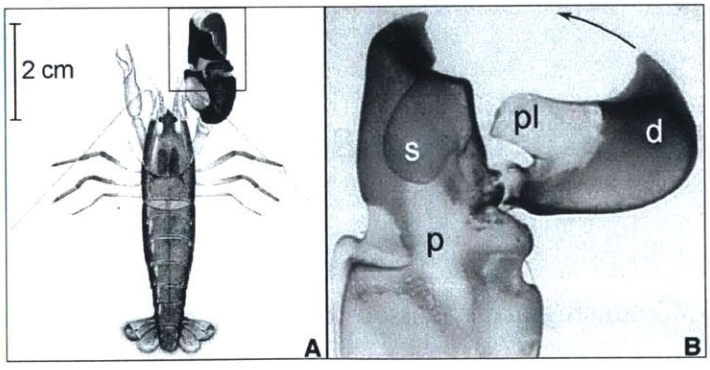

Figure 2-2: Diagram of snapping shrimp and claw snapping mechanism [20].

Their impulse-like non-Gaussian noise hinders the performance of linear correlators (which work well with Gaussian noise), but can also be used as an illumination

source in ambient noise imaging [17]. The frequency range of the snapping noise is 2-300kHz [10] and the sound pressure of these acoustic sources can be higher than $190dB\ re\ 1m\ 1\mu Pa$ [2]. Figure 2-3 shows the sound pressure of a snap, measured at 4 cm away, and sequence of the snapping motion taken at 40,500 fps [20].

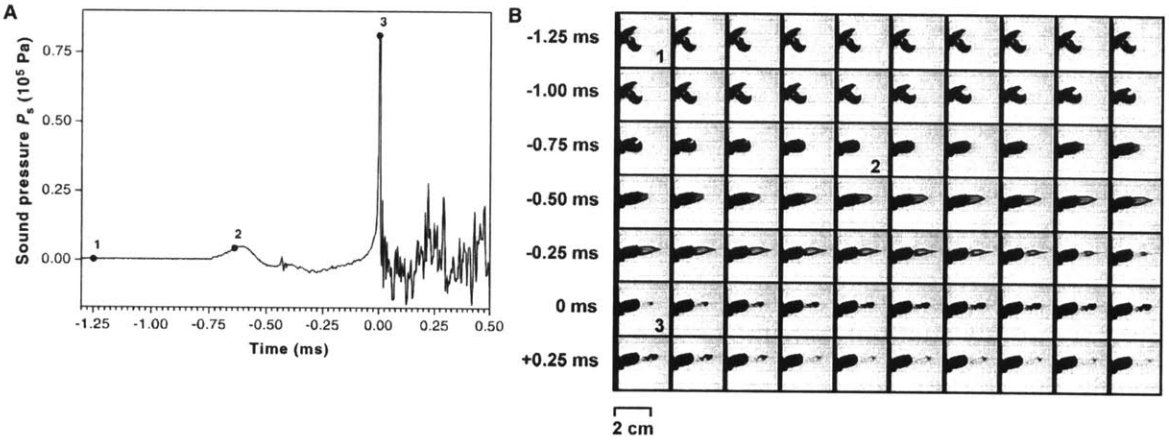

Figure 2-3: Sound pressure profile of the snapping shrimp, measured at 4cm away [20].

2.2 Experiment Setup

A 2D sensing array is used to capture the surrounding sound pressure. This Ambient Noise Imaging (ANI) system is called Remotely Operated Mobile Ambient Noise Imaging System (ROMANIS). A photo of it is shown in Figure 2-4. Its sensing array is composed of 508 sensors, populated on a circular plane of 1.3m in diameter. It weighs about half a ton and has a resolution of 1degree at $60kHz$ [19]. The bandwidth is 25-85kHz, and the weight is 650kg [19].

Figure 2-4: Remotely Operated MobileAmbient Noise Imaging System, developed by Acoustic Research Laboratory (ARL) at the Tropical Marine Science Institute under the National University of Singapore [19].

Each of these sensors measures the sound pressure at a rate of 196 kSa/a [9], which creates a time series data. Not all sensors of ROMANIS were able to provide reading during the measurement – Figure 2-5 shows ROMANIS will all sensors number, and Figure 2-6 shows ROMANIS with numbered sensors that were providing readings.The data is then read, processed by a custom software [19]. ROMANIS is able to produce images of targets at a distance of up to 120m [9].

The measurements taken by ROMANIS are uncaliberated sound pressure and the sound pressures are relative. To convert them to absolute pressure, there is a conversion factor is multiplied to the uncaliberated readings. This calibration is provided in Appendix A.

2.3 Mathematical Model

2.3.1 Signal Model

To simulate the system, also known as the 'forward operator' in sparse sensing literature, the signals of the shrimp and the acoustic propagation need to be modeled. As mentioned previously, the snaps made by the shrimps are impulse-like, shown by Figure

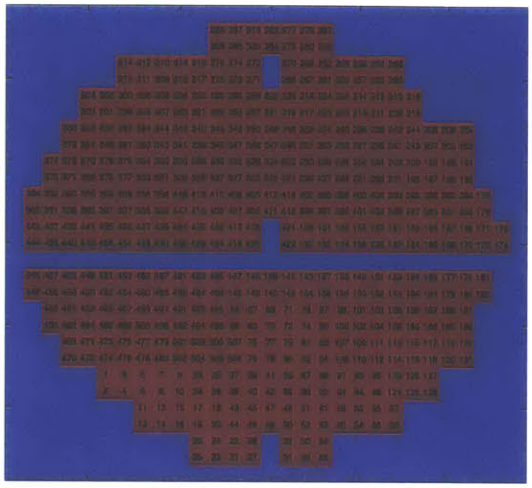

Figure 2-5: ROMANIS with all sensors labeled.

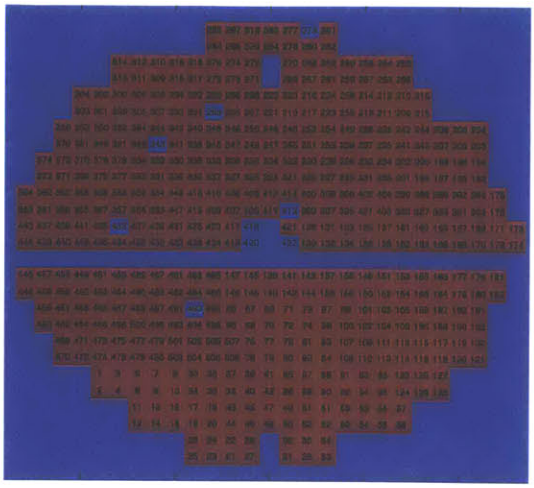

Figure 2-6: ROMANIS with all working sensors labeled.

2-3. In this work, the acoustic signals from the shrimp are approximated as impulses.

Our collaborators in Singapore, Professor Mandar Chitre, and Too Yuen Min helped us developed the following formalism and sampling scheme for simulation: A space of azimuth angles, ϕ from -10 degrees into 10 degrees, and elevation angles, θ from -10 degrees to 10 degrees are partitioned to 1 degree sections. A range, Γ of 1 frame (data point) into 100 frames was partitioned to 1 frame sections. There could be many more frames, but due to the limit in compuation power, it is constrained to 100 in simulation. In other words, a cone-like shaped space is divided into smaller truncated cones, or frustums. This partion scheme is decided based on the angular resolution of the system (1 degree), and the level of similarity between the system propagation vectors.

To reduce the dimension of the signal representation, the three dimensional space partitions are stacked together vertically into a column vector. This column vector iterates through the space in the following way: elevation angle, azimuth angle, and range. When a signal is present in one of the frustums, a '1' is present in the corresponding vector location to represent it. Figure 2-7 shows a shrimp in 3D, and Figure 2-8 shows the signal representation of the shrimp in 1D vector form.

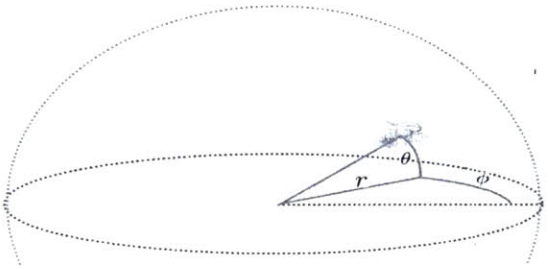

Figure 2-7: A shrimp located in 3D space.

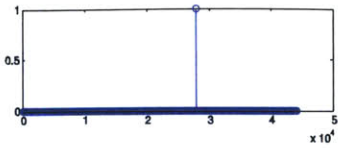

Figure 2-8: Signal representation of the shrimp in 1D vector form.

2.3.2 System Model

The model for the system is formed based on the delay (in frames) in the signal propagating to each sensor. The number of frames is found by the following equation:

$$\tau = \Gamma - \frac{x \times sin(\phi)cos(\theta) + y \times sin(\theta)}{c} \times Fs \tag{2.1}$$

[15], where τ is the time delay in frames, Fs, the sampling rate is 196000Hz), $c = 1540\frac{m}{s}$ is the speed of sound in water, x is the horizontal distance from each sensor to the middle of the phase array, y is the vertical distance from each sensor to the middle of the phase array. The reduction in magnitude of the sound pressure in propagation has not been accounted for in this model for simplification purposes and also partly because the change is small compared to the scale of the problem setup.

This delay is calculated for each sensor to each signal, forming matrix A. This matrix is multiplied by the signal vector to produce the response on the receiver. To make the system more realistic, white Gaussian noise is added to simulate the noise from the electronics.

Here, the mirror image of the sound generated from the reflection at the surface of water has been disregarded. Ideally, the sources emit spherical waves. However, to simplify the model and the calculation, linear approximations are made to the spherical spread of the acoustic waves.

One thing to note is that the similarity between column vectors could make the

forward operator ill-posed. If the response vectors of two signals are very similar, and substantial noise is present in the system, the algorithms would be less likely to trace back to the true signals, because the characteristics of their responses are masked by noise. When two vectors are close together, their response vectors are similar because the difference in the times their signals reach each of the sensors are very small.

Chapter 3

Convex Optimization

Optimization is used in many real world applications where the best solution is desired out of many that satisfy the requirement. Some applications are portfolio optimization, and data fitting [5]– all of these applications require some kind of optimization with respect to certain constraints, which is similar to our problem which will explained in more detail later in the section . This chapter will discuss convex optimization [5], and the CVX Matlab software.

3.1 Optimization Problems

3.1.1 Problem Characteristics

Optimization problems hold the following form:

$$\begin{aligned} &\text{minimize} && f_0(x) \\ &\text{subject to} && f_i(x) \leq b_i,\ i = 1, \ldots, m \end{aligned} \tag{3.1}$$

where x is the vector being minimized, f_0 is the objective function, f_i is the constraint function, and b_i is the bound for the constraints. In other words, among all the solutions that satisfy $f_i(x) \leq b_i$, we would like the x that produces the smallest $f_0(x)$ [5].

A convex optimization problem is an optimization problem that satisfies the follow-

ing inequality:

$$f_i(\alpha x + \beta y) \leq \alpha f_i(x) + \beta f_i(y) \tag{3.2}$$

where $x, y \in \mathbf{R}^n$, and $\alpha, \beta \in \mathbf{R}$, satisfying $\alpha + \beta = 1, \alpha \geq 0, \beta \geq 0$. In other words, the functions $f_0, \ldots f_m$ in equation 3.1 need to be convex [5].

There are two main classes of convex optimization problems: least square and linear programming. Least square problems have the following problem setup:

$$\text{minimize} f_0(x) = ||Ax - b||_2^2 = \Sigma_{i=1}^{k}(a_i^T x - b_i)^2 \tag{3.3}$$

where a^T are rows of A. Notice that it does not have a constraint function. The solution is as follows:

$$(A^T A)x = A^T b. \tag{3.4}$$

Linear programming problems have the following setup:

$$\begin{aligned} &\text{minimize} \quad c^T x \\ &\text{subject to} \quad a_i^T \leq b_i, \quad i = 1, \ldots, m \end{aligned} \tag{3.5}$$

[5]. In correspondence with the scope of this problem and the system mentioned in Chapter 2, the linear programming model is used for the problem setup.

3.1.2 Problem Setup

The convex optimization problem setup for this work is:

$$\begin{aligned} &\text{minimize} \quad ||x||_1 \\ &\text{subject to} \quad ||y - Ax||_2 \leq \lambda ||N||_2 \end{aligned} \tag{3.6}$$

where λ is a constant weight factor, N is the noise vector, y is the measurement, and A is the system matrix. In reference to the linear programming setup, c from 3.5is the ones vector, outputing the l_1 norm of x.

3.2 Theory

To verify the validity of using the linear programming problem setup in convex optimization, the functions $||x||_1$ and $||y - Ax||_2$ need to be shown to be convex.

3.2.1 Convexity

As Boyd and Vandenberghe explain, an affine set S is one that contains the line through any two distinct points in the set 3.7. That is, S contains the linear combinations of any number of points in it, with the sum of the linear coefficients being 1. For example, the solution set of linear equations $S = x : Ax = b$ is affine. A geometric interpretation is show in Figure 3-1.

$$x = \theta x_1 + (1 - \theta x_2) \tag{3.7}$$

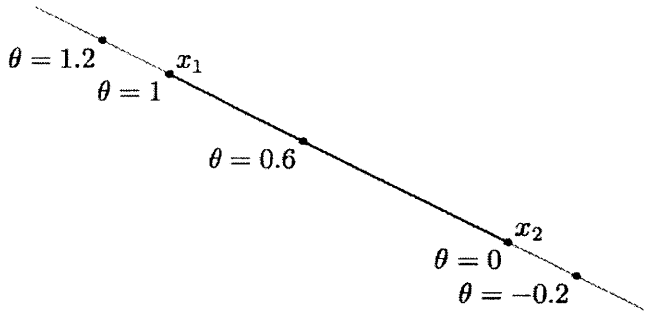

Figure 3-1: Illustration of an affine set [5].

A convex set S is one that contains line segment between any points in it. Some examples of convex sets are shown in Figure 3-2

$$x = \theta x_1 + (1 - \theta)x_2 \quad 0 \leq \theta \leq 1 \tag{3.8}$$

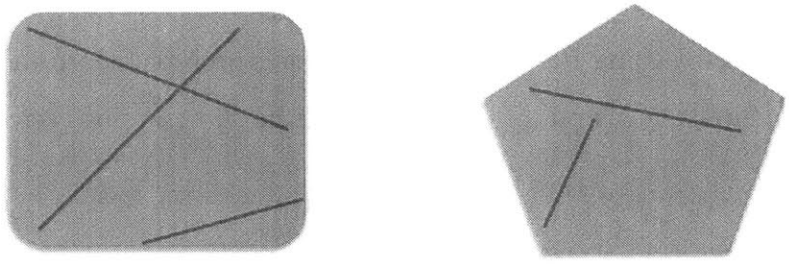

Figure 3-2: Examples of convex sets; the line segment between any two points is contained in the set.

All affine sets are convex sets, since they have the line segments constructed from any two points from it (as defined previously of convex sets).

An affine function is one such that it is a sum of linear functions and constants 3.9, where S is a convex set. For example, functions of the form $Ax + b$ are affine. Affine functions preserve convexity [5].

$$fS = f(x)|x \in S \tag{3.9}$$

A function f is convex if the domain of f is a convex set and for all x, $y \in \mathbf{dom}f$, and $0 \leq \theta \leq 1$ in

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y). \tag{3.10}$$

[5]

Here, the convexity of the setup of the problem is shown:

The norm function is convex, for norms greater than or equal to 1, that is:

$$\begin{aligned} &if \quad \mathbf{u}, \mathbf{v} \in \mathbf{R}^n \quad and \quad p \in [1, \infty) \\ &then \quad ||\mathbf{u}+\mathbf{v}||_p \leq ||\mathbf{u}||_p + ||\mathbf{v}||_p, \end{aligned} \tag{3.11}$$

for all $\mathbf{u}$ and $\mathbf{v}$.

This can be shown in the following way: Powers of absolute values, $f(x) = |x|^p$,

are convex. By the convexity requirement,

$$\begin{aligned} ||\theta\mathbf{u}+(1-\theta)\mathbf{v}||_p^p &= \Sigma_{i=1}^n|\theta u_i+(1-\theta)v_i|^p \\ &\leq \Sigma_i^n\theta|u_i|^p+(1-\theta)|v_i|^p \\ &= \theta||\mathbf{u}||_p^p+(1-\theta)||\mathbf{v}||_p^p \end{aligned} \tag{3.12}$$

It is apparent that $y-Ax$ is convex, because it is affine. More formally,

$$y-A(\theta x_1+(1-\theta)x_2)=\theta(y-Ax_1)+(1-\theta)(y-Ax_2).$$

3.3 CVX

CVX is a Matlab package that models and solves disciplined convex programs. In particular, it supports linear programming problems, and functions like l_1 norms, which are used in the problem setup [13]. As mentioned in Chapter 1, disciplined convex programming is a method of formulating problem setup according to a set of rules proposed by Grant, Boyd, and Ye [11]. CVX contains many solvers, and the results for this work were produced by the SDPT3 solver.

Chapter 4

Matching Pursuit

Matching pursuit is an algorithm that allows for adaptive signal representation. It decomposes a signal into its most important signature components, also called atoms. The set of atoms form a redundant dictionary [14].

In Chapter 1, it was mentioned that a typical sinusoidal signal is sparse in the frequency domain and its dictionary could be composed of Fourier series. However, it does poorly for a signal that is localized in time, because the coefficients would be spread out among the whole basis. One may suggest that it's possible to just represent the signal in the time basis, however, if the signal is a combination of one that is concentrated in time and another that is concentrated in frequency, it would be difficult to represent this combination sparsely. Matching pursuit provides a way for flexible decomposition for any signal, that results in only using a few atoms to represent the most important structures in the signal. Atoms are waveforms that adapt to the structures of the signal [14]. In this case, these atoms have features that adapt to the unique nature of the signal. In this work, the atoms that make up the dictionary are the unique propagation vectors of the signals each possible location.

4.1 Theory

Given a complete dictionary in which the signal is sparse, matching pursuit expands the signal over a small subset of waveforms that belongs to the dictionary. Let $\mathscr{D}$ be the

dictionary, and $\mathscr{D} = (g_\gamma)_\gamma$ that is g_γ is a vector in the dictionary, and f be the signal. Matching pursuit decomposes f into a combination of a set of vectors selected from $\mathscr{D}$. f can be represented by

$$f = < f, g_{\gamma 0} > g_{\gamma 0} + Rf \tag{4.1}$$

where Rf is the residual after projecting f onto $g_{\gamma 0}$. Since $g_{\gamma 0}$ and Rf are orthogonal,

$$||f||^2 = | < f, g_{\gamma 0} > |^2 + ||Rf||^2. \tag{4.2}$$

In order to express f with the most representative vectors, f is projected onto all of them, and matching pursuit picks the $g_{\gamma 0}$ that maximizes $| < f, g_{\gamma 0} > |$. After projecting f onto such $g_{\gamma 0}$, the same is done to Rf, the residual after the projection. Then Rf is projected on to the set of vectors in the dictionary, excluding the one that has already been picked out and a new residual is formed. This is repeated until a stop criterion is reached [14]. To better illustrate this point, each subsequent residual can be found by

$$R^{n+1}f = R^n f - < R^n f, g_{\gamma n} > g_{\gamma n}. \tag{4.3}$$

And f can be represented by a sum of the m vectors that's picked out:

$$f = \Sigma_{n=0}^{m-1} < R^n f, g_{\gamma n} > g_{\gamma n} + R^m f. \tag{4.4}$$

This decomposition is nonlinear, but it conserves energy, and is guaranteed to converge [14].

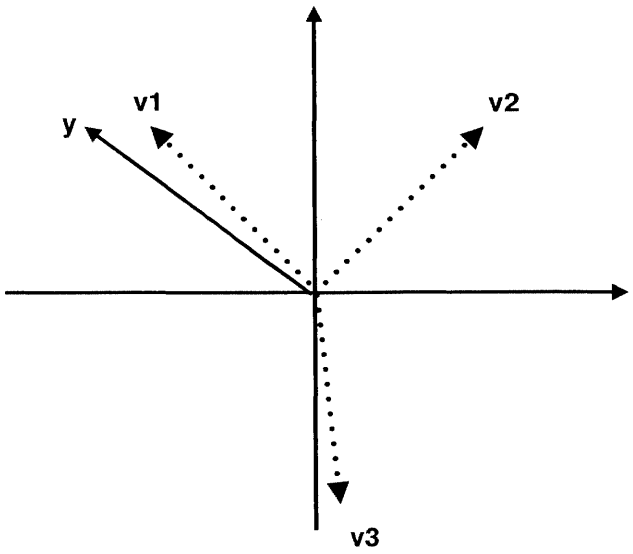

Figure 4-1: $v1, v2, v3$ are vectors in the dictionary, and y is the function that we would like to decompose.

Figure 4-1 shows y, and the atoms $v1, v2, v3$. Here, $v1$ would be chosen first, because the dot product between $v1$ and y is the largest of all dot products with vectors in the dictionary.

4.2 Orthogonal Matching Pursuit

Orthogonal Matching Pursuit (OMP) is a modified matching pursuit algorithm. Regular matching pursuit has a dictionary that consists of vectors that are not mutually orthogonal. Therefore, when residuals are subtracted, it can result in introducing components that are not orthogonal to the previous projected vectors. Orthogonal matching pursuit is different from matching pursuit in that the residual is always orthogonal to the chosen set of basis.

OMP updates existing coefficients by applying the projection operator to the residual so that the residual is orthogonal to the selected atoms. This allows a improved convergence rate of OMP, compared to matching pursuit for nonorthogonal dictionaries [18], which is why OMP is usually preferred over matching pursuit, and is used in

this work.

OMP computes the projection operator P as follows:

$$P = \Phi(\Phi * \Phi)^{-1}\Phi* \tag{4.5}$$

, where Φ is the set of already chosen atoms, and updates the residual by

$$R^{m+1}f = (I - P)R^m f \tag{4.6}$$

[18]. Similar to matching pursuit, OMP conserves energy, which is important for its convergence.

In [16], the orthongonality is achieved by projecting the residual to the set of chosen atoms in order again, and reevaluate the dot products each time.

4.2.1 Problem Setup

OMP is implemented to solve the follwing system:

$$y = Ax + N,$$

with the prior that x is sparse, and N is noise. The inputs to this algorithm are y,A, and k. k is the estimation for sparsity, and tells the algorithm how many signals to try to find, it is not present in this general setup of the problem.

Chapter 5

Compressive Sensing

Compressive sensing is a technique that allows for solving underdetermined linear systems given some priors, with fewer measurements than the typical requirement for signal sampling. It provides accurate estimations for solutions and is used widely as a signal processing technique.

5.1 Theory

5.1.1 Nyquist-Shannon Sampling Criteria

Traditional sampling methods require a sampling rate of at least twice the highest signal frequency for accurate signal reconstruction. Sampling at a rate lower than the Nyquist-Shannon sampling theorem could result in aliasing. Take the following example in Figure 5-1 the original signal is in blue, if we sample this signal using the points shown, the red signal could be constructed, which is incorrect.

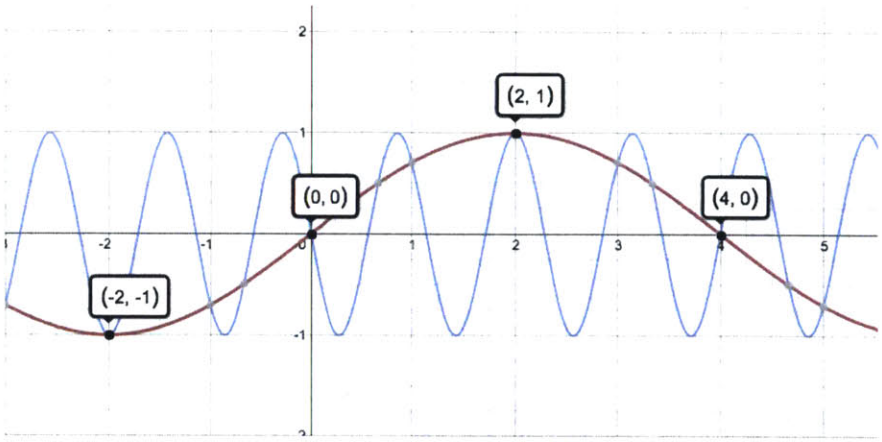

Figure 5-1: The original signal is in blue, the red is a possible signal constructed from sampled points shown.

Compressive sensing states that the Nyquist criteria can be violated, as long as certain conditions such as signal sparsity and incoherence (to be defined more precisely later) can be used to compensate for the limited measurements and, thus, still recover the signal accurately.

5.1.2 Sparsity and Measurement Requirement

One prior of compressive sensing is that the signal needs to be sparse. A signal is sparse when there are many zeros in its coefficients. In other words, a sparse signal f, can be represented by:

$$f = \Sigma_{i=1}^{n} x_i \psi_i, \tag{5.1}$$

where x_i are coefficients of $< f, \psi_i >$ and ψ_i are basis functions chosen from the basis Ψ. Sparse here implies that the set of coefficients X, have few nonzero x_i elements. In some applications, such as images, there may be many nonzero coefficients in the sparse basis, but it would be alright to discard some of the smaller coefficients, and still obtain a good representation of the original. Image compression methods such as lossy JPEG compression exploit this feature for effective storage of images.

The sparsity of the signal dictates the number of measurements that is needed for

good signal reconstruction. Candes [6] has formulated the relationship between the number of measurements required and the sparsity of the signal to correctly reconstruct:

$$m \geq C\mu^2 S log(n), \tag{5.2}$$

where m is the number of measurements, C is a constant, μ is the maximum of the coherence between the signal and the field of detection, S is the number of nonzeros in the signal, and n is the total length of the signal [8].

5.1.3 Incoherence

Having a sparse signal is not enough for compressive sensing to work. Another important factor that the system needs to satisfy is that the signal needs to be non-sparse in the measurement domain. In particular, there needs to be incoherence in mapping the signal to the measurement. This system needs to be able to spread out each signal, so that even with very few measurements, the characteristics of the original is still captured.

Figure 5-2 demonstrates this concept. The left colored column represents a vector that is the linear combination of the products of sparse coefficients (represented by the colors on the right) and their basis vectors. The colored columns in the matrix shows the atoms that the sparse entries picked out. The left column, the measurement, has the spread of each sparse entry. The shape of the matrix (having more columns than rows) confirms that this is an undertermined system, or an undersampling process, which cannot be solved by typical matrix inversions.

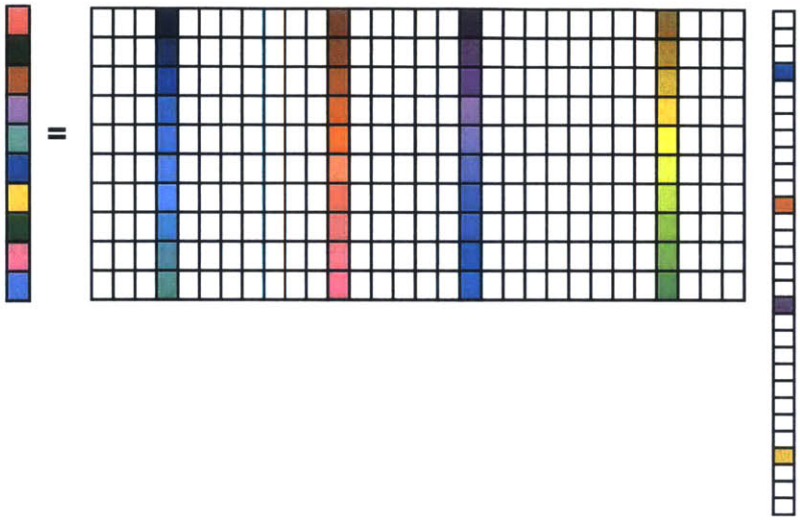

Figure 5-2: The measurement captures the spread of the sparse signal, transformed by the basis vectors.

Coherence can be computed as follows:

$$\mu(\Phi, \Psi) = \sqrt{n}\max_{1 \leq k,j \leq n} | < \phi_k, \psi_j > |, \tag{5.3}$$

where Φ is the sensing basis and Ψ is the representation basis [8]. It measures the correlation between Ψ and Φ. One can imagine that certain systems will not satisfy the incoherence requirement. For example, a one to one mapping system will result in large coherence between the sensing and sparse fields. Certainly if we were to wish to reconstruct the sparse signal accurately, all measurements would be needed in order to capture the original signal, since the measurement field is sparse itself as well. In addition to the incoherence of the sensing matrix, the Restricted Isometry Property (RIP) is useful to the robustness of compressed sensing, allows for accurate reconstruction under the presence of noise. To understand RIP better, let's look at the following equation

$$(1 - \delta_S)||x||_{l2}^2 \leq ||Ax||_{l2}^2 \leq (1 + \delta_S)||x||_{l2}^2, \tag{5.4}$$

where A is a matrix, and S are positive integers and δ is the isometry constant; this

equation holds for all x that is sparse. When δ_S is not too close to one, A is said to obey the RIP order S, and preserves the Euclidean length of the S-sparse signals, hence the nullspace of A does not contain S-sparse vectors [8]. This is important because it ensures that we can recontruct some of these vectors.

5.1.4 Recovery

Compressed sensing recovers via the following convex optimization:

$$\min||x||_1 \quad \text{subject to} ||Ax - y||_2 \leq \epsilon, \tag{5.5}$$

where ϵbounds the energy of the noise.

5.2 TwIST

Two-Step Iterative Shrinkage Thresholding (TwIST) is an algorithm developed by Bioucas-Dias and Figueiredo [4] that provides solutions to inverse linear problems. It solves many image restoration and compressed sensing problems. Its approach is to solve the minimization problem:

$$f(x) = \frac{1}{2}||y - Ax||^2 + \lambda\Phi(x), \tag{5.6}$$

where y, A, x are the same as defined in the previous sections, λ is a weight vector, Φ is a regularization function, and $\frac{1}{2}$ is an energy matching coefficient. Minimizing this objective function allows for a balance in the solution between the closeness of x to solving $y = Ax$, and some optimization factor, which could be some normalization function. This balance is needed because often in real world applications, there are noises captured in the measurement, therefore an exact solution to $y = Ax$ will not be accurate. Minimizing the energy or magnitude at the same time ensures that overall the signal is low energy or sparse. The weighing factor λ controls how much we would like to minimized one factor over the other. As mentioned, some typical regularization functions

are l_1, l_2 norms, or total variation, which is used widely in image reconstruction.

TwIST was developed on the foundations of iterative reweighted shrinkage (IRS) and iterative shrinkage/thresholding (IST) algorithms.

IST algorithms updates the solution by:

$$\mathbf{x}_{t+1} = (1-\beta)\mathbf{x}_t + \beta\Psi_\lambda(\mathbf{x}_t + \mathbf{K}^T(\mathbf{y}-\mathbf{K}\mathbf{x}_t)) \tag{5.7}$$

, where $\beta > 0$, and Ψ_λ is the Moreau proximal mapping defined as

$$\Psi_\lambda(y) = \text{argmin x} \quad \frac{||Ax-y||^2}{2} + \lambda\Phi(x). \tag{5.8}$$

IRS iterates based on the follow equation:

$$\mathbf{x}_{t+1} = \text{solution}\mathbf{A}_t\mathbf{x} = b \tag{5.9}$$

, where $b = K^T y$ and $A_t = \lambda D_t + K^T K$, and D_t is a diagonal matrix that depends on x_t and Φ

TwIST combines IST and IRS: initialize the solution as $\mathbf{x}_0$,

$$\mathbf{x}_1 = \Gamma_\lambda(\mathbf{x}_0) \tag{5.10}$$

, where

$$\Gamma(\mathbf{x}_0) = \Psi_\lambda(x + A^T(y - A\mathbf{x})) \tag{5.11}$$

and find each iteration thereafter as

$$\mathbf{x}_{t+1} = (1-\alpha)\mathbf{x}_{t-1} + (\alpha-\beta)\mathbf{x}_t + \beta\Gamma_\lambda(\mathbf{x}_t) \tag{5.12}$$

,where α, β, β_0 are weight constants. TwIST iterates based on two previous steps, rather than just one. TwIST also converges faster than IST [3].

5.2.1 Problem Setup

The parameters of the acoustic localization problem complies with the incoherence and sparsity requirements.

The minimization function implemented in this work is:

$$f(x) = \frac{1}{2}||y - Ax||^2 + \lambda||x||_1 \tag{5.13}$$

Compressed sensing finds the sparsest solutions to the problem. For a multi-dimensional sensing matrix and measurement, compressed sensing strives for the solution that gives the smallest dimensionality. In other words, given a solution that falls onto a hyper-plane, compressed sensing will favor the one that lies on the axis, minimizing the l_0 norm. Therefore intuitively, we would like to set the regularization function Φ to be $||x||_0$, however in this work, $\Phi = ||x||_1$ is employed. This choice was taken because in practice, solving minimization of l_0 norm is NP-hard. This can be seen from the following logic: to find the solution for a 1-sparse signal that is N dimensioned, we need to examine all the values, which results in $\binom{N}{1}$ possibilities. Similarly for a signal that is m-sparse, there are $\binom{N}{k}$ possibilites. The overalL complexity is $\Sigma_{i=1}^{N}\binom{N}{i}$, which is NP hard.

The alternative solution is to minimize the l_1 norm, which minimizes the magnitude of the solution, and is covex (also shown in Chapter 3 under Convexity) and much easier to solve. Theoretical results have shown that under sufficient conditions, the optimal solution to l_1 norm minimization is also optimal to l_0 norm minimization [7].

Chapter 6

Results

In this chapter, results on simulations and real data are shown. First, the system and signal described in Chapter 2 are implemented in simulation. This was done to examine the feasibility of the three algorithms: CVX, OMP, and TwIST, in reconstructing the signals. CVX, OMP, and TwIST are compared in their robustness to noise, sparsity, and a combination of both. These comparisons were made because we would like to know how real-world factors such as the amount of noise in the measurement, and the number of shrimps in the space affect the accuracy these methods. Then, the three algorithms were implemented on the real data from ROMANIS.

6.1 Simulation

Simulations are made based on the system explained in Chapter 2. The following sections show the results of the three algorithms's robustness to noise, sparsity, and their combination.

6.1.1 Robustness to Noise

There are many sources of noise in the experiment. Noise in this our case is any unwanted signal. For example, there is noise resulting from signals emitted by sources other than the snapping shrimps, such as other types of marine animals; there also is

noise from the electronics of the experimental system. Since noise is a component that needs to be taken into account in the real world system, we would like to find out how different levels of noise affect the signal reconstruction accuracy of the algorithms.

Noise is usually quantified by signal-to-noise ratio (SNR). SNR is usually expressed in the log decibel scale, since signals can have a large dynamic range. SNR is given by:

$$\text{SNR} = 10log_{10}(\frac{P_{signal}}{P_{noise}}), \tag{6.1}$$

where P is the power. A SNR of 0 would indicate that the power of signal is equal to the power of noise.

Figures 6-1,6-2,and 6-3 show how each method performs at different noise levels. There were 3 shrimps (signals) in each trial. The performance is evaluated upon the accuracy in the reconstruction of the locations of the signals. The blue line indicates the number of correct locations of signal in reconstruction (true positives), and the red line indicates the number of incorrect locations (false positives) the reconstruction output.

A quick look at the results shows that the algorithms are performing well even at negative SNR, when the power of the noise is greater than the power of signal. This can seem misleading, but can be explained by the nature of the signal. The signal is very sparse, and hence the power is low, whereas the noise is non-sparse, and the power across the domain can be greater. Therefore, a second axis showing the peak-signal-to-noise ratio is also shown on the plots. PSNR takes into account of the peak value of the signal. PSNR can be calculated by:

$$\text{PSNR} = 10log_{10}(\frac{peakvalue^2}{MSE}), \tag{6.2}$$

where MSE is the mean squared error. All three methods have made many errors at low PSNR values.

The results show that the three algorithms fail at around $-20dB$ SNR, or around $5dB$ PSNR. The results for CVX fluctuates a lot, which is due to the limited number of trials conducted at low SNR. CVX took many iterations to converge at low SNR, and given the limited computation power, only a few trials were conducted (more trials were

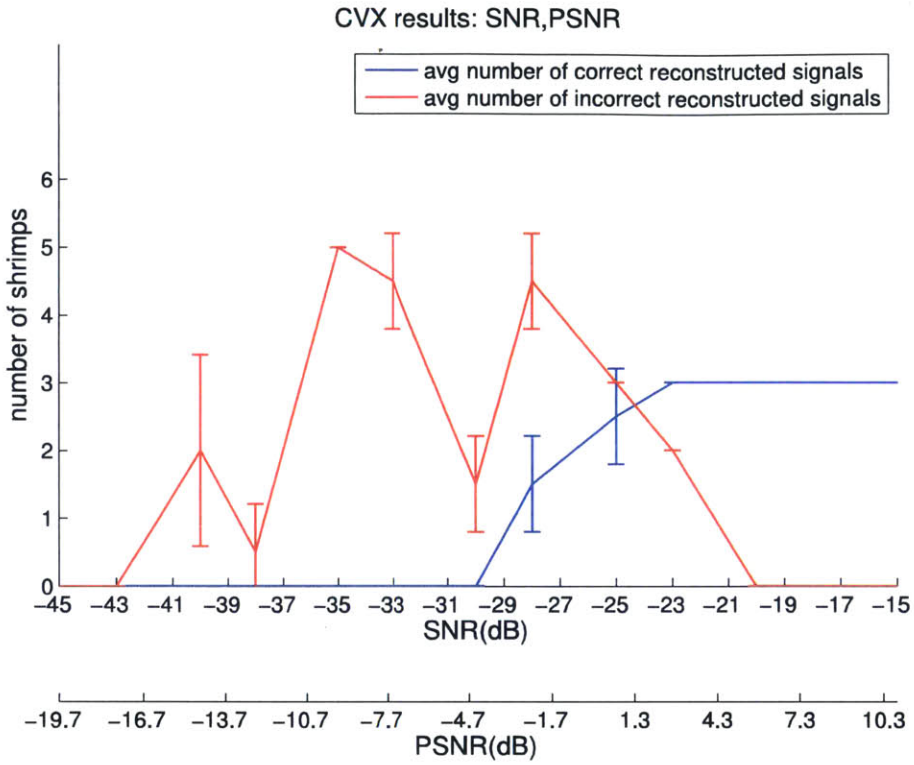

Figure 6-1: Performance of CVX in presence of noise.

conducted at higher SNR). However, from the plot, we can determine at what SNR CVX starts to fail. One thing to point out about CVX problem setup is that an estimate of noise level is given, and this is not done for OMP or TwIST problem setups. In the next section, the effect of the knowledge of noise in CVX setups is shown when dealing with real data.

OMP plot shows seemingly symmetric curves. This feature is resulted from the algorithm setup – OMP requires an input of the number of signals the user is trying to find. And in these trials, the exact number of signals were fed into the system (this also explains the small error bars when SNR is high and when it's low). This might not seem like an appropriate solution when applied to real world data when the number of signals is unknown, Figures 6-4 and 6-5 demonstrates that this is not a concern. Figure 6-4 shows the results for reconstruction given the correct number of signals (3), Figure 6-5 shows the results for the same measurement, given tenfold the correct number of signals (30). It is obvious that Figure 6-5 has three significant signals, and many very

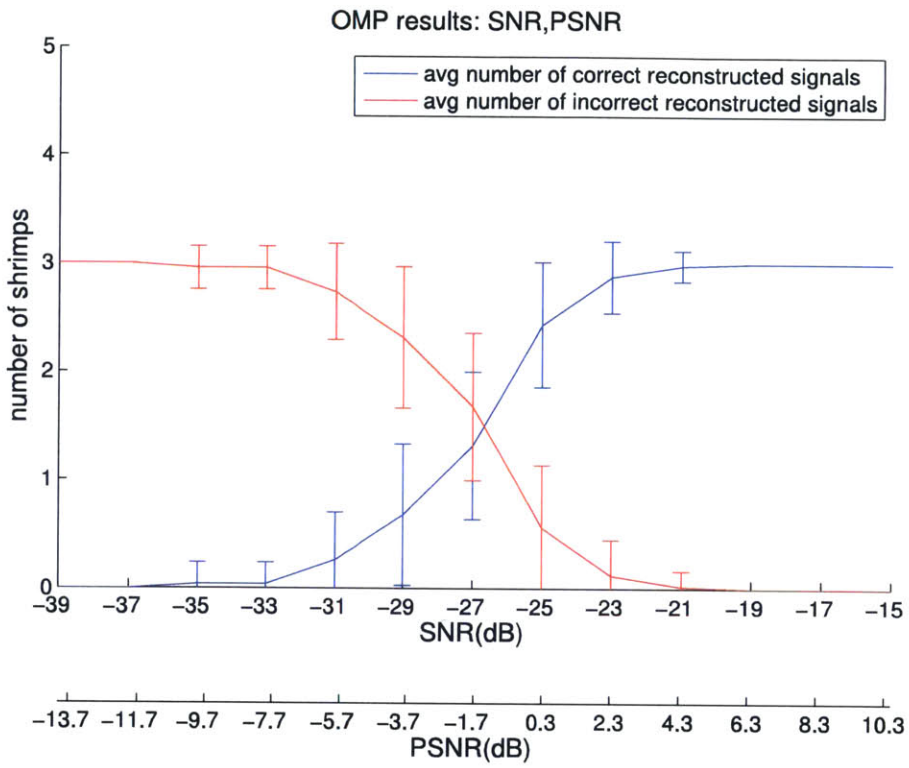

Figure 6-2: Performance of OMP in presence of noise.

small ones. These three signals are the correct ones. The smaller signals are the extraneous ones that the algorithm was trying to find because we forced it to find this many signals. The extraneous signals can be easily prevented when processing real data by setting a threshold factor.

The results from TwIST shows a great increase in the number of false positives right after $-20dB$ SNR. It fails very quickly, with growing error at lower SNR. From these three results, OMP fails slower than TwIST and CVX.

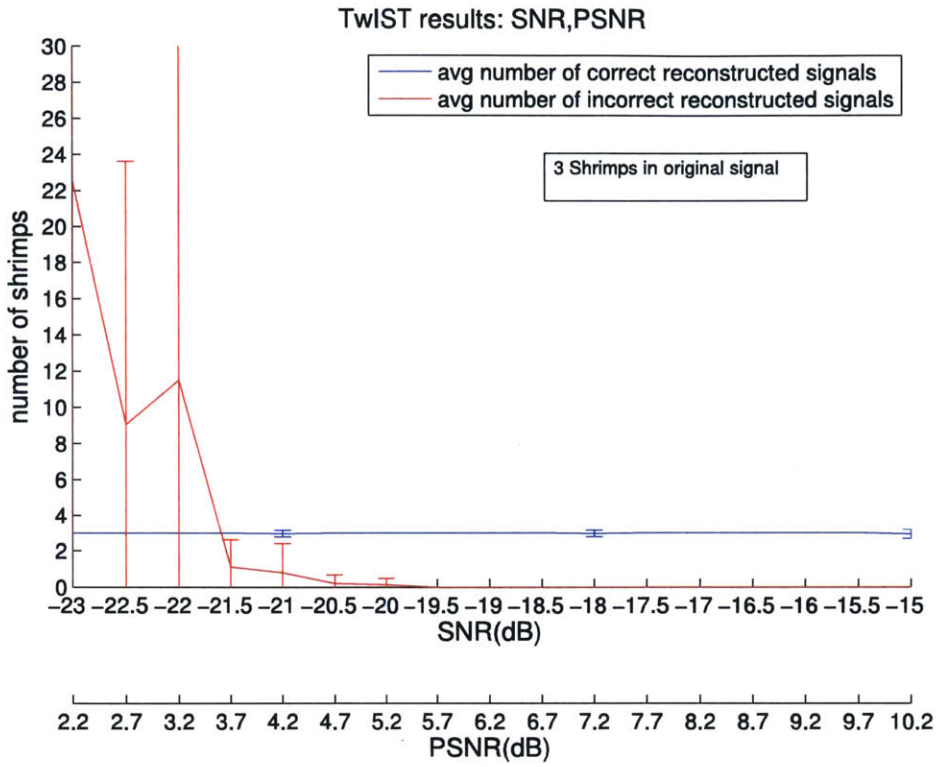

Figure 6-3: Performance of TwIST in presence of noise.

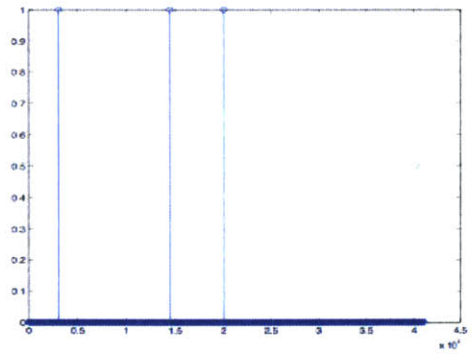

Figure 6-4: OMP reconstruction given exact number of shrimps.

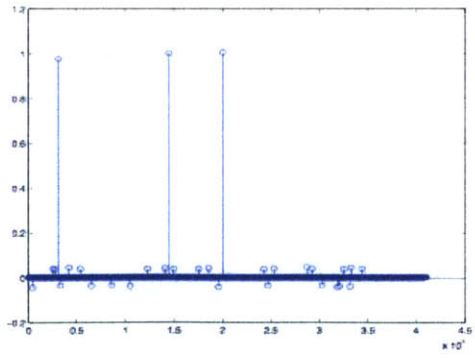

Figure 6-5: OMP reconstruction given 10x number of shrimps.

6.1.2 Robustness to Sparsity

In addition to how the methods perform under noise, we would also like to explore how they perform at different levels of sparsity. Research on directionality of acoustic signals from shrimps has shown that the signals are roughly isotropic, or uniform in all directions [17]. Therefore in this set of simulations, different number of shrimps are placed randomly in space in a unform distribution. The three algorithms are then tested in their robustness to sparsity, and no noise is added. TwIST and OMP specifically have sparsity requirements – both work under the assumption that the original signal is sparse.

Figures 6-6, and 6-7 give a sense of how sparse the space would look like with 1000 shrimps present (2% non-sparse) and 8000 shrimps present (18% non-sparse). Figures 6-8, 6-9, and 6-10 show the robustness of the methods with respect to signal sparsity. Same as previous results, the y-axis is the number of shrimps, the green line is the ground truth of the number of shrimps in original signal, the blue line is the number of correct recontructed locations (true positives), the red line number of incorrect reconstructed locations (false positives), the black line (only apparent for TwIST results) is the total number of reconstructed signals (sum of true positives and false positives). The sparsity ranges from 2% to 18%.

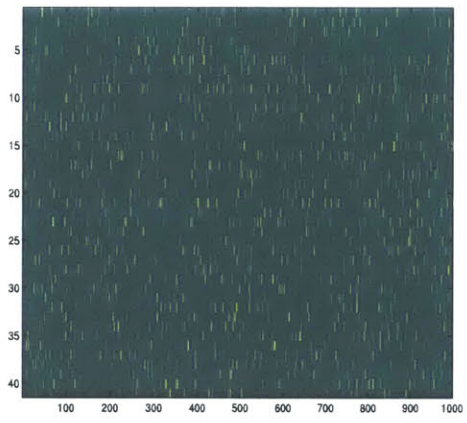

Figure 6-6: Illustration of 2% sparsity in 2D space.

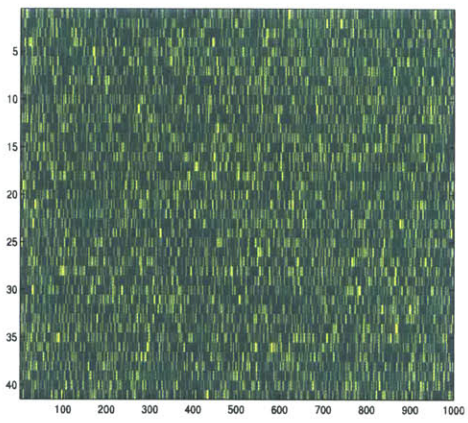

Figure 6-7: Illustration of 18% sparsity in 2D space.

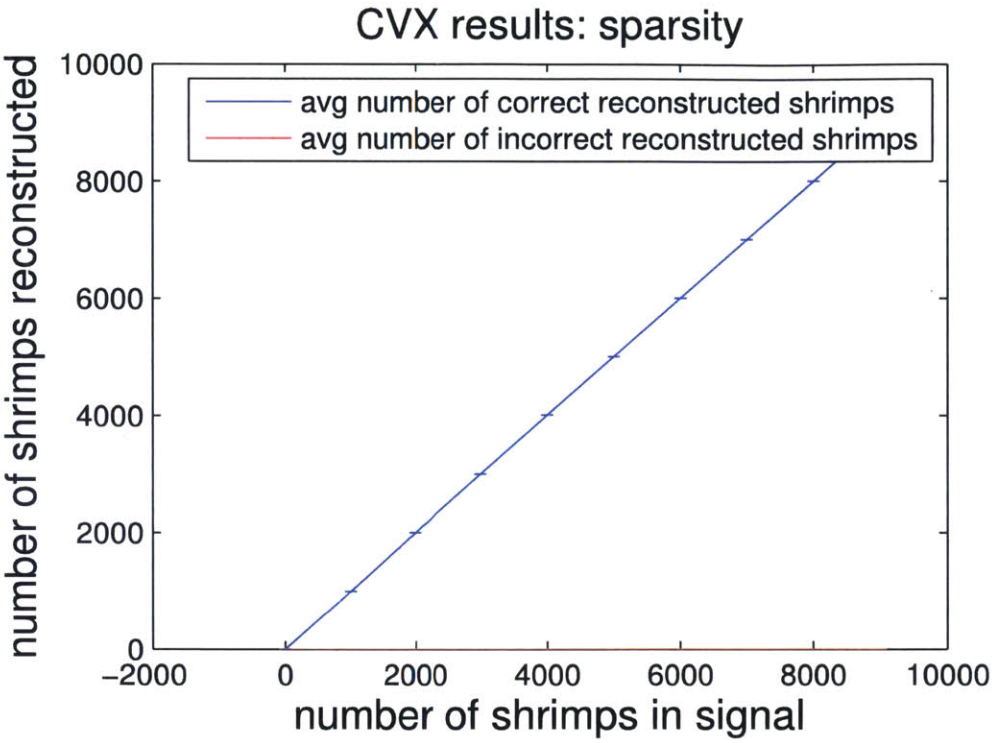

Figure 6-8: CVX performance at different levels of sparsity.

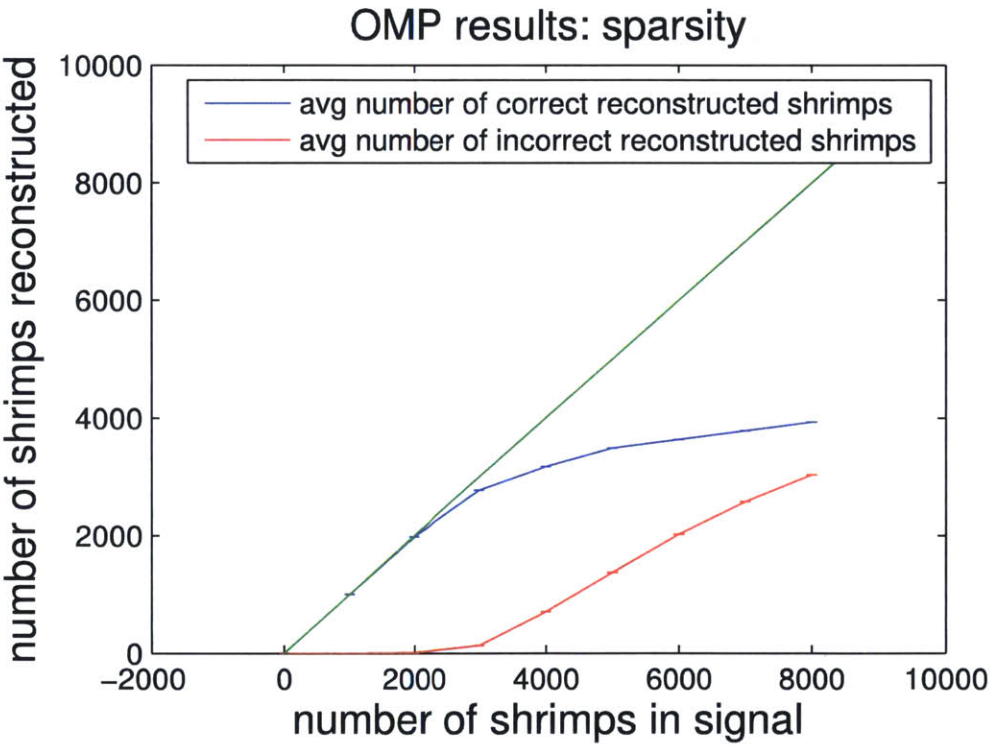

Figure 6-9: OMP performance at different levels of sparsity.

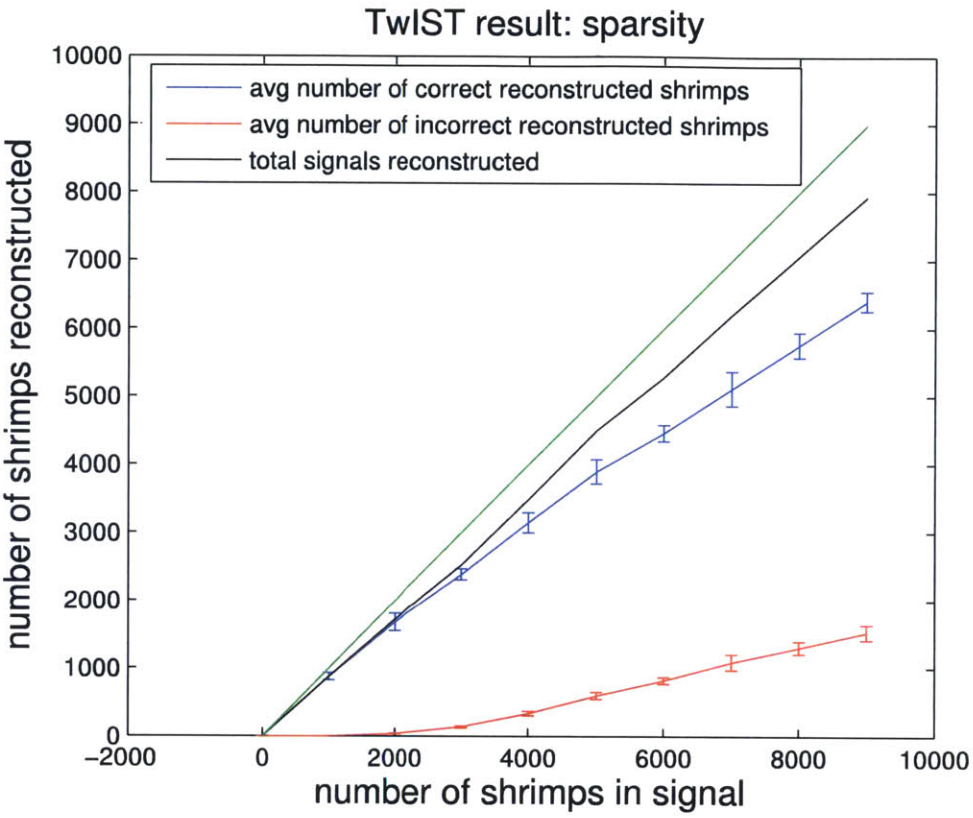

Figure 6-10: TwIST performance at different levels of sparsity.

The results show that CVX has correctly reconstructed all signals regardless of sparsity, while OMP and TwIST reconstruct with some error at low sparsity. OMP reconstructs well up to 7% (3000 shrimps), at which point it starts to fail rather quickly. TwIST performs slightly worst than OMP at 7% and lower, but does a better job at lower sparsity. The total number of reconstructions that TwIST also retrieves is close to the ground truth, without knowing the exact number as compared to OMP.

6.1.3 Robustness to Noise and Sparsity Combination

The previous two subsections show the ideal cases when each factor noise or sparsity is isolated. In reality, both of these factors would be present. Therefore, we would like to see how the methods perform under the presence of both, shown in Figures 6-11, 6-12, and 6-13. Here the x-axis is the sparsity level, and noise levels are represented by lines of different colors. The y-axis here is the percentage of correctness in reconstruction.

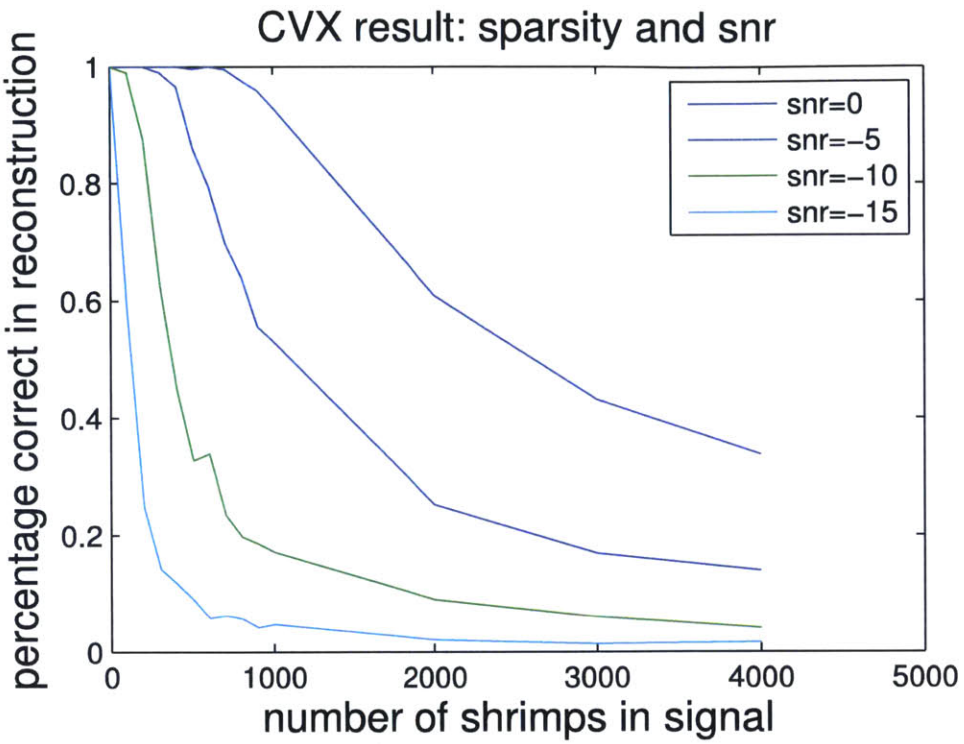

Figure 6-11: Performance of CVX with respect to sparsity and noise.

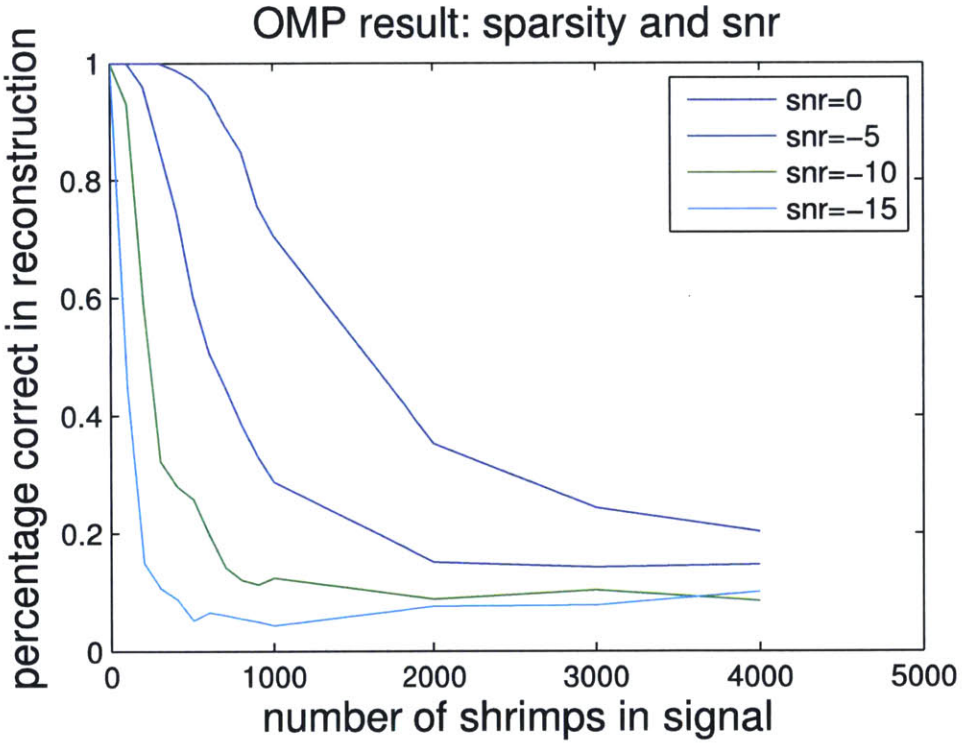

Figure 6-12: Performance of OMP with respect to sparsity and noise.

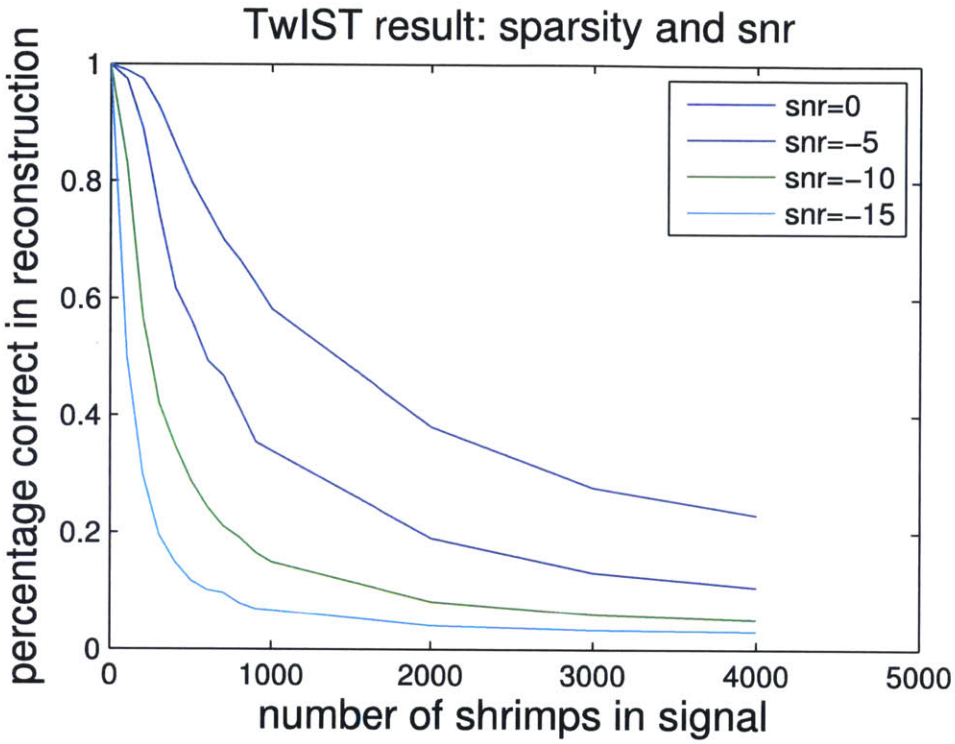

Figure 6-13: Performance of TwIST with respect to sparsity and noise.

The plots show that at low noise level (high SNR), CVX performs the best with vary sparsity, and at high noise level (low SNR), all methods perform similarly.

TwIST, fine-tuning

Before we move on to conclude that CVX is the best out of all algorithms, there is a variable in TwIST that can be optimized but has been fixed in the past. This parameter is λ, the weight factor of the regularization function. λ is used to vary the importance between the regularization function and the closeness of x to the solution of $y = Ax$. It makes sense that as the power of noise approach 0, the true x is equal to the solution to $y = Ax$. On the other hand, as the level of noise increases, more weight should be placed on the regularization function, to emphasize the sparsity so as to, hopefully, beat the noise. This weight affects both noise and sparsity levels.

In this subsection, we want to see how varying the λ changes the performance of TwIST at different sparsity level given a constant amount of noise. In addition, we compare the performance of TwIST using the best λ with previous results of CVX and OMP.

Since the performance between the 3 methods varied the most at $0dB$, λ is optimized at this noise level.

Figure 6-14 shows how TwIST performs at varous λ values. To not just set λ to any arbituary value but rather make it dependent on the measuremnt, we compute it by $\lambda = \tau \max|A^T y|$, where τ is a constant and we use $A^T y$ as an estimate for the initial solution [1].

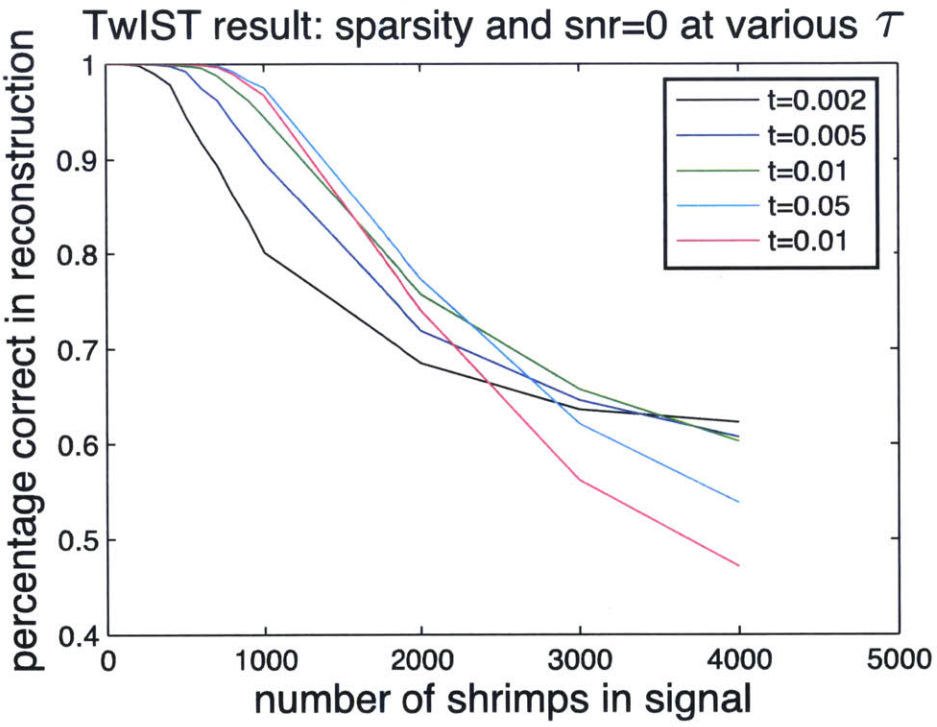

Figure 6-14: TwIST performace at $0dB$ SNR, varing sparisty levels and tau.

The plot shows that a τ value of 0.05 works well at low sparsity levels. This value is chosen because we do expect that there is sparsity in the shrimp population in the ocean. Figure 6-15 shows a comparison of all three algorithms, with TwIST optimized. It is apparent that TwIST outperforms CVX and OMP. Other optimization parameters can be set for OMP, such as the limit on the residual, however, it would affect how well OMP performs under noisy and non-sparse conditions.

[1] This estimation was developed by Yi Liu

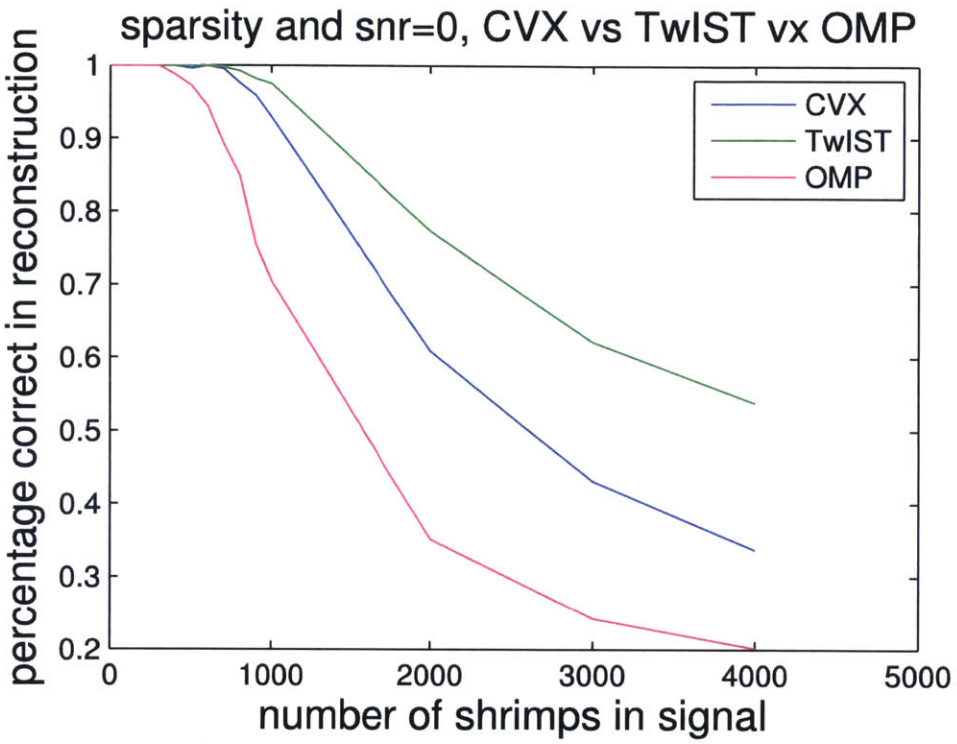

Figure 6-15: Performance of CVX, OMP, TwIST at 0dB SNR, varying sparsity (TwIST optimized).

6.2 Real Data

After examining the methods in simulation and understanding their robustness to noise and sparsity, the three methods were then applied to the real data acquired by ROMANIS. Figure 6-16 shows some real time data. The x-axis is the time in frames, and the y-axis is uncalibrated pressure. A section of this data was taken for evaluation using the three algorithms, it is denoted by the box in the graph.

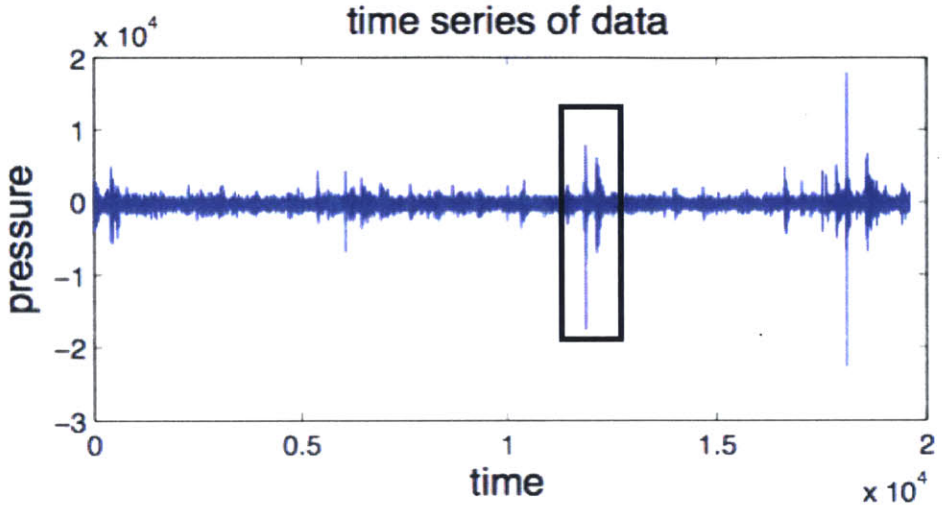

Figure 6-16: Data provided by ROMANIS.

6.2.1 Recontruction on Real Data

For reconstruction, it is assumed that the shrimps are stationary in the short moment, and they are emitting signal at the same instance. CVX was not able to converge on this data. It is possible that part of this non-convergence is due to its requirement on the input of noise. Although different amounts of noise were given as input to try out (they were computed as a function of signal power, at 0.01 times signal power all the way to 100 times), CVX did not converge. This inability to converge has to do with the fact that the noise is unknown. In simulation, CVX was able to converge when the estimated noise level is close to the true noise level. Therefore, even though different levels of noise were tried as inputs, CVX could not converge unless it's very close to true noise, which is unknown.

However, OMP and Twist were able to converge, and produced similar results, shown in Figures 6-16 and 6-18. There are small discrepancies in the magnitude of each signal in the reconstruction, however, the signal locations matched up exactly between the results from OMP and TwIST. Even though we do not have ground truth – it is very difficult to know which shrimp made which signals in the vast ocean, this consistency across two methods is very promising.

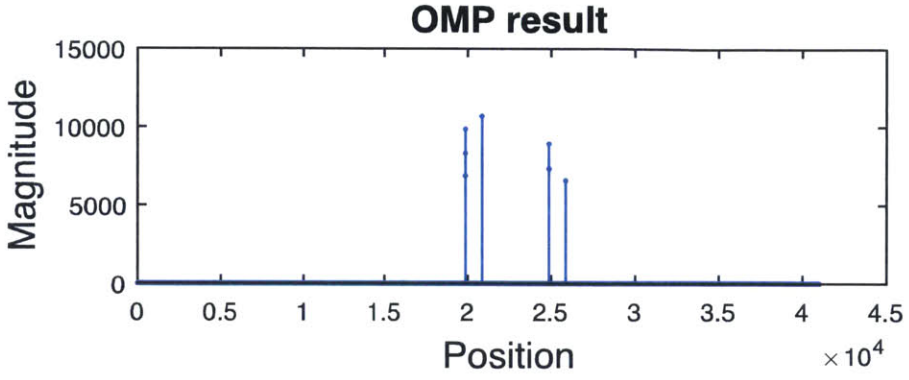

Figure 6-17: Reconstruction results on real data from OMP.

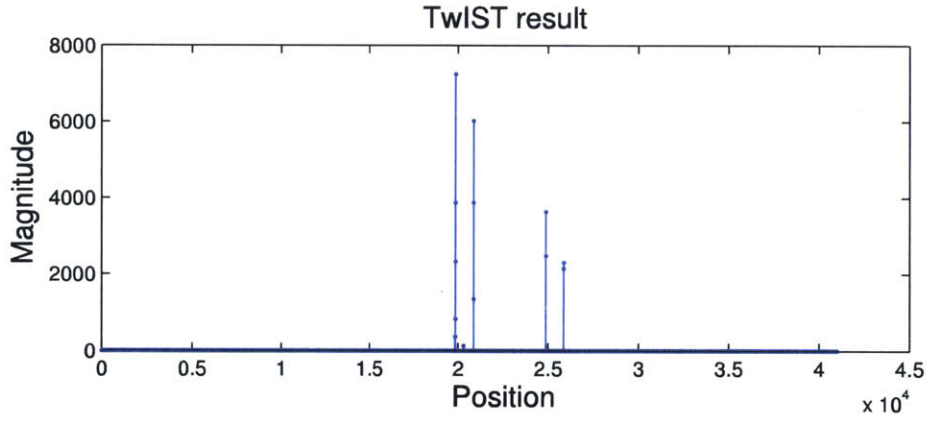

Figure 6-18: Reconstruction results on real data from TwIST.

Since the number of shrimps present is unknown, less sparse solutions are also produced. Figures 6-19 and 6-20 show less sparse results obtained by OMP and TwIST. Once again, the results line up, showing promising consistency.

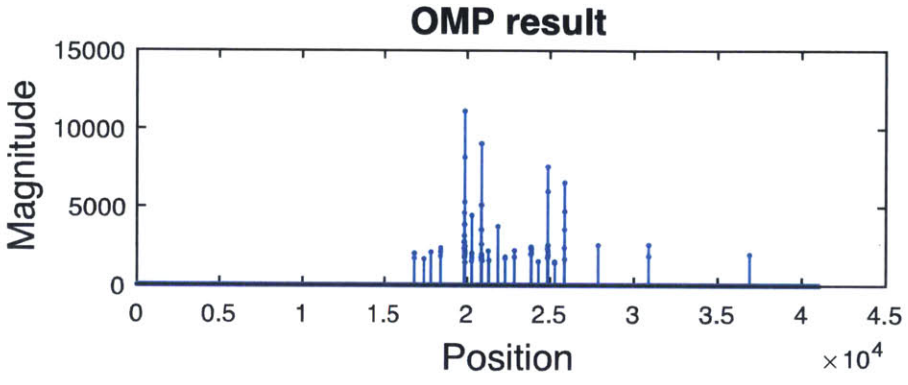

Figure 6-19: Reconstruction results on real data from OMP, less sparse solution.

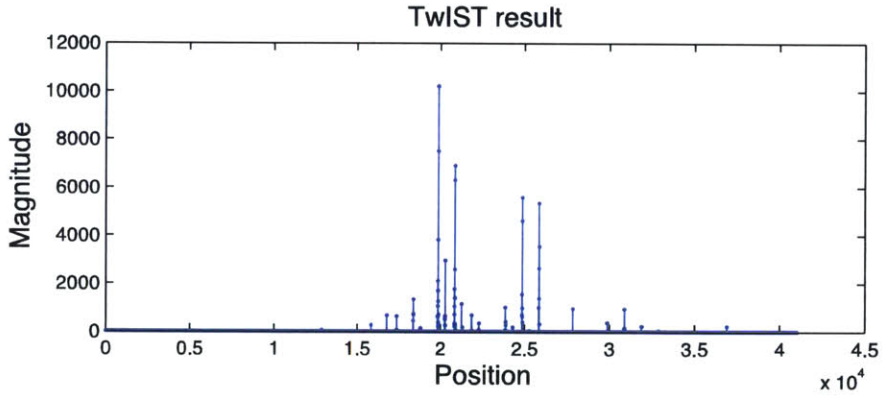

Figure 6-20: Reconstruction results on real data from TwIST, less sparse solution.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

Snapping shrimps living in shallow waters make impulse-like acoustic signals. These signals are a source of noise for underwater acoustic technologis. Since the signals are highly non-Gaussian, they are difficult to remove by linear correlators. At the same time, the signals emitted by the snapping shrimps show isotropic properties, which supports the idea that the signals can be appropriate sources for ambient noise illumination.

This work has developed a theoretical approximation for the snapping signals, and the sensing system which encapusulates the propagation as well. The formulation of this research problem has deem it an undertermined problem. Three common approaches to solving this type of problem are introduced, and an algorithm from each has been picked out for implementation. The algorithms are: CVX, OMP, and TwIST.

The three methods were first implemented in simulation and tested on their robustness to noise, and sparsity. They were then tested on real data set, in which OMP and TwIST showed promising consistency in their results, while CVX was unable to converge. Since there is no available information on ground truth, the consistency is best measure we can get.

7.2 Future Work

There is still space for future work on this topic. The system model can be made to be more accurate – for example, it can take into account of the spreading losses due to the expansion of the spherical source wave, and transmission losses due to the absorption and viscosity of the sea water. A good first order approximation of the attenuation loss is the spherical spreading. In addition, a better system can be created by expanding the ranges of coordinates – have ϕ, θ, and Γ to be the range of what ROMANIS can detect. The frequncy of the snaps is also not constant, and this affects the travel time of the signals, and this is another area to look into for future improvements.

The model can be extended to another dimension: time. Setting up a system matrix that takes into account of the time difference between the snaps of shrimps is more accurate to real life situations. Although at the same time, this requires much more computational power, and should probably be done with parallel computing.

Bibliography

[1] Snapping shrimps. http://www.wildsingapore.com/wildfacts/crustacea/othercrust/shrimp/alpheidae.htm. Accessed: 2016-04-02.

[2] Whitlow WL Au and Kiara Banks. The acoustics of the snapping shrimp synalpheus parneomeris in kaneohe bay. *The Journal of the Acoustical Society of America*, 103(1):41–47, 1998.

[3] José M Bioucas-Dias and Mário AT Figueiredo. A new twist: two-step iterative shrinkage/thresholding algorithms for image restoration. *Image Processing, IEEE Transactions on Image Processing*, 16(12):2992–3004, 2007.

[4] José M Bioucas-Dias and Mário AT Figueiredo. Two-step algorithms for linear inverse problems with non-quadratic regularization. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 1, pages I–105. IEEE, 2007.

[5] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[6] Emmanuel Candes and Justin Romberg. Sparsity and incoherence in compressive sampling. *Inverse problems*, 23(3):969, 2007.

[7] Emmanuel Candes, Mark Rudelson, Terence Tao, and Roman Vershynin. Error correction via linear programming. In *Foundations of Computer Science, 2005. FOCS 2005. 46th Annual IEEE Symposium on*, pages 668–681. IEEE, 2005.

[8] Emmanuel J Candes and Michael B Wakin. An introduction to compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):21–30, 2008.

[9] Mandar Chitre. Acoustic sensing in snapping shrimp dominated environments. In *Proceedings of 20th International Congress on Acoustics, ICA*, 2010.

[10] Mandar A Chitre, John R Potter, and Sim-Heng Ong. Optimal and near-optimal signal detection in snapping shrimp dominated ambient noise. *Oceanic Engineering, IEEE Journal of*, 31(2):497–503, 2006.

[11] Michael Grant, Stephen Boyd, and Yinyu Ye. *Disciplined convex programming*. Springer, 2006.

[12] Michael Grant, Stephen Boyd, and Yinyu Ye. Cvx: Matlab software for disciplined convex programming, 2008.

[13] Michael Grant, Stephen Boyd, and Yinyu Ye. cvx users' guide, 2009.

[14] Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *Signal Processing, IEEE Transactions on*, 41(12):3397–3415, 1993.

[15] Too Yuen Min and Mandar Chitre. Localization of impulsive sources in the ocean using the method of images. In *Oceans-St. John's, 2014*, pages 1–6. IEEE, 2014.

[16] Yagyensh Chandra Pati, Ramin Rezaiifar, and PS Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, pages 40–44. IEEE, 1993.

[17] John R Potter and Teong Beng Koay. Do snapping shrimp chorus in time or cluster in space? temporal-spatial studies of high-frequency ambient noise in singapore waters. In *European Conference on Underwater Acoustics*, pages 0–2, 2000.

[18] Laura Rebollo-Neira and David Lowe. Optimized orthogonal matching pursuit approach. *signal processing Letters, IEEE*, 9(4):137–140, 2002.

[19] P Venugopalan, Mandar A Chitre, Eng Teck Tan, John Potter, Koay Tecing Beng, Sheldon B Ruiz, and Soo Pieng Tan. Ambient noise imaging-first deployments of romanis and preliminary data analysis. In *OCEANS 2003. Proceedings*, volume 2, pages 882–888. IEEE, 2003.

[20] Michel Versluis, Barbara Schmitz, Anna von der Heydt, and Detlef Lohse. How snapping shrimp snap: through cavitating bubbles. *Science*, 289(5487):2114–2117, 2000.

[21] WildSingapore. Smooth snapping shrimp, 2008. snapshrimp.jpg.