University of California

Los Angeles

Applications of Stochastic Simulation and Compressed Sensing to Large Systems

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Mathematics

by

Farzin Barekat

2014

Abstract of the Dissertation

Applications of Stochastic Simulation and Compressed Sensing to Large Systems

by

Farzin Barekat

Doctor of Philosophy in Mathematics

University of California, Los Angeles, 2014

Professor Russel Caflisch, Chair

In this dissertation, three new algorithms for three distinct problems are proposed. The three distinct problems considered here have applications to stochastic modeling, compressive sensing, and numerical solutions of partial differential equations. A common aspect of these problems is that to obtain accurate results require an ever increasing number of unknown variables. Since the proposed algorithms are more efficient than the state of the art methods used for these problems, the use of these new algorithms allows one to compute solutions to these problems with substantially higher accuracy. In addition, some theoretical analysis is provided relating to the investigated problems and the proposed algorithms.

The dissertation of Farzin Barekat is approved.

Chris Anderson

Marek Biskup

Yingnian Wu

Russel Caflisch, Committee Chair

University of California, Los Angeles

2014

To my parents,
Masoud and Sohila, for
their love, care, and sacrifice;
and to my brother, Fardin,
who made my life more beautiful.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I am indebted to Professor Russel Caflisch for his supervision and constant support. I am also indebted to Professor Stanley Osher for guiding and motivating me throughout completion of this dissertation. Without them, none of this dissertation would have been possible. I am extremely thankful of these two professors for revitalizing in me the exuberance of doing mathematics and making the Ph.D. program a superb experience for me.

I would like to thank my other collaborators: Damek Davis, Rongjie Lai, Vidvuds Ozoliņš, and Ke Yin. It's been a real pleasure and a great experience working with them. Some of the work presented here are the result of joint collaboration.

I would also like to thank my committee members Professors Chris Anderson, Marek Biskup, and Yingnian Wu for helpful discussions and useful insights.

Vita

2006	Silver medal in the 47th International Mathematical Olympiad.
2008–2010	NSERC USRA Scholarship for Summer Research.
2010	B.Sc. Combined Honours in Mathematics and Physics, University of British Columbia; Vancouver, Canada.
2010-2011	NSERC Postgraduate Scholarship PGS M.
2010–2013	Teaching Assistant, Mathematics Department, UCLA.
2011–present	NSERC Postgraduate Scholarship PGS D.
2013–2014	Teaching Assistant, Anderson School of Management, UCLA.
2012–present	Research Assistant, Mathematics Department, UCLA.
2013	PhD Intern, PIMCO.

Publications

F. Barekat, V. Reiner, S. van Willigenburg. *Skew Schur function irreducibility, note to go with "Coincidences among skew Schur functions"*, Adv. Math., 220 (2009), pp. 1655–1656.

F. Barekat, S. van Willigenburg. *Composition of transpositions and equality of*

ribbon Schur Q-functions, The Electronic Journal of Combinatorics, 16 (2009), pp. 1–28.

R.P. Anstee, F. Barekat, A. Sali. *Small Forbidden Configurations V: Exact Bounds for 4x2 cases*, Studia Sci. Math. Hun., 48 (2011), pp. 1–22.

F. Barekat, R. Caflisch. *Simulation with Fluctuating and Singular Rates*, Communications in Computational Physics, to appear.

R.P. Anstee, F. Barekat. *Design Theory and Some Non-simple Forbidden Configurations*, submitted to Journal of Combinatorial Designs, 27pp.

F. Barekat. *On the Consistency of Compressed Modes for Variational Problems*, submitted to SIAM Journal of Math. Analysis.

F. Barekat, R. Caflisch, and S. Osher, *On the Support of Compressed Modes*, submitted to SIAM Journal of Math. Analysis.

F. Barekat and S. Osher, *A time continuation based fast approximate algorithm for compressed sensing related optimization*, submitted to Mathematics of Computation.

CHAPTER 1

Introduction

An important aspect of scientific computing is the use of discrete approximation of equations associated with mathematical models. A general feature of discrete approximations is that the accuracy obtained is directly related to the number of unknowns used in the approximations. For many applications computing solutions to the discrete equations as the number of unknowns increases introduces significant numerical and computational challenges. There are problems for which available algorithms perform satisfactorily for a moderate number of unknowns; however, these algorithms become very inefficient and practically useless when the size of the problems increases. As a result, such problems remain unsolved when high accuracy is sought. This phenomena has made the pursuit for faster and more efficient algorithms a hot topic of research in science.

The general theme of this dissertation is to devise new algorithms and methodologies for three different problems, in very distinct settings, that are more efficient than the state of the art methods used for these problems. The three distinct problems considered here have applications to stochastic modeling, compressive sensing, and numerical solutions of partial differential equations.

Chapter 2 presents a method to generate independent samples for a general random variable, either continuous or discrete. The algorithm is an extension of the Acceptance Rejection method, and it is particularly useful for kinetic simulation in which the rates are dynamically updated in time and have singular limits, as occurs for example in simulation of recombination interactions in a plasma.

Although it depends on some additional requirements, the new method is easy to implement and rejects less samples than the Acceptance Rejection method.

Chapter 3 introduces a fast approximate algorithm to optimize an augmented version of the Basis Pursuit problem and subsequently find the solution to the Compressed Sensing problem. The methodology is to first solve the Lagrangian dual formulation of the problem and then use the result to find an approximate solution to the primal problem. Although we emphasize that the proposed algorithm finds an approximate solution, numerical experiments show that the algorithm perfectly recovers the solution when the solution is relatively sparse with respect to the number of measurements. In these scenarios, the recovery is extremely fast compared to other available methods. Numerical experiments also demonstrate that the algorithm exhibits a sharp phase transition in success rate of recovery of the solution to Compressed Sensing problems as sparsity of solution varies. The algorithm proposed here is parameter free (except a tolerance parameter due to numerical machine precision), and very easy to implement.

In Chapter 4 some theoretical result for Compressed Modes (CMs) are presented. Compressed modes are solutions of the Laplace equation with a potential and a subgradient term. The subgradient term comes from addition of an L^1 term in the corresponding variational principle. This chapter contains two analyses of compressed modes. First, we provide theoretical consistency results for compressed modes. It is proven that as the L^1 regularization term in certain non-convex variational optimization problems vanishes, the solution of the optimization problem and the corresponding eigenvalues converge to a unitary transformation of eigenfunctions and the eigenvalues of the Hamiltonian, respectively. Second, we establish that compressed modes have compact support and find an upper bound on the volume of the support.

Chapter 5 presents a fast algorithm for projecting a given function to the set of shift orthogonal functions (i.e. the set containing functions with unit L^2 norm

that are orthogonal to their prescribed shifts). The algorithm can be parallelized easily and its computational complexity is bounded by $O(M \log(M))$, where M is the number of coefficients used for storing the input. To derive the algorithm, a particular class of basis functions called Shift Orthogonal Basis Functions are introduced and some theory regarding them is developed.

CHAPTER 2

Simulation with Fluctuating and Singular Rates

Kinetic transport for a gas or plasma involves particle interactions such as collisions, excitation/deexcitation and ionization/recombination. Simulation of these interactions is most often performed using the Direct Simulation Monte Carlo (DSMC) method [5] or one of its variants, in which the actual particle distribution is represented by a relatively small number of numerical particles, each of which is characterized by state variables, such as position x and energy E. Interactions between the numerical particles are performed by random selection of the interacting particles and the interaction parameters, depending on the interaction rates. Correctly sampling these interactions involves several computational challenges: First the number N of particles can be large (e.g., $N = 10^6$) and the number of possible interaction events can be even larger (e.g., N^k for $k = 2$ or 3). Second, the interaction probabilities vary throughout the simulation since interactions change the state of the interacting particles. These two difficulties are routinely overcome using Acceptance Rejection sampling. Third, the interaction rates can be nearly singular, for example in a recombination event between an ion and two electrons (described in more detail in Section 2.4). This creates a wide range of interaction rates that makes Acceptance Rejection computationally intractable. Figure 2.1 illustrates these challenges and how different methods can handle them. The sampling method presented here, which we call Reduced Rejection, was developed to overcome the challenges of a large number of interaction events with fluctuating and singular rates.

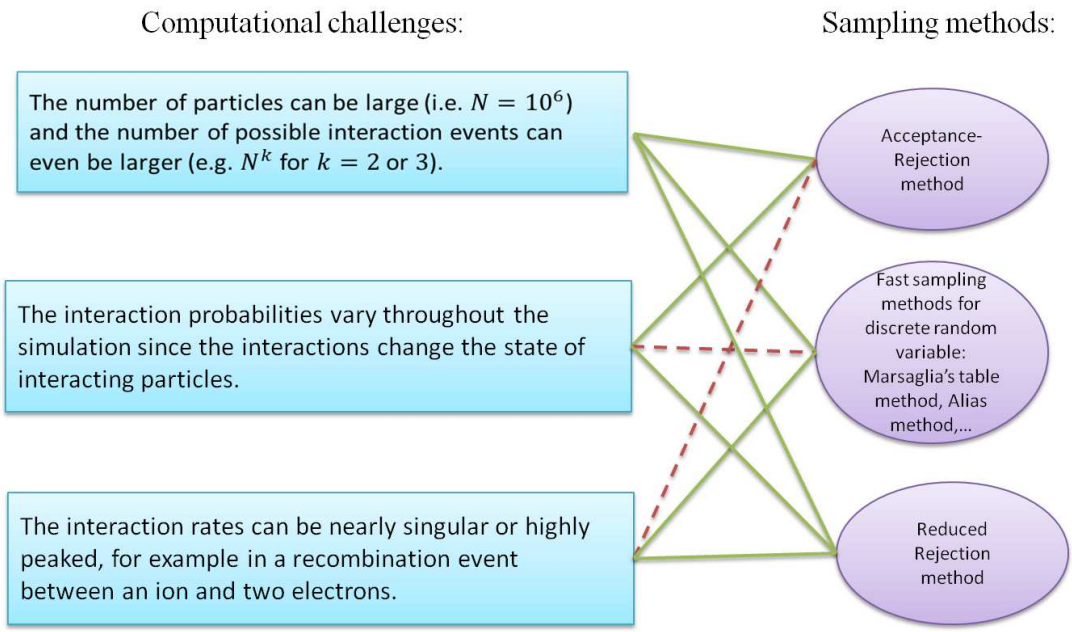

Figure 2.1: This figure illustrates the computational challenges involved in sampling interactions of numerical particles, and how different methods can handle them. Broken line represents challenges for which the method becomes computationally inefficient, whereas, the solid line represents the challenges for which the method is still computationally efficient.

Simulation of kinetics requires sampling methods that generate independent samples. This rules out Markov Chain Monte Carlo schemes, such as Metropolis–Hastings, Gibbs sampling, and Slice sampling. Although these methods are very powerful and are used very often, this chapter focuses on sampling methods that generate independent samples.

There are several efficient algorithms for simulation of discrete random variables, notably Marsaglia's table method [41] and the Alias method [62, 63]. However, these methods require pre-processing time and, therefore, are not efficient for sampling from a random variable whose probability function changes during the simulation. For continuous random variables there are several different algorithms; nevertheless, each of these algorithms has its own constraints. For example, Inverse Transform Sampling method requires knowledge of the cumulative distribution function and evaluation of its inverse, Box-Muller only applies to a normal distribution, and Ziggurat algorithm [42] can be used for random variables that have monotone decreasing (or symmetric unimodal) density function.

An algorithm of choice for general (both continuous and discrete) random variables that generates independent samples and does not require preprocessing time is Acceptance Rejection method (see for example [11]). Let $q(x)$ be a real-valued function on the sample space. Let $I[q]$ denote the expectation of function $q(x)$ with respect to some given measure. By sampling according to function $q(x)$ we mean to sample using the probability distribution function $q(x)/I[q]$. We say function $q(x)$ *encloses* function $p(x)$ if $p(x) \leq q(x)$ for all x in the sample space. The idea of Acceptance Rejection method is to find a proposal function $q(x)$ that encloses function $p(x)$. Suppose we already have a mechanism to sample according to $q(x)$, then Acceptance Rejection algorithm enables us to sample according to $p(x)$. In most cases the constant function is used as the proposal function $q(x)$. The main drawback of Acceptance Rejection method is that it might reject many samples. Indeed the ratio of the number of rejected samples to the number of

accepted samples is approximately equal to the ratio of the area between curves $q(x)$ and $p(x)$ to the area under the curve $p(x)$.

For many given distributions, finding a good proposal function that encloses it without leading to many rejected samples is difficult. One extension to Acceptance Rejection method is Adaptive Rejection Sampling [30]. The basic idea of Adaptive Rejection Sampling is to construct proposal function $q(x)$ that encloses the given distribution by concatenating segments of one or more exponential distributions. As the algorithm proceeds, it successively updates the proposal function $q(x)$ to correspond more closely to the given distribution. Another extension to Acceptance Rejection method is the Economical method [23]. This method is basically a generalization of Alias method for continuous distributions. In this method, one needs to define a specific transformation that maps $\{x : p(x) > q(x)\}$ to $\{x : p(x) \leq q(x)\}$. Although this method produces no rejection, finding the required transformation is difficult in general.

In the Reduced Rejection method we sample according to a given function $p(x)$ based on a proposal function $q(x)$. In contrast to the Acceptance Rejection method, Reduced Rejection sampling does not require $q(x)$ to enclose $p(x)$ (i.e. it allows $p(x) > q(x)$ for some x). On the other hand, Reduced Rejection sampling requires some extra knowledge about the functions $p(x)$ and $q(x)$.

The Reduced Rejection sampling method can be applied to a wide range of sampling problems (for both continuous and discrete random variables) and in many examples is more efficient than customary methods (three examples are provided in Sections 2.3, 2.4 and 2.5). In particular, Reduced Rejection sampling requires no pre-processing time and consequently is suitable for simulations in which $p(x)$ is changing constantly (see Section 2.2 for an elaboration on this point and Sections 2.4 and 2.5 for examples of simulations with fluctuating $p(x)$). Also in situations where $p(x)$ has singularities or is highly peaked in certain regions, Reduced Rejection sampling can be very efficient.

The next section describes the Reduced Rejection sampling and proves its validity. Section 2.2 compares Reduced Rejection sampling to other methods (including other generalizations of Acceptance Rejection), highlights advantages of Reduced Rejection sampling in comparison to other methods, and points out some of the challenges in applying Reduced Rejection sampling. In Section 2.3, Reduced Rejection sampling is demonstrated on a simple example. In Section 2.4, Reduced Rejection sampling is applied to an example motivated from plasma physics, for which other sampling methods cannot be used efficiently. In Section 2.5, we make some comments on how to apply Reduced Rejection in the context of stochastic chemical kinetics.

2.1 Reduced Rejection Sampling

Consider a sample space Ω with Lebesgue measure μ on Ω, and two functions $q, p : \Omega \to \mathbb{R}$. Denote

$$I[q] = \int_\Omega q(x)d\mu(x), \qquad I[p] = \int_\Omega p(x)d\mu(x).$$

By sampling from Ω according to $p(x)$ we mean sampling from Ω using probability distribution function $p(x)/I[p]$. Partition sample space Ω into two sets $\mathcal{S}$ and $\mathcal{L}$:

$$\mathcal{L} = \{x \in \Omega : p(x) > q(x)\}, \qquad \mathcal{S} = \{x \in \Omega : p(x) \leq q(x)\}.$$

Reduced Rejection sampling is a method for sampling from Ω according to $p(x)$ using an auxiliary function $q(x)$. It depends on the following:

- The values of $I[q]$, $I[p]$ and $\int_\mathcal{L}(p(x) - q(x))d\mu(x)$. Note that the last value is needed only for "Algorithm II", see Subsection 2.1.1.
- A mechanism to sample from Ω according to $q(x)$.
- A mechanism to sample from $\mathcal{L}$ according to $p(x) - q(x)$.

Whereas the Acceptance Rejection method for sampling from $p(x)$ requires a function $q(x)$ that encloses $p(x)$ (i.e., $0 \leq p(x) \leq q(x)$ for all $x \in \Omega$), the Reduced Rejection sampling algorithm is a generalization of the Acceptance Rejection method, that allows $p(x) > q(x)$ for some x. The Reduced Rejection sampling algorithm is detailed in Section 2.1.1, and its validity as a method for sampling from Ω according to $p(x)$ is demonstrated in Section 2.1.2.

2.1.1 The Reduced Rejection sampling algorithm

The Reduced Rejection sampling method consists of two algorithms (i.e., two different algorithms) depending on the relative values of $I[p]$ and $I[q]$. Flow charts for these two algorithms are presented in Figures 2.2 and 2.3. The outcome of each algorithm is a value z that is an independent sample from Ω according to $p(x)$.

Algorithm I: $\mathbf{I[p] \geq I[q]}$.

Proceed as follows:

i) With probability $(I[p]-I[q])/I[p]$, sample x_0 from $\mathcal{L}$ according to $p(x)-q(x)$ and accept $z = x_0$.

ii) Otherwise (with probability $(I[q]/I[p])$, sample x_0 from Ω according to $q(x)$.

   a) If $x_0 \in \mathcal{L}$, accept $z = x_0$.

   b) If $x_0 \in \mathcal{S}$, accept $z = x_0$ with probability $p(x_0)/q(x_0)$.

iii) If x_0 was not accepted, then sample a new value of x_1 from $\mathcal{L}$ according to $p(x) - q(x)$ and accept $z = x_1$.

Algorithm II: $\mathbf{I[p] < I[q]}$.

Proceed as follows until a value z is accepted:

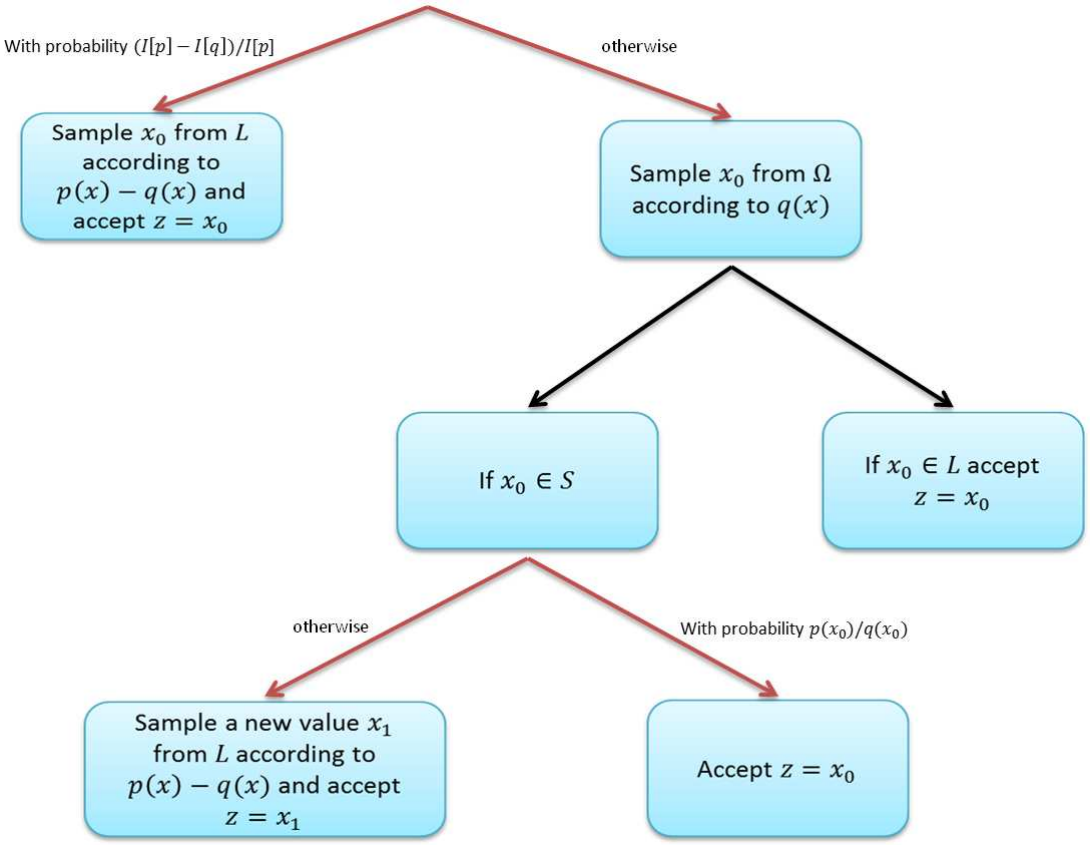

Figure 2.2: Flow chart of Algorithm I of the Reduced Rejection sampling method attributing to the case $I[p] \geq I[q]$.

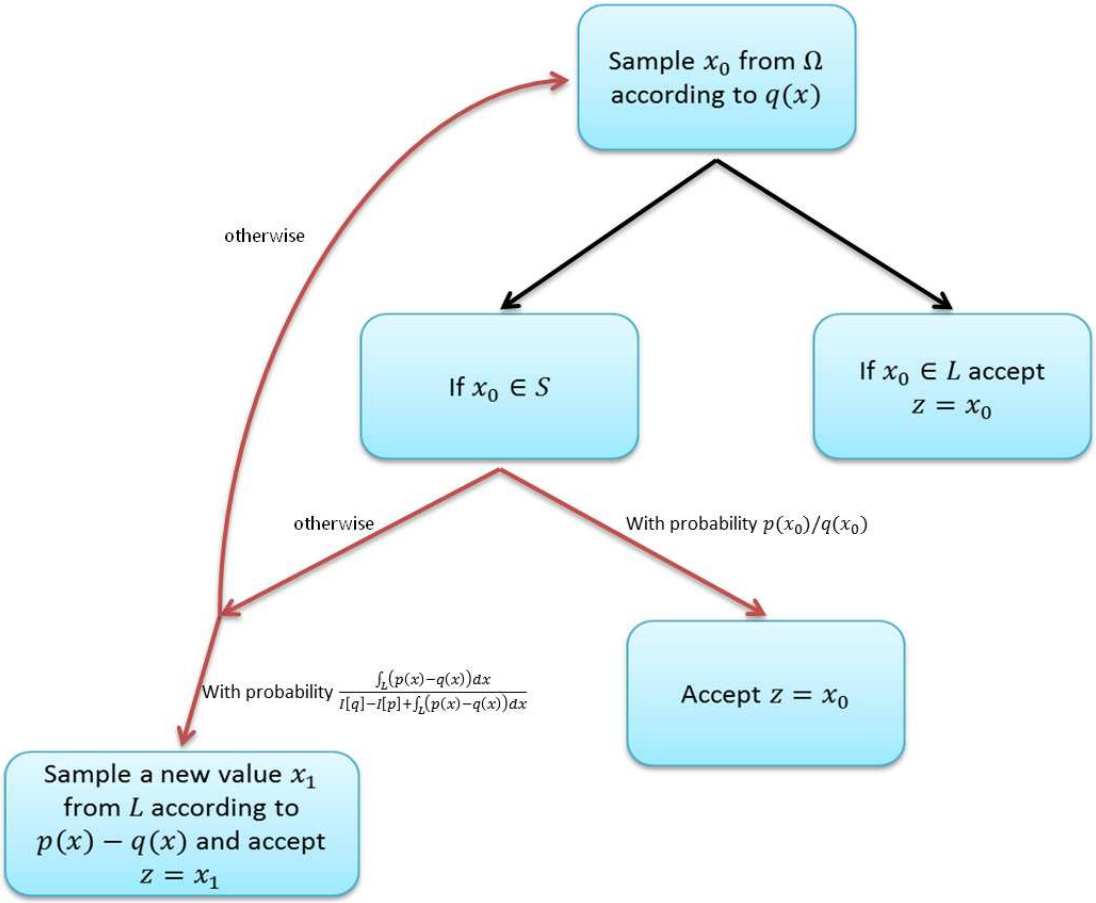

Figure 2.3: Flow chart of Algorithm II of the Reduced Rejection sampling method attributing to the case $I[p] < I[q]$.

i) Sample x_0 from Ω according to $q(x)$.

ii) If $x_0 \in \mathcal{L}$, accept $z = x_0$.

iii) If $x_0 \in \mathcal{S}$, accept $z = x_0$ with probability $p_a = p(x_0)/q(x_0)$,

iv) If x_0 was not accepted, then

a) With probability p_a select x_1 from $\mathcal{L}$ according to $p(x)-q(x)$ and accept $z = x_1$, in which

$$\begin{aligned} p_a &= \frac{\int_{\mathcal{L}}(p(x) - q(x))d\mu(x)}{\int_{\mathcal{S}}(q(x) - p(x))d\mu(x)} \\ &= \frac{\int_{\mathcal{L}}(p(x) - q(x))d\mu(x)}{I[q] - I[p] + \int_{\mathcal{L}}(p(x) - q(x))d\mu(x)}. \end{aligned} \tag{2.1}$$

b) Otherwise (i.e., with probability $1-p_a$), return to (i) without accepting a value of z.

As described in Algorithms I and II, Reduced Rejection samples from p through the following steps: On $\mathcal{L}$, treat p as a mixture $p = q + (p - q)$ and sample from q and $p - q$ with the correct probabilities; and on $\mathcal{S}$, sample from p by sampling from q and accepting the sample with probability p/q. Rejected samples in $\mathcal{S}$ correspond to the region B in Figure 2.4, and the region A is where q does not enclose p. If $|A| > |B|$ (i.e., Algorithm I) then all of the rejected samples can be replaced by samples from A; if $|A| < |B|$ (i.e., Algorithm II) then a portion of the rejected samples can be replaced by samples from A, and for the remainder, the algorithm is repeated as in Acceptance Rejection.

2.1.2 Validity of the Reduced Rejection sampling

In this subsection we show the correctness of the Reduced Rejection sampling method. As the method is different for Algorithms I and II, we prove the correctness for each algorithm separately.

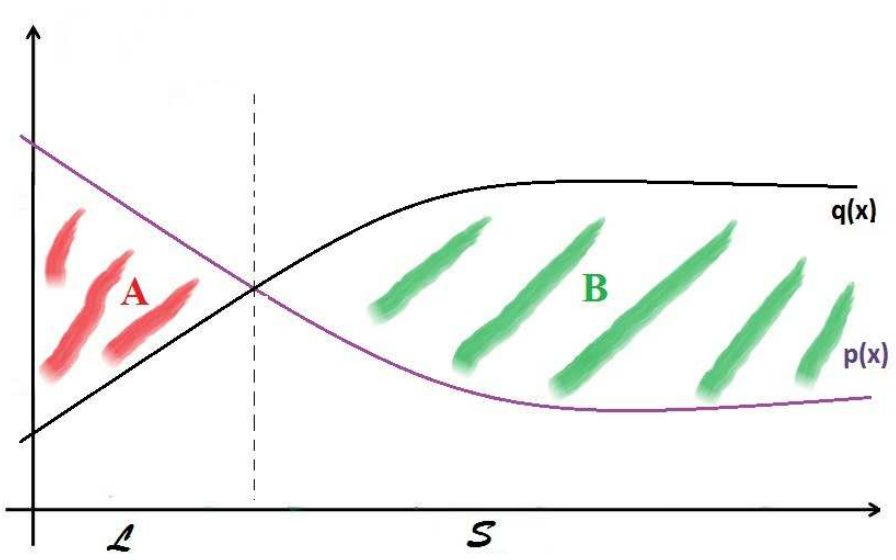

Figure 2.4: This figure illustrates the Reduced Rejection method. Region A is where q does not enclose p, and region B is where samples are rejected. Rejected samples from region B can be replaced by samples from region A if $|A| > |B|$; otherwise (if $|A| < |B|$), some of the rejected samples lead to repetition of the algorithm.

Proof for Algorithm I: For each $z \in \Omega$, show that the algorithm of Algorithm I returns in dz with probability distribution $p(z)d\mu(z)/I[p]$.

If $z \in \mathcal{S}$, then part (ii) must have been selected, z must have been sampled in (ii) and it must have been accepted in case (ii.b). Therefore, the probability of returning z is

$$\begin{aligned} & \Pr[\text{(ii) selected}]\ \Pr[z \text{ sampled in (ii)}]\ \Pr[z \text{ accepted in (ii.b)}] \\ = & \frac{I[q]}{I[p]} \times \frac{q(z)d\mu(z)}{I[q]} \times \frac{p(z)}{q(z)} \\ = & \frac{p(z)d\mu(z)}{I[p]}. \end{aligned} \tag{2.2}$$

Also note that for every $x_0 \in \mathcal{S}$, after x_0 is selected in (ii.b) with probability $\frac{q(x)}{I[q]}d\mu(x)$, the probability of reaching (iii) is $\frac{q(x_0)-p(x_0)}{q(x_0)}$. Thus the total probability of reaching (iii) after selecting (ii) is

$$\Pr[\text{reaching (iii)}|\text{(ii) selected}] = \int_{\mathcal{S}} \frac{q(x)-p(x)}{q(x)} \frac{q(x)}{I[q]} d\mu(x) = \frac{\int_{\mathcal{S}}(q(x)-p(x))d\mu(x)}{I[q]}. \tag{2.3}$$

Next suppose that $z \in \mathcal{L}$. The probability that z is returned from (i) is

$$\begin{aligned} \Pr[z \text{ returned from (i)}] = & \Pr[\text{(i) selected}]\ \Pr[z \text{ sampled in (i)}] \\ = & \frac{(I[p]-I[q])}{I[p]} \times \frac{(p(z)-q(z))d\mu(z)}{\int_{\mathcal{L}}(p(x)-q(x))d\mu(x)}. \end{aligned} \tag{2.4}$$

The probability that z was returned from (ii.a) is

$$\begin{aligned} \Pr[z \text{ returned from (ii.a)}] = & \Pr[\text{(ii) selected}]\ \Pr[z \text{ sampled in (ii.a)}] \\ = & \frac{I[q]}{I[p]} \times \frac{q(z)d\mu(z)}{I[q]} \\ = & \frac{q(z)d\mu(z)}{I[p]}. \end{aligned} \tag{2.5}$$

Also, using equation (2.3), the probability that z was returned from (iii) is

$$\begin{aligned}
&\Pr[z \text{ returned from (iii)}] \\
&\quad = \Pr[\text{(ii) selected}]\ \Pr[\text{reaching (iii)} \mid \text{(ii) selected}]\ \Pr[z \text{ sampled from } \mathcal{L} \text{ in (iii)}] \\
&\quad = \frac{I[q]}{I[p]} \times \left(\frac{\int_{\mathcal{S}} (q(x) - p(x)) d\mu(x)}{I[q]} \right) \times \frac{(p(z) - q(z)) d\mu(z)}{\int_{\mathcal{L}} (p(x) - q(x)) d\mu(x)} \\
&\quad = \frac{\int_{\mathcal{S}} (q(x) - p(x)) d\mu(x)}{I[p]} \frac{(p(z) - q(z)) d\mu(z)}{\int_{\mathcal{L}} (p(x) - q(x)) d\mu(x)}. \qquad (2.6)
\end{aligned}$$

Finally, using equations (2.4), (2.5), and (2.6), the probability of returning z is

$$\begin{aligned}
&\Pr[z \text{ returned from (i)}] + \Pr[z \text{ returned from (ii.a)}] + \Pr[z \text{ returned from (iii)}] \\
&\quad = \frac{(I[p] - I[q])}{I[p]} \frac{(p(z) - q(z)) d\mu(z)}{\int_{\mathcal{L}} (p(x) - q(x)) d\mu(x)} + \frac{q(z) d\mu(z)}{I[p]} + \\
&\qquad\qquad + \frac{(p(z) - q(z)) d\mu(z)}{\int_{\mathcal{L}} (p(x) - q(x)) d\mu(x)} \frac{\int_{\mathcal{S}} (q(x) - p(x)) d\mu(x)}{I[p]} \\
&\quad = \frac{(p(z) - q(z)) d\mu(z)}{I[p] \int_{\mathcal{L}} (p(x) - q(x)) d\mu(x)} \left(I[p] - I[q] + \int_{\mathcal{S}} (q(x) - p(x)) d\mu(x) \right) + \frac{q(z) d\mu(z)}{I[p]} \\
&\quad = \frac{(p(z) - q(z)) d\mu(z)}{I[p] \int_{\mathcal{L}} (p(x) - q(x)) d\mu(x)} \left(\int_{\mathcal{L}} (p(x) - q(x)) d\mu(x) \right) + \frac{q(z) d\mu(z)}{I[p]} \\
&\quad = \frac{p(z) d\mu(z)}{I[p]}. \qquad (2.7)
\end{aligned}$$

Hence, by (2.2) and (2.7), whether $z \in \mathcal{S}$ or $z \in \mathcal{L}$, the probability of returning z is equal to $p(z)d\mu(z)/I[p]$. This completes the proof for Algorithm I. ∎

Proof for Algorithm II: For each $z \in \Omega$, show that the algorithm in Algorithm II returns in dz with probability distribution $p(z)d\mu(z)/I[p]$. The algorithm consists of some number of cycles, each consisting of steps (i)-(iv), until a value z is accepted. We first calculate the probability that z is accepted within one of the cycles.

Suppose that $z \in \mathcal{S}$. Then z must be sampled in (i) and accepted in (iii).

Thus, the probability of returning z in (iii) is

$$\begin{aligned}\Pr[z \text{ returned from (iii)}] =& \Pr[z \text{ sampled in (i)}] \Pr[z \text{ accepted in (iii)}] \\ =& \frac{q(z)d\mu(z)}{I[q]} \times \frac{p(z)}{q(z)} \\ =& \frac{p(z)d\mu(z)}{I[q]}. \end{aligned} \tag{2.8}$$

Also note that for every $x_0 \in \mathcal{S}$,which is chosen with probability $\frac{q(x_0)d\mu(x_0)}{I[q]}$, the probability that it is not accepted in (iii) is $\frac{q(x_0)-p(x_0)}{q(x_0)}$. Thus the total probability of not returning an element of $\mathcal{S}$ in (iii), which is the same as the probability of reaching (iv), is

$$\Pr[\text{reaching (iv)}] = \int_{\mathcal{S}} \frac{q(x)-p(x)}{q(x)} \frac{q(x)}{I[q]} d\mu(x) = \int_{\mathcal{S}} \frac{q(x)-p(x)}{I[q]} d\mu(x). \tag{2.9}$$

Next suppose that $z \in \mathcal{L}$. The probability that z is accepted in (ii) is

$$\Pr[z \text{ returned from (ii)}] = \frac{q(z)d\mu(z)}{I[q]}. \tag{2.10}$$

For z to be returned from (iv.a), the algorithm must reach (iv), then go to (iv.a) and then select z in (iv.a). This has probability

$$\begin{aligned}&\Pr[z \text{ returned from (iv.a)}] \\ &=\Pr[\text{reach (iv)}] \Pr[\text{go to (iv.a)}] \Pr[\text{z sampled in (iv.a)}] \\ &= \left(\int_{\mathcal{S}} \frac{q(x)-p(x)}{I[q]} d\mu(x) \right) \times \frac{\int_{\mathcal{L}} (p(x)-q(x)) d\mu(x)}{\int_{\mathcal{S}} (q(x)-p(x)) d\mu(x)} \times \frac{(p(z)-q(z))d\mu(z)}{\int_{\mathcal{L}} (p(x)-q(x)) d\mu(x)} \\ &= \frac{(p(z)-q(z))d\mu(z)}{I[q]}. \end{aligned} \tag{2.11}$$

Now using equations (2.10) and (2.11), the probability of returning z in a cycle is

$$\begin{aligned}\Pr[z \text{ returned}] =& \Pr[z \text{ returned from (ii)}] + \Pr[z \text{ returned from (iv.a)}] \\ =& \frac{q(z)d\mu(z)}{I[q]} + \frac{(p(z)-q(z))d\mu(z)}{I[q]} \\ =& \frac{p(z)d\mu(z)}{I[q]}. \end{aligned} \tag{2.12}$$

Equations (2.8) and (2.12) imply that, whether $z \in \mathcal{S}$ or $z \in \mathcal{L}$, the probability that z is returned in a cycle is $p(z)d\mu(z)/I[q]$. Integrating over all the samples in Ω, we deduce that the probability that a sample is returned in a cycle is $I[p]/I[q]$. Consequently, probability that no sample point is returned in a cycle is $1 - I[p]/I[q]$. Because the cycle is repeated until a sample point is returned, we conclude that the probability that the algorithm returns z is equal to

$$\sum_{k=1}^{\infty} \left(1 - \frac{I[p]}{I[q]}\right)^{k-1} \frac{p(z)d\mu(z)}{I[q]} = \frac{p(z)d\mu(z)}{I[p]}.$$

This completes the proof for Algorithm II. ∎

Note that the efficiency of Algorithm II is nominally the same as acceptance rejection, i.e. the probability of a rejection is $1 - I[p]/I[q]$. Actually it can be significantly better because $I[q]$ can be smaller, since $q < p$ is allowed. Also, note that if $I[p] = I[q]$, then Algorithms I and II are the same.

2.2 Comparison of Reduced Rejection and Other Sampling Methods

One of the important features of Reduced Rejection sampling is that it requires no preprocessing time. This is particularly useful for *dynamic simulation*; i.e., simulation in which the probability distribution function $p(x)$ may change after each sample (see Section 2.4 for an example from plasma physics). For dynamic simulation, fast discrete sampling methods such as Marsaglia's table method or the Alias method, are not suitable as they require preprocessing time after each change in $p(x)$. Although, the Acceptance Rejection method requires no preprocessing time and can be used for dynamic simulation, it may require changes in $q(x)$ if $p(x)$ changes, which is usually not difficult, and it becomes very inefficient when the ratio of the area under function $p(x)$ to the area under proposal function $q(x)$ is small. Moreover, adaptive rejection sampling is not efficient, because the

process of adapting $q(x)$ to $p(x)$ starts over whenever $p(x)$ changes.

The Reduced Rejection sampling method can be thought of as an extension of the Acceptance Rejection method. In particular when the proposal function $q(x)$ encloses $p(x)$ (i.e., $q(x) \geq p(x)$ for all $x \in \Omega$ so that $\mathcal{L} = \emptyset$) the Reduced Rejection sampling method reduces to Acceptance Rejection method. The advantage of Reduced Rejection sampling over Acceptance Rejection method is that the proposal function $q(x)$ does not need to enclose function $p(x)$; i.e., it allows $q(x) < p(x)$ for some x. This is very useful in dynamic simulation as it can accommodate changes in $p(x)$ without requiring changes in $q(x)$. Moreover, Reduced Rejection sampling may result in fewer rejected samples than Acceptance Rejection does, especially if $p(x)$ has singularities or is highly peaked.

There are several challenges in implementing the Reduced Rejection sampling method. The main challenge is the need to sample from set Ω according to $q(x)$ and from set $\mathcal{L}$ according to $p(x) - q(x)$, which can be performed by various sampling methods.

Another challenge in using Reduced Rejection sampling is the need to know the values of $I[q]$, $I[p]$ and $\int_{\mathcal{L}}(p(x) - q(x))d\mu(x)$ (but note that the last value is only for Algorithm II). In many situations, these values are readily available or can be calculated during the simulation.

2.3 Example 1: Reduced Rejection Sampling for a Random Variable with Singular Density

In this section, Reduced Rejection sampling method is applied to a simple problem. Let $\Omega = (0, 1)$ and sample according to

$$p(x) = \frac{1}{\sqrt{x}} + \frac{1}{\sqrt[5]{1-x}} \tag{2.13}$$

which has singularities at 0 and 1. Using inverse transform sampling, it is easy to sample according to $1/\sqrt{x}$ or $1/\sqrt[5]{1-x}$, but inverse transform cannot be easily applied to (2.13) as it requires finding the root of an eighth degree polynomial. We apply Reduced Rejection sampling to this problem by setting $q(x) = 1/\sqrt{x}$. Observe that $\mathcal{L} = \Omega = (0, 1)$. As mentioned earlier, inverse transform sampling is easily used to sample according to $q(x)$ and according to $p(x) - q(x)$. The Reduced Rejection sampling is very fast and yields no unwanted sample points.

This example is equivalent to sampling from a mixture and can be extended to sampling from a probability density $p(x)$ that is a sum $p = p_1 + p_2 + \ldots + p_n$, if there is a method for sampling from each p_k separately and the integrals $I[p_k]$ are all known.

2.4 Example 2: Reduced Rejection Sampling for a Stochastic Process with Fluctuating and Singular Rates

In this section, we apply Reduced Rejection sampling to an idealized problem motivated by plasma physics. As discussed in Subsection 2.4.5, the unique features of this problem makes other sampling methods inefficient to use.

2.4.1 Statement of the stochastic process and the simulation algorithm

The example presented here is a simplified version of simulation for recombination by a collision of two electrons with an ion, in which one of the electrons is absorbed into the atom and the other electron is scattered away. For incident electron energies E_1 and E_2, the recombination rate is proportional to $(E_1 E_2)^{-1/2}$ [49, 72], which can become singular if electrons of low energy are involved. This is an obstacle to kinetic simulation of recombination by electron impact in a plasma.

Our goal is to simulate the evolution of the following system: Consider N par-

ticles labeled $1, \ldots, N$. To each particle i we associate a number $x_i \in (0, 1)$, called the state of particle i (and corresponding to electron energy in the recombination problem). Occasionally, where it does not cause confusion, we use x_i to refer to particle i. We refer to the set $\Gamma = \{x_1, \ldots, x_n\}$ as the configuration of the system. For every pair of states x_i and x_j, $T_{i,j}$ is a random variable with an exponential distribution with parameter $(x_i x_j)^\alpha$, in which α is a fixed constant between 0 and 1. $T_{i,j}$ is the time for collision between particle i and j which randomly occurs with rate $1/(x_i x_j)^\alpha$. After a scattering event occurs, say for the pair $\{k, l\}$, the values of states x_k and x_l are replaced by new values x'_k and x'_l; consequently, the distribution of $T_{i,j}$ changes if either of i and j is equal to k or l.

We will consider a simple updating mechanism for the states after each collision. In the simulations presented below, the updated values of x'_k and x'_l are chosen independently and uniformly at random from $(0, 1)$, without dependence on x_k and x_l. This choice is made for simplicity and because the stationary distribution can be calculated for this choice (see Section 2.4.2), but we expect that Reduced Rejection sampling would work equally well for more complex collision rules. Indeed the Algorithm 2.4.1 described below and the more detailed algorithm presented in Section 2.4.4 do not depend on the collision rules.

First we introduce some notation and make a few observations. Set $s_i = 1/x_i^\alpha$ and $s = \sum_i s_i$. Let $T(\lambda)$ denote an exponential random variable with parameter λ (with rate $1/\lambda$); then $T(\lambda) = \mu T(\mu\lambda)$ for any scalar μ. We will use

$$\frac{s_i}{s}\frac{s_j}{s} T(1/s^2) = T(1/(s_i s_j))$$

in the following algorithm, which is a variant of the Kinetic Monte Carlo (KMC) algorithm (also known as the residence-time algorithm or the n-fold way or the Bortz-Kalos-Lebowitz (BKL) algorithm [6]), that simulates the system described above. This algorithm chooses collisions, by choosing two particles separately out of the N number of particles, rather than choosing a pair of particles out of the

N^2 number of pairs.

Algorithm 2.4.1 *1. Start from $t = 0$.*

2. Choose time Δt by sampling from an exponential distribution with rate s^2.

3. Choose index k with probability s_k/s.

4. Choose index l with probability s_l/s.

5. At time $t + \Delta t$ collision between particles k and l occurs.

6. Update states x_k and x_l according to the updating mechanism and the update value of s.

7. Set $t = t + \Delta t$ and start over from 2.

We use Reduced Rejection sampling in Subsection 2.4.5 to perform steps 3 and 4 in the above algorithm. We also explain why other methods of sampling would be inefficient in these circumstances. To verify that our simulation is working properly, we perform the following test.

Let $g(x_1, \ldots, x_N)$ be a real-valued function on the configuration space, with expectation of g over configurations of the system denoted by $E[g]$. For a simple collision rule and specific g we can find the value of $E[g]$ analytically, as shown in Subsection 2.4.2. Consequently, the difference between the numerical and analytic results provides a measure of the accuracy of the simulation as discussed at the end of Subsection 2.4.5.

2.4.2 Theoretical results

Think of the system's evolution as a random walk on the configurations of the system. Suppose the updating process is that if states x_k and x_l collide, then states x'_k and x'_l are chosen independently uniformly at random from $(0, 1)$. In

this section, we find the stationary probability distribution for this random walk and the value of $E[g]$ for two functions g. Observe that for the updating mechanism considered here, the random walk is irreducible; that is, it can go from any configuration to any other configuration (i.e. if for example the updating mechanism had additional constraints, such as $x'_k + x'_l = x_k + x_l$, then the random walk would not be irreducible since it could reach only those configurations whose sum of the states is the same as the sum of the states of the starting configuration).

For every configuration $\Gamma = \{x_1 \ldots, x_N\}$, set

$$\pi_\Gamma := \frac{(x_1 \cdots x_N)^\alpha}{Z} \left(\sum_{i,j} \frac{1}{(x_i x_j)^\alpha} \right),$$

where Z is the normalizing constant so that $\int_\Gamma \pi_\Gamma \, \mathrm{d}\mathbf{x} = 1$. We will show that π_Γ is density of the stationary probability distribution for the system.

Suppose the current configuration of the system is $\Gamma = \{x_1, \ldots, x_N\}$. According to steps 3 and 4 in Algorithm 2.4.1, the probability of collision occurring between states x_k and x_l, with $k \neq l$, is proportional to $s_k s_l = 1/(x_k x_l)^\alpha$. Let $P(\Gamma'|\Gamma)$ denote the transition probability density of going from configuration Γ to configuration Γ'. Then

$$P(\Gamma'|\Gamma) = \begin{cases} \frac{1/(x_k x_l)^\alpha}{\sum_{i,j} 1/(x_i x_j)^\alpha} & \text{if } \Gamma' = \{x'_k, x'_l\} \cup \Gamma \setminus \{x_k, x_l\} \text{ for some } k \neq l. \\ 0 & \text{otherwise.} \end{cases}$$

Now it is straightforward to verify the detailed balance equation

$$\pi_\Gamma P(\Gamma'|\Gamma) = \pi_{\Gamma'} P(\Gamma|\Gamma').$$

Therefore, π_Γ is the unique stationary distribution of the random walk. Uniqueness follows from the fact that the random walk is irreducible.

The normalizing constant Z for probability distribution π_Γ can be calculated by

$$Z = \int_{(0,1)^N} (x_1 \cdots x_N)^\alpha \left(\sum_{i,j} \frac{1}{(x_i x_j)^\alpha} \right) dx_1 \cdots dx_N = \frac{\binom{N}{2}}{(\alpha+1)^{N-2}}.$$

Hence for any function $g(x_1, \ldots, x_N)$,

$$E[g] = \frac{(\alpha+1)^{N-2}}{\binom{N}{2}} \int_{(0,1)^N} g(x_1, \ldots, x_N)(x_1 \cdots x_N)^\alpha \left(\sum_{i,j} \frac{1}{(x_i x_j)^\alpha} \right) dx_1 \cdots dx_N.$$

Some tedious algebra leads to the following proposition:

Proposition 2.4.2 *Using the above notations and assumptions:*

a) $E[g] = \frac{\alpha+1}{\alpha+2}(N-2) + 1$ *when* $g(x_1, \ldots, x_N) = x_1 + \cdots + x_N$.

b) $E[g] = \frac{\alpha+1}{\alpha+3}(N-2) + \frac{2}{3}$ *when* $g(x_1, \ldots, x_N) = x_1^2 + \cdots + x_N^2$.

2.4.3 Simulation issues

In this section we make some remarks about the challenges involved in simulating this system.

The main challenge of sampling in this dynamic simulation is that the s_i's are changing after each collision. Consequently, the sampling method should require small or zero preprocessing time. For this reason, discrete sampling methods such as Marsaglia's table method or the Alias method are not very efficient for this problem.

Next consider using Acceptance Rejection method based on uniform sampling from 1 to n for the proposal distribution (i.e., q constant). As mentioned earlier, the changing distribution property of the problem is not very detrimental for Acceptance Rejection. On the other hand, the singularity in the rates at $x_i = 0$ can lead to a large constant for q, for which there will be many rejected samples, so that the method is inefficient.

Moreover, there seems to be no other clear choice for the proposal distribution q other than a constant. Note that the sampling is from a discrete set of probabilities s_i/s with little control over their values; for example the s_i's are not monotonically

ordered. This is quite different from sampling a single random variable from the density $p(x) = x^{-\alpha}$.

2.4.4 Use of the Reduced Rejection algorithm

In this section we explain how to use Reduced Rejection Sampling to perform steps 3 and 4 in Algorithm 2.4.1. Reduced Rejection sampling can be readily used in this dynamic simulation. Even though the values of the s_i's change after each collision, they do not change drastically; in each collision at most two of the s_i's change. Starting at time 0, we set $q_i = p_i = s_i$. After each collision, we update the values of p_i's to $p_i = s_i$, but do not change the values of q_i's. Note that we can easily update the value of $I[p]$ after each collision and keep track of set $\mathcal{L} = \{i : p_i > q_i\}$ by comparing the updated values of p_i's to their corresponding values of q_i's. Moreover, the size of set $\mathcal{L}$ changes by at most 2 after each collision (but it can also decrease after some collisions).

We use Marsaglia's table method to sample according to q_i's. Since we do not update q_i's after each collision, the preprocessing time in Marsaglia's table method is only required for the first sampling and not for the subsequent samplings. To sample from $\mathcal{L}$ according to $p_i - q_i$, we use Acceptance Rejection with uniform distribution for the proposal distribution. As long as the size of set $\mathcal{L}$ is not too big, the sampling from $\mathcal{L}$ is not very time consuming. To prevent $\mathcal{L}$ from getting too large, we reset the values of q_i's to $q_i = p_i = s_i$, which sets $\mathcal{L}$ to be empty, whenever the size of $\mathcal{L}$ exceeds a predetermined number M.

The size of M is important for the performance of the algorithm. If M is too small, then there are many updates of the q_i's, each of which requires preprocessing time for Marsaglia's table method. On the other hand, if M is too big, then $\mathcal{L}$ is large and costly to sample from by Acceptance Rejection. Our computational experience shows that setting M equal to a multiple of $\sqrt{N}$ is a good choice. It

might be better for the reinitialization criterion to be based on the efficiency of the sampling from $\mathcal{L}$ (i.e., the fraction of rejected samples when using acceptance rejection on $\mathcal{L}$), rather than the size of $\mathcal{L}$.

2.4.5 Numerical result

We simulated the evolution of the system under the conditions outlined in Subsection 2.4.2 with $N = 10^4$, and $\alpha = 0.5$. We start with a random configuration at time $t = 0$. The simulation is based the Reduced Rejection sampling method, using Marsaglia's table method and the Acceptance Rejection method as described above. After each collision, we evaluate $g(x_1, \ldots, x_N) = x_1 + \cdots + x_N$ and take the average to get an estimate for $E[g]$. Each result is produced by taking an arithmetic average of five independent runs. Figure 2.5 compares the results for $E[g]$ from Reduced Rejection sampling with those from the Acceptance Rejection method. The results of Figure 2.5 show excellent agreement between the values of $E[g]$ as a function of the number of collisions from the two methods, which provides a validity check for Reduced Rejection sampling.

The advantage of Reduced Rejection sampling is demonstrated in Figure 2.6 which shows a log-log plot of the processing time as a function of the number of collisions, for Reduced Rejection sampling and Acceptance Rejection. The results show that Reduced Rejection sampling is much faster than the Acceptance Rejection method. In fact, for n collisions, the computational time scales as $O(n)$ for Reduced Rejection sampling, and as $O(n^{3/2})$ for Acceptance Rejection, in the range $10^4 \leq n \leq 10^6$. For small values of n, the initial pre-processing step of Marsaglia's table method dominates the computational time. For $n > 10^4$, however, the pre-processing time (including the multiple pre-processing steps due to reinitialization) is not a significant part of the computational time. The average number of reinitialization steps for Reduced Rejection sampling is $(0, 0, 0, 3.7, 53.1)$ for $n = (10^2, 10^3, 10^4, 10^5, 10^6)$, respectively. Another interesting advantage of

the Reduced Rejection sampling is that the variance of the processing time for independent runs is much smaller in the Reduced Rejection sampling than it is in the Acceptance Rejection method.

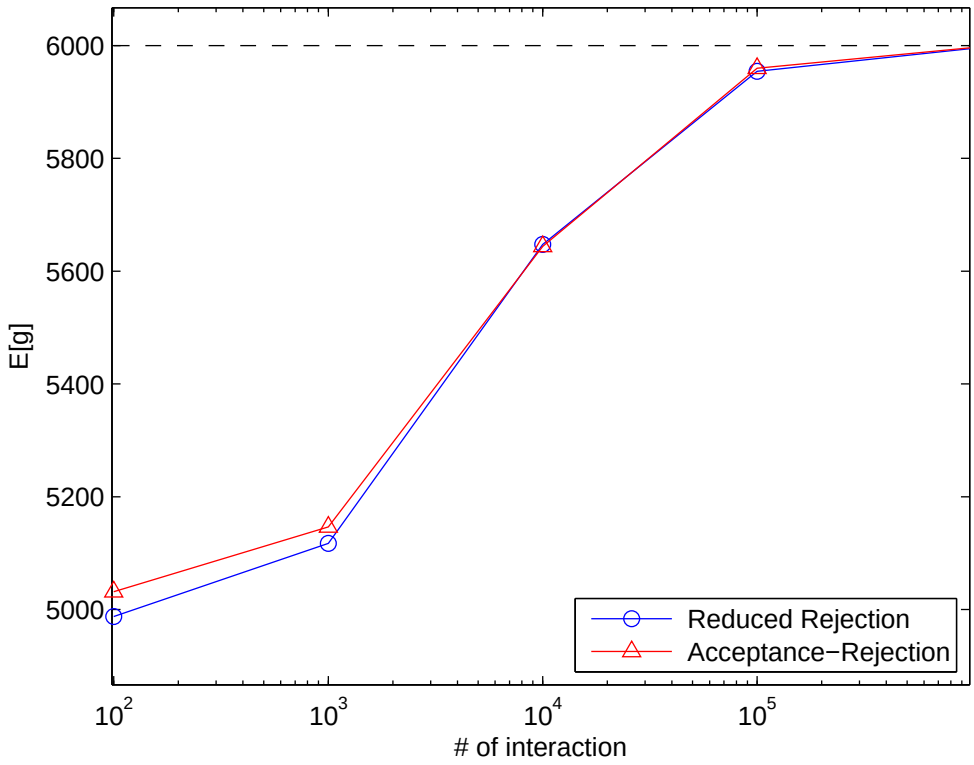

Figure 2.5: Theoretical (dashed line) and estimated values (solid lines) of $E[g]$ using different number of collisions. Here $g(x_1, \ldots, x_N) = x_1 + \cdots + x_N$, $N = 10^4$, and $\alpha = 0.5$. Also $M = 4000$ for the Reduced Rejection sampling. The theoretical value of $E[g]$ is 5999.8. The estimated value of $E[g]$ after 10^6 collisions using Reduced Rejection sampling and Acceptance Rejection methods were, respectively, 5994.59 and 5996.35. The reported result is the average of 5 independent runs.

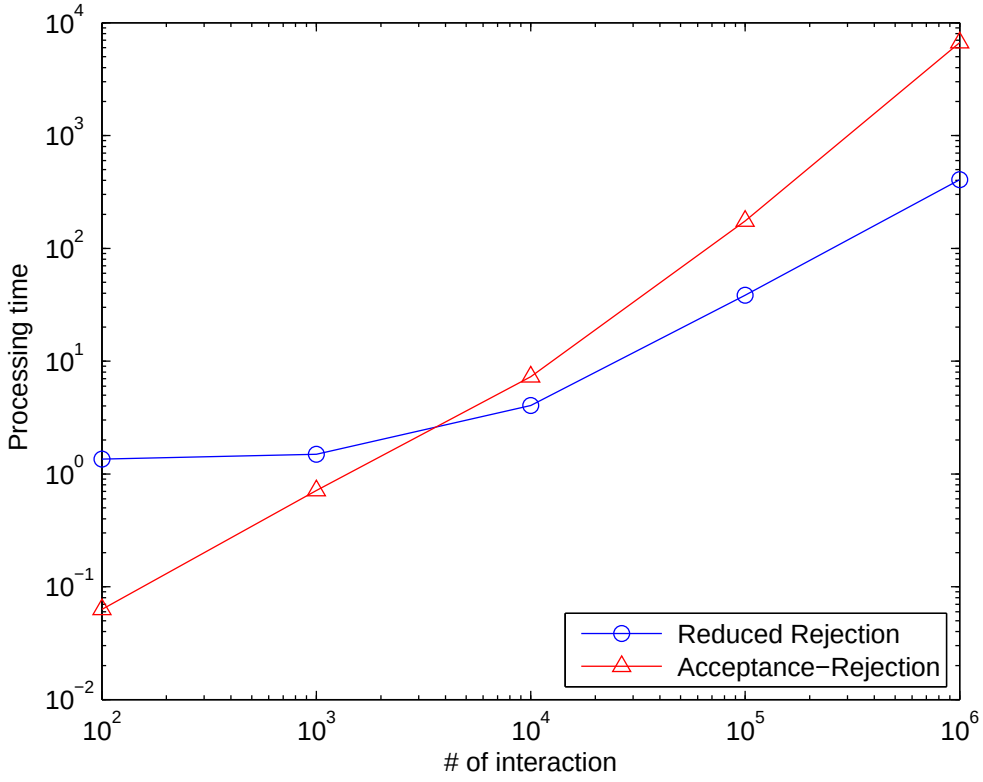

Figure 2.6: A Loglog plot of the processing time for Acceptance Rejection and Reduced Rejection sampling. Here $g(x_1, \ldots, x_N) = x_1 + \cdots + x_N$, $N = 10^4$ and $\alpha = 0.5$. Also $M = 4000$ for the Reduced Rejection sampling. The reported processing time is the average of 5 independent runs.

2.5 Example 3: Stochastic Simulation of Chemical Kinetics

In this section we describe how we can use Reduced Rejection in the context of stochastic chemical kinetics. The term stochastic simulation in chemical kinetics typically refers to a Monte Carlo procedure to numerically simulate the time evolution of a well-stirred chemically reacting system. The first Stochastic Simulation Algorithm, called the Direct Method, was presented in [32]. The Direct Method is computationally expensive and there have been many adaptations of this algorithm to achieve greater speed in simulation. The first-reaction method, also in [32], is an equivalent formulation of the Direct Method. The next-reaction method [28] is an improvement over the first-reaction method, using a binary-tree structure to store the reaction times. The Modified Direct Method [16] and Sorting Direct Method [45] speed up the Direct Method by indexing the reactions in such a way that reactions with larger propensity function tend to have a lower index value. Recently, some new Stochastic Simulation Algorithms, called partial-propensity methods, were introduced that work only for elementary chemical reactions (i.e. reactions with at most two different reactants) (see [54, 55, 56]). Nevertheless, note that it is possible to decompose any non-elementary reaction into combination of elementary reactions. There are also approximate Stochastic Simulation Algorithms, such as tau-leaping and slow-scale, that provide better computational efficiency in exchange for sacrificing some of the exactness in the Direct Method (see [31] and the references therein for more details).

Next we give a brief review of stochastic simulation in chemical kinetics. An excellent reference with more detailed explanation is [31]. Using the same notation and terminology as in [31], consider a well-stirred system of molecules of N chemical species $\{S_1, \ldots, S_N\}$, which interact through M chemical reactions $\{R_1, \ldots, R_M\}$. Let $X_i(t)$ denote the number of molecules of species S_i in the sys-

tem at time t. The goal is to estimate the state vector $\mathbf{X}(t) \equiv (X_1(t), \ldots, X_N(t))$ given the system is initially in state $\mathbf{X}(0) = \mathbf{x}_0$.

Similarly to Section 2.4, when the system is in state $\mathbf{x}$, the time for reaction R_j to occur is given by an exponential distribution whose rate is the propensity function $a_j(\mathbf{x})$. When reaction R_j occurs, the state of the system changes from $\mathbf{x}$ to $\mathbf{x} + (v_{1j}, \ldots, v_{Nj})$, where v_{ij} is the change in the number of S_i molecules when one reaction R_j occurs.

Estimating the propensity functions is not an easy task in general. As noted, the values of the propensity functions depend on the state of the system. For example, if R_i and R_j are, respectively, the unimolecular reaction $S_1 \to \mathsf{product}(s)$ and bimolecular reaction $S_1 + S_2 \to \mathsf{product}(s)$, then $a_i(\mathbf{x}) = c_i x_1$ and $a_j(\mathbf{x}) = c_j x_1 x_2$ for some constants c_i and c_j. Therefore, the propensity functions of the reactions are changing throughout the simulation. Moreover, if for some chemical species the magnitude of their population differs drastically from others, we expect the values of propensity functions to be very non-uniform.

For every state $\mathbf{x}$, define

$$a(\mathbf{x}) = \sum_{i=1}^{M} a_i(\mathbf{x}).$$

To simulate the chemical kinetics of the system the following algorithm is used, which resembles Algorithm 2.4.1 in Section 2.4.

Algorithm 2.5.1 *1. Start from time $t = 0$ and state $\mathbf{x} = \mathbf{x}_0$.*

2. Choose time Δt by sampling from an exponential distribution with rate $a(\mathbf{x})$.

3. Choose index k with probability $a_k(\mathbf{x})/a(\mathbf{x})$.

4. At time $t + \Delta t$ reaction R_k occurs.

5. Update time $t = t + \Delta t$, state $\mathbf{x} = \mathbf{x} + (v_{1k}, \ldots, v_{Nk})$ and start over from 2.

In the original Direct Method [32] step 3 in the above Algorithm 2.5.1 is performed by choosing number r uniformly at random in the unit interval and setting

$$k = \text{ the smallest integer satisfying } \sum_{i=1}^{k} a_i(\mathbf{x}) > ra(\mathbf{x}). \tag{2.14}$$

However, when we have many reactions with a wide range of propensity function values presented in the system, a scenario that is very common in biological models, the above procedure of using partial sums becomes computationally expensive. As noted earlier, some methods, such as the Modified Direct Method [16] and Sorting Direct Method [45], index the reactions in a smart way that reduces the average number of terms summed in equation (2.14), which results in computational efficiency.

We propose a different approach to performing step 3 in Algorithm 2.5.1 using the Acceptance Rejection or Reduced Rejection method. The approach is very similar to what was done in Section 2.4. To be specific, we can use Acceptance Rejection for step 3 in the following way: let

$$\bar{a}(\mathbf{x}) = \max_i a_i(\mathbf{x}).$$

Until an index is accepted, select index k uniformly at random from $\{1, \ldots, N\}$ and accept it with probability $a_k(\mathbf{x})/\bar{a}(\mathbf{x})$; otherwise, discard k and repeat. When an index is accepted step 3 in Algorithm 2.5.1 is completed. Typically for chemical reactions R_j, most of v_{ij}'s are zero; therefore, we can efficiently update the value of $\bar{a}(\mathbf{x})$ at each iteration of Algorithm 2.5.1.

However, as in Section 2.4, if the values of $a_i(\mathbf{x})$'s are very non-uniform (for example, when the population of some chemical species differ drastically from that of other species in the system) the Acceptance Rejection method becomes inefficient due to rejection of many samples. In these circumstances, the Reduced Rejection algorithm can be readily used in a very similar way as it was used in Section 2.4. We expect that the use of the Reduced Rejection algorithm in these

circumstances would greatly improve the computational efficiency of the exact Stochastic Simulation Algorithms.

2.6 Conclusions and Future Directions

In this chapter we introduced a new Reduced Rejection sampling method that can be used to generate independent samples for a discrete or continuous random variable. The strength of this method is most evident for applications in which Acceptance Rejection method is inefficient; namely, the probability distribution of the sample random variable is highly peaked in certain regions or has singularities. It is also useful when the probabilities are dynamically updated, so that discrete methods that requiring preprocessing are inefficient. In particular, the Reduced Rejection sampling method is expected to perform well on kinetic simulation of electron-impact recombination in a plasma, which is difficult to simulate by other methods.

The preliminary examples in this chapter are meant to illustrate these advantages of the Reduced Rejection sampling method. They provide evidence of improvement in computation time using the Reduced Rejection sampling versus Acceptance Rejection method. These examples also provide some insights on implementation of the method.

One possible direction for future research is the nested use of Reduced Rejection sampling methods. For the most difficult step - sampling from $\mathcal{L}$ according to $p(x) - q(x)$ - we propose to apply the Reduced Rejection sampling method again using a new proposal function. In essence, this would use one Reduced Rejection sampling method inside another Reduced Rejection sampling method.

CHAPTER 3

A Time Continuation Based Fast Approximate Algorithm for Compressed Sensing Related Optimization

This chapter introduces a fast approximate algorithm to numerically solve a class of optimization problems that are related to the Compressed Sensing problem. The methodology is to first find an approximate solution to the Lagrangian dual formulation of the problem and then use the result to find an approximate solution to the primal problem.

The use of Lagrangian dual formulation to numerically solve optimization problems is ubiquitous in optimization literatures. The ideas behind many optimization algorithms can be easily understood in the language of their dual formulation. For example, the Dual Ascent method (see for example [7]) can be thought of as performing an explicit gradient method on the dual problem. Also, considering that the action of proximal operator is equivalent to an implicit gradient (i.e. backward Euler) step, then methods such as Augmented Lagrangian method [35, 53] (i.e. also known as the method of multipliers) can be thought as performing an implicit gradient method on the dual problem.

It is also well understood that for some optimization problems, regularizing the objective function may significantly simplify the problem for numerical computations. Sometimes regularizing the primal problem may also result into an easier-to-handle dual formulation (i.e. see section 3.1 for more details).

Consider the following minimization problem, also known as Basis Pursuit problem (i.e. see for example [17]),

$$\min_{y} \|y\|_1 \quad \text{subject to} \quad \mathbf{A}y = \mathbf{b}. \tag{3.1}$$

Throughout this chapter, $|\cdot|$ stands for the Euclidean norm $\|\cdot\|_2$ and $\mathbf{A}$ for an $m \times n$ matrix with $m \ll n$. Let $R(t, y)$ be some specific function that incorporates the constraint $\mathbf{A}y = \mathbf{b}$. It has recently been noted in [20] that if the Basis Pursuit problem is relaxed by adding regularization $\frac{1}{2\mu}|y|^2$ and penalty term $R(t, y)$:

$$\underset{y \in \mathbb{R}^m}{\operatorname{argmin}} \left\{ \|y\|_1 + \frac{1}{2\mu}|y|^2 + R(t, y) \right\}, \tag{3.2}$$

then the Lagrangian dual problem is of the form:

$$\underset{\mathbf{s} \in \mathbb{R}^n}{\operatorname{argmax}} \left\{ \langle \mathbf{b}, \mathbf{s} \rangle - \frac{\mu}{2}|\mathsf{shrink}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}})|^2 - tH(|\mathbf{s}|) \right\}. \tag{3.3}$$

For two vectors with same length $\vec{x}$ and $\vec{v}$, the shrink operator is defined by

$$\mathsf{shrink}(\vec{x}, \vec{v})_i = \max(|x_i| - v_i, 0)\frac{x_i}{|x_i|}.$$

Here function $H(\cdot)$ is determined by $R(t, y)$; for example, when $R(t, y) = \frac{1}{2t}|\mathbf{A}y - \mathbf{b}|^2$ and $R(t, y) = -\sqrt{t^2 - |\mathbf{A}y - \mathbf{b}|^2}$, then $H(|\mathbf{s}|) = \frac{1}{2}|\mathbf{s}|^2$ and $H(|\mathbf{s}|) = \sqrt{1 + |\mathbf{s}|^2}$, respectively.

We also note that the idea of using duality for (3.2) was first used in [70], for $t = 0$ and then in [20] to obtain formula (3.3). We note that there is an interesting link between this duality formulation and viscosity solutions for certain Hamilton-Jacobi equations, pointed out in [20] (see also [21]).

This chapter proposes a new and practical approximate algorithm that uses the duality correspondence between (3.2) and (3.3) to find an approximate solution to the Compressed Sensing problem. The algorithm is named *Time Continuation Compressed Sensing*, or TCCS for short. The TCCS algorithm consists of two parts. The first part of the algorithm finds an approximate solution to the

constrained problem

$$\underset{\mathbf{s}}{\operatorname{argmax}} \{\langle \mathbf{b}, \mathbf{s} \rangle - tH(|\mathbf{s}|)\} \quad \text{subject to} \quad -\overrightarrow{\mathbf{1}} \leq \mathbf{A}^T\mathbf{s} \leq \overrightarrow{\mathbf{1}}. \tag{3.4}$$

The second part of the TCCS algorithm uses the Lagrangian dual correspondence between (3.2) and (3.3), and a first order approximation in terms of $1/\mu$ to find an approximate solution to problem

$$\underset{y \in \mathbb{R}^m}{\operatorname{argmin}} \{\|y\|_1 + R(t, y)\}, \tag{3.5}$$

Although the TCCS algorithm does not find the exact optimal argument for problem (3.5) (or (3.4)), it performs very well. Section 3.4 provides numerical evidence that shows that when the TCCS algorithm is applied to Compressed Sensing problems, it perfectly recovers the solution when it is relatively sparse. Moreover, the TCCS algorithm is faster than some of the fastest methods used for solving Compressed Sensing problems (see Section 3.4 for more details). Another advantage of using the TCCS algorithm for Compressed Sensing problems is that it is parameter free.

We also derive some theoretical results that are important in their own rights. In particular, an alternative proof for the *exact regularization property* introduced in [70] and [27] is presented. It is shown in Subsection 3.2.3 how the solution of problem (3.4) can be used to yield an approximate solution to primal problem (3.2) for sufficiently large μ.

Some readers might think that the idea used in the first part of the TCCS algorithm resembles the Simplex method or the Interior method used in optimization. However, the resemblance is misleading. Suppose a polytope Ω is defined by

$$\Omega = \{\mathbf{s} \in \mathbb{R}^m : a_i^T\mathbf{s} \leq 1, \text{ for } i = 1 \ldots, n\}.$$

For every point $\mathbf{s} \in \mathbb{R}^m$, let $J(\mathbf{s})$ be the set of indices $i \in \{1, \ldots, n\}$ for which $a_i^T\mathbf{s} \geq 1$. We call $J(\mathbf{s})$ the set of *violations* of $\mathbf{s}$. We sometimes use J instead

of $J(\mathbf{s})$ when this does not cause confusion. Let $\mathbf{A}_J$ be the matrix formed by columns of $\mathbf{A}$ whose index belongs to J.

In each iteration of the Simplex method, we move from an extreme point (a point on the boundary of Ω for which the corresponding matrix $\mathbf{A}_J$ has rank m) to another extreme point. On the other hand, in each iteration of the Interior method, the corresponding points are inside Ω, and therefore their set of violations is empty (i.e. by convention $\mathbf{A}_J$ has rank zero for these points). However, in the TCCS algorithm, we start with a point inside the polytope Ω and at each iteration move to a new point on a face of Ω in such a way that the rank of $\mathbf{A}_J$ increases by at least one. As a consequence the algorithm stops after a maximum of m iterations.

The algorithm also resembles LARS [25] and adaptive inverse scale space [10], but is certainly different from these compressed sensing related algorithms.

Indeed, the algorithm used for the first part of the TCCS algorithm, belongs to a class of techniques called Homotopy method (see, for example, [65] and [66]), suitably adapted for the particular problem at hand. Let

$$g_\tau(\mathbf{s}) = \langle \mathbf{b}, \mathbf{s} \rangle - \tau H(|\mathbf{s}|).$$

In essence, we generate a sequence $\tau_0 > \tau_1 > \cdots > \tau_\ell = t$ and iteratively find approximate maximizers $\mathbf{s}^1_{opt}, \ldots, \mathbf{s}^\ell_{opt}$, respectively, for functions $g_{\tau_0}(\mathbf{s}), \ldots, g_{\tau_\ell}(\mathbf{s})$ subject to constraint $-\overrightarrow{\mathbf{1}} \leq \mathbf{A}^T\mathbf{s} \leq \overrightarrow{\mathbf{1}}$. As a result, $\mathbf{s}^\ell_{opt}$ yields a good approximate maximizer for (3.4) .

The main contribution of this chapter is to introduce an algorithm that perfectly recovers the solution of Compressed Sensing problems when the solution is relatively sparse with respect to the number of measurements. For this regime of sparsity, the recovery is extremely fast compare to other algorithms being used. Moreover, the TCCS algorithm has a fixed number of iterations. Also, the TCCS algorithm exhibits a sharp phase transition in the success rate of recovery of so-

lution when sparsity of solution varies.

The reminder of this chapter is organized as follows: Section 3.1 presents a quick overview of different regularizations of the Basis Pursuit problem and their corresponding Lagrangian dual formulation. Section 3.2 provides some theoretical result that are used in developing the TCCS algorithm. Section 3.3 describes the general idea of the TCCS algorithm. Section 3.4 contains numerical evidence for the good performance of the TCCS algorithm. Section 3.5 contains some concluding remarks. In Appendix B pseudo-codes for the TCCS algorithm are presented.

3.1 Augmented Problems and Lagrangian Duality Formulation

The goal of compressed sensing is to find sparse solutions to equation $\mathbf{A}y = \mathbf{b}$, where $\mathbf{A}$ is an $m \times n$ matrix with $m \ll n$. The Compressed Sensing problem is formulated as

$$\min_y \|y\|_0 \quad \text{subject to} \quad \mathbf{A}y = \mathbf{b}, \tag{3.6}$$

where $\|y\|_0$ counts the number of nonzero entries in y.

It turns out (i.e. see [13]) that the sparse solution is often the same as the solution to the Basis Pursuit problem (3.1). The lack of smoothness of the objective function in (3.1) poses a challenge for numerical minimization. Moreover, the Lagrangian dual of (3.1),

$$\max_{\mathbf{s}} \langle \mathbf{b}, \mathbf{s} \rangle \quad \text{subject to} \quad \mathbf{A}^T\mathbf{s} \leq \overrightarrow{\mathbf{1}}. \tag{3.7}$$

is a constrained and non-strictly convex problem.

Problem (3.1) can be relaxed in several ways: For $\mu > 0$, one can introduce a

regularization term and consider the problem

$$\min_y \left\{ \|y\|_1 + \frac{1}{2\mu}|y|^2 \right\} \quad \text{subject to} \quad \mathbf{A}y = \mathbf{b}. \tag{3.8}$$

It has been shown in [70] and [27] that the above problem has the exact regularization property: there exist a large fixed μ_0, depending on $\mathbf{A}$ and $\mathbf{b}$, such that for $\mu > \mu_0$, the solution to (3.8) is the same as the solution to (3.1). In Corollary 3.2.7 an alternative proof for the exact regularization property is provided. Moreover, as noted in [38], the Lagrange dual problem of (3.8) is unconstrained and differentiable; consequently, many classical techniques such as Nesterov's acceleration method [47], Barzilai–Borwein step sizes [4], and non monotone line search can be used to speed up computation.

Alternatively, one can add a penalty term to (3.1) and consider the following problem (known in the literature as LASSO [60] or Basis Pursuit Denoising [19]):

$$\min_y \left\{ \|y\|_1 + \frac{1}{2t}|\mathbf{A}y - \mathbf{b}|^2 \right\}. \tag{3.9}$$

For certain applications, $\mathbf{b}$ may contain noise; which makes solving (3.9) more preferable than solving (3.1). The TCCS algorithm proposed in this chapter yields approximate solution to problems of the type (3.9).

The primal problem (3.9) is unconstrained; however, its Lagrangian dual problem is constrained. In contrast, the primal problem (3.8) is constrained, whereas, its Lagrangian dual problem is unconstrained.

Motivated by the above two relaxation procedure, next turn to problem (also known as Augmented Lasso),

$$\min_y \left\{ \|y\|_1 + \frac{1}{2\mu}|y|^2 + \frac{1}{2t}|\mathbf{A}y - \mathbf{b}|^2 \right\}, \tag{3.10}$$

and a related problem

$$\min_y \left\{ \|y\|_1 + \frac{1}{2\mu}|y|^2 - \sqrt{t^2 - |\mathbf{A}y - \mathbf{b}|^2} \right\} \quad \text{subject to} \quad |\mathbf{A}y - \mathbf{b}|^2 \leq t. \tag{3.11}$$

As it is shown in Appendix A, the Lagrangian dual of the above problems is of the form

$$\max_{\mathbf{s}\in\mathbb{R}^m} \left\{ \langle \mathbf{b}, \mathbf{s} \rangle - \frac{\mu}{2} |\mathsf{shrink}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}})|^2 - tH(|\mathbf{s}|) \right\}, \tag{3.12}$$

with $H(x) = x^2/2$ for problem (3.10), and $H(x) = \sqrt{1+x^2}$ for problem (3.11). Hence, the Lagrangian dual formulation can be used to solve problems (3.10) and (3.11) in the following way (see for example [21]): First compute

$$\mathbf{s}^* = \operatorname*{argmax}_{\mathbf{s}\in\mathbb{R}^m} \left\{ \langle \mathbf{b}, \mathbf{s} \rangle - \frac{\mu}{2} |\mathsf{shrink}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}})|^2 - tH(|\mathbf{s}|) \right\}, \tag{3.13}$$

then compute

$$y^* = \mu \cdot \mathsf{shrink}(\mathbf{A}^T\mathbf{s}^*, \overrightarrow{\mathbf{1}}).$$

As it turns out,

- if $H(x) = \frac{1}{2}x^2$, then y^* solves (3.10),
- if $H(x) = \sqrt{1+x^2}$, then y^* solves (3.11).

Moreover [20, 21] demonstrate that,

$$\varphi(\mathbf{b}, t) = \sup_{\mathbf{s}} \left\{ \langle \mathbf{b}, \mathbf{s} \rangle - \frac{\mu}{2} |\mathsf{shrink}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}})|^2 - tH(|\mathbf{s}|) \right\}$$

is the solution to the initial value Hamilton-Jacobi equation

$$\begin{cases} \varphi_t + H(D_{\mathbf{b}}\varphi) = 0 \\ \varphi(\mathbf{b}, 0) = \inf_y \left\{ \|y\|_1 + \frac{1}{2\mu}|y|^2 \right\} \quad \text{such that} \quad \mathbf{A}y = \mathbf{b}. \end{cases}$$

3.2 Theoretical Results

This section establishes some theoretical results. In Subsections 3.2.1 and 3.2.2 we study the trajectory of the solutions for problems (3.3) and (3.4), respectively, as we vary parameter t. In Subsection 3.2.3, a first order approximation in terms of $1/\mu$ is used to establish relation between the solution of (3.4) and the solution of the of the primal problem (3.2) when μ is large.

Indeed, in the rest of the chapter, we consider the more general problems

$$\mathbf{s}_{opt}(t) = \text{argmax}\{\langle \mathbf{b}, \mathbf{s}\rangle - tH(|\mathbf{s}|)\} \quad \text{subject to } \mathbf{A}^T\mathbf{s} \leq \overrightarrow{\mathbf{1}}, \tag{3.14}$$

and

$$\mathbf{s}^*(t,\mu) = \underset{\mathbf{s}\in\mathbb{R}^m}{\text{argmax}} \left\{ \langle \mathbf{b}, \mathbf{s}\rangle - \frac{\mu}{2}|\mathcal{S}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}})|^2 - tH(|\mathbf{s}|) \right\}, \tag{3.15}$$

where operator $\mathcal{S}$ is defined by $\mathcal{S}(\vec{x}, \vec{v})_i = \max(x_i - v_i, 0)$ and $\mathbf{A}^T$ is such that the polytope

$$\Omega = \{\mathbf{s} : \mathbf{A}^T\mathbf{s} \leq \overrightarrow{\mathbf{1}}\}$$

is bounded. Here, $\mathbf{b}$, μ and t are fixed, $\mathbf{A}$ is $m \times n$ matrix with $m \ll n$. The following assumptions about function H are made:

1. H is a function from positive real numbers to positive real numbers.
2. H is strictly convex and strictly increasing.
3. H is differentiable with h denoting its derivative.
4. h is a composition of polynomials and radicals.
5. h maps bounded sets to bounded sets.

Remark 3.2.1 *Assumption 4 in above is used in Lemma 3.2.3 and Assumption 5 is used in Theorem 3.2.4. One might be able to weaken these assumptions.*

Remark 3.2.2 *The functions H given in Section 3.1, satisfy all these assumptions.*

To see that the new formulation yields the original problem, observe that if $\tilde{\mathbf{A}}$ denotes the concatenation of $\mathbf{A}$ and $-\mathbf{A}$ along their columns; that is

$$\tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{A} & | & -\mathbf{A} \end{bmatrix}, \tag{3.16}$$

then $|\mathsf{shrink}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}})| = |\mathcal{S}(\tilde{\mathbf{A}}^T\mathbf{s}, \overrightarrow{\mathbf{1}})|$. Therefore, the solution to (3.15), with $\tilde{\mathbf{A}}$ in place of $\mathbf{A}$, is the same as the solution to (3.13). Moreover, if $y = \mu \cdot \mathsf{shrink}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}})$ and $\tilde{y} = \mu \cdot \mathcal{S}(\tilde{\mathbf{A}}^T\mathbf{s}, \overrightarrow{\mathbf{1}})$, then

$$y = \begin{bmatrix} \mathbf{I} & | & -\mathbf{I} \end{bmatrix} \tilde{y}, \tag{3.17}$$

where $\mathbf{I}$ is the identity matrix.

3.2.1 Unconstrained dual problem

This subsection investigates the solution to problem (3.14).

Set

$$f_\tau(\mathbf{s}) = \langle \mathbf{b}, \mathbf{s} \rangle - \tau H(|\mathbf{s}|) - \frac{\mu}{2}|\mathcal{S}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}})|^2. \tag{3.18}$$

Observe that

$$\nabla f_\tau(\mathbf{s}) = \mathbf{b} - \tau h(|\mathbf{s}|)\frac{\mathbf{s}}{|\mathbf{s}|} - \mu\mathbf{A}\mathcal{S}(\mathbf{A}^T\mathbf{s}, \overrightarrow{\mathbf{1}}).$$

Because of the concavity of functions $f_\tau(\mathbf{s})$ and since $f_\tau(\mathbf{s})$ is bounded above when $\mu > 0$, we know that $\mathbf{s}^* = \mathbf{s}^*(\tau, \mu) = \mathrm{argmax}_{\mathbf{s}}(f_\tau(\mathbf{s}))$ is either the origin or satisfies

$$0 = \mathbf{b} - \tau h(|\mathbf{s}^*|)\frac{\mathbf{s}^*}{|\mathbf{s}^*|} - \mu\mathbf{A}\mathcal{S}(\mathbf{A}^T\mathbf{s}^*, \overrightarrow{\mathbf{1}}).$$

It is clear that for nontrivial cases, the optimal argument $\mathbf{s}^*$ is not at the origin. For that reason, we usually discard the origin in the discussion that follows.

For every point $\mathbf{s} \in \mathbb{R}^m$, let $J(\mathbf{s})$ be the set of indices i for which

$$a_i^T\mathbf{s} \geq 1.$$

Here a_i^T denotes the i^{th} row of matrix A^T. We call $J(\mathbf{s})$ the set of *violations* of $\mathbf{s}$. We sometimes use J instead of $J(\mathbf{s})$ where this does not cause confusion. Let $\mathbf{A}_J^T$ be the matrix formed by appending the rows of $\mathbf{A}^T$ whose index belong to J. Let $\mathbf{A}_J$ be the transpose of $\mathbf{A}_J^T$; that is, the matrix formed by appending the columns of $\mathbf{A}$ whose index belongs to J. Now, the previous equation is equivalent to

$$0 = \mathbf{b} - \tau h(|\mathbf{s}^*|)\frac{\mathbf{s}^*}{|\mathbf{s}^*|} - \mu\mathbf{A}_J(\mathbf{A}_J^T\mathbf{s}^* - \overrightarrow{\mathbf{1}}).$$

Rearranging implies that

$$\mathbf{s}^* = \left(\tau\frac{h(|\mathbf{s}^*|)}{|\mathbf{s}^*|}\mathbf{I} + \mu\mathbf{A}_J\mathbf{A}_J^T\right)^{-1}(\mathbf{b} + \mu\mathbf{A}_J\overrightarrow{\mathbf{1}}), \tag{3.19}$$

where $\mathbf{I}$ is the identity matrix. The matrix inverse in equation (3.19) always exist for $\tau > 0$, $0 < \mu < \infty$ and $\mathbf{s}^*$ not the origin[1].

Consider SVD decomposition

$$\mathbf{A}_J^T = U\Sigma\begin{bmatrix} V^T \\ W^T \end{bmatrix},$$

where rows of V^T (i.e. $\{v_1^T, \ldots, v_k^T\}$) and W^T (i.e. $\{w_1^T \ldots, w_{m-k}^T\}$) correspond, respectively, to nonzero and zero singular values of $\mathbf{A}_J^T$. In particular, $\text{span}\{a_i\}_{i\in J} = \text{span}\{v_i\}_{i=1}^k$. Moreover, $\{v_1, \ldots, v_k, w_1 \ldots, w_{m-k}\}$ form an orthonormal set of basis. Observe that,

$$\mathbf{A}_J^T = \sum_{i=1}^k u_i\sigma_i v_i^T \qquad \text{and} \qquad \mathbf{A}_J\mathbf{A}_J^T = \sum_{i=1}^k \sigma_i^2 v_i v_i^T.$$

Note that,

$$\tau\frac{h(|\mathbf{s}^*|)}{|\mathbf{s}^*|}\mathbf{I} + \mu\mathbf{A}_J\mathbf{A}_J^T = \tau\frac{h(|\mathbf{s}^*|)}{|\mathbf{s}^*|}\sum_{i=1}^{m-k} w_i w_i^T + \sum_{i=1}^k(\mu\sigma_i^2 + \tau\frac{h(|\mathbf{s}^*|)}{|\mathbf{s}^*|})v_i v_i^T. \tag{3.20}$$

Taking an inverse and substituting in (3.19) yields that

$$\mathbf{s}^* = \frac{|\mathbf{s}^*|}{\tau h(|\mathbf{s}^*|)}(\sum_{i=1}^{m-k} w_i w_i^T)(\mathbf{b} + \mu\mathbf{A}_J\overrightarrow{\mathbf{1}}) + \sum_{i=1}^k \frac{1}{\mu\sigma_i^2 + \tau h(|\mathbf{s}^*|)/|\mathbf{s}^*|} v_i v_i^T(\mathbf{b} + \mu\mathbf{A}_J\overrightarrow{\mathbf{1}}).$$

The w_i's are in the space perpendicular to $\text{span}\{a_i\}_{i\in J}$, $w_i^T\mathbf{A}_J = 0$. Therefore, the above equation simplifies to

$$\mathbf{s}^*(\tau,\mu) = \frac{|\mathbf{s}^*|}{\tau h(|\mathbf{s}^*|)}(\sum_{i=1}^{m-k} w_i w_i^T)\mathbf{b} + \sum_{i=1}^k \frac{1}{1 + \frac{\tau}{\mu\sigma_i^2}h(|\mathbf{s}^*|)/|\mathbf{s}^*|}\frac{v_i v_i^T}{\sigma_i^2}(\mathbf{b}/\mu + \mathbf{A}_J\overrightarrow{\mathbf{1}}). \tag{3.21}$$

[1]One way to see this is noting that eigenvalues of the matrix are strictly positive due to positivity of h. See also equation (3.20).

This is one of the main equations that is used throughout this chapter. Equation (3.21) describes an implicit equation for the trajectory of optimal argument $\mathbf{s}^*$ as function of τ and μ. Note that $\mathbf{A}_J$, σ_i's, w_i's and v_i's depend on the set of violations of $\mathbf{s}^*$, $J(\mathbf{s}^*)$. Also equation (3.21) is valid for $0 < \tau$ and $0 \leq \mu$ (since the matrix in equation (3.19) must be invertible).

At first glance, equation (3.21) might not seem very helpful, as it is an implicit equation. However, many useful properties can be inferred from this equation. What follows is not very practical to implement; however, it is useful in deriving some theoretical results. The TCCS algorithm inspired by these ideas is presented in Section 3.3.

For every set J, τ and μ, write the implicit equation

$$\mathbf{s} = \frac{|\mathbf{s}|}{\tau h(|\mathbf{s}|)}(\sum_i w_i w_i^T)\mathbf{b} + \sum_i \frac{1}{1 + \frac{\tau}{\mu\sigma_i^2} h(|\mathbf{s}|)/|\mathbf{s}|} \frac{v_i v_i^T}{\sigma_i^2}(\mathbf{b}/\mu + \mathbf{A}_J \overrightarrow{\mathbf{1}}). \qquad (3.22)$$

However, note that $\mathbf{s}$ in the above equation is not necessarily the optimal solution because in equation (3.21), the set of violations J (i.e. consequently $\mathbf{A}_J^T$, σ_i's, w_i's and v_i's) implicitly depends on τ and μ.

For every set J and $i \in \{1, \ldots, n\}$, let $\Theta_i(J)$ be the set of τ of positive real numbers for which there exist $\mathbf{s}$ that satisfies both equation (3.22) and $a_i^T \mathbf{s} = 1$. The importance of set $\Theta_i(J)$ will be discussed shortly. However, we first show the following lemma:

Lemma 3.2.3 *For fixed μ, J and i, the set $\Theta_i(J)$ consists of a finite number of intervals in $\mathbb{R}$.*

Proof: First take the Euclidean norm of both sides of equation (3.22) and simplify. Orthogonality of the basis $\{w_1, \ldots, w_{m-k}, v_1, \ldots, v_k\}$ considerably simplifies the expression. Perform a series of algebraic manipulations (depending on the form of function h), to reduce the expression into the form

$$P(|\mathbf{s}|, \tau) = 0, \qquad (3.23)$$

where P is a polynomial in two variables.

Next suppose that $\mathbf{s}$ satisfies $a_i^T\mathbf{s} = 1$. Multiply both side of equation (3.22) by a_i^T. The LHS equals to 1. Again, perform a series of algebraic manipulation to reduce the expression into the form

$$Q(|\mathbf{s}|, \tau) = 0, \tag{3.24}$$

where Q is a polynomial. Thus if τ is chosen such that $\mathbf{s}$ satisfies (3.22) and $a_i^T\mathbf{s} = 1$, then $|\mathbf{s}|$ and τ must satisfy both (3.23) and (3.24). As it is well known (and goes by the name Bezout's theorem), polynomials P and Q have finite number of intersections, unless they have a common component. In either case the set of τ for which there exist $|\mathbf{s}|$ such that pair $(|\mathbf{s}|, \tau)$ satisfies both (3.23) and (3.24) consist of a finite number of intervals in $\mathbb{R}$. The result follows. ■

Let us now move on to partition $\mathbb{R}^m$ into regions where the points of each region have the same set of violations. Clearly there are a finite number of regions (although many of them). Now going back to equation (3.21) recall that $\mathbf{A}_J^T$, σ_i's, w_i's and v_i's depend on the set of violations of $\mathbf{s}^*$. Fix μ and think of $\mathbf{s}^*$ as a function of τ (to ease the notation we use $\mathbf{s}^*(\tau)$ in place of $\mathbf{s}^*(\tau, \mu)$ and $J(\tau)$ in place of $J(\mathbf{s}^*(\tau, \mu))$. Let τ_0 be sufficiently large so that $\mathbf{s}^*(\tau_0)$ is close to the origin and $J(\tau_0)$ is empty. Set $\tau = \tau_0$.

Since the set of violations is empty, equation (3.21) implies that

$$\mathbf{s}^*(\tau) = \frac{|\mathbf{s}^*(\tau)|}{\tau h(|\mathbf{s}^*(\tau)|)}\mathbf{b}. \tag{3.25}$$

The above equation describes trajectory for $\mathbf{s}^*(\tau)$ in the region where the set of violations of the optimal argument is empty.

Next, slowly decrease τ from τ_0. As $\tau \to t$, by continuity $\mathbf{s}^*(\tau) \to \mathbf{s}^*(t)$. Equation (3.25) continues to hold until the set of violations of $\mathbf{s}^*(\tau)$ changes; that is, $\mathbf{s}^*(\tau)$ enters into a new region. Let τ_1 denote the largest τ for which this happens. Continue this process.

Suppose $\tau = \tau_r$ for some $r = 1, 2, \ldots$. Again by continuity, for $\tau < \tau_r$, $\mathbf{s}^*(\tau)$ follows a trajectory prescribed by equation (3.21) (using $J(\tau_r)$ in place of J), until the set of violations of $\mathbf{s}^*(\tau)$ changes; that is $\mathbf{s}^*(\tau)$ enters into another region at $\tau = \tau_{r+1}$. Therefore, as τ decreases from τ_r, τ_{r+1} is the first τ for which $\mathbf{s}^*(\tau)$ passes through a plane given by $\{\mathbf{s} : a_k^T \mathbf{s} = 1\}$ for some $k \in \{1, \ldots, n\}$. Thus, τ_{r+1} is equal to the largest element smaller than τ_r that belongs to $\Theta_k(J(\tau_r))$ for some k, where τ_r and τ_{r+1} are not on the same interval in $\Theta_k(J(\tau_r))$ (i.e. If τ_r and τ_{r+1} are on the same interval of $\Theta_k(J(\tau_r))$, it means that for $\tau_{r+1} \leq \tau \leq \tau_r$, $\mathbf{s}^*(\tau)$ lies on the plane given by $\{\mathbf{s} : a_k^T \mathbf{s} = 1\}$, consequently, the set of violations of $\mathbf{s}^*(\tau)$ does not change due to k). This process is continued until τ reaches t.

The important observation is that because of Lemma 3.2.8 and since there are finite number of regions (i.e. different set of violations) and indices i, the above process does not continue indefinitely. Therefore, there exist a sequence $\tau_0 > \tau_1 > \cdots > \tau_N > \tau_{N+1} = 0$, where for $\tau \in (\tau_{i+1}, \tau_i]$ $\mathbf{s}^*(\tau)$ lies in the same region.

The above observations are used to prove several results.

Theorem 3.2.4 *For every μ there exist $t_c > 0$ and a matrix $\mathbf{A}_J$ with rank m such that for $0 < \tau \leq t_c$*

$$\mathbf{s}^*(\tau, \mu) = \sum_i \frac{1}{1 + \frac{\tau}{\mu\sigma_i^2} h(|\mathbf{s}^*|)/|\mathbf{s}^*|} \frac{v_i v_i^T}{\sigma_i^2} (\mathbf{b}/\mu + \mathbf{A}_J \overrightarrow{\mathbf{1}}). \tag{3.26}$$

Proof: In the discussions above, set $t_c = \tau_N$ and $\mathbf{A}_J = \mathbf{A}_{J(\tau_N)}$. Then equation (3.21) yields that for $\tau \in (0, t_c]$,

$$\mathbf{s}^*(\tau, \mu) = \frac{|\mathbf{s}^*|}{\tau h(|\mathbf{s}^*|)} (\sum_i^{m-k} w_i w_i^T)\mathbf{b} + \sum_i^k \frac{1}{1 + \frac{\tau}{\mu\sigma_i^2} h(|\mathbf{s}^*|)/|\mathbf{s}^*|} \frac{v_i v_i^T}{\sigma_i^2} (\mathbf{b}/\mu + \mathbf{A}_J \overrightarrow{\mathbf{1}}),$$

where $\mathbf{A}_J$, w_i's, v_i's and σ_i's are fixed. However, we know that $\mathbf{s}^*(\tau, \mu)$ is bounded as $\tau \to 0$. Therefore, from assumption 5 about function h, $h(|\mathbf{s}^*|)$ is bounded as $\tau \to 0$ and unless $(\sum_i^{m-k} w_i w_i^T)\mathbf{b} = 0$, the first summand in the above equation

would blow up as τ approaches 0; which yields a contradiction. Thus it musty be the case that $k = m$; that is, $\mathbf{A}_J^T$ has m nonzero singular values, which implies that $\mathbf{A}_J$ has rank m. The result follows. ∎

Remark 3.2.5 *The values of t_c and matrix $\mathbf{A}_J$ in Theorem 3.2.4 are μ dependent.*

Theorem 3.2.6 *For μ sufficiently large, $\mu \times \mathcal{S}(\mathbf{A}^T\mathbf{s}^*(0,\mu), 1)$ is independent of μ.*

Proof: The result of Theorem 3.2.4 implies that as τ approaches 0, the optimal argument $\mathbf{s}^*(\tau,\mu)$ eventually satisfies equation (3.26). Although at $\tau = 0$ there might not be a unique optimal argument $\mathbf{s}^*$; however, from equation (3.26), one of them is given by

$$\mathbf{s}^*(0,\mu) = \sum_i \frac{v_i v_i^T}{\sigma_i^2}(\mathbf{b}/\mu + \mathbf{A}_J \overrightarrow{\mathbf{1}}), \tag{3.27}$$

where $J = J(\mathbf{s}^*(0,\mu))$ denote the set of violations of $\mathbf{s}^*(0,\mu)$. Recall that $\mathbf{A}_J$, v_i's and σ_i's depend on the value of μ. Next think of μ as variable. As μ increases, $\mathbf{s}^*(0,\mu)$ follows a trajectory given by (3.27). Using a similar argument to the one used for decreasing τ shows that there exist μ_c such that for $\mu \geq \mu_c$, $\mathbf{s}^*(0,\mu)$ stays in the same region.

Let $\mu \geq \mu_c$. Observe that $\mathbf{s}^*(0,\mu)$ still satisfies equation (3.27); however, $\mathbf{A}_J$, v_i's and σ_i's no longer depend on μ and are fixed. From the definition of the violation set J, we can conclude that the only rows of $\mathcal{S}(\mathbf{A}^T\mathbf{s}^*(t,\mu), \overrightarrow{\mathbf{1}})$ that are nonzero are exactly the elements of J. Thus, it suffices to show the result for $\mu \times \mathcal{S}(\mathbf{A}_J^T\mathbf{s}^*(0,\mu), 1)$.

Now recall that $\mathbf{A}_J^T = \sum_j u_j \sigma_j v_j^T$. Thus

$$\mathbf{A}_J^T\mathbf{s}^*(0,\mu) = (\sum_j u_j\sigma_j v_j^T)(\sum_i \frac{v_i v_i^T}{\sigma_i^2}(\mathbf{b}/\mu + \mathbf{A}_J\overrightarrow{\mathbf{1}})) = \sum_i \frac{u_i v_i^T}{\sigma_i}(\frac{\mathbf{b}}{\mu} + \mathbf{A}_J\overrightarrow{\mathbf{1}}). \tag{3.28}$$

In the limiting case when μ approaches ∞, $\mathbf{s}^*(0,\mu)$ approaches to the boundary of Ω. Indeed, when $\mu = \infty$, $\mathbf{s}^*(0,\infty)$ is the optimal solution of the linear

programming problem

$$\underset{\mathbf{s}}{\text{argmax}} \ \langle \mathbf{b}, \mathbf{s} \rangle \quad \text{subject to} \quad \mathbf{A}^T \mathbf{s} \leq \overrightarrow{\mathbf{1}}. \tag{3.29}$$

Therefore, $\mathbf{A}_J^T \mathbf{s}^*(0, \infty) = \overrightarrow{\mathbf{1}}$. By formally substituting ∞ in equation (3.28), we conclude that $\overrightarrow{\mathbf{1}} = \sum_i \frac{u_i v_i^T}{\sigma_i} \mathbf{A}_J \overrightarrow{\mathbf{1}}$. Substituting this back into the above equation, implies that

$$\mathbf{A}_J^T \mathbf{s}^*(0, \mu) = (\sum_i \frac{u_i v_i^T}{\sigma_i}) \frac{\mathbf{b}}{\mu} + \overrightarrow{\mathbf{1}}.$$

Recall that by the definition of violation set J, entries of $\mathbf{A}_J^T \mathbf{s}^*(0, \mu)$ are greater or equal to one. Therefore, entries of $(\sum_i \frac{u_i v_i^T}{\sigma_i}) \frac{\mathbf{b}}{\mu}$ are greater or equal to zero. Hence,

$$\mu \times \mathcal{S}(\mathbf{A}_J^T \mathbf{s}^*(0, \mu), 1) = (\sum_i \frac{u_i v_i^T}{\sigma_i}) \mathbf{b}.$$

Since the right hand side is the same for all $\mu \geq \mu_c$, the result follows. Indeed, one can easily show that $\sum_i \frac{u_i v_i^T}{\sigma_i}$ is the pseudo-inverse of $\mathbf{A}_J$, $\mathbf{A}_J^\dagger$.

As a bonus, we also have that for sufficiently large μ, the nonzero elements of

$$\mu \times \mathcal{S}(\mathbf{A}^T \mathbf{s}^*(0, \mu), 1)$$

are equal to $\mathbf{A}_J^\dagger \mathbf{b}$, where J is the set of active constraints for the optimal solution of the linear programming problem (3.29). ■

The results of the above theorem shows the exact regularization property described in [70] and [27].

Corollary 3.2.7 *There exist fixed large μ_c such that for $\mu \geq \mu_c$, the solution to problem* (3.8) *is independent of μ. Indeed, let J denote the set of active constraints of the maximizer of the linear programming problem*

$$\underset{\mathbf{s}}{\text{argmax}} \ \langle \mathbf{b}, \mathbf{s} \rangle \quad \textit{subject to} \quad \tilde{A}^T \mathbf{s} \leq \overrightarrow{\mathbf{1}},$$

where $\tilde{A}$ is given by (3.16)*. Let $\tilde{y}$ be a column vector with $2n$ entries where n is the number of columns of A. Moreover, those entries of $\tilde{y}$ whose index belong to J*

are given by $\tilde{y}(J) = \tilde{\mathbf{A}}_J^{\dagger}\mathbf{b}$*, and the other entries of* $\tilde{y}$ *are zero. Then the minimizer of problem* (3.8) *is given by*

$$\begin{bmatrix} \mathbf{I} & | & -\mathbf{I} \end{bmatrix} \tilde{y}.$$

Proof: The Lagrangian dual correspondence in Section 3.1 shows that the solution of (3.8) is equal to $\mu \cdot \mathsf{shrink}(\mathbf{A}^T\mathbf{s}^*, \overrightarrow{\mathbf{1}})$ where $\mathbf{s}^*$ is the solution of (3.13) with $t = 0$. Now use the result of Theorem 3.2.6 and the relation (3.17). ∎

3.2.2 Constrained dual problem

This section analyzes the solution to the constrained problem (3.14). This problem can be viewed as the limiting case of problem (3.15) as μ approaches ∞. Many of the ideas used in this section are similar to the previous section. However, the formulas in this case become simpler and the results enables us to devise a fast algorithm for finding the optimal argument.

Let

$$g_\tau(\mathbf{s}) = \langle \mathbf{b}, \mathbf{s} \rangle - \tau H(|\mathbf{s}|),$$

and set

$$\mathbf{s}_{opt}(\tau) = \underset{\mathbf{s}}{\operatorname{argmax}}\, g_\tau(\mathbf{s}) \qquad \text{subject to} \qquad \mathbf{A}^T\mathbf{s} \leq \overrightarrow{\mathbf{1}}.$$

As in previous section, for any subset J of $\{1, \ldots, n\}$, let $\mathbf{A}_J^T$ denotes the matrix formed from rows of $\mathbf{A}^T$ whose index belong to J. The following lemma is essential in the analysis done in this subsection:

Lemma 3.2.8 *Suppose* $\tau > 0$ *and* J *is a fixed subset of* $\{1, \ldots, n\}$*. If function* $g_\tau(\mathbf{s})$ *has unique global max on the set* $\{\mathbf{s} : \mathbf{A}_J^T\mathbf{s} = \overrightarrow{\mathbf{1}}\}$*, then this global maximum is given by*

$$\mathbf{s}_J(\tau) := \underset{\mathbf{s}:\mathbf{A}_J^T\mathbf{s}=\overrightarrow{\mathbf{1}}}{\operatorname{argmax}}\, g_\tau(\mathbf{s}) = F(\tau, J)\overrightarrow{\alpha}(J) + \overrightarrow{\beta}(J) \tag{3.30}$$

where $\overrightarrow{\alpha}(J) := \mathbf{b} - \mathbf{A}_J(\mathbf{A}_J^T\mathbf{A}_J)^\dagger\mathbf{A}_J^T\mathbf{b}$ *is the projection of* $\mathbf{b}$ *on the space* $\{\mathbf{s} : \mathbf{A}_J^T\mathbf{s} = \overrightarrow{\mathbf{1}}\}$*, and* $\overrightarrow{\beta}(J) := \mathbf{A}_J(\mathbf{A}_J^T\mathbf{A}_J)^\dagger\overrightarrow{\mathbf{1}}$*. Here,* F *is some function that is determined by values of* τ*, the set* J *and the function* h*.*

Proof: Since the set $\{\mathbf{s} : \mathbf{A}_J^T\mathbf{s} = \overrightarrow{\mathbf{1}}\}$ is convex and g_τ is a concave function that has a global maximum on this set, it suffices to solve the first order conditions for the Lagrangian

$$\mathcal{L}(\mathbf{s}, \overrightarrow{\lambda}) = \langle \mathbf{b}, \mathbf{s} \rangle - \tau H(|\mathbf{s}|) + \overrightarrow{\lambda}^T(\mathbf{A}_J^T\mathbf{s} - \overrightarrow{\mathbf{1}})$$

to find the global maximum.

First order conditions with respect to $\mathbf{s}$ imply that

$$\mathbf{b} - \tau\frac{h(|\mathbf{s}_J|)}{|\mathbf{s}_J|}\mathbf{s}_J + \mathbf{A}_J\overrightarrow{\lambda}^* = 0, \tag{3.31}$$

and first order conditions with respect to $\overrightarrow{\lambda}$ imply that

$$\mathbf{A}_J^T\mathbf{s}_J = \overrightarrow{\mathbf{1}}. \tag{3.32}$$

Multiplying both sides of (3.31) by $\mathbf{A}_J^T$, using (3.32), and rearranging yields that

$$\overrightarrow{\lambda}^* = (\mathbf{A}_J^T\mathbf{A}_J)^\dagger\left(\tau\frac{h(|\mathbf{s}_J|)}{|\mathbf{s}_J|}\overrightarrow{\mathbf{1}} - A_J^T\mathbf{b}\right).$$

Substituting the above into (3.31) and simplifying, yields that

$$\mathbf{s}_J = \frac{|\mathbf{s}_J|}{\tau h(|\mathbf{s}_J|)}\overrightarrow{\alpha} + \overrightarrow{\beta}, \tag{3.33}$$

where $\overrightarrow{\alpha} = \mathbf{b} - \mathbf{A}_J(\mathbf{A}_J^T\mathbf{A}_J)^\dagger\mathbf{A}_J^T\mathbf{b}$ and $\overrightarrow{\beta} = \mathbf{A}_J(\mathbf{A}_J^T\mathbf{A}_J)^\dagger\overrightarrow{\mathbf{1}}$. It is straightforward to verify that $\overrightarrow{\alpha}$ is the projection of vector $\mathbf{b}$ onto the space $\{\mathbf{s} : \mathbf{A}_J^T\mathbf{s} = 1\}$, and $\overrightarrow{\alpha}$ and $\overrightarrow{\beta}$ are perpendicular to each other. Now taking Euclidean norm square of (3.33), implies that

$$|\mathbf{s}_J|^2 = \frac{|\mathbf{s}_J|^2}{\tau^2 h(|\mathbf{s}_J|)^2}|\overrightarrow{\alpha}|^2 + |\overrightarrow{\beta}|^2.$$

The above equation and properties of function h, yield that $|\mathbf{s}_J|$ is the root of some polynomial that depends on τ and J. Substituting these roots into the RHS of

equation (3.33) yields an explicit formula for $\mathbf{s}_J$. Moreover, since it is assumed that $g_\tau(\mathbf{s})$ has global maximum on the set (and global maximum is unique for $\tau > 0$ due to strict concavity), it can be concluded that J and τ uniquely determine the value of $\frac{|\mathbf{s}_J|}{\tau h(|\mathbf{s}_J|)}$. It follows that,

$$\mathbf{s}_J(\tau) = F(\tau, J)\overrightarrow{\alpha}(J) + \overrightarrow{\beta}(J).$$

■

Example 3.2.9 *In the case* $H(|\mathbf{s}|) = |\mathbf{s}|^2/2$*:*

$$\mathbf{s}_J(\tau) = \frac{1}{\tau}\overrightarrow{\alpha}(J) + \overrightarrow{\beta}(J).$$

Example 3.2.10 *In the case* $H(|\mathbf{s}|) = \sqrt{1 + |\mathbf{s}|^2}$*:*

$$\mathbf{s}_J(\tau) = \sqrt{\frac{1 + |\overrightarrow{\beta}(J)|^2}{\tau^2 - |\overrightarrow{\alpha}(J)|^2}}\overrightarrow{\alpha}(J) + \overrightarrow{\beta}(J).$$

Next we analyze the path that the optimal argument $\mathbf{s}_{opt}(\tau)$ traverse as τ decreases. When τ is sufficiently large, $\mathbf{s}_{opt}(\tau)$ lies in the interior of Ω. To find $\mathbf{s}_{opt}(\tau)$, gradient of $g_\tau(\mathbf{s})$ is set equal to zero:

$$\mathbf{b} - \tau\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\mathbf{s}_{opt} = 0.$$

Taking norm from both sides and simplifying yields that

$$\mathbf{s}_{opt}(\tau) = \frac{1}{|\mathbf{b}|}h^{-1}\left(\frac{|\mathbf{b}|}{\tau}\right)\mathbf{b} = F(\tau, \emptyset)\alpha(\emptyset). \tag{3.34}$$

The second equality in the above equation comes from $\mathbf{b} = \overrightarrow{\alpha}(\emptyset)$. It is written here to highlight that the above equation is a special case of equation (3.30) when J is the empty set.

Now decrease τ slowly. Note that $\mathbf{s}_{opt}(\tau)$ follows the path given by equation (3.34) until it hits a boundary of polytope Ω. Let τ_1 denote the first time that this happens. As a consequence, for $t \leq \tau \leq \tau_1$, the global maximum of $g_\tau(\mathbf{s})$

occurs outside of Ω if there was no constraints. Hence, because g_τ is a concave function and Ω is a convex set, $\mathbf{s}_{opt}(\tau)$ must be on a boundary of Ω for $\tau \leq \tau_1$.

Let J_1 denote the set of indices i for which $a_i^T \mathbf{s}_{opt}(\tau_1) = 1$. In the language of linear programming, J_1 is called the set of active constraints for point $\mathbf{s}_{opt}(\tau_1)$. From the discussion above, as τ decreases from τ_1, $\mathbf{s}_{opt}(\tau)$ moves along faces of Ω. Suppose the set of constraints K_1 determines the first face of Ω along which $\mathbf{s}_{opt}(\tau)$ moves for $\tau \leq \tau_1$. Clearly $K_1 \subset J_1$. From Lemma 3.2.8, for $\tau \leq \tau_1$, $\mathbf{s}_{opt}(\tau)$ moves along the path given by

$$F(\tau, K_1)\overrightarrow{\alpha}(K_1) + \overrightarrow{\beta}(K_1),$$

until it hits another face of Ω. Let τ_2 denote the first time that this happens; that is, τ_2 is the largest $\tau < \tau_1$ for which the set of active constrained of $\mathbf{s}_{opt}(\tau)$ is different than K_1. Let J_2 denote the set of active constraints of $\mathbf{s}_{opt}(\tau_2)$. Again, as τ decreases from τ_2, $\mathbf{s}_{opt}(\tau)$ moves along a face of Ω. Let K_2 denote the set of constraints that determines this face. Again, $K_2 \subset J_2$, and repeat as before.

The above process is continued until τ reaches t. From the arguments in Section 3.2.1, there exist a sequence

$$\tau_0 > \tau_1 > \cdots > \tau_N > \tau_{N+1} = 0, \tag{3.35}$$

where for $\tau \in (\tau_{i+1}, \tau_i]$, $\mathbf{s}_{opt}(\tau)$ lies on the same face of the polytope. Therefore, $\mathbf{s}_{opt}(\tau)$ moves along finite number of faces until τ becomes equal to t. As mentioned earlier, the constrained problem can be viewed as the limiting case of an unconstrained problem as μ approaches ∞. This insight and Theorem 3.2.4, yield the following corollary:

Corollary 3.2.11 *There exist $t_c > 0$, such that for $0 < \tau \leq t_c$ the optimal argument $\mathbf{s}_{opt}(\tau)$ for constrained problem*

$$\text{argmax}\{\langle \mathbf{b}, \mathbf{s} \rangle - \tau H(|\mathbf{s}|)\} \quad \textit{subject to } \mathbf{A}^T \mathbf{s} \leq \overrightarrow{\mathbf{1}},$$

are all the same and lie on an extreme point of the polytope Ω.

Proof: Formally substituting $\mu = \infty$ in equation (3.26) of Theorem 3.2.4, we observe that for $0 < \tau \leq t_c$, $\mathbf{s}_{opt}(\tau)$ is independent of τ. Furthermore, since $\mathbf{A}_{J(t_c)}$ has rank m, $\mathbf{s}_{opt}(t_c)$ is an extreme point of polytope Ω.

3.2.3 Approximate solution of the augmented problem

This subsection describes how $\mathbf{s}_{opt}(t)$ can be used to find an approximation for $y^*(t,\mu) = \mu\mathcal{S}(\mathbf{A}^T\mathbf{s}^*(t,\mu), \overrightarrow{\mathbf{1}})$ when μ is sufficiently large. The relationship is established using first order approximation in terms of $1/\mu$.

Note that formally, $\mathbf{s}_{opt}(t) = \mathbf{s}^*(t,\infty)$. For ease of notation, $\mathbf{s}^*$ and $\mathbf{s}_{opt}$ are used occasionally in place of $\mathbf{s}^*(t,\mu)$ and $\mathbf{s}_{opt}(t)$, respectively, in the remainder of this subsection. Using arguments similar to the one used in proof of Theorem 3.2.6, show that there exist μ_c such that for $\mu \geq \mu_c$, $\mathbf{s}^*(t,\mu)$ and $\mathbf{s}_{opt}$ have the same set of violations[2]. Suppose $\mu > \mu_c$. Assume that $\mathbf{s}_{opt}$ does not lie in the interior of Ω as in that case $y^*(t,\mu) = \vec{0}$ and there is nothing to show.

Assume that we can perform Taylor expansions,

$$\mathbf{s}^*(t,\mu) = \mathbf{s}_{opt} + \frac{1}{\mu}\Psi + O(1/\mu^2), \tag{3.36}$$

and

$$\frac{|\mathbf{s}^*(t,\mu)|}{h(|\mathbf{s}^*(t,\mu)|)} = \frac{|\mathbf{s}_{opt}|}{h(|\mathbf{s}_{opt}|)} + \frac{1}{\mu}\Gamma + O(1/\mu^2). \tag{3.37}$$

[2]Even though for $\mu \geq \mu_c$ the set of violations of $\mathbf{s}^*(t,\mu)$ are the same, $\mathbf{s}_{opt}$ is a limit point of sequence $\mathbf{s}^*(t,\mu)$ as $\mu \to \infty$, and might have a larger set of violations. However here, as we increase μ, the problem becomes more constrained and therefore it is expected that the set of violations to decrease. So it is assumed that $\mathbf{s}_{opt}$ and $\mathbf{s}^*(t,\mu)$, for $\mu \geq \mu_c$, have the same set of violations.

Recall from equation (3.21) that,

$$\begin{aligned}\mathbf{s}^* &= \frac{|\mathbf{s}^*|}{th(|\mathbf{s}^*|)}(\sum_{i=1}^{m-k} w_i w_i^T)\mathbf{b} + \sum_{i=1}^{k} \frac{1}{1+\frac{t}{\mu\sigma_i^2}h(|\mathbf{s}^*|)/|\mathbf{s}^*|}\frac{v_i v_i^T}{\sigma_i^2}(\mathbf{b}/\mu + \mathbf{A}_J\overrightarrow{\mathbf{1}}) \\ &= \frac{|\mathbf{s}^*|}{th(|\mathbf{s}^*|)}(\sum_{i=1}^{m-k} w_i w_i^T)\mathbf{b} + \sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^2}\mathbf{A}_J\overrightarrow{\mathbf{1}} + \frac{1}{\mu}\sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^2}\mathbf{b} \\ &\qquad - \frac{t}{\mu}\frac{h(|\mathbf{s}^*|)}{|\mathbf{s}^*|}\sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^4}\mathbf{A}_J\overrightarrow{\mathbf{1}} + O(1/\mu^2)\end{aligned}$$

Substituting (3.36) and (3.37) into the above expression, and equating coefficients of $1/\mu$ on both sides of the equality yields that

$$\Psi = \frac{1}{t}\Gamma\sum_{i=1}^{m-k} w_i w_i^T\mathbf{b} + \sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^2}\mathbf{b} - t\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^4}\mathbf{A}_J\overrightarrow{\mathbf{1}}.$$

Multiplying both sides with $\mathbf{A}_J^T = \sum_i u_i\sigma_i v_i^T$ and using orthogonality of v_i's to w_i's, we have

$$\mathbf{A}_J^T\Psi = \mathbf{A}_J^T\sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^2}\mathbf{b} - t\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\mathbf{A}_J^T\sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^4}\mathbf{A}_J\overrightarrow{\mathbf{1}}.$$

From definition of violation set J, one concludes that the only rows of $\mathcal{S}(\mathbf{A}^T\mathbf{s}^*, \overrightarrow{\mathbf{1}})$ that are nonzero are exactly the elements of J. Therefore, it suffices to evaluate $\mathcal{S}(\mathbf{A}_J^T\mathbf{s}^*, \overrightarrow{\mathbf{1}})$. Furthermore, because $\mathbf{s}_{opt}$ is on the boundary of Ω and J is the set of violations of both $\mathbf{s}_{opt}$ and $\mathbf{s}^*$,

$$\mathbf{A}_J^T\mathbf{s}_{opt} = \overrightarrow{\mathbf{1}}, \qquad \text{and} \qquad \mathbf{A}_J^T\mathbf{s}^* \geq \overrightarrow{\mathbf{1}}.$$

Putting all these together,

$$\begin{aligned}\vec{0} \leq \mathbf{A}_J^T\mathbf{s}^* - \overrightarrow{\mathbf{1}} &= \mathbf{A}_J^T(\mathbf{s}_{opt} + \frac{1}{\mu}\Psi + O(1/\mu^2)) - \overrightarrow{\mathbf{1}} = \\ &= \frac{1}{\mu}\mathbf{A}_J^T\sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^2}\mathbf{b} - \frac{t}{\mu}\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\mathbf{A}_J^T\sum_{i=1}^{k}\frac{v_i v_i^T}{\sigma_i^4}\mathbf{A}_J\overrightarrow{\mathbf{1}} + O(1/\mu^2).\end{aligned}$$

Hence, nonzero elements of $y^*(t,\mu)$, for $\mu \geq \mu_c$, are given by

$$\begin{aligned}\mu\mathcal{S}(\mathbf{A}_J^T\mathbf{s}^*(t,\mu), \overrightarrow{\mathbf{1}}) &= \mathbf{A}_J^T \sum_{i=1}^{k} \frac{v_i v_i^T}{\sigma_i^2}\mathbf{b} - t\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\mathbf{A}_J^T \sum_{i=1}^{k} \frac{v_i v_i^T}{\sigma_i^4}\mathbf{A}_J \overrightarrow{\mathbf{1}} + O(1/\mu) \\ &= \mathbf{A}_J^T(\mathbf{A}_J\mathbf{A}_J^T)^\dagger\mathbf{b} - t\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\mathbf{A}_J^T((\mathbf{A}_J\mathbf{A}_J^T)^\dagger)^2\mathbf{A}_J \overrightarrow{\mathbf{1}} + O(1/\mu) \\ &= \mathbf{A}_J^\dagger\mathbf{b} - t\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\mathbf{A}_J^\dagger(\mathbf{A}_J^T)^\dagger \overrightarrow{\mathbf{1}} + O(1/\mu),\end{aligned}$$

where properties of the pseudo-inverse were used for the last equality.

Corollary 3.2.12 *In problem* (3.2)*, for each* t*, there exist fixed large* μ_c *such that for* $\mu \geq \mu_c$*, the solution to the corresponding problem is given as follows:*

Let J *denote the set of violations of the maximizer of*

$$\mathbf{s}_{opt} = \underset{\mathbf{s}}{\operatorname{argmax}} \{\langle \mathbf{b}, \mathbf{s}\rangle - tH(|\mathbf{s}|)\} \qquad \textit{subject to} \qquad \tilde{\mathbf{A}}^T\mathbf{s} \leq \overrightarrow{\mathbf{1}},$$

where $\tilde{\mathbf{A}}$ *is given by* (3.16)*. Let* $\tilde{y}$ *be a column vector with* $2n$ *entries where* n *is the number of columns of* $\mathbf{A}$*. Moreover, those entries of* $\tilde{y}$ *whose index belong to* J *are given by*

$$\tilde{y}(J) = \tilde{\mathbf{A}}_J^\dagger\mathbf{b} - t\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\tilde{\mathbf{A}}_J^\dagger(\tilde{\mathbf{A}}_J^T)^\dagger \overrightarrow{\mathbf{1}} + O(1/\mu),$$

and the other entries of $\tilde{y}$ *are zero. Then the minimizer of problem* (3.2) *is given by*

$$y^*(t,\mu) = \begin{bmatrix}\mathbf{I} & | & -\mathbf{I}\end{bmatrix}\tilde{y}.$$

Proof: The Lagrangian dual correspondence in Section 3.1 shows that the solution of problem (3.2) is equal to $\mu \cdot \mathsf{shrink}(\mathbf{A}^T\mathbf{s}^*, \overrightarrow{\mathbf{1}})$, where $\mathbf{s}^*$ is the solution of (3.13). Now use the result of this section and the relation (3.17). ∎

3.3 The TCCS Algorithm

This section describes the TCCS algorithm. The TCCS algorithm consists of two parts. The first part of the algorithm finds an approximate solution to the

constrained problem (3.14). This part is described in Subsection 3.3.1. The second part of the algorithm takes the result of the first part of the algorithm and then uses the result developed in Subsection 3.2.3 to yield an approximate solution to problem (3.5). This part of the algorithm is described in Subsection 3.3.2. Pseudo-codes for the TCCS algorithm are provided in Appendix B.

3.3.1 First part

For the first part of the TCCS algorithm we devise a fast algorithm that gives an approximate solution for constrained problem (3.14). Other methods could be used for this part of the TCCS algorithm. Nevertheless, we found the algorithm described here to be effective and fast for optimizing constrained problems of form (3.14).

The idea of the algorithm is to trace out an approximation to the path of $\mathbf{s}_{opt}(\tau)$, as τ decreases, which was described in Section 3.2.2. Denote the approximate path by $\tilde{\mathbf{s}}(\tau)$. To construct this path, one proceeds very similarly to the way the trajectory $\mathbf{s}_{opt}(\tau)$ was determined at the end of Section 3.2.2. The only difference is that whenever $\tilde{\mathbf{s}}(\tau)$ hits a new face of polytope Ω, say at τ_i, J_i is used as proxy for K_i and proceed. As a result, the trajectory of $\tilde{\mathbf{s}}(\tau)$ might be different than $\mathbf{s}_{opt}(\tau)$. This approximation is made to avoid cumbersome computations. The details of the algorithm are provided below.

The algorithm is initialized by finding $\tilde{\tau}_0$ large enough so that

$$\frac{1}{|\mathbf{b}|} h^{-1}\left(\frac{|\mathbf{b}|}{\tilde{\tau}_0}\right)\mathbf{b}$$

lies in the interior of Ω. Set $\tilde{\mathbf{s}}(\tilde{\tau}_0)$ to be equal to the above expression. Set $\tilde{J} = \emptyset$ and $\tau = \tilde{\tau}_0$. This completes the initialization of the algorithm. Next the algorithm goes through a loop as described below.

Suppose $\tau = \tilde{\tau}_r$, and $\tilde{J} = \tilde{J}_r$ denotes the set of active constraints for $\tilde{\mathbf{s}}(\tilde{\tau}_r)$. Note that $a_i^T \overrightarrow{\alpha}(\tilde{J}_r) = 0$ for $i \in \tilde{J}_r$, because $\alpha(\tilde{J}_r)$ is the projection of $\mathbf{b}$ on the

space $\{\mathbf{s} : \mathbf{A}_{\tilde{J}_r}^T \mathbf{s} = 1\}$. Therefore, when τ is decreasing from $\tilde{\tau}_r$ and $\tilde{\mathbf{s}}(\tau)$ is moving along $\overrightarrow{\alpha}(\tilde{J}_r)$, it is still the case that $a_i^T \tilde{\mathbf{s}}(\tau) = 1$ for $i \in \tilde{J}_r$. Thus, constraints given by $\tilde{J}_r$ continue to be active for $\tilde{\mathbf{s}}(\tau)$.

Let L be the complement of $\tilde{J}_r$; that is $L = \{1, \ldots, n\} \setminus \tilde{J}_r$. Let $\tilde{\tau}_{r+1}$ denote the largest τ smaller than $\tilde{\tau}_r$ such that the set of active constraints of

$$\tilde{\mathbf{s}}(\tau) = F(\tau, \tilde{J}_r)\overrightarrow{\alpha}(\tilde{J}_r) + \overrightarrow{\beta}(\tilde{J}_r) = \tilde{\mathbf{s}}(\tilde{\tau}_r) + \Big(F(\tau, \tilde{J}_r) - F(\tilde{\tau}_r, \tilde{J}_r)\Big)\overrightarrow{\alpha}(\tilde{J}_r) \quad (3.38)$$

is different than $\tilde{J}_r$. From the discussion above, $\tilde{\tau}_{r+1}$ is the largest τ less than $\tilde{\tau}_r$ so that for some $i \in L$, $a_i^T \tilde{\mathbf{s}}(\tau) = 1$, where $\tilde{\mathbf{s}}(\tau)$ is given by equation (3.38). Finding $\tilde{\tau}_{r+1}$ is problem specific and depends on function $F(\tau, J)$. However, for many examples of H, $\tilde{\tau}_r$ can be found, either analytically or numerically, very efficiently. Set $\tau = \tilde{\tau}_{r+1}$ and $\tilde{J} = \tilde{J}_{r+1}$, and repeat the above process.

In this way the sequence $\tilde{\tau}_0 > \tilde{\tau}_1 > \cdots$ is generated. In general this sequence is different than the sequence (3.35), except at $\tilde{\tau}_0 = \tau_0$ and $\tilde{\tau}_1 = \tau_1$. Also note that, at each step of the loop the rank of $\mathbf{A}_{\tilde{J}}^T$ strictly increases. Because of this, at $\tau = \tilde{\tau}_j$, for some $j \leq m$, $\tilde{\mathbf{s}}(\tau)$ reaches an extreme point of Ω. For $\tau < \tilde{\tau}_j$, $\tilde{\mathbf{s}}(\tau)$ stays at the same extreme point, and the algorithm outputs the same result. Thus, the algorithm goes through at most m loops. This feature makes this algorithm to be very efficient (in particular when $m \ll n$) in comparison to other available algorithms for optimizing problems of the form (3.14).

Also note that even though the first part of the algorithm outputs an approximate solution $\tilde{\mathbf{s}}(t)$ to constrained problem (3.14), using KKT conditions it is easy to verify whether $\tilde{\mathbf{s}}(t)$ is indeed the optimal solution $\mathbf{s}_{opt}(t)$ or not.

3.3.2 Second part

This subsection explains how $\mathbf{s}_{opt}(t)$ obtained in the first part of the TCCS algorithm is used to obtain an approximate solution to problem (3.5). Let $y_{opt}(t)$ denote the solution to problem (3.5). As $\mu \to \infty$, the solution to problem (3.2),

$y^*(t,\mu)$, provides a good approximate for $y_{opt}(t)$. Hence, in view of Corollary 3.2.12, an approximation for $y_{opt}(t)$ using $\mathbf{s}_{opt}(t)$ is found in the following way:

Let J denote the set of violations of $\mathbf{s}_{opt}(t)$. Suppose $\tilde{y}_{opt}$ is a column vector with $2n$ entries where n is the number of columns of $\mathbf{A}$. Set those entries of $\tilde{y}_{opt}$ whose index belong to J by

$$\tilde{y}_{opt}(J) = \tilde{\mathbf{A}}_J^{\dagger}\mathbf{b} - t\frac{h(|\mathbf{s}_{opt}|)}{|\mathbf{s}_{opt}|}\tilde{\mathbf{A}}_J^{\dagger}(\tilde{\mathbf{A}}_J^T)^{\dagger}\overrightarrow{\mathbf{1}},$$

and the other entries of $\tilde{y}_{opt}$ are set to zero. Now

$$\begin{bmatrix}\mathbf{I} & | & -\mathbf{I}\end{bmatrix}\tilde{y}_{opt}$$

provides a good approximation for $y_{opt}(t)$.

3.3.3 Application to compressed sensing

Observe that problem (3.5) reduces to Basis Pursuit problem (3.1) when $t = 0$. Therefore, by what was mentioned in Section 3.1, when $t = 0$, the TCCS algorithm is expected to output a sparse solution for Compressed Sensing problem (3.6). Section 3.4 provides numerical evidence in support of this.

To summarize, the TCCS algorithm approaches the Compressed Sensing problem (3.6) in the following way. We try to find an approximate optimizer for problem (3.1). To that end, we generate sequence $\tau_0 > \tau_1 > \cdots > \tau_\ell = 0$ and iteratively find approximate optimizers $\tilde{\mathbf{s}}(\tau_i)$ to problem (3.5) with τ_i in place of t. Finally, we use the result of Corollary 3.2.12 to find an approximate solution to problem (3.5).

One might question the necessity of introducing parameter μ in the methodology used to find a solution to Basis Pursuit problem (3.1). Indeed, the value of μ does not appear in the TCCS algorithm (i.e. in the second part of the TCCS algorithm only makes the assumption that μ is sufficiently large). Nevertheless, parameter μ was important in establishing Corollary 3.2.12 and identifying a good

approximate for $y_{opt}(t)$ in Subsection 3.3.2.

On the other hand, introducing the parameter t (and eventually setting it equal to 0), enables us to use the Homotopy method to find an approximate solution to the linear programming problem

$$\underset{\mathbf{s}}{\text{argmax}} \ \langle \mathbf{b}, \mathbf{s} \rangle \quad \text{subject to} \quad -\overrightarrow{\mathbf{1}} \leq \mathbf{A}^T\mathbf{s} \leq \overrightarrow{\mathbf{1}}. \tag{3.39}$$

3.4 Numerical Results

This section compares the performance of the TCCS algorithm applied to Compressed Sensing problems with the LBSB algorithm [68] and OMP algorithm [52, 40, 61]. It is important to note that the TCCS algorithm is developed to solve the more general problem of (3.5) (i.e. not only the Basis Pursuit problem (3.1), which corresponds to $t = 0$ in (3.5)); whereas, LBSB and OMP algorithm can be used only for problems (3.1) and (3.6), respectively.

For the numerical experiments, entries of the matrix $\mathbf{A}$ are generated from independent and identically distributed normal distributions. The "true signal" y_{real} with a given sparsity level is also generated from independent and identically distributed normal distributions. Vector $\mathbf{b}$ is obtained from y_{real} via $\mathbf{b} = \mathbf{A}y_{real}$. Define the *sparsity level* of vector y_{real} to be the percentage ratio of the number of nonzero components of y_{real} with respect to the number of measurements (i.e. m). That is,

$$\text{sparsity level of } y_{real} = \frac{\#\text{ nonzero components of } y_{real}}{m} \times 100\%.$$

In each setting, 10 independent trials are ran and the average processing time and the average relative error (i.e., $|y - y_{real}|/|y_{real}|$) of each method for the 10 trials are reported.

3.4.1 TCCS versus LBSB

Here the performance of the TCCS algorithm and the LBSB algorithm are compared. The LBSB algorithm is one of the fastest methods used for Compressed Sensing problems and in [68], extensive numerical experiments comparing this method with respect to other methods were done.

In each test, we generate a true signal y_{real} by first determining its support (i.e. the nonzero entries) at random and then generate each entry of the support using independent and identically distributed normal distributions. For the LBSB method, we set $\alpha = 10$, $\lambda = 0.4$ and the maximum number of iterations equal to 2000. The TCCS algorithm applied to Compressed Sensing problem does not require any parameter to be set.

Tables 3.1 and 3.2 show comparison of the LBSB algorithm and the TCCS algorithm when sparsity level is 2% and 10%, respectively.

m	n	LBSB time	LBSB error	TCCS time	TCCS error
200	10000	5.65	7.04e-03	0.13	1.07e-15
200	50000	56.71	1.21e-01	0.68	1.22e-15
200	100000	114.34	4.16e-01	1.55	9.03e-16
400	10000	11.22	1.66e-03	0.51	2.00e-15
400	50000	104.82	2.83e-02	2.65	1.53e-15
400	100000	229.97	1.11e-01	6.04	1.23e-15

Table 3.1: The comparison of the performance of the TCCS algorithm and the LBSB algorithm. "Time" and "error", respectively, refer to processing time (in seconds) and relative error for each method. Here, sparsity level is 2% and each row is the result of 10 independent trials.

For the numerical experiments that are presented in Tables 3.1 and 3.2, the matrix $\mathbf{A}$ was chosen to be very narrow and wide, to highlight the advantage of

m	n	LBSB time	LBSB error	TCCS time	TCCS error
200	10000	11.36	5.81e-02	2.22	2.62e-03
200	50000	56.72	3.59e-01	32.10	7.54e-01
200	100000	115.50	4.27e-01	71.05	3.49e-01
400	10000	22.23	3.80e-03	4.60	2.03e-15
400	50000	113.28	1.57e-01	83.21	4.77e-01
400	100000	230.65	3.62e-01	286.82	6.62e-01

Table 3.2: The comparison of the performance of the TCCS algorithm and the LBSB algorithm. "Time" and "error" , respectively, refer to processing time (in seconds) and relative error for each method. Here, sparsity level is 10% and each row is the result of 10 independent trials.

the TCCS algorithm over the LBSB algorithm in this regime, both in processing time and relative error of the solution. Remarkably, as shown in Table 3.1, when sparsity level is low, the TCCS algorithm perfectly recovers the sparse solution in all 10 independent trials.

In the next set of experiments, for different values of m and n, we compare the performance of the LBSB algorithm and the TCCS algorithm over different range of sparsity levels. The result are shown in Figures 3.1, 3.2 and 3.3.

As it can be seen from the figures, for low sparsity levels, the TCCS algorithm is again both much faster and more accurate in recovering the solution than the LBSB algorithm. Indeed, as observed earlier, for low sparsity levels, the TCCS algorithm perfectly recovers the sparse solution in all independent trials. For higher sparsity levels, the relative error of the TCCS algorithm and the LBSB algorithm is comparable. At these sparsity levels, the TCCS algorithm performs faster than the LBSB algorithm for very small ratios m/n, whereas, the converse becomes true for higher ratios of m/n (i.e. compare the panels on the right in Figure 3.2). It is also noteworthy that as the sparsity level increases, the TCCS

algorithm exhibits a sharp phase transition in the success rate of recovery of the sparse solution.

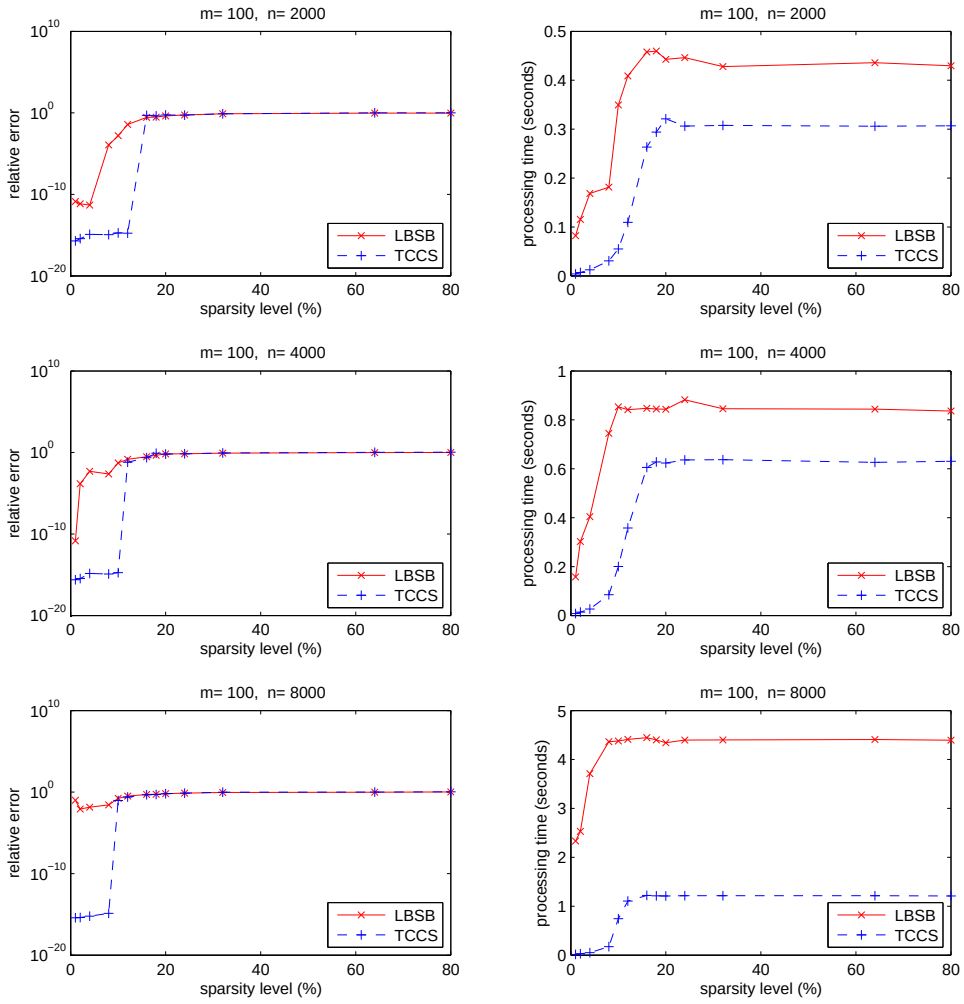

Figure 3.1: Performance comparison of the LBSB algorithm and the TCCS algorithm in terms of processing time and relative error for different sparsity levels. Here $m = 100$ and $n = 2000, 4000, 8000$. The panels on the left show semilog plot of the relative error, while the right panels show plot of processing time (in seconds). Each data point is the average of 10 independent trials.

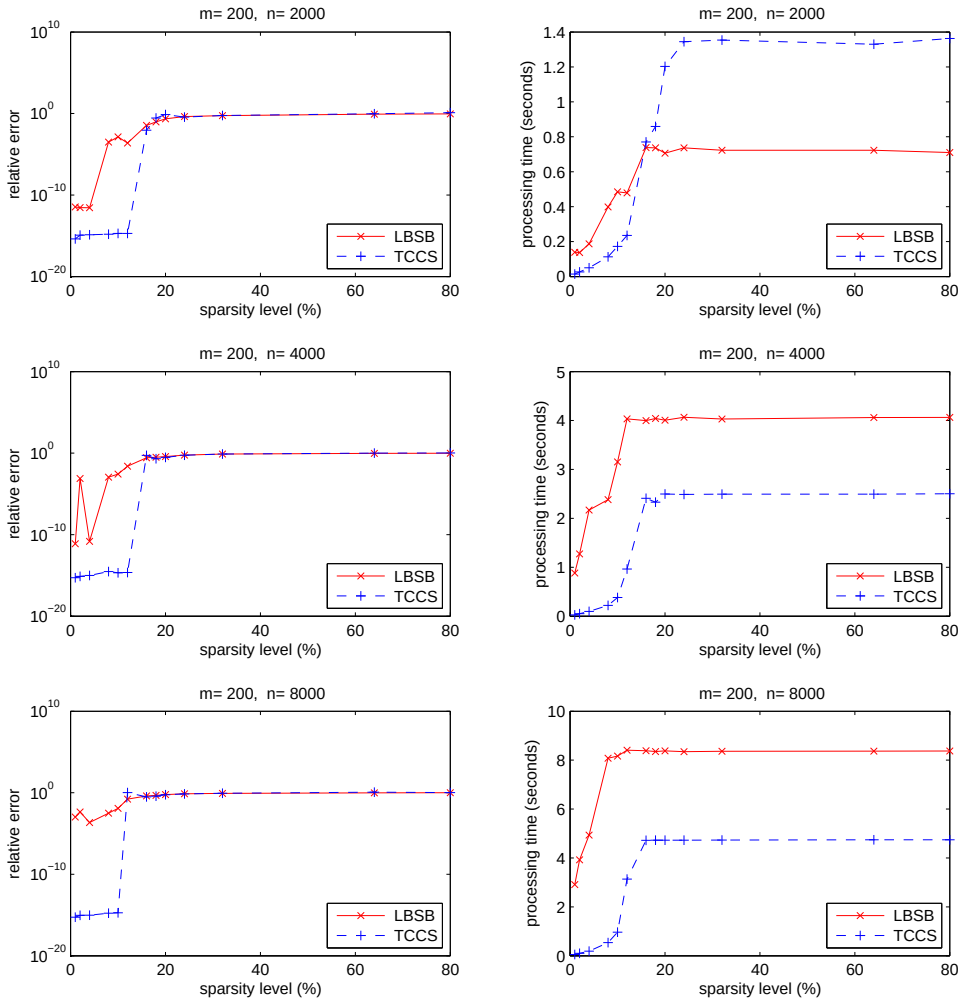

Figure 3.2: Performance comparison of the LBSB algorithm and the TCCS algorithm in terms of processing time and relative error for different sparsity levels. Here $m = 200$ and $n = 2000, 4000, 8000$. The panels on the left show semilog plot of the relative error, while the right panels show plot of processing time (in seconds). Each data point is the average of 10 independent trials.

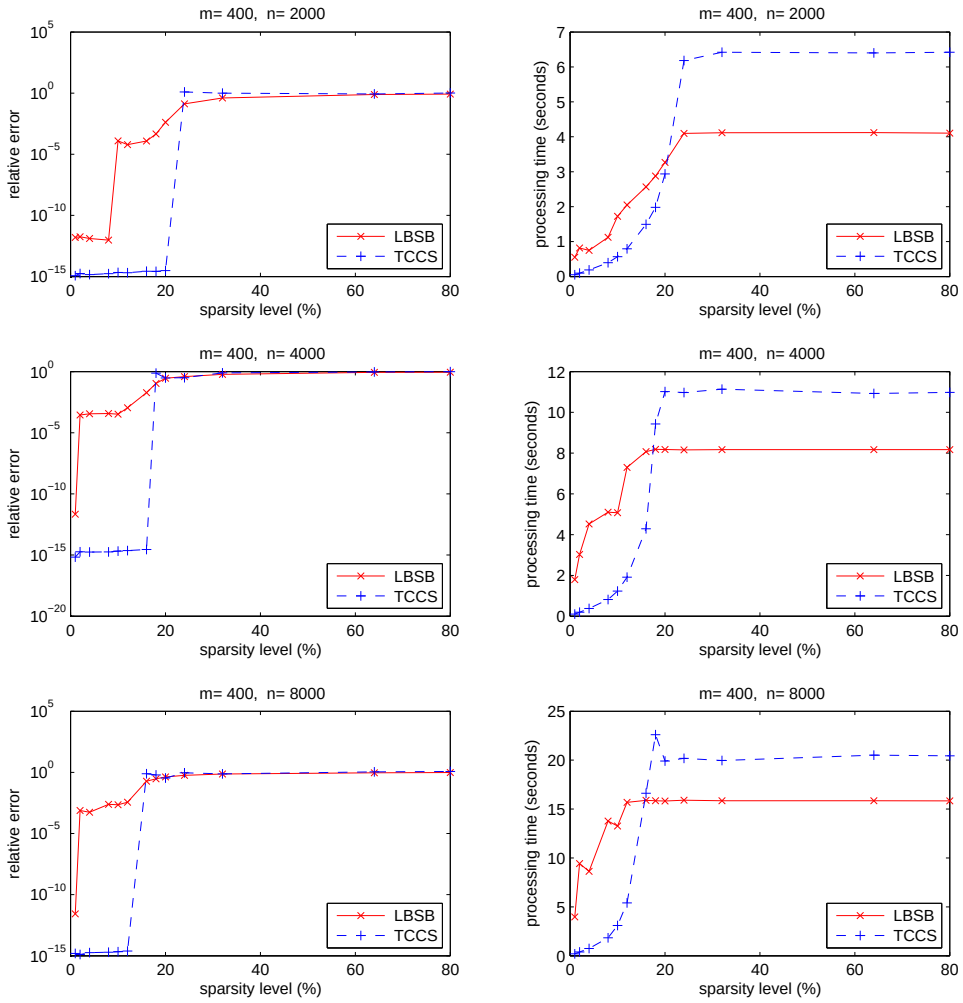

Figure 3.3: Performance comparison of the LBSB algorithm and the TCCS algorithm in terms of processing time and relative error for different sparsity levels. Here $m = 400$ and $n = 2000, 4000, 8000$. The panels on the left show semilog plot of the relative error, while the right panels show plot of processing time (in seconds). Each data point is the average of 10 independent trials.

3.4.2 TCCS versus OMP

Here the difference between the TCCS algorithm and the OMP algorithm is investigated numerically. The OMP algorithm has been noted for its speed and ease of implementation. Furthermore, [61] provides some theoretical and numerical evidence for the reliability of the OMP algorithm.

It is important to note that the methodologies of these two algorithms are quite different despite some superficial similarities in their pseudo-codes. The OMP algorithm is a greedy based method that solves the Compressed Sensing problem by iteratively increasing the support of y with components whose correlation to the current residual is maximum. On the other hand, the TCCS algorithm uses a Homotopy approach to trace the optimal solution of the dual problem.

We run two sets of experiments similar to the numerical experiments done in [10]. Because both the TCCS and the OMP algorithms run very fast, only the relative error of the two algorithms are reported. In the first experiment, we generate the true signal y_{real} in the same fashion as in Subsection 3.4.1. The results are shown in Figure 3.4.

Although the OMP algorithm outperforms the TCCS algorithm in Figure 3.4, we will see that this is not the case in general. For the second set of experiments, we generate the true signal y_{real} by first determining its support (i.e. the nonzero entries) at random and then set each entry of the support to be $+1$ or -1 at random. The results are shown in Figure 3.5. As it can be seen, the TCCS algorithm outperforms the OMP algorithm in this example.

3.5 Conclusions

In this chapter we presented the TCCS algorithm for finding approximate solutions to problem (3.5). In particular, by setting $t = 0$, the algorithm outputs a good

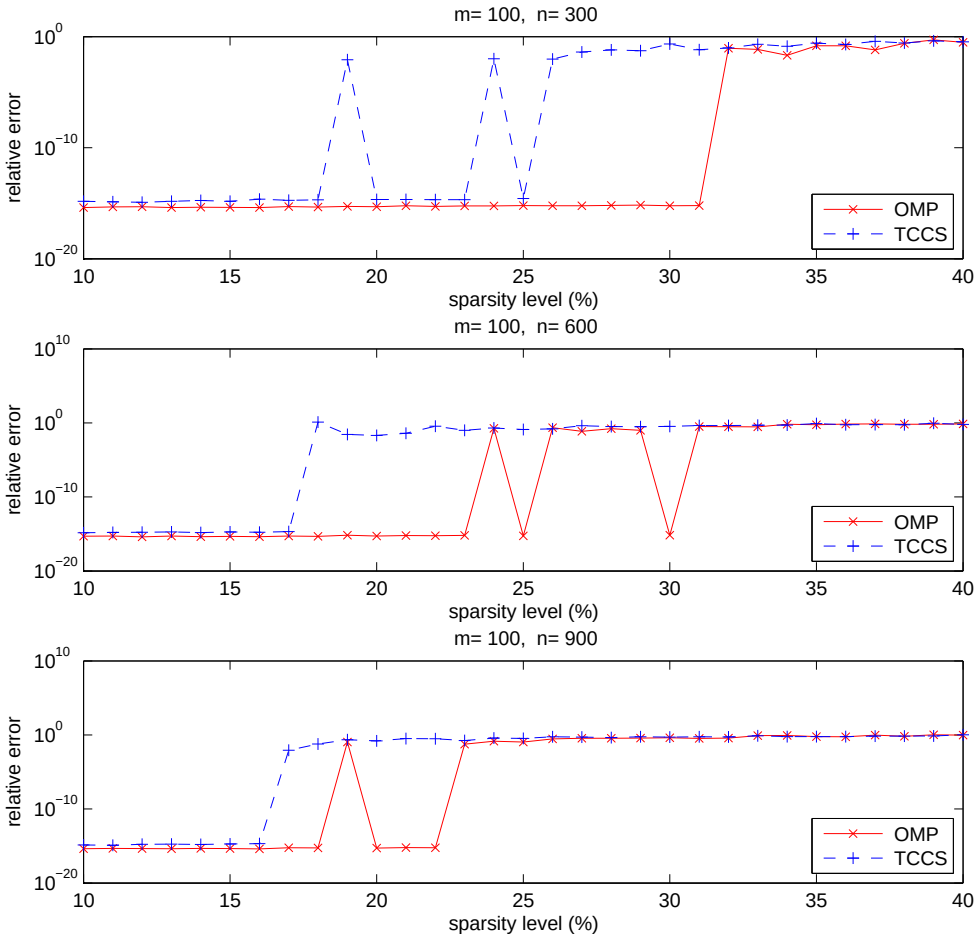

Figure 3.4: Comparison of the relative error of the OMP algorithm and the TCCS algorithm for different sparsity levels. Here the entries of the support of the true signal are chosen to be independent and identically distributed random variables. Each data point is the arithmetic average of 10 independent trials.

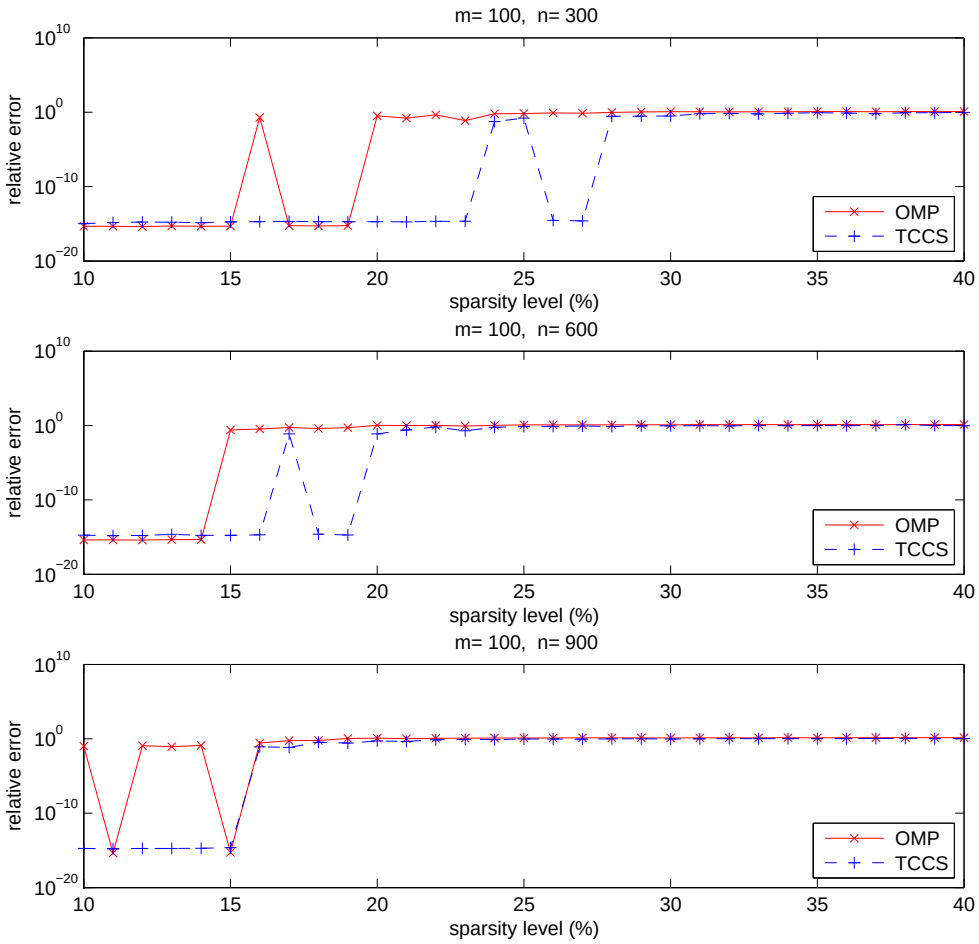

Figure 3.5: Comparison of the relative error of the OMP algorithm and the TCCS algorithm for different sparsity levels. Here the entries of the support of the true signal are chosen from $+1$ or -1 at random. Each data point is the arithmetic average of 10 independent trials.

approximate for the solution to the Compressed Sensing problem (3.6). Numerical results show that the algorithm perfectly recovers the solution to the Compressed Sensing problem when the solution is relatively sparse with respect to the number of measurements. For this regime of sparsity, the algorithm recovers the solution extremely fast.

Another advantage of the TCCS algorithm is that it is parameter free except a tolerance parameter that is due to numerical machine precision. In many other optimization methods, one needs to introduce extra parameters whose value greatly affect the performance of the algorithm.

We also observed a sharp phase transition in the success rate of recovery of the solution of the Compressed Sensing problem by the algorithm as the sparsity of the solution varies. This suggest a theoretical analysis should be conducted relating to this phenomena.

CHAPTER 4

Theoretical Analysis of Compressed Modes

Spatial localization occurs naturally in many problems from physics and other disciplines. A new methodology was developed in [50, 51] using sparsity techniques to obtain localized functions that are approximate solutions to a class of problems in mathematical physics that can be recast as variational optimization problems.

After introduction of L^1 regularization in [50, 51], the variational formulation of the Schrödinger equation of quantum mechanics, localized functions called "compressed modes" (CMs) were constructed by solution of the resulting new non-convex optimization problem. Numerical computation showed the CMs have many desirable features; for example, the energy calculated using CMs approximates the ground state energy of the system. Moreover, there is no requirement to cut off the resulting CMs "by hand". In addition, the ideas of [50] were used in [51] to generate a new set of spatially localized orthonormal functions, called "compressed plane waves" (CPWs), with multi resolution capabilities adapted for the Laplace operator.

Several analytic treatments of CMs and CPWs have been performed. In this chapter we present the analysis that was done in [2] and [3]. In [69], CMs were proved to form a complete basis set in their corresponding vector spaces, which is necessary for use of CMs in numerical computation.

Use of the ℓ_1 norm as a constraint or penalty term to achieve sparsity has attracted considerable attention in a variety of fields including compressed sensing [24, 15], matrix completion [57], phase retrieval [14], etc. Recently, use of sparsity

techniques began in physical sciences (see for example [46]) and partial differential equations (see for example [58]). In all these examples sparsity means that in the representation of a corresponding vector or function in terms of a well-chosen set of modes (i.e. a basis or dictionary), most coefficients are zero. However, it was proven in [8] and [9] that for certain elliptic and parabolic PDEs, insertion of L^1 terms into the quantity to be optimized would result in solutions having compact support (i.e. which can be thought as spatial sparsity) for the solutions. In [50, 51], L^1 norm regularization is used to achieve spatial sparsity for variational problems in mathematics and physics. In more recent work, [12] presents qualitative analysis of elliptic and parabolic PDEs with subgradient terms that come from L^1 regularization in the variational formulation of the problem.

In this chapter we present two main results are presented. First the consistency results for compressed modes (CMs) are proven. In particular, we show that as $\mu \to \infty$ the approximate energy calculated using CMs converges from above to the actual energy of the system. This is done in Section 4.1. More importantly, we show in Section 4.2 that under some necessary assumptions on the spectrum of the Hamiltonian, as $\mu \to \infty$, CMs converge to a unitary transformation of the eigenfunctions in L^2 norm. Finally, we verify a conjecture stated in [50].

The second main result of this chapter is to analytically prove the spatial localization property of CMs that was observed numerically in [50]. We show that as the coefficient increases for the L^1 regularization term in the variational formulation of the Schrödinger equation, the volume of the support of CMs shrinks.

To be specific, let $\{\psi_i\}_{i=1}^N$ denote the CMs, corresponding to number N and regularization parameter μ, that solve the L^1 regularization of the variational formulation of the Schrödinger equation in $\mathbb{R}^d$:

$$\{\psi_1, \ldots, \psi_N\} = \underset{\tilde{\psi}_1,\ldots,\tilde{\psi}_N}{\operatorname{argmin}} \sum_{i=1}^{N} \left(\frac{1}{\mu} \|\tilde{\psi}_i\|_1 + \langle \tilde{\psi}_i, \hat{H}\tilde{\psi}_i \rangle \right) \quad \text{s.t.} \quad \langle \tilde{\psi}_j, \tilde{\psi}_k \rangle = \delta_{jk}. \tag{4.1}$$

Here, $\hat{H} = -\frac{1}{2}\Delta + V(\mathbf{x})$ is the Hamiltonian operator corresponding to potential

$V(\mathbf{x})$, L^1 norm is defined as $\|f\|_1 = \int_\Omega |f| d\mathbf{x}$, and $\langle f, g\rangle = \int_\Omega f^* g d\mathbf{x}$ $(\Omega \subset \mathbb{R}^d)$.

In this chapter, for the ease of exposition we assume that $\Omega = [0, L]^d$ with periodic boundary conditions. Other boundary conditions can be handled in a similar way. For simplicity, we assume that ψ_i's are real functions and equal to zero at the boundary of Ω; generalization to complex-valued functions is straightforward. Furthermore, we assume that potential $V(\mathbf{x})$ in the Hamiltonian $\hat{H}$ is bounded above by a finite number; that is,

$$\|V\|_\infty := \sup_\Omega |V(\mathbf{x})| < \infty.$$

These assumptions, in particular, imply that the spectrum of the Hamiltonian $\hat{H}$ is discrete and its eigenvalues grow to positive infinity. In section 4.4, we make several remarks about the existence of minimizers for (4.1). Note that because of the orthonormality condition, the space of feasible functions in (4.1) is not a convex set and many convex optimization techniques cannot be applied here.

Denote the eigenfunctions of the Hamiltonian operator $\hat{H}$ by $\phi_1, \phi_2, \ldots$, and assume that they are ordered so that their corresponding eigenvalues obey $\lambda_1 \leq \lambda_2 \leq \ldots$. Observe that $\{\phi_i\}_{i=1}^N$ are a solution to the optimization problem:

$$\{\phi_1, \ldots, \phi_N\} = \underset{\tilde{\phi}_1, \ldots, \tilde{\phi}_N}{\operatorname{argmin}} \sum_{i=1}^N \langle \tilde{\phi}_i, \hat{H}\tilde{\phi}_i\rangle \qquad \text{s.t.} \qquad \langle \tilde{\phi}_j, \tilde{\phi}_k\rangle = \delta_{jk}. \tag{4.2}$$

As shown in [50], compressed modes have many desirable features. In particular, consider the $N \times N$ matrix $\langle \Psi_N^T, \hat{H}\Psi_N\rangle$ with the (j,k)-th entry defined by $\langle \psi_j, \hat{H}\psi_k\rangle$ and let $(\sigma_1, \ldots, \sigma_N)$ denote its eigenvalues listed in non-decreasing order. In [50], it was conjectured that as $\mu \to \infty$, σ_i's converge to λ_i's. In Theorem 4.2.5 we verify this conjecture. We also show that as $\mu \to \infty$, CMs $\{\psi_i\}_{i=1}^N$ converge to a unitary transformation of eigenfunctions $\{\phi_i\}_{i=1}^N$ in the L^2 norm. In Theorem 4.4.1 we show that for sufficiently small μ,

$$|\text{supp}(\psi_i)| \leq C\mu^{2d/(4+d)} \qquad \text{for } i = 1, \ldots, N,$$

where C is a constant. This result, in particular, verifies the observations in [50] that the smaller μ, the more localized are the corresponding CMs.

The remainder of this chapter consists of the following: Section 4.1 shows that the energy associated with CMs converges from above to the actual energy of the system as $\mu \to \infty$. Section 4.2 contains the first main result of this chapter; that is, as $\mu \to \infty$, $\{\psi_i\}_{i=1}^N$ converge to a unitary transformation of $\{\phi_i\}_{i=1}^N$ and eigenvalues $(\sigma_1, \ldots, \sigma_N)$ converge to $(\lambda_1, \ldots, \lambda_N)$. Section 4.3 provides an analytic formula for the first compressed mode when $V(\mathbf{x}) = 0$. Section 4.4 contains the second main result of this chapter and establishes the asymptotic upper bound on the volume of the support of compressed modes mentioned above. Section 4.5 includes several remarks on the CMs corresponding to different regularization parameter μ and a conjecture.

4.1 Convergence of Energies

Define the energy associated with CMs $\{\psi_i\}_{i=1}^N$ by,

$$E = \sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle = \mathrm{Tr}(\langle \Psi_N^T, \hat{H}\Psi_N \rangle) = \sigma_1 + \cdots + \sigma_N. \tag{4.3}$$

This section shows that E converges to the ground state energy $E_0 = \sum_{j=1}^N \lambda_j$ as $\mu \to \infty$. Although, the result of this section can be readily deduced from Theorem 4.2.5, it is included here for its independent interest and simplicity of argument.

First observe that

$$E_0 = \lambda_1 + \cdots + \lambda_N = \sum_{i=1}^{N} \langle \phi_i, \hat{H}\phi_i \rangle \leq \sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle = \sigma_1 + \cdots + \sigma_N = E, \tag{4.4}$$

where (4.2) was used for the inequality and equation (4.3) was used for the last equality. Next choose μ large enough such that

$$\frac{1}{\mu} \sum_{i=1}^{N} \|\phi_i\|_1 < \epsilon.$$

Then

$$E = \sigma_1 + \cdots + \sigma_N = \sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle \leq \frac{1}{\mu} \sum_{i=1}^{N} \|\psi_i\|_1 + \sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle$$
$$\leq \frac{1}{\mu} \sum_{i=1}^{N} \|\phi_i\|_1 + \sum_{i=1}^{N} \langle \phi_i, \hat{H}\phi_i \rangle < \epsilon + \lambda_1 + \cdots + \lambda_N = \epsilon + E_0, \tag{4.5}$$

where (4.3) was used for the first equality and (4.1) was used for the second inequality. From equations (4.4) and (4.5) it follows that

$$E \downarrow E_0 \qquad \text{as } \mu \to \infty.$$

4.2 Consistency Results for L^1 Regularization

This section contains the first main result of this chapter. In Theorem 4.2.3, we show that as $\mu \to \infty$, the solutions to the regularized optimization problem (4.1) converge to a unitary transformation of the eigenfunctions in L^2 norm. Furthermore, in Theorem 4.2.5, we show that as $\mu \to \infty$, the eigenvalues of matrix $\langle \Psi_N^T, \hat{H}\Psi_N \rangle$ converges to the first N eigenvalues of the Hamiltonian $\hat{H}$. This provides an affirmative answer to the conjecture stated in [50].

First we establish the following lemma. There are several different proofs known for this lemma. Here we present a proof that uses a so-called Cholesky decomposition.

Lemma 4.2.1 *Suppose that for $i, j = 1, \ldots, N$*

$$\sum_{k=1}^{\infty} a_{ik}^* a_{jk} = \delta_{ij}. \tag{4.6}$$

Then, for any k,

$$\sum_{i=1}^{N} |a_{ik}|^2 \leq 1.$$

Proof: It suffices to show the result for $k = 1$ (i.e. by relabeling the indices, the result would follow for other k's). Let C be an $N \times N$ matrix whose ij-th entry is given by

$$C_{ij} = \sum_{l=2}^{\infty} a_{il}^* a_{jl}. \tag{4.7}$$

By construction, C is hermitian. We claim that C is also positive semi-definite matrix. Indeed, for any vector $\mathbf{x} = (x_1, \ldots, x_N)^T$,

$$\begin{aligned} \mathbf{x}^* C \mathbf{x} &= \sum_{i,j=1}^{N} x_i^* C_{ij} x_j = \sum_{i,j=1}^{N} x_i^* x_j \sum_{l=2}^{\infty} a_{il}^* a_{jl} \\ &= \sum_{l=2}^{\infty} \left(\sum_{i=1}^{N} x_i a_{il} \right)^* \left(\sum_{j=1}^{N} x_i a_{jl} \right) = \sum_{l=2}^{\infty} \left| \sum_{i=1}^{N} x_i a_{il} \right|^2 \geq 0. \end{aligned}$$

Using a so-called Cholesky decomposition, there exists a lower diagonal matrix L (i.e. not necessarily unique as C is semi-definite) such that $C = \bar{L}\bar{L}^\dagger$. Thus,

$$C_{ij} = \sum_{m=1}^{N} L_{im}^* L_{jm}.$$

In particular, comparing with equation (4.7), one concludes that for $i, j = 1 \ldots, N$,

$$\sum_{m=1}^{N} L_{im}^* L_{jm} = \sum_{l=2}^{\infty} a_{il}^* a_{jl}$$

For $i = 1, \ldots, N$, set $L_{i0} := a_{i1}$ and $\vec{L}_i = (L_{i0}, L_{i1}, \ldots, L_{iN})$. Assumption (4.6) and the above equality yield that

$$\langle \vec{L}_i, \vec{L}_j \rangle = \delta_{ij} \qquad \text{for} \qquad i, j = 1, \ldots, N.$$

Note that $\{\vec{L}_i\}_{i=1}^N$ are $(N+1)$-dimensional vectors. There exist vector $\vec{L_0} = (L_{00}, L_{01}, \ldots, L_{0N})$ such that $\{\vec{L}_i\}_{i=0}^N$ is an orthonormal basis in $\mathbb{R}^{N+1}$. Form $(N+1)\times(N+1)$ matrix M whose ij-th entry M_{ij} is equal to L_{ij} for $i, j = 0, \ldots, N$. Observe that M is unitary as its rows are orthonormal. Hence the columns of M are orthonormal as well. In particular,

$$1 = \sum_{i=0}^{N} |M_{i0}|^2 = \sum_{i=0}^{N} |L_{i0}|^2 = |L_{00}|^2 + \sum_{i=1}^{N} |a_{i1}|^2,$$

and so

$$\sum_{i=1}^{N} |a_{i1}|^2 = 1 - |L_{00}|^2 \leq 1.$$

This completes the lemma. ∎

Lemma 4.2.2 *Suppose $\lambda_N < \lambda_{N+1}$. For any $\epsilon > 0$ there exist $\epsilon_0 > 0$ such that*

$$\left| \sum_{i=1}^{N} \langle g_i, \hat{H} g_i \rangle - \sum_{i=1}^{N} \langle \phi_i, \hat{H} \phi_i \rangle \right| < \epsilon_0 \qquad \textit{with} \qquad \langle g_i, g_j \rangle = \delta_{i,j}$$

implies that $\|g_i - \varphi_i\|_2 < \epsilon$, for $i = 1 \ldots, N$, where $\varphi_1 \ldots, \varphi_N$ is a unitary transformation of $\phi_1 \ldots, \phi_N$.

Proof: Since $\{\phi_i\}_{i=1}^{\infty}$ form a set of basis in L^2, for every $i = 1 \ldots, N$:

$$g_i = \sum_{k=1}^{\infty} a_{ik} \phi_k.$$

Moreover, the assumptions on the g_i's imply that

$$\sum_{k=1}^{\infty} a_{ik}^* a_{jk} = \delta_{ij}. \tag{4.8}$$

By spectral decomposition and using Dirac's bra-ket notation,

$$\hat{H} = \sum_{l=1}^{\infty} \lambda_l |\phi_l\rangle\langle\phi_l|.$$

Thus, for $i = 1 \ldots, N$,

$$\langle g_i, \hat{H} g_i \rangle = \langle g_i, \sum_{l=1}^{\infty} \lambda_l |\phi_l\rangle\langle\phi_l | g_i \rangle = \sum_{l=1}^{\infty} \lambda_l \langle g_i, \phi_l\rangle\langle\phi_l, g_i\rangle = \sum_{l=1}^{\infty} \lambda_l |a_{il}|^2.$$

Summing over i on both sides of the above equality yields that

$$\sum_{i=1}^{N} \langle g_i, \hat{H} g_i \rangle = \sum_{i=1}^{N} \sum_{l=1}^{\infty} \lambda_l |a_{il}|^2 = \sum_{l=1}^{\infty} \lambda_l \sum_{i=1}^{N} |a_{il}|^2 = \sum_{l=1}^{\infty} \lambda_l b_l,$$

where

$$b_l := \sum_{i=1}^{N} |a_{il}|^2.$$

Observe that b_l's satisfy the following properties:

$$0 \leq b_l \leq 1 \quad \text{for} \quad l = 1, \ldots, \tag{4.9}$$

and

$$\sum_{l=1}^{\infty} b_l = N. \tag{4.10}$$

The first property is deduced from Lemma 4.2.1. To see the second property note that

$$\sum_{l=1}^{\infty} b_l = \sum_{l=1}^{\infty} \sum_{i=1}^{N} |a_{il}|^2 = \sum_{i=1}^{N} \sum_{l=1}^{\infty} |a_{il}|^2 = \sum_{i=1}^{N} 1 = N.$$

Now observe that

$$\begin{aligned}
&\sum_{i=1}^{N} \langle g_i, \hat{H} g_i \rangle - \sum_{i=1}^{N} \langle \phi_i, \hat{H} \phi_i \rangle \\
&\qquad = \sum_{l=1}^{\infty} \lambda_l b_l - (\lambda_1 + \cdots + \lambda_N) \\
&\qquad \geq (b_1 - 1)\lambda_1 + \cdots + (b_N - 1)\lambda_N + (\sum_{l=N+1}^{\infty} b_l)\lambda_{N+1} \\
&\qquad = (b_1 - 1)\lambda_1 + \cdots + (b_N - 1)\lambda_N + (N - b_1 - \cdots - b_N)\lambda_{N+1} \\
&\qquad = (1 - b_1)(\lambda_{N+1} - \lambda_1) + \cdots + (1 - b_N)(\lambda_{N+1} - \lambda_N),
\end{aligned} \tag{4.11}$$

where the nondecreasing ordering of λ_i's was used in the third line, and (4.10) was used in the fourth line. Now from (4.9) and nondecreasing ordering of λ_i's, each of the terms in summation (4.11) is positive. Hence

$$\left| \sum_{i=1}^{N} \langle g_i, \hat{H} g_i \rangle - \sum_{i=1}^{N} \langle \phi_i, \hat{H} \phi_i \rangle \right| \geq (1 - b_1)(\lambda_{N+1} - \lambda_1) + \cdots + (1 - b_N)(\lambda_{N+1} - \lambda_N).$$

The assumption that λ_{N+1} is strictly greater than $\lambda_1, \ldots, \lambda_N$ yields that for every $\epsilon_1 > 0$, there exist ϵ_0 such that if the LHS of the above inequality is smaller than ϵ_0, then

$$1 - b_l < \epsilon_1 \quad \text{for} \quad l = 1, \ldots, N.$$

Moreover, equation (4.10) implies that

$$\sum_{l=N+1}^{\infty} b_l < N\epsilon_1 \Longrightarrow \sum_{l=N+1}^{\infty}\sum_{i=1}^{N} |a_{il}|^2 = \sum_{i=1}^{N}\sum_{l=N+1}^{\infty} |a_{il}|^2 < N\epsilon_1.$$

In particular for $i = 1 \ldots, N$:

$$\sum_{l=N+1}^{\infty} |a_{il}|^2 < N\epsilon_1. \tag{4.12}$$

Next we will show that $N \times N$ matrix $\{a_{ik}\}_{i,k=1}^{N}$ is "almost" unitary in the sense that

$$\left|\delta_{ij} - \sum_{k=1}^{N} a_{ik}^* a_{jk}\right| = \left|\sum_{k=N+1}^{\infty} a_{ik}^* a_{jk}\right| \leq \left(\sum_{k=N+1}^{\infty} |a_{ik}|^2\right)^{1/2} \left(\sum_{k=N+1}^{\infty} |a_{jk}|^2\right)^{1/2}$$
$$< (N\epsilon_1)^{1/2}(N\epsilon_1)^{1/2} = N\epsilon_1,$$

where (4.8) was used for the first equality, Cauchy-Schwarz was used for the first inequality, and (4.12) was used for the second inequality. One can orthonormalize the rows of matrix $\{a_{ik}\}_{i,k=1}^{N}$ using Gram-Schmidt process, to form a new unitary matrix $\{a'_{ik}\}_{i,k=1}^{N}$. Indeed, because of the above inequality, for any $\epsilon_2 > 0$, one may choose ϵ_1 small enough such that

$$\sum_{k=1}^{N} |a_{ik} - a'_{ik}|^2 < \epsilon_2 \qquad \text{for every} \qquad i = 1, \ldots, N. \tag{4.13}$$

For $i = 1, \ldots, N$, set

$$\varphi_i = \sum_{k=1}^{N} a'_{ik}\phi_k.$$

Observe that

$$\|g_i - \varphi_i\|_2^2 = \sum_{k=1}^{N} |a_{ik} - a'_{ik}|^2 + \sum_{k=N+1}^{\infty} |a_{ik}|^2 < \epsilon_2 + N\epsilon_1,$$

where (4.13) and (4.12) were used for the inequality.

Hence, for any $\epsilon > 0$, there exist ϵ_1 and ϵ_2 small enough such that $\|g_i - \varphi_i\|_2 < \epsilon$ for $i = 1, \ldots, N$. The result follows. ∎

Now, the main results of this section are presented:

Theorem 4.2.3 *Assume $\lambda_{N+1} > \lambda_N$. For every $\epsilon > 0$, there exist μ_0 such that for $\mu > \mu_0$, the solutions to the regularized optimization problem* (4.1) *satisfy*

$$\|\psi_i - \varphi_i\|_2 < \epsilon \qquad for \qquad i = 1 \ldots, N,$$

where $\varphi_1 \ldots, \varphi_N$ is some unitary transformation of $\phi_1 \ldots, \phi_N$.

Proof: For given ϵ, choose ϵ_0 as indicated by Lemma 4.2.2. Choose μ_0 large enough such that

$$\frac{1}{\mu_0} \sum_{i=1}^{N} \|\phi_i\|_1 < \epsilon_0. \tag{4.14}$$

Let $\mu > \mu_0$. Observe that

$$\begin{aligned} \sum_{i=1}^{N} \langle \phi_i, \hat{H}\phi_i \rangle \leq \sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle &\leq \frac{1}{\mu} \sum_{i=1}^{N} \|\psi_i\|_1 + \sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle \\ &\leq \frac{1}{\mu} \sum_{i=1}^{N} \|\phi_i\|_1 + \sum_{i=1}^{N} \langle \phi_i, \hat{H}\phi_i \rangle. \end{aligned}$$

where (4.2) was used for the first inequality, and (4.1) was used for the last inequality. Hence, using (4.14), one conclude that

$$\sum_{i=1}^{N} \langle \phi_i, \hat{H}\phi_i \rangle \leq \sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle \leq \epsilon_0 + \sum_{i=1}^{N} \langle \phi_i, \hat{H}\phi_i \rangle,$$

which implies that

$$\left| \sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle - \sum_{i=1}^{N} \langle \phi_i, \hat{H}\phi_i \rangle \right| < \epsilon_0. \tag{4.15}$$

Now applying Lemma 4.2.2, completes the proof. ∎

Remark 4.2.4 *The assumption $\lambda_{N+1} > \lambda_N$ is essential. Indeed, otherwise there is a possibility that ψ_N converges to ϕ_{N+1} in L^2 norm and the result of Theorem 4.2.3 would clearly not hold.*

Theorem 4.2.5 *Using the above notations, the eigenvalues $(\sigma_1, \ldots, \sigma_N)$ of matrix $\langle \Psi_N^T, \hat{H}\Psi_N^T \rangle$ converge, as $\mu \to \infty$, to the eigenvalues $(\lambda_1, \ldots, \lambda_N)$ of the Hamiltonian $\hat{H}$.*

Proof: Using the same notation as before, let φ_i, for $i = 1, \ldots, N$, be the unitary transformation of the eigenfunctions $\phi_1, \ldots, \phi_N$ as described by Theorem 4.2.3. Let $\langle \Phi_N^T, \hat{H}\Phi_N \rangle$ be the $N \times N$ matrix with (j,k)-th entry equal to $\langle \varphi_j, \hat{H}\varphi_k \rangle$. The objective is to show that matrix $\langle \Psi_N^T, \hat{H}\Psi_N \rangle$ converges to matrix $\langle \Phi_N^T, \hat{H}\Phi_N \rangle$ entry-wise. This would complete the proof because matrix $\langle \Phi_N^T, \hat{H}\Phi_N \rangle$ has the same eigenvalues $(\lambda_1, \ldots, \lambda_N)$ as the Hamiltonian (i.e., functions $\{\varphi_i\}_{i=1}^N$ are a unitary transformation of the eigenfunctions $\{\phi_i\}_{i=1}^N$). To that end, it suffices to show that

$$\langle \psi_i, \hat{H}\psi_j \rangle \to \langle \varphi_i, \hat{H}\varphi_j \rangle \qquad \text{for } i, j = 1, \ldots, N.$$

Suppose

$$\varphi_i = \sum_{k=1}^{N} a'_{ik}\phi_k \qquad \text{for } i = 1, \ldots, N,$$

and

$$\psi_i = \sum_{k=1}^{\infty} a_{ik}\phi_k \qquad \text{for } i = 1, \ldots, N.$$

The result of Theorem 4.2.3 yields that

$$\|\psi_i - \varphi_i\|_2^2 = \sum_{k=1}^{N} |a_{ik} - a'_{ik}|^2 + \sum_{k=N+1}^{\infty} |a_{ik}|^2 \to 0, \qquad \text{as} \qquad \mu \to \infty. \tag{4.16}$$

This also implies that as $\mu \to \infty$, for $i = 1 \ldots, N$,

$$a_{ik} \to a'_{ik} \text{ when } 1 \le k \le N \quad \text{ and } \quad a_{ik} \to 0 \text{ when } k > N. \tag{4.17}$$

Since $\sum_{i=1}^N \langle \phi_i, \hat{H}\phi_i \rangle = \sum_{i=1}^N \langle \varphi_i, \hat{H}\varphi_i \rangle$, from the proof of Theorem 4.2.3 (i.e. equation (4.15)), one can conclude that

$$\sum_{i=1}^{N} \langle \psi_i, \hat{H}\psi_i \rangle \to \sum_{i=1}^{N} \langle \varphi_i, \hat{H}\varphi_i \rangle \qquad \text{as} \qquad \mu \to \infty.$$

Rewriting the above expression,

$$\sum_{i=1}^{N}\sum_{k=1}^{\infty} \lambda_k |a_{ik}|^2 \to \sum_{i=1}^{N}\sum_{k=1}^{N} \lambda_k |a'_{ik}|^2 \qquad \text{as} \qquad \mu \to \infty.$$

Relationships (4.17) yield that for any finite $M > N$, $\sum_{i=1}^{N}\sum_{k=1}^{M}\lambda_k|a_{ik}|^2$ converges to $\sum_{i=1}^{N}\sum_{k=1}^{N}\lambda_k|a'_{ik}|^2$ as $\mu \to \infty$. Hence,

$$\sum_{i=1}^{N}\sum_{k=M}^{\infty}\lambda_k|a_{ik}|^2 \to 0 \qquad \mu \to \infty. \tag{4.18}$$

Now for $i, j = 1, \ldots, N$, consider

$$\begin{aligned}\langle\psi_i, \hat{H}\psi_j\rangle - \langle\varphi_i, \hat{H}\varphi_j\rangle &= \sum_{k=1}^{\infty}\lambda_k a^*_{ik}a_{jk} - \sum_{k=1}^{N}\lambda_k a'^*_{ik}a'_{jk} \\ &= \sum_{k=1}^{N}\lambda_k(a^*_{ik}a_{jk} - a'^*_{ik}a'_{jk}) + \sum_{k=N+1}^{\infty}\lambda_k a^*_{ik}a_{jk} \end{aligned} \tag{4.19}$$

The first summation in line (4.19) goes to zero as $\mu \to \infty$, because of (4.17). If all λ_k's are negative (i.e. and therefore bounded, as they are arranged in nondecreasing order), then a simple application of Cauchy-Schwarz and using (4.16), implies that the second summation in line (4.19) also goes to zeros as $\mu \to \infty$. Otherwise, there exist finite M greater than N such that for $k > M$, λ_k is positive. Next write the second summation in line (4.19) as

$$\sum_{k=N+1}^{M}\lambda_k a^*_{ik}a_{jk} + \sum_{k=M+1}^{\infty}\lambda_k a^*_{ik}a_{jk}.$$

Again the first summation in the above line goes to zero as $\mu \to \infty$ because of (4.17). For the second summation in the above expression, note that by Cauchy-Schwarz

$$\left|\sum_{k=M+1}^{\infty}\lambda_k a^*_{ik}a_{jk}\right| \leq \left(\sum_{k=M+1}^{\infty}\lambda_k|a_{ik}|^2\right)^{1/2}\left(\sum_{k=M+1}^{\infty}\lambda_k|a_{jk}|^2\right)^{1/2}.$$

The reason that we use λ_k instead of $|\lambda_k|$ on the RHS is due to the assumption that λ_k's are positive for $k > M$. Equation (4.18), in particular, yields that the RHS of the above expression goes to zero. Thus, it is shown that the expression on line (4.19) goes to 0 as $\mu \to \infty$. The result follows. ∎

Remark 4.2.6 *Observe that Theorem 4.2.5 does not immediately follow from the result of Theorem 4.2.3. This is due to the fact that the Hamiltonian $\hat{H}$ is not generally a bounded operator on L^2 functions.*

4.3 First Compressed Mode with Zero Potential

This section provides an analytic formula for the first compressed modes, for potential $V(\mathbf{x}) = 0$. This formula was derived in [50, equation 9], under symmetry and positivity assumptions which are proved here. The result is used later in the proof of Theorem 4.4.1.

Let $\Omega = [0, L]^d$. Denote

$$\|f\|_p = \left(\int_\Omega |f|^p \, \mathrm{d}\mathbf{x} \right)^{1/p},$$

and define functional $J : L^2(\Omega) \to \mathbb{R}^+$ by

$$J[f] := \frac{1}{\mu} \|f\|_1 + \langle f, \hat{H} f \rangle.$$

By the first compressed mode ψ, we mean the minimizer of $J[\psi]$ subject to the constraint that $\|\psi\|_2 = 1$.

Proposition 4.3.1 *For $V(\mathbf{x}) = 0$, the first compressed mode ψ is symmetric, i.e. $\psi(\mathbf{x}) = \psi(|\mathbf{x}|)$, and can be taken to be nonnegative.*

Proof: First we show that the first compressed mode can be taken to be non-negative. Take the function ψ, which may have positive and negative values, and replace it by $|\psi|$. Since this does not change the values of $|\psi|$ and $|\nabla\psi|$, it does not change the values of the L^2 norm or of J. It follows that $|\psi|$ is also a minimizer of J.

Next we show that nonnegative ψ is symmetric. Let ψ^* denote the "symmetric decreasing rearrangement" of ψ (i.e. see for example [39]). By construction ψ^* is symmetric and it is well known that

$$\|\psi\|_p = \|\psi^*\|_p \qquad \text{for } 1 \leq p. \tag{4.20}$$

Moreover, from so-called Pólya-Szegő inequality we conclude that

$$\|\nabla\psi^*\|_2 \leq \|\nabla\psi\|_2. \tag{4.21}$$

The inequality sign in the above expression is strict unless ψ is symmetric. Equations (4.20) and (4.21) imply that $\|\psi^*\|_2 = 1$ and $J[\psi^*] \leq J[\psi]$. Since ψ is the minimizer of J, it must be the case that ψ is symmetric. ∎

Next, using the fact that ψ is spherically symmetric, we find explicit solution for the first compressed mode when $V(\mathbf{x}) = 0$; that is, the minimizer of the variational problem:

$$\psi_1 = \underset{\psi}{\operatorname{argmin}} \frac{1}{\mu}\int_\Omega |\psi| d\mathbf{x} - \frac{1}{2}\int_\Omega \psi\Delta\psi d\mathbf{x} \qquad \text{s.t.} \qquad \int_\Omega \psi(\mathbf{x})^2 d\mathbf{x} = 1. \tag{4.22}$$

First consider $d = 1$. The Euler-Lagrange equation in $1D$ is

$$-\partial_x^2\psi_1 + \frac{1}{\mu}\text{sign}(\psi_1) = \lambda\psi_1. \tag{4.23}$$

Using the result of Proposition 4.3.1 and without loss of generality assume that ψ_1 is symmetric around $x = L/2$. Then, the solution of (4.22) is

$$\psi_1 = \begin{cases} \dfrac{1}{\lambda\mu}[1 + \cos(\sqrt{\lambda}(x - L/2))] & \text{if} \quad |x - L/2| \leq l, \\ 0 & \text{if} \quad l \leq |x - L/2| \leq L, \end{cases} \tag{4.24}$$

where $l = \pi/\sqrt{\lambda}$ and $\lambda = (3\pi)^{2/5}\mu^{-4/5}$. Here ψ_1 has compact support $[L/2 - l, L/2 + l]$ if μ is small enough satisfying $l = \pi/\sqrt{\lambda} < L$. Note that $\psi_1 = \partial_x\psi_1 = 0$ and $\partial_x^2\psi_1$ has a jump of $-\mu^{-1}$ at the boundary $x = L/2 + l$ of the support of ψ_1, which are all consistent with equation (4.23). From this simple 1D example, it is clear that L_1 regularization can naturally truncate solutions to the variational problem given by equation (4.22). Moreover, we also observe that the smaller μ will provide a smaller region of compact support.

The 1D solution (4.24) can be generalized to dimension $d > 1$, as

$$\psi_1 = \begin{cases} \dfrac{1}{\lambda\mu}(1 - U_0^{-1}U(\sqrt{\lambda}|\mathbf{x} - \mathbf{x}_0|)) & \text{if} \quad |\mathbf{x} - \mathbf{x}_0| \leq l, \\ 0 & \text{if} \quad l \leq |\mathbf{x} - \mathbf{x}_0| \leq L, \end{cases} \tag{4.25}$$

in which $\mathbf{x}_0$ is the center of the cube $[0, L]^d$ and $U(\mathbf{y}) = U(r = |\mathbf{y}|)$ (for $\mathbf{y} \in \mathbb{R}^d$) is the solution of $\Delta U = -U$, i.e.,

$$r^2\partial_r^2 U + (d - 1)r\partial_r U + r^2 U = 0, \tag{4.26}$$

and $U_0 = U(r_0)$, $l = r_0/\sqrt{\lambda}$, $\lambda = \mu^{-4/(d+4)}U_1^{2/(d+4)}$. Here r_0 is the smallest (nonnegative) solution of $\partial_r U(r) = 0$ and $U_1 = \int_{|\mathbf{y}|<r_0}(1 - U_0^{-1}U(|\mathbf{y}|))^2 \, \mathrm{d}\mathbf{y}$ in which $\mathbf{y}$ is in $\mathbb{R}^d$. For $d = 2$, $U(r) = J_0(r)$ is the 0-th Bessel function of the first kind, and for $d = 3$, $U(r) = \mathrm{sinc}(r) = \sin(r)/r$.

The following proposition follows directly from this formula:

Proposition 4.3.2 *When potential $V(\mathbf{x}) = 0$ for all $\mathbf{x} \in \Omega$ and μ is sufficiently small, the volume of the support of the first compressed mode ψ_1 is proportional to $\mu^{2d/(4+d)}$, where the proportionality constant depends only on d. Moreover,*

$$\int_\Omega \left(\frac{1}{\mu}|\psi_1| + \frac{1}{2}|\nabla\psi_1|^2 \right) d\mathbf{x} = C_1\mu^{-4/(4+d)},$$

where C_1 is some fixed constant depending on d.

Proof: From equation (4.25), observe that support of the first compressed mode is a sphere of radius l. Since l is proportional to $\mu^{2/(4+d)}$, the volume of the sphere is proportional to $\mu^{2d/(4+d)}$.

The second part of the proposition also follows from equation (4.25) by straightforward calculations.

4.4 Upper Bounds on the Volume of Support of Compressed Modes

This section contains the second main result of this chapter. In Theorem 4.4.1, we establish an asymptotic upper bound on the volume of the support of compressed modes (CMs) in terms of regularization parameter μ. This proves that CMs are spatially localized.

The first N compressed modes $\{\psi_i\}_{i=1}^N$ are defined by (4.1). Using integration

by parts, we see that ψ_i's solve the following constrained optimization problem

$$\{\psi_1, \ldots, \psi_N\} = \underset{\tilde{\psi}_1,\ldots,\tilde{\psi}_N}{\operatorname{argmin}} \sum_{i=1}^{N} \int_\Omega \left(\frac{1}{\mu}|\tilde{\psi}_i| + \frac{1}{2}|\nabla\tilde{\psi}_i|^2 + V(\mathbf{x})\tilde{\psi}_i{}^2 \right) d\mathbf{x}$$
$$\text{s.t.} \qquad \int_\Omega \tilde{\psi}_j\tilde{\psi}_k d\mathbf{x} = \delta_{jk}. \tag{4.27}$$

Because we are interested in solutions related to physics and engineering, we assume that the above variational problem has solutions that are continuous in Ω. In general, to ensure that a constrained variational problem has a minimizer, the Lagrangian and the functions that define the constraints need to satisfy certain conditions. Typical conditions required to prove existence of minimizers are coercivity condition and convexity of Lagrangian in terms of the differentials. It is easy to see that the Lagrangian in variational problem (4.27) satisfies these two conditions. Moreover, the functions that define the constraints in problem (4.27) are well behaved and bounded above by quadratic functions. Full details on the existence of minimizers to problem (4.27) can be found in standard textbooks on variational calculus (see for example proof of theorem 1 in section 8.4.1 of [26] for general outline of how the proof proceeds).

On the other hand, we do not make any claim on the uniqueness of solutions in variational problem (4.27). In particular, if we eliminate the L^1 term in problem (4.27) (i.e. formally set $\mu = \infty$), then any unitary transformation of solutions $\{\psi_i\}_{i=1}^N$ is also a minimizer to the variational problem.

Theorem 4.4.1 *Using the above notation, there exist $\mu_0 > 0$, depending on values of L, N, and d, such that for $\mu < \mu_0$ the corresponding compressed modes $\{\psi_i\}_{i=1}^N$ satisfy*

$$|supp(\psi_i)| \leq C\mu^{2d/(4+d)} \qquad for \qquad i = 1, \ldots, N.$$

Here, C is a constant whose value depends on N, d, and $\|V\|_\infty$.

Proof: The proof consist of four parts:

Step 1: Finding Euler-Lagrange equations

For any function u, let $p(u)$ denote an element of subdifferential of $|u|$, that is

$$p(u) = \begin{cases} 1 & \text{if } u > 0 \\ \in [-1, 1] & \text{if } u = 0 \\ -1 & \text{if } u < 0. \end{cases}$$

From the theory of variational calculus with constraints (i.e. see for example [26, Chapter 8]) we know that the solutions of (4.27) are weak solutions of the following system of nonlinear boundary value problem:

For $i = 1, \ldots, N$,

$$\frac{1}{\mu} p(\psi_i) + (2V(\mathbf{x}) - 2\lambda_i)\psi_i - \Delta\psi_i - \sum_{j \neq i} \lambda_{ij}\psi_j = 0 \qquad \text{in } \Omega \tag{4.28}$$

where constants λ_i and λ_{ij} (with $\lambda_{ij} = \lambda_{ji}$) are Lagrange multipliers corresponding to orthonormality constraints

$$\int_\Omega \psi_i^2 = 1, \qquad \text{and} \qquad \int_\Omega \psi_i\psi_j = 0, \qquad \text{for } i, j = 1, \ldots, N. \tag{4.29}$$

Satisfying system of Euler-Lagrange equations (4.28) is a necessary but not sufficient condition for solutions of (4.27).

Step 2: Upper bounds for λ_i, $\|\psi_i\|_1$, and $\|\nabla\psi_i\|_2$

For each i multiply both sides of equation (4.28) by $\psi_i(\mathbf{x})$ and integrate over domain Ω:

$$\int_\Omega \left(\frac{1}{\mu} p(\psi_i)\psi_i + (2V(\mathbf{x}) - 2\lambda_i)\psi_i^2 - (\Delta\psi_i)\psi_i - \sum_{j \neq i} \lambda_{ij}\psi_j\psi_i \right) d\mathbf{x} = 0,$$

which, using orthonormality conditions (4.29) and integration by parts, implies that

$$\frac{1}{\mu} \int_\Omega |\psi_i| d\mathbf{x} + 2\int_\Omega V(\mathbf{x})\psi_i^2 d\mathbf{x} - 2\lambda_i + \int_\Omega |\nabla\psi_i|^2 d\mathbf{x} = 0.$$

Therefore,

$$\lambda_i = \frac{1}{2\mu}\int_\Omega |\psi_i| d\mathbf{x} + \int_\Omega V(\mathbf{x})\psi_i^2 d\mathbf{x} + \frac{1}{2}\int_\Omega |\nabla\psi_i|^2 d\mathbf{x}. \tag{4.30}$$

Next, let $\{f_i\}_{i=1}^{i=N}$ be the compressed modes when potential $V(\mathbf{x})$ is zero everywhere; that is:

$$\{f_1,\ldots,f_N\} = \underset{\tilde{f}_1,\ldots,\tilde{f}_N}{\operatorname{argmin}} \sum_{i=1}^{N}\int_\Omega (\frac{1}{\mu}|\tilde{f}_i| + \frac{1}{2}|\nabla\tilde{f}_i|^2) d\mathbf{x} \quad \text{s.t.} \quad \int_\Omega \tilde{f}_j\tilde{f}_k d\mathbf{x} = \delta_{jk}. \tag{4.31}$$

Observe that

$$\begin{aligned}\sum_{i=1}^{N}\int_\Omega (\frac{1}{\mu}|\psi_i| + \frac{1}{2}|\nabla\psi_i|^2) d\mathbf{x} - N\|V\|_\infty &\le \sum_{i=1}^{N}\int_\Omega (\frac{1}{\mu}|\psi_i| + \frac{1}{2}|\nabla\psi_i|^2 + V(\mathbf{x})\psi_i^2) d\mathbf{x} \\ &\le \sum_{i=1}^{N}\int_\Omega (\frac{1}{\mu}|f_i| + \frac{1}{2}|\nabla f_i|^2 + V(\mathbf{x})f_i^2) d\mathbf{x} \\ &\le \sum_{i=1}^{N}\int_\Omega (\frac{1}{\mu}|f_i| + \frac{1}{2}|\nabla f_i|^2) d\mathbf{x} + N\|V\|_\infty,\end{aligned} \tag{4.32}$$

where we used definition (4.27) in the second line. For the first and third line, we used orthonormality of f_j's and ψ_j's to conclude that for each j

$$|\int_\Omega V(\mathbf{x})\psi_j^2| \le \int_\Omega |V(\mathbf{x})|\psi_j^2 \le \|V\|_\infty \quad \text{and} \quad |\int_\Omega V(\mathbf{x})f_j^2| \le \int_\Omega |V(\mathbf{x})|f_j^2 \le \|V\|_\infty. \tag{4.33}$$

Now from Proposition 4.3.2, we know that when potential $V(\mathbf{x})$ is zero everywhere, the first compressed mode f has support whose volume is proportional to $\mu^{2d/(4+d)}$. It follows that for μ sufficiently small, N disjoint copies (i.e. translates) of f can be placed in Ω, and these N functions are a solution for problem (4.31). So Proposition 4.3.2 implies that for μ sufficiently small:

$$\sum_{i=1}^{N}\int_\Omega (\frac{1}{\mu}|f_i| + \frac{1}{2}|\nabla f_i|^2) d\mathbf{x} = C_1 N\mu^{-4/(4+d)},$$

for some fixed constant C_1, depending on d. Using this in equation (4.32) and

rearranging, we have

$$\sum_{i=1}^{N}\int_{\Omega}\frac{1}{\mu}|\psi_i|d\mathbf{x}+\sum_{i=1}^{N}\int_{\Omega}\frac{1}{2}|\nabla\psi_i|^2d\mathbf{x}\leq C_1N\mu^{-4/(4+d)}+2N\|V\|_{\infty}.$$

Because each of the summands in the left hand side of above inequality is positive, there exist constant C_2 (i.e. depending on d, N and $\|V\|_{\infty}$) such that

$$\int_{\Omega}\frac{1}{\mu}|\psi_i|d\mathbf{x}\leq C_2\mu^{-4/(4+d)}\quad\text{and}\quad\int_{\Omega}|\nabla\psi_i|^2d\mathbf{x}\leq C_2\mu^{-4/(4+d)}\quad\text{for } i=1,\ldots,N. \tag{4.34}$$

Moreover, substituting the above inequalities into (4.30) and using (4.33), it follows that for small enough μ,

$$\lambda_i\leq\frac{C_2}{2}\mu^{-4/(4+d)}+\|V\|_{\infty}+\frac{C_2}{2}\mu^{-4/(4+d)}\leq C_3\mu^{-4/(4+d)}. \tag{4.35}$$

Step 3: Upper bounds for λ_{ij}'s

Fix i. For $k\neq i$ multiply both sides of equation (4.28) by $\psi_k(\mathbf{x})$ and integrate over Ω:

$$\int_{\Omega}\left(\frac{1}{\mu}p(\psi_i)\psi_k+(2V(\mathbf{x})-2\lambda_i)\psi_i\psi_k-(\Delta\psi_i)\psi_k-\sum_{j\neq i}\lambda_{ij}\psi_j\psi_k\right)d\mathbf{x}=0,$$

which, using orthonormality conditions (4.29) and integration by parts, implies that

$$\frac{1}{\mu}\int_{\Omega}p(\psi_i)\psi_kd\mathbf{x}+2\int_{\Omega}V(\mathbf{x})\psi_i\psi_kd\mathbf{x}+\int_{\Omega}(\nabla\psi_i)\cdot(\nabla\psi_k)d\mathbf{x}-\lambda_{ik}=0.$$

Therefore,

$$\lambda_{ik}=\frac{1}{\mu}\int_{\Omega}p(\psi_i)\psi_kd\mathbf{x}+2\int_{\Omega}V(\mathbf{x})\psi_i\psi_kd\mathbf{x}+\int_{\Omega}(\nabla\psi_i)\cdot(\nabla\psi_k)d\mathbf{x}. \tag{4.36}$$

In view of (4.34),

$$\left|\frac{1}{\mu}\int_{\Omega}p(\psi_i)\psi_kd\mathbf{x}\right|\leq\frac{1}{\mu}\int_{\Omega}|\psi_k|d\mathbf{x}\leq C_2\mu^{-4/(4+d)}.$$

Also, note that by Cauchy-Schwarz and orthonormality of ψ_i and ψ_k,

$$\begin{aligned}\left|\int_\Omega V(\mathbf{x})\psi_i\psi_k d\mathbf{x}\right| &\leq \|V\|_\infty \int_\Omega |\psi_i||\psi_k| d\mathbf{x} \\ &\leq \|V\|_\infty \left(\int_\Omega \psi_i^2 d\mathbf{x}\right)^{1/2} \left(\int_\Omega \psi_k^2 d\mathbf{x}\right)^{1/2} \\ &= \|V\|_\infty.\end{aligned}$$

Finally, using Cauchy-Schwarz and equation (4.34),

$$\begin{aligned}\int_\Omega (\nabla\psi_i)\cdot(\nabla\psi_k) d\mathbf{x} &\leq \left(\int_\Omega |\nabla\psi_i|^2 d\mathbf{x}\right)^{1/2} \left(\int_\Omega |\nabla\psi_k|^2 d\mathbf{x}\right)^{1/2} \\ &\leq (C_2\mu^{-4/(4+d)})^{1/2}(C_2\mu^{-4/(4+d)})^{1/2} = C_2\mu^{-4/(4+d)}.\end{aligned}$$

Substituting, the last three inequalities into equation (4.36), shows that for small enough μ

$$\lambda_{ik} \leq C_2\mu^{-4/(4+d)} + 2\|V\|_\infty + C_2\mu^{-4/(4+d)} \leq C_4\mu^{-4/(4+d)}. \tag{4.37}$$

Note that constant C_4 depends on d, N, and $\|V\|_\infty$.

Step 4: Bounding the volume of the support of ψ_i's

For each i multiply both sides of equation (4.28) by

$$\operatorname{sgn}(\psi_i) = \begin{cases} 1 & \text{if } \psi_i > 0 \\ 0 & \text{if } \psi_i = 0 \\ -1 & \text{if } \psi_i < 0 \end{cases}$$

and integrate over domain Ω. It follows that

$$\begin{aligned}\frac{1}{\mu}|\operatorname{supp}(\psi_i)| + \int_\Omega (2V(\mathbf{x}) - 2\lambda_i)|\psi_i| d\mathbf{x} - \int_\Omega \operatorname{sgn}(\psi_i)\Delta\psi_i d\mathbf{x} \\ - \sum_{j\neq i} \lambda_{ij} \int_\Omega \psi_j \operatorname{sgn}(\psi_i) d\mathbf{x} = 0.\end{aligned} \tag{4.38}$$

Define

$$\Omega^+ = \{\mathbf{x} \in \Omega : \psi_i(\mathbf{x}) > 0\},$$

and

$$\Omega^- = \{\mathbf{x} \in \Omega : \psi_i(\mathbf{x}) < 0\}.$$

According to Green's formula

$$\int_{\Omega^+} \Delta\psi_i d\mathbf{x} = \int_{\partial\Omega^+} \frac{\partial\psi_i}{\partial\nu} dS \leq 0,$$

where ν is outward pointing unit normal vector along $\partial\Omega^+$. Since ψ_i is positive in Ω^+ and becomes zero on $\partial\Omega^+$ (i.e. recall that ψ_i is continuous), the RHS of above expression is not positive. In order to apply Green's theorem, we need $\partial\Omega^+$ to be continuously differentiable. Nevertheless, if this is not the case, we approximate function ψ_i by a sequence of functions for which the corresponding boundary is C^1, and we can still conclude the above inequality.

With a similar argument, we have that

$$\int_{\Omega^-} \Delta\psi_i d\mathbf{x} \geq 0.$$

Hence,

$$\int_{\Omega} \text{sgn}(\psi)\Delta\psi_i d\mathbf{x} = \int_{\Omega^+} \Delta\psi_i d\mathbf{x} - \int_{\Omega^-} \Delta\psi_i d\mathbf{x} \leq 0.$$

Using the above inequality in (4.38) and rearranging, we conclude that

$$\begin{aligned}
&\frac{1}{\mu}|\text{supp}(\psi_i)| \\
&\leq \int_\Omega (2\lambda_i - 2V(\mathbf{x}))|\psi_i| d\mathbf{x} + \sum_{j\neq i} \lambda_{ij} \int_\Omega |\psi_j| d\mathbf{x} \\
&\leq (2\lambda_i + 2\|V\|_\infty) \int_\Omega |\psi_i| d\mathbf{x} + \sum_{j\neq i} \lambda_{ij} \int_\Omega |\psi_j| d\mathbf{x} \\
&\leq (2C_3\mu^{-4/(4+d)} + 2\|V\|_\infty)(\mu C_2 \mu^{-4/(4+d)}) + (N-1)C_4\mu^{-4/(4+d)}(\mu C_2\mu^{-4/(4+d)}),
\end{aligned}$$

where the last line comes from equations (4.35), (4.37) and (4.34). Hence, for each $i = 1, \ldots, N$, when μ is sufficiently small

$$|\text{supp}(\psi_i)| \leq C\mu^{2d/(4+d)},$$

where C is a constant depending on N, d, and $\|V\|_\infty$. This completes the proof of Theorem 4.4.1. ∎

Remark 4.4.2 *In view of Proposition 4.3.2, the asymptotic bound of Theorem 4.4.1 is tight for the case $V(\mathbf{x}) = 0$.*

Remark 4.4.3 *Note that the upper bound estimate for the volume of the support of compressed modes is independent of the value of L. The value of L becomes important only in determining the value of μ_0. Indeed, as $L \to \infty$, $\mu_0 \to \infty$.*

4.5 Effect of Regularization Term

As noted earlier, the regularization parameter μ controls how spatial localization of the corresponding compressed modes are. Indeed, in Section 4.4 we demonstrated that the volume of the support of compressed modes is bounded above by a quantity that depends on μ. It is of interest to establish relationships between different sets of compressed modes corresponding to different values of regularization parameter μ. To be specific, let $\{\psi_i^{(1)}\}_{i=1}^N$ and $\{\psi_i^{(2)}\}_{i=1}^N$ be the CMs corresponding to parameters μ_1 and μ_2, with $\mu_1 < \mu_2$. That is,

$$\{\psi_1^{(1)}, \ldots, \psi_N^{(1)}\} = \underset{\tilde{\psi}_1,\ldots,\tilde{\psi}_N}{\operatorname{argmin}} \sum_{i=1}^{N} \frac{1}{\mu_1}\|\tilde{\psi}_i\|_1 + \langle \tilde{\psi}_i, \hat{H}\tilde{\psi}_i\rangle \quad \text{s.t.} \quad \langle \tilde{\psi}_j, \tilde{\psi}_k\rangle = \delta_{jk}, \quad (4.39)$$

and

$$\{\psi_1^{(2)}, \ldots, \psi_N^{(2)}\} = \underset{\tilde{\psi}_1,\ldots,\tilde{\psi}_N}{\operatorname{argmin}} \sum_{i=1}^{N} \frac{1}{\mu_2}\|\tilde{\psi}_i\|_1 + \langle \tilde{\psi}_i, \hat{H}\tilde{\psi}_i\rangle \quad \text{s.t.} \quad \langle \tilde{\psi}_j, \tilde{\psi}_k\rangle = \delta_{jk}, \quad (4.40)$$

Proposition 4.5.1 *Using the above notation,*

$$\sum_{i=1}^{N} \|\psi_i^{(1)}\|_1 \leq \sum_{i=1}^{N} \|\psi_i^{(2)}\|_1.$$

Proof: By way of contradiction, assume that

$$\sum_{i=1}^{N} \|\psi_i^{(2)}\|_1 < \sum_{i=1}^{N} \|\psi_i^{(1)}\|_1.$$

Multiplying both sides by $(1/\mu_1 - 1/\mu_2) > 0$, gives

$$\sum_{i=1}^{N}(\frac{1}{\mu_1} - \frac{1}{\mu_2})\|\psi_i^{(2)}\|_1 < \sum_{i=1}^{N}(\frac{1}{\mu_1} - \frac{1}{\mu_2})\|\psi_i^{(1)}\|_1. \tag{4.41}$$

On the other hand, minimality property of (4.40) implies

$$\sum_{i=1}^{N}\frac{1}{\mu_2}\|\psi_i^{(2)}\|_1 + \langle\psi_i^{(2)}, \hat{H}\psi_i^{(2)}\rangle \leq \sum_{i=1}^{N}\frac{1}{\mu_2}\|\psi_i^{(1)}\|_1 + \langle\psi_i^{(1)}, \hat{H}\psi_i^{(1)}\rangle. \tag{4.42}$$

Adding both sides of equations (4.41) and (4.42), yields

$$\sum_{i=1}^{N}\frac{1}{\mu_1}\|\psi_i^{(2)}\|_1 + \langle\psi_i^{(2)}, \hat{H}\psi_i^{(2)}\rangle \leq \sum_{i=1}^{N}\frac{1}{\mu_1}\|\psi_i^{(1)}\|_1 + \langle\psi_i^{(1)}, \hat{H}\psi_i^{(1)}\rangle.$$

This last equation, contradicts the minimality property of (4.39). The result follows. ∎

The above proposition shows that as μ decreases, the sum of the L^1-norm of the corresponding compressed modes decreases monotonically. We conjecture that the sum of the volume of the support of compressed modes also decreases monotonically as μ decreases:

Conjecture 4.5.2 *Using the above notation,*

$$\sum_{i=1}^{N}|\text{supp}(\psi_i^{(1)})| \leq \sum_{i=1}^{N}|\text{supp}(\psi_i^{(2)})|.$$

4.6 Conclusions

This chapter presented two analytic treatments of compressed modes, which were introduced in [50] as a way of finding spatially localized approximate solutions for the Schrödinger operator.

First, we proved consistency results for compressed modes. It is shown that under some necessary assumptions on the spectrum of the Hamiltonian, as the regularization term in the non-convex optimization problem (4.1) vanishes, the

compressed modes converge to a unitary transformation of the eigenfunctions of the Hamiltonian. We also verify a conjecture in [50].

Second, we established an asymptotic upper bound on the volume of the support of compressed modes. This proves that compressed modes are spatially localized.

CHAPTER 5

Theory and Fast Algorithms for Shift Orthogonal Functions

This chapter presents a theoretical analysis of a particular class of basis functions called Shift Orthogonal Basis Functions. These basis functions are used to develop a fast algorithm for projecting a given function to the set of shift orthogonal functions (i.e. the set containing functions with unit L^2 norm that are orthogonal to their prescribed shifts). The algorithm can be parallelized easily and its computational complexity is bounded by $O(M \log(M))$, where M is the number of coefficients used for storing the input.

Let $\Omega = [0, L_1] \times \cdots \times [0, L_d]$ be a bounded, periodic domain. Let $\mathbf{w} = (w_1, \cdots, w_d) \in \mathbb{R}_+^d$ be a basis of a d-dimensional lattice, and define

$$\Gamma_{\mathbf{w}} = \{\mathbf{jw} := (j_1 w_1, \cdots, j_d w_d) \mid \mathbf{j} = (j_1, \cdots, j_d) \in \mathbb{Z}^d\}. \tag{5.1}$$

Note that to make the boundary of Ω periodic, we require that L_i be divisible by w_i, for $i = 1, \ldots, d$. Endow Ω with the usual inner product

$$\langle f, g \rangle = \int_\Omega f^* g \, \mathrm{d}\mathbf{x}.$$

For any function $f(\mathbf{x}) \in L^2(\Omega)$, we say f is *shift orthogonal*, if it satisfies *shift orthogonality constraints*:

$$\int_\Omega f(\mathbf{x})^* f(\mathbf{x} - \mathbf{jw}) d\mathbf{x} = \delta_{\mathbf{j}0} \qquad \text{for all } \mathbf{jw} \in \Gamma_{\mathbf{w}}. \tag{5.2}$$

Shift orthogonality constraints arise naturally in many applications of science and engineering. However, due to the numerical and theoretical challenges in

imposing the shift orthogonality constraints, these constraints have not received much attention in the literature.

In this chapter we propose a very fast algorithm for finding a shift orthogonal function that is closest in L^2 norm to a given function (i.e. projection to the set of shift orthogonal functions). Note that shift orthogonal functions constitute a set and not a vector space. There are many potential applications for the algorithm described in this chapter. As an example, in Section 5.4 we demonstrate how the algorithm increases the speed in computing Compressed Plane Waves (CPWs) described in [51]. Further applications of the algorithm will be investigated in subsequent research projects.

In order to devise the algorithm we introduce the notion of Shift Orthogonal Basis Functions (SOBFs) and develop some theory for them. For one dimensional periodic domain $[0, L_1]$ we call a family of functions

$$\{\xi^i_j(x)\}^{i=\infty, j=L_1/w_1-1}_{i=1,j=0},$$

with $\xi^i_j(x) = \xi^i_0(x - jw_1)$, *Shift Orthogonal Basis Functions (SOBFs)* if they form an orthonormal basis for space $L^2([0, L_1])$. We refer to superscript index i as the *depth index*, and subscript index j as the *shift index*. For higher dimensional periodic domains, a complete set of functions is called SOBFs if they are formed from the tensor product of one dimensional SOBFs. There might be other analogous functions that are not formed from tensor products, but they are not studied here. Section 5.3 describes a concrete example of SOBFs with certain nice properties.

It turns out that SOBFs have been of interest to quantum mechanics and signal analysis communities (i.e. although, the orthonormal basis considered in these literatures are usually defined in domain $\mathbb{R}^d$, instead of periodic bounded domains). In [67, 59], a numerical example of SOBFs, called phase space Wannier functions, are provided that are exponentially localized in time and frequency domain and for which the matrix of the Laplacian operator is near-diagonal (i.e.

see [59, equations 2.1, 2.3 and 2.5]). Reference [22] provides a simple description of certain SOBFs, called Wilson bases, whose Fourier transform have specific bimodal form (see [22, equations 1.9a and 1.9b]). An explicit example of Wilson basis with exponential decay in time and frequency domain is also constructed in [22] using rapidly converging superpositions of Gaussians (i.e. however, the Laplacian matrix is no longer near-diagonal for this basis). Other examples of Wilson basis are presented in [37] and [1].

The remainder of this chapter consists of the following: Section 5.1 develops some theory regarding SOBFs that is used in devising the fast algorithm. Section 5.2 contains the description of the fast algorithm and highlights some of its important properties including its computational complexity. Section 5.3 describes a concrete example of SOBFs, which we call Shift Orthogonal Plane Waves (SOPWs), that have certain nice properties. Section 5.4 gives an application of the fast algorithm in generating CPWs. Section 5.5 presents some concluding remarks. Appendices C, D and E contain proofs for properties of SOPWs mentioned in Section 5.3.

5.1 Shift Orthogonal Basis Functions (SOBFs)

This section develops some theory for SOBFs in multidimensional domains. The objective in this section is to develop methods that are specialized to shift orthogonality and not to develop new tools for tensor analysis. In particular, Lemma 5.1.2 provides a useful characterization of shift orthogonal functions. For the ease of exposition, the concepts are illustrated for 2D domain. The results to other dimensions can be extended in the trivial way.

By scaling, it is assumed without loss of generality that $\Omega = [0, L_1] \times [0, L_2]$ and the length of the shift in each coordinate is unit length (i.e. L_1 and L_2 are replaced by L_1/w_1 and L_2/w_2, respectively). Because SOBFs in 1D form an orthonormal

basis, tensor product can be used to form an orthonormal basis for 2D. That is, any function $f \in L^2(\Omega)$ can be expressed by

$$f(x_1, x_2) = \sum_{i_1,i_2=1}^{\infty} \sum_{j_1=0,j_2=0}^{L_1-1,L_2-1} a_{j_1,j_2}^{i_1,i_2} \xi_{j_1}^{i_1}(x_1) \xi_{j_2}^{i_2}(x_2). \tag{5.3}$$

Remark 5.1.1 *In what follows i_1 and i_2 and their primes (i.e. i_1', i_2'', etc.) take value from $1, 2, \ldots$. Indices j_1, s_1 and their primes take value from the set $\{0, \ldots, L_1 - 1\}$. Indices j_2 and s_2 and their primes take value from the set $\{0, 1, \ldots, L_2 - 1\}$. Also addition and subtraction for indices j_1, s_1 and their primes are performed modulo L_1. Similarly, addition and subtraction for indices j_2, s_2 and their primes are performed modulo L_2.*

Note that

$$\begin{aligned} f(x_1 - s_1, x_2 - s_2) &= \sum_{i_1,i_2=1}^{\infty} \sum_{j_1=0,j_2=0}^{L_1-1,L_2-1} a_{j_1,j_2}^{i_1,i_2} \xi_{j_1}^{i_1}(x_1 - s_1) \xi_{j_2}^{i_2}(x_2 - s_2) \\ &= \sum_{i_1,i_2=1}^{\infty} \sum_{j_1=0,j_2=0}^{L_1-1,L_2-1} a_{j_1,j_2}^{i_1,i_2} \xi_{j_1+s_1}^{i_1}(x_1) \xi_{j_2+s_2}^{i_2}(x_2) \\ &= \sum_{i_1,i_2=1}^{\infty} \sum_{j_1=0,j_2=0}^{L_1-1,L_2-1} a_{j_1-s_1,j_2-s_2}^{i_1,i_2} \xi_{j_1}^{i_1}(x_1) \xi_{j_2}^{i_2}(x_2). \end{aligned} \tag{5.4}$$

Also, suppose

$$g(x_1, x_2) = \sum_{i_1,i_2=1}^{\infty} \sum_{j_1=0,j_2=0}^{L_1-1,L_2-1} b_{j_1,j_2}^{i_1,i_2} \xi_{j_1}^{i_1}(x_1) \xi_{j_2}^{i_2}(x_2).$$

Because SOBFs in 2D form an orthonormal set,

$$\int g(x_1, x_2)^* f(x_1 - s_1, x_2 - s_2) \, \mathrm{d}\mathbf{x} = \sum_{i_1,i_2=1}^{\infty} \sum_{j_1=0,j_2=0}^{L_1-1,L_2-1} {b_{j_1,j_2}^{i_1,i_2}}^* a_{j_1-s_1,j_2-s_2}^{i_1,i_2}. \tag{5.5}$$

We use a specific multi-index notation

$$(i_1, i_2; j_1, j_2)_{i_1=1,i_2=1,j_1=0,j_2=0}^{i_1=\infty,i_2=\infty,j_1=L_1-1,j_2=L_2-1},$$

to refer to the entries of an infinite dimensional vector. To be rigorous, there is a (non-unique) one-to-one and onto mapping

$$\rho : \mathbb{N} \times \mathbb{N} \times \{0, \ldots, L_1 - 1\} \times \{0, \ldots, L_2 - 1\} \to \mathbb{N}.$$

Indeed, by $(i_1, i_2; j_1, j_2)$, we mean $\rho((i_1, i_2; j_1, j_2))$; however, to avoid cumbersome notation, we just use $(i_1, i_2; j_1, j_2)$ to refer to the positive integer instead. As it will be seen shortly, i_1 and j_1 are related to depth index and shift index for the first coordinate, respectively. Similarly, i_2 and j_2 are related to the depth index and shift index for the second coordinate, respectively. We also use multi-index notation

$$(s_1, s_2)_{s_1=0, s_2=0}^{s_1=L_1-1, s_2=L_2-1},$$

to refer to the entries of a vector of length $L_1 L_2$. Again, to be rigorous, there is a (non-unique) one-to-one and onto mapping

$$\tilde{\rho} : \{0, \ldots, L_1 - 1\} \times \{0, \ldots, L_2 - 1\} \to \{1, 2, \ldots, L_1 L_2\}.$$

Indeed, by (s_1, s_2), we mean $\tilde{\rho}((s_1, s_2))$; however, to avoid cumbersome notation, we just use (s_1, s_2) to refer to the elements of $\{1, \ldots, L_1 L_2\}$.

In view of (5.3), a function f can be represented in SOBFs basis using infinite dimensional vector

$$\mathbf{SOBF}(f)(i_1, i_2; j_1, j_2) := a_{j_1, j_2}^{i_1, i_2}. \tag{5.6}$$

Also for any infinite dimensional vector $a \in \mathbb{C}^{\mathbb{N}}$, define

$$\mathbf{ISOBF}(a) := \sum_{i_1, i_2=1}^{\infty} \sum_{j_1=0, j_2=0}^{L_1-1, L_2-1} a(i_1, i_2; j_1, j_2) \xi_{j_1}^{i_1}(x_1) \xi_{j_2}^{i_2}(x_2). \tag{5.7}$$

For any infinite dimensional vector v and all pairs (s_1, s_2), define transformation $S(s_1, s_2)$ in the following way:

$$\tilde{v} = S(s_1, s_2) v \qquad \text{if and only if} \qquad \tilde{v}(i_1, i_2; j_1, j_2) = v(i_1, i_2; j_1 - s_1, j_2 - s_2).$$

Note that from (5.4),

$$\mathbf{SOBF}(f(x_1 - s_1, x_2 - s_2)) = S(s_1, s_2) \mathbf{SOBF}(f(x_1, x_2)).$$

In view of (5.5), for $s_1, s_1' = 0, \ldots, L_1 - 1$ and $s_2, s_2' = 0, \ldots, L_2 - 1$:

$$\langle g(x_1 - s_1, x_2 - s_2), f(x_1 - s_1', x_2 - s_2') \rangle = \langle S(s_1, s_2) \mathbf{SOBF}(g), S(s_1', s_2') \mathbf{SOBF}(f) \rangle. \tag{5.8}$$

Define Set of Shift Orthogonal as follows:

$$\begin{aligned}\mathcal{SSO} &= \{v \in \mathbb{C}^{\mathbb{N}} : \langle v, S(s_1, s_2)v\rangle = \delta_{0s_1}\delta_{0s_2} \text{ for all } s_1 \text{ and } s_2\} \\ &= \{v \in \mathbb{C}^{\mathbb{N}} : \langle S(s_1', s_2')v, S(s_1, s_2)v\rangle = \delta_{s_1's_1}\delta_{s_2's_2} \text{ for all } s_1,\ s_1',\ s_2 \text{ and } s_2'\}.\end{aligned}$$

Observe that $\mathcal{SSO}$ is only a set and not a subspace. Moreover, equation (5.8) implies that $f(x_1, x_2)$ is shift orthogonal if and only if $\mathbf{SOBF}(f) \in \mathcal{SSO}$ (i.e. see Lemma 5.1.2).

Define $\mathcal{B}$-transform to be operator $\mathcal{B} : \mathbb{C}^{\mathbb{N}} \to \mathbb{C}^{\mathbb{N}}$ defined by

$$\mathcal{B}(v)(i_1, i_2; j_1, j_2) = \sum_{\ell_1,\ell_2} e^{i2\pi(\frac{j_1}{L_1}, \frac{j_2}{L_2})\cdot(\ell_1,\ell_2)} v(i_1, i_2; \ell_1, \ell_2). \tag{5.9}$$

The intuition for $\mathcal{B}$-transform is that for every fixed i_1 and i_2, if $v(i_1, i_2;\ :\ ,\ :\ )$ and $\mathcal{B}(v)(i_1, i_2;\ :\ ,\ :\ )$ are thought as $L_1 \times L_2$ matrices then

$$\mathcal{B}(v)(i_1, i_2;\ :\ ,\ :\ ) = L_1L_2\mathcal{F}_{2D}^{-1}(v(i_1, i_2;\ :\ ,\ :\ ),$$

where $\mathcal{F}_{2D}^{-1}$ is the 2D discrete inverse Fourier transform. The inverse of $\mathcal{B}$-transform, $\mathcal{B}^{-1} : \mathbb{C}^{\mathbb{N}} \to \mathbb{C}^{\mathbb{N}}$, is

$$\mathcal{B}^{-1}(v)(i_1, i_2; j_1, j_2) = \frac{1}{L_1L_2} \sum_{\ell_1,\ell_2} e^{-i2\pi(\frac{j_1}{L_1}, \frac{j_2}{L_2})\cdot(\ell_1,\ell_2)} v(i_1, i_2; \ell_1, \ell_2).$$

Similar to the above, if $v(i_1, i_2;\ :\ ,\ :\ )$ and $\mathcal{B}^{-1}(v)(i_1, i_2;\ :\ ,\ :\ )$ are thought as $L_1 \times L_2$ matrices then $\mathcal{B}^{-1}$-transform can be written as

$$\mathcal{B}^{-1}(v)(i_1, i_2;\ :\ ,\ :\ ) = \frac{1}{L_1L_2}\mathcal{F}_{2D}(v(i_1, i_2;\ :\ ,\ :\ )),$$

where $\mathcal{F}_{2D}$ is the 2D discrete Fourier transform.

The importance of $\mathcal{B}$-transform appears in the following two lemmas:

Lemma 5.1.2 *For given function f, the followings are equivalent:*

1. *f is shift orthogonal.*

2. $\mathbf{SOBF}(f) \in \mathcal{SSO}$.

3. *for all* $(j_1, j_2) \in \{0, \ldots, L_1 - 1\} \times \{0, \ldots, L_2 - 1\}$,

$$\|\mathcal{B}(\mathbf{SOBF}(f))(\ :\ ,\ :\ ; j_1, j_2)\|_2 = 1.$$

Lemma 5.1.3 *For given functions f and g, the followings are equivalent:*

1. *for all $s_1, s_1' = 0, \ldots, L_1 - 1$ and $s_2, s_2' = 0, \ldots, L_2 - 1$,*

$$\langle g(x_1 - s_1, x_2 - s_2), f(x_1 - s_1', x_2 - s_2') \rangle = 0.$$

2. *for all $s_1 = 0, \ldots, L_1 - 1$ and $s_2 = 0, \ldots, L_2 - 1$,*

$$\langle \mathbf{SOBF}(g), S(s_1, s_2)\mathbf{SOBF}(f) \rangle = 0.$$

3. *for all $j_1 = 0, \ldots, L_1 - 1$ and $j_2 = 0, \ldots, L_2 - 1$,*

$$\Big\langle \mathcal{B}(\mathbf{SOBF}(g))(\ :\ ,\ :\ ; j_1, j_2), \mathcal{B}(\mathbf{SOBF}(f))(\ :\ ,\ :\ ; j_1, j_2) \Big\rangle = 0.$$

In order to prove the above two lemmas, we need to introduce some more notations and concepts. We use multi-index notation to define matrices in the following way: Identify entries of matrix A by $A(\cdot|\cdot)$, where the index to the left of "|" determines the row number of the entry and the index to the right of "|" determines the column number of the entry. Let W be the $L_1L_2 \times L_1L_2$ matrix defined by

$$W(s_1, s_2|j_1, j_2) = \frac{1}{\sqrt{L_1 L_2}} e^{-i2\pi(\frac{s_1}{L_1}, \frac{s_2}{L_2}) \cdot (j_1, j_2)}.$$

Let W_∞ to be infinite dimensional square matrix defined by

$$W_\infty(i_1, i_2; s_1, s_2|i_1', i_2'; j_1, j_2) = \frac{1}{\sqrt{L_1 L_2}} e^{-i2\pi(\frac{s_1}{L_1}, \frac{s_2}{L_2}) \cdot (j_1, j_2)} \delta_{i_1 i_1'} \delta_{i_2 i_2'}.$$

Finally, for any infinite dimensional vector v, let $\mathrm{circ}(v)$ be the $L_1L_2 \times \mathbb{N}$ matrix defined by

$$\mathrm{circ}(v)(s_1, s_2|i_1, i_2; j_1, j_2) = (S(s_1, s_2)v)(i_1, i_2; j_1, j_2) = v(i_1, i_2; j_1 - s_1, j_2 - s_2).$$

Note that W and W_∞ are unitary matrices (i.e. although the latter is infinite dimensional and by unitary we mean that $W_\infty W_\infty^\dagger$ is infinite dimensional diagonal matrix whose diagonal entries are one). To see this, observe that for example,

$$\begin{aligned}
&(W_\infty W_\infty^\dagger)(i_1,i_2;s_1,s_2|i_1',i_2';s_1',s_2')\\
=&\sum_{i_1'',i_2''}\sum_{s_1'',s_2''} W_\infty(i_1,i_2;s_1,s_2|i_1'',i_2'';s_1'',s_2'')W_\infty^\dagger(i_1'',i_2'';s_1'',s_2''|i_1',i_2';s_1',s_2')\\
=&\sum_{i_1'',i_2''}\sum_{s_1'',s_2''} W_\infty(i_1,i_2;s_1,s_2|i_1'',i_2'';s_1'',s_2'')\overline{W_\infty(i_1',i_2';s_1',s_2'|i_1'',i_2'';s_1'',s_2'')}\\
=&\sum_{i_1'',i_2''}\sum_{s_1'',s_2''} \frac{1}{\sqrt{L_1L_2}}e^{-i2\pi(\frac{s_1}{L_1},\frac{s_2}{L_2})\cdot(s_1'',s_2'')}\delta_{i_1i_1''}\delta_{i_2i_2''}\frac{1}{\sqrt{L_1L_2}}e^{i2\pi(\frac{s_1'}{L_1},\frac{s_2'}{L_2})\cdot(s_1'',s_2'')}\delta_{i_1'i_1''}\delta_{i_2'i_2''}\\
=&\sum_{i_1''}\delta_{i_1i_1''}\delta_{i_1'i_1''}\sum_{i_2''}\delta_{i_2i_2''}\delta_{i_2'i_2''}\sum_{s_1'',s_2''}\frac{1}{L_1L_2}e^{i2\pi(\frac{s_1'-s_1}{L_1},\frac{s_2'-s_2}{L_2})\cdot(s_1'',s_2'')}\\
=&\delta_{i_1i_1'}\delta_{i_2i_2'}\delta_{s_1s_1'}\delta_{s_2s_2'}.
\end{aligned}$$

Another important property that W and W_∞ have is that if $\mathrm{circ}(v)$ is multiplied on left by W and on right by $W_\infty^\dagger$, then

$$(W\mathrm{circ}(v)W_\infty^\dagger)(s_1,s_2|i_1,i_2;j_1,j_2)=\delta_{s_1j_1}\delta_{s_2j_2}\mathcal{B}(v)(i_1,i_2;j_1,j_2). \tag{5.10}$$

To see the above, note that

$$\begin{aligned}
&(W\mathrm{circ}(v)W_\infty^\dagger)(s_1,s_2|i_1,i_2;j_1,j_2)\\
=&\sum_{j_1',j_2'}\sum_{i_1',i_2',s_1',s_2'} W(s_1,s_2|j_1',j_2')\mathrm{circ}(v)(j_1',j_2'|i_1',i_2';s_1',s_2')W_\infty^\dagger(i_1',i_2';s_1',s_2'|i_1,i_2;j_1,j_2)\\
=&\sum_{j_1',j_2'}\sum_{i_1',i_2'}\sum_{s_1',s_2'}\frac{e^{-i2\pi(\frac{s_1}{L_1},\frac{s_2}{L_2})\cdot(j_1',j_2')}}{\sqrt{L_1L_2}}v(i_1',i_2';s_1'-j_1',s_2'-j_2')\frac{e^{i2\pi(\frac{j_1}{L_1},\frac{j_2}{L_2})\cdot(s_1',s_2')}}{\sqrt{L_1L_2}}\delta_{i_1i_1'}\delta_{i_2i_2'}\\
=&\sum_{j_1',j_2'}\sum_{\ell_1,\ell_2}\frac{e^{-i2\pi(\frac{s_1}{L_1},\frac{s_2}{L_2})\cdot(j_1',j_2')}}{\sqrt{L_1L_2}}v(i_1,i_2;\ell_1,\ell_2)\frac{e^{i2\pi(\frac{j_1}{L_1},\frac{j_2}{L_2})\cdot(\ell_1+j_1',\ell_2+j_2')}}{\sqrt{L_1L_2}}\\
=&\sum_{j_1',j_2'}\frac{1}{L_1L_2}e^{i2\pi(\frac{j_1-s_1}{L_1},\frac{j_2-s_2}{L_2})\cdot(j_1',j_2')}\sum_{\ell_1,\ell_2}v(i_1,i_2;\ell_1,\ell_2)e^{i2\pi(\frac{j_1}{L_1},\frac{j_2}{L_2})\cdot(\ell_1,\ell_2)}\\
=&\delta_{s_1j_1}\delta_{s_2j_2}\mathcal{B}(v)(i_1,i_2;j_1,j_2),
\end{aligned}$$

where (5.9) was used for the last equality. Equality (5.10) is very significant: it shows how $\mathrm{circ}(v)$ can be turned into a "pseudo-diagonal" matrix using unitary matrices W and W_∞. Equation (5.10) is used extensively, in the remainder of this section.

Finally, for any two infinite dimensional vectors a and b:

$$(\mathrm{circ}(b)\mathrm{circ}(a)^\dagger)(s_1,s_2|j_1,j_2) = \overline{\langle S(s_1,s_2)b, S(j_1,j_2)a\rangle}. \tag{5.11}$$

To do this, observe that

$$\begin{aligned}
&(\mathrm{circ}(b)\mathrm{circ}(a)^\dagger)(s_1,s_2|j_1,j_2)\\
=&\sum_{i_1,i_2,j_1',j_2'} \mathrm{circ}(b)(s_1,s_2|i_1,i_2;j_1',j_2')\mathrm{circ}(a)^\dagger(i_1,i_2;j_1',j_2'|j_1,j_2)\\
=&\sum_{i_1,i_2,j_1',j_2'} \mathrm{circ}(b)(s_1,s_2|i_1,i_2;j_1',j_2')\overline{\mathrm{circ}(a)(j_1,j_2|i_1,i_2;j_1',j_2')}\\
=&\sum_{i_1,i_2,j_1',j_2'} (S(s_1,s_2)b)(i_1,i_2;j_1',j_2')\overline{(S(j_1,j_2)a)(i_1,i_2;j_1',j_2')}\\
=&\overline{\langle S(s_1,s_2)b, S(j_1,j_2)a\rangle}.
\end{aligned}$$

Now we are ready to prove Lemmas 5.1.2 and 5.1.3.

Proof of Lemma 5.1.2: The equivalence of 1 and 2 follows easily from (5.8) (i.e. with $g = f$) and the definition of $\mathcal{SSO}$. It remains to prove the equivalence between 2 and 3:

Set $v = \mathbf{SOBF}(f)$. Equation (5.11) (i.e., with $a = b = v$) implies that $v \in \mathcal{SSO}$ if and only if $(\mathrm{circ}(v)\mathrm{circ}(v)^\dagger)(s_1,s_2|j_1,j_2) = \delta_{s_1j_1}\delta_{s_2j_2}$ for all s_1, j_1, s_2 and j_2 (i.e. matrix $\mathrm{circ}(v)\mathrm{circ}(v)^\dagger$ is the $L_1L_2 \times L_1L_2$ identity matrix).

Next let

$$V = W\mathrm{circ}(v)W_\infty^\dagger.$$

Compute $VV^\dagger$ in two ways: On one hand, because W_∞ is unitary

$$VV^\dagger = W\mathrm{circ}(v)W_\infty^\dagger W_\infty \mathrm{circ}(v)^\dagger W^\dagger = W\mathrm{circ}(v)\mathrm{circ}(v)^\dagger W^\dagger. \tag{5.12}$$

On the other hand, (5.10) yields that

$$\begin{aligned}
&VV^\dagger(s_1,s_2|j_1,j_2)\\
=&\sum_{i_1,i_2,j_1',j_2'} V(s_1,s_2|i_1,i_2;j_1',j_2')V^\dagger(i_1,i_2;j_1',j_2'|j_1,j_2)\\
=&\sum_{i_1,i_2,j_1',j_2'} V(s_1,s_2|i_1,i_2;j_1',j_2')\overline{V(j_1,j_2|i_1,i_2;j_1',j_2')}\\
=&\sum_{i_1,i_2,j_1',j_2'} \mathcal{B}(v)(i_1,i_2;j_1',j_2')\delta_{s_1j_1'}\delta_{s_2j_2'}\overline{\mathcal{B}(v)(i_1,i_2;j_1',j_2')\delta_{j_1j_1'}\delta_{j_2j_2'}}\\
=&\sum_{i_1,i_2} |\mathcal{B}(v)(i_1,i_2;j_1,j_2)|^2\delta_{s_1j_1}\delta_{s_2j_2}\\
=&\|\mathcal{B}(v)(\ :\ ,\ :\ ;j_1,j_2)\|_2^2\ \delta_{s_1j_1}\delta_{s_2j_2}. \qquad (5.13)
\end{aligned}$$

Now if $\mathrm{circ}(v)\mathrm{circ}(v)^\dagger$ is the identity matrix (i.e. $v \in \mathcal{SSO}$), then from (5.12) and W being unitary, one concludes that $VV^\dagger$ is the identity matrix. Hence, by (5.13), it must be the case that

$$\|\mathcal{B}(v)(\ :\ ,\ :\ ;j_1,j_2)\|_2 = 1,$$

for all $(j_1,j_2) \in \{0,\ldots,L_1-1\}\times\{0,\ldots,L_2-1\}$.

Conversely, if the above holds, then by (5.13), $VV^\dagger$ is the identity matrix. Therefore, because W is a unitary matrix, $W^\dagger VV^\dagger W$ would be the identity matrix as well. Equation (5.12), yields that

$$W^\dagger VV^\dagger W = \mathrm{circ}(v)\mathrm{circ}(v)^\dagger.$$

Hence, $\mathrm{circ}(v)\mathrm{circ}(v)^\dagger$ is the identity matrix, which implies $v \in \mathcal{SSO}$. ∎

Proof of Lemma 5.1.3: The equivalence of 1 and 2 follows easily from (5.8) and the fact that

$$\langle \mathbf{SOBF}(g), S(s_1,s_2)\mathbf{SOBF}(f)\rangle = 0 \quad \text{for all } s_1 \text{ and } s_2,$$

is equivalent to

$$\langle S(s_1,s_2)\mathbf{SOBF}(g), S(s_1',s_2')\mathbf{SOBF}(f)\rangle = 0 \quad \text{for all } s_1,\ s_1',\ s_2 \text{ and } s_2'. \qquad (5.14)$$

It remains to show that (5.14) is equivalent to 3 in Lemma 5.1.3. Let $b = \mathbf{SOBF}(g)$ and $a = \mathbf{SOBF}(f)$. Equation (5.11), shows that (5.14) is equivalent to $(\mathrm{circ}(b)\mathrm{circ}(a)^\dagger)(s_1, s_2|j_1, j_2) = 0$ for all s_1, j_1, s_2 and j_2 (i.e. matrix $\mathrm{circ}(b)\mathrm{circ}(a)^\dagger$ is the $L_1L_2 \times L_1L_2$ zero matrix).

Next let

$$B = W\mathrm{circ}(b)W_\infty^\dagger \qquad \text{and} \qquad A = W\mathrm{circ}(a)W_\infty^\dagger.$$

Compute $BA^\dagger$ in two ways: On one hand, because W_∞ is unitary

$$BA^\dagger = W\mathrm{circ}(b)W_\infty^\dagger W_\infty \mathrm{circ}(a)^\dagger W^\dagger = W\mathrm{circ}(b)\mathrm{circ}(a)^\dagger W^\dagger. \tag{5.15}$$

On the other hand, (5.10) yields that

$$\begin{aligned}
&BA^\dagger(s_1, s_2|j_1, j_2) \\
=&\sum_{i_1,i_2,j_1',j_2'} B(s_1, s_2|i_1, i_2; j_1', j_2')A^\dagger(i_1, i_2; j_1', j_2'|j_1, j_2) \\
=&\sum_{i_1,i_2,j_1',j_2'} B(s_1, s_2|i_1, i_2; j_1', j_2')\overline{A(j_1, j_2|i_1, i_2; j_1', j_2')} \\
=&\sum_{i_1,i_2,j_1',j_2'} \mathcal{B}(b)(i_1, i_2; j_1', j_2')\delta_{s_1j_1'}\delta_{s_2j_2'}\overline{\mathcal{B}(a)(i_1, i_2; j_1', j_2')\delta_{j_1j_1'}\delta_{j_2j_2'}} \\
=&\sum_{i_1,i_2} \mathcal{B}(b)(i_1, i_2; j_1, j_2)\overline{\mathcal{B}(a)(i_1, i_2; j_1, j_2)}\delta_{s_1j_1}\delta_{s_2j_2} \\
=&\overline{\langle \mathcal{B}(b)(\ :\ ,\ :\ ; j_1, j_2), \mathcal{B}(a)(\ :\ ,\ :\ ; j_1, j_2)\rangle}\ \delta_{s_1j_1}\delta_{s_2j_2}.
\end{aligned} \tag{5.16}$$

Now if $\mathrm{circ}(b)\mathrm{circ}(a)^\dagger$ is the zero matrix, then (5.15) implies that $BA^\dagger$ is the zero matrix. Hence, by (5.16), it must be the case that

$$\langle \mathcal{B}(b)(\ :\ ,\ :\ ; j_1, j_2), \mathcal{B}(a)(\ :\ ,\ :\ ; j_1, j_2)\rangle = 0,$$

for all $(j_1, j_2) \in \{0, \ldots, L_1 - 1\} \times \{0, \ldots, L_2 - 1\}$; which shows 3 in Lemma 5.1.3.

Conversely, if 3 holds, then by (5.16), $BA^\dagger$ is the zero matrix. Therefore, $W^\dagger BA^\dagger W$ would also be the zero matrix. Equation (5.15) yields that

$$W^\dagger BA^\dagger W = \mathrm{circ}(b)\mathrm{circ}(a)^\dagger.$$

Hence, $\text{circ}(b)\text{circ}(a)^\dagger$ is the zero matrix, which implies (5.14). ∎

For any function $g \in L^2(\Omega)$, let Πg denote the projection of g into the set of shift orthogonal functions; that is,

$$\Pi g := \underset{f}{\text{argmin}} \, \|g - f\|_2 \qquad \text{subject to } f \text{ being shift orthogonal.}$$

Indeed using SOBFs basis,

$$\Pi g = \mathbf{ISOBF}(Proj_{\mathcal{SSO}}(\mathbf{SOBF}(g))),$$

where for any $b \in \mathbb{C}^{\mathbb{N}}$,

$$Proj_{\mathcal{SSO}}(b) := \underset{v}{\text{argmin}} \, \|b - v\|_2 \qquad \text{subject to} \qquad v \in \mathcal{SSO}.$$

Observe that in the above two definitions the minimum arguments are not necessarily unique, and Πg and $Proj_{\mathcal{SSO}}(b)$ are sets.

Define operator $\Theta : \mathbb{C}^{\mathbb{N}} \to \mathbb{C}^{\mathbb{N}}$ by

$$\Theta(v)(:,:,j_1,j_2) = \begin{cases} \vec{e} & \text{if } v(:,:,j_1,j_2) = \vec{0} \\ \frac{v(:,:,j_1,j_2)}{\|v(\,:\,,\,:\,,j_1,j_2)\|_2} & \text{otherwise,} \end{cases}$$

where $\vec{e}$ is a fixed infinite dimensional real vector with unit L^2 norm.

Lemma 5.1.4 *For any $b \in \mathbb{C}^{\mathbb{N}}$,*

$$\mathcal{B}^{-1}(\Theta(\mathcal{B}(b))) \in Proj_{\mathcal{SSO}}(b).$$

Proof: Suppose $v \in \mathcal{SSO}$ and set

$$p = \mathcal{B}(b) \qquad \text{and } q = \mathcal{B}(v).$$

Note that,

$$\begin{aligned} \|b - v\|_2^2 = \frac{1}{L_1 L_2}\|\text{circ}(b) - \text{circ}(v)\|_F^2 &= \frac{1}{L_1 L_2}\|W\text{circ}(b)W_\infty^\dagger - W\text{circ}(v)W_\infty^\dagger\|_F^2 \\ &= \frac{1}{L_1 L_2}\|\mathcal{B}(b) - \mathcal{B}(v)\|_2^2 = \frac{1}{L_1 L_2}\|p - q\|_2^2, \end{aligned}$$

where equalities similar to (5.10) were used for the second last equality. Now minimizing $\|p-q\|_2$ amounts to solving L_1L_2 subproblems: for every $j_1 = 0, \ldots, L_1-1$ and $j_2 = 0, \ldots, L_2-1$, solve

$$\text{argmin} \sum_{i_1,i_2} |p(i_1,i_2;j_1,j_2) - q(i_1,i_2;j_1,j_2)|^2 \quad \text{subject to } \sum_{i_1,i_2} |q(i_1,i_2;j_1,j_2)|^2 = 1. \tag{5.17}$$

The constraints in the above subproblems are due to $v \in \mathcal{SSO}$ and equivalence of 2 and 3 in Lemma 5.1.2. The solutions to the above subproblems are exactly

$$\begin{cases} q(:,:;j_1,j_2) = p(:,:;j_1,j_2)/\|p(:,:;j_1,j_2)\|_2 & \text{if } \|p(:,:;j_1,j_2)\|_2 \neq 0 \\ \text{any infinite dimensional complex vector with unit } L^2 \text{ norm} & \text{if } \|p(:,:;j_1,j_2)\|_2 = 0, \end{cases}$$

that is, projection of $p(:,:;j_1,j_2)$ into an infinite dimensional ball of radius 1. In particular, set $q = \Theta(p)$ (i.e. choose a fixed real valued vector in the second case above), in which case, $v = \mathcal{B}^{-1}(\Theta(p))$ would be an element of $Proj_{\mathcal{SSO}}(b)$. ■

Remark 5.1.5 *Theorem 2.2 in paper [18] and definition of operator Θ, yields that if an infinite dimensional vector b is real valued, then $\mathcal{B}^{-1}(\Theta(\mathcal{B}(b)))$ is also real valued. This is why in the case $v(:,:,j_1,j_2) = \vec{0}$, we assign to operator Θ the fixed infinite dimensional vector $\vec{e}$; which is real valued and has unit L^2 norm. Indeed, (as it is apparent in the proof of Lemma 5.1.4) had we defined operator Θ to output all infinite dimensional (complex) vector of unit L^2 norm in the case $v(:,:,j_1,j_2) = \vec{0}$, then $\mathcal{B}^{-1}(\Theta(\mathcal{B}(b)))$ would have been a set equal to $Proj_{\mathcal{SSO}}(b)$; however, some elements of $\mathcal{B}^{-1}(\Theta(\mathcal{B}(b)))$ would have been complex valued.*

5.2 Fast Algorithm for Projection to the Set of Shift Orthogonal Functions

For computational purposes, only a finite number of SOBFs basis are used to represent a function. In this section, assume that for the first coordinate all

SOBFs basis whose depth index is smaller or equal to N_1 and for the second coordinate all SOBFs basis whose depth index is smaller or equal to N_2 are used to denote functions in $L^2(\Omega)$. That is,

$$g(x_1, x_2) = \sum_{i_1=1, i_2=1}^{\infty} \sum_{j_1=0, j_2=0}^{L_1-1, L_2-2} b_{j_1,j_2}^{i_1,i_2} \xi_{j_1}^{i_1}(x_1) \xi_{j_2}^{i_2}(x_2) \approx \sum_{i_1, i_2=1}^{N_1, N_2} \sum_{j_1=0, j_2=0}^{L_1-1, L_2-1} b_{j_1,j_2}^{i_1,i_2} \xi_{j_1}^{i_1}(x_1) \xi_{j_2}^{i_2}(x_2).$$

The analysis done in Section 5.1 can be adapted for this situation by simple modification. In particular, index i_1 (and i_1') takes value from $1, \ldots, N_1$ instead of $1, 2, \ldots$ and index i_2 (and i_2') takes value from $1, \ldots, N_2$ instead of $1, 2, \ldots$.

The adapted definition for set $\mathcal{SSO}$ with finite depth indices is

$$\mathcal{SSO}(N_1 N_2) = \{v \in \mathbb{C}^{N_1 N_2 L_1 L_2} : \langle v, S(s_1, s_2) v \rangle = \delta_{0 s_1} \delta_{0 s_2} \text{ for all } s_1 \text{ and } s_2\}.$$

As noted earlier, an important question that arises in optimization problems that involve shift orthogonality constraints is to find Πg for a given function g; that is, find a shift orthogonal function f that minimizes $\|g - f\|_2$. When functions are expressed in terms of tensor product of one dimensional SOBFs basis (i.e., with corresponding depth indices smaller or equal to N_1 and N_2), then the question is equivalent to: given $\vec{b} \in \mathbb{C}^{N_1 N_2 L_1 L_2}$, solve

$$Proj_{\mathcal{SSO}(N_1 N_2)}(\vec{b}) = \operatorname{argmin} \|\vec{b} - \vec{v}\|_2 \qquad \text{subject to} \qquad \vec{v} \in \mathcal{SSO}(N_1 N_2). \quad (5.18)$$

Result of Lemma 5.1.4 in Section 5.1 implies that the solution to problem (5.18) can be obtained using the procedure in Algorithm 1.

Algorithm 1: Projection to $\mathcal{SSO}(N_1N_2)$

Input: $\vec{b}$

Output: $\vec{v} = Proj_{\mathcal{SSO}(N_1N_2)}(\vec{b})$

for $i_1 = 1, \ldots, N_1$ *and* $i_2 = 1, \ldots, N_2$ **do**
  $p(i_1, i_2; \ : \ , \ : \ ) = L_1 L_2 \mathcal{F}_{2D}^{-1}(b(i_1, i_2; \ : \ , \ : \ ))$; // $p = \mathcal{B}(b)$.

for $j_1 = 0, \ldots, L_1 - 1$ *and* $j_2 = 0, \ldots, L_2 - 1$ **do**
  if $\|p(:, :; j_1, j_2)\|_2 \neq 0$ **then**
    $q(:, :; j_1, j_2) = p(:, :; j_1, j_2)/\|p(:, :; j_1, j_2)\|_2$; // $q = \mathcal{B}(v)$.
  else
    $q(:, :; j_1, j_2) = \vec{e}_1$; // $\vec{e}_1$ is the first canonical basis of $\mathbb{R}^{N_1N_2}$.

for $i_1 = 1, \ldots, N_1$ *and* $i_2 = 1, \ldots, N_2$ **do**
  $v(i_1, i_2; \ : \ , \ : \ ) = \frac{1}{L_1 L_2} \mathcal{F}_{2D}(q(i_1, i_2; \ : \ , \ : \ ))$; // $v = \mathcal{B}^{-1}(q)$.

Then, projection of g to the set of shift orthogonal functions is

$$\Pi g = \mathbf{ISOBF}(Proj_{\mathcal{SSO}(N_1N_2)}(\mathbf{SOBF}(g))).$$

All the results that were developed in Section 5.1 for dimension 2 can also be easily adapted for domains with other dimensions. For example suppose $\Omega = [0, L_1] \times [0, L_2] \times [0, L_3]$ (i.e. using appropriate scaling, it is assumed that the length of the shift along each coordinate is 1), and let

$$g(x_1, x_2, x_3) \approx \sum_{i_1=1, i_2=1, i_3=1}^{N_1, N_2, N_3} \sum_{j_1=0, j_2=0, j_3=0}^{L_1-1, L_2-1, L_3-1} b_{j_1, j_2, j_3}^{i_1, i_2, i_3} \xi_{j_1}^{i_1}(x_1) \xi_{j_2}^{i_2}(x_2) \xi_{j_3}^{i_3}(x_3).$$

Three dimensional version of Algorithm 1 is:

Algorithm 2: Projection to $\mathcal{SSO}(N_1N_2N_3)$

Input: $\vec{b}$

Output: $\vec{v} = Proj_{\mathcal{SSO}(N_1N_2N_3)}(\vec{b})$

for $i_1 = 1, \ldots, N_1$, $i_2 = 1, \ldots, N_2$ *and* $i_3 = 1, \ldots, N_3$ **do**
    $p(i_1, i_2, i_3; \ : \ , \ : \ , \ : \ ) = L_1L_2L_3\mathcal{F}_{3D}^{-1}(b(i_1, i_2, i_3; \ : \ , \ : \ , \ : \ ))$;
    // $p = \mathcal{B}(b)$.

for $j_1 = 0, \ldots, L_1 - 1$, $j_2 = 0, \ldots, L_2 - 1$ *and* $j_3 = 0, \ldots, L_3 - 1$ **do**
    if $\|p(:, :, :; j_1, j_2, j_3)\|_2 \neq 0$ **then**
        $q(:, :, :; j_1, j_2, j_3) = p(:, :, :; j_1, j_2, j_3)/\|p(:, :, :; j_1, j_2, j_3)\|_2$;
        // $q = \mathcal{B}(v)$.
    else
        $q(:, :, :; j_1, j_2, j_3) = \vec{e}_1$; // $\vec{e}_1$ is the first canonical basis of $\mathbb{R}^{N_1N_2N_3}$.

for $i_1 = 1, \ldots, N_1$, $i_2 = 1, \ldots, N_2$ *and* $i_3 = 1, \ldots, N_3$ **do**
    $v(i_1, i_2, i_3; \ : \ , \ : \ , \ : \ ) = \frac{1}{L_1L_2L_3}\mathcal{F}_{3D}(q(i_1, i_2, i_3; \ : \ , \ : \ , \ : \ ))$;
    // $v = \mathcal{B}^{-1}(q)$.

Finally, the one dimensional version of Algorithm 1 for domain $\Omega = [0, L]$ (i.e. again using appropriate scaling, it is assumed that the length of the shift is 1) and

$$g(x) \approx \sum_{i=1}^{N} \sum_{j=0}^{L-1} b_j^i \xi_j^i(x),$$

is the following:

Algorithm 3: Projection to $\mathcal{SSO}(N)$

Input: $\vec{b}$

Output: $\vec{v} = Proj_{\mathcal{SSO}(N)}(\vec{b})$

for $i = 1, \ldots, N$ **do**
  $p(i; \ : \ ) = L\mathcal{F}_{1D}^{-1}(b(i; \ : \ ))$; // $p = \mathcal{B}(b)$.
for $j = 0, \ldots, L-1$ **do**
  if $\|p(\ : \ ; j)\|_2 \neq 0$ **then**
    $q(\ : \ ; j) = p(\ : \ ; j)/\|p(\ : \ ; j)\|_2$; // $q = \mathcal{B}(v)$.
  else
    $q(\ : \ ; j) = \vec{e}_1$; // $\vec{e}_1$ is the first canonical basis of $\mathbb{R}^N$.
for $i = 1, \ldots, N$ **do**
  $v(i; \ : \ ) = \frac{1}{L}\mathcal{F}_{1D}(q(i; \ : \ ))$; // $v = \mathcal{B}^{-1}(q)$.

Figure 5.1 shows projection of a constant, an absolute value, a Gaussian, and a sine function on the set of shift orthogonal functions using Algorithm 3.

5.2.1 Computational complexity and important features of the algorithm

This sections describes the computational complexity of Algorithm 1 and highlights some of the important properties of this algorithm. The results stay the same for domains with dimensions other than $d = 2$.

Let $M = L_1 L_2 N_1 N_2$ be the size of input vector $\vec{b}$; which indicates the number of coefficients used to represent the given function. Algorithm 1 consists of three "for" loops. Each iteration in the first and the last "for" loop can be computed using $O(L_1 L_2 \log(L_1 L_2))$ operations via inverse Fast Fourier Transform and Fast Fourier Transform, respectively. Each iteration in the second "loop" can be done

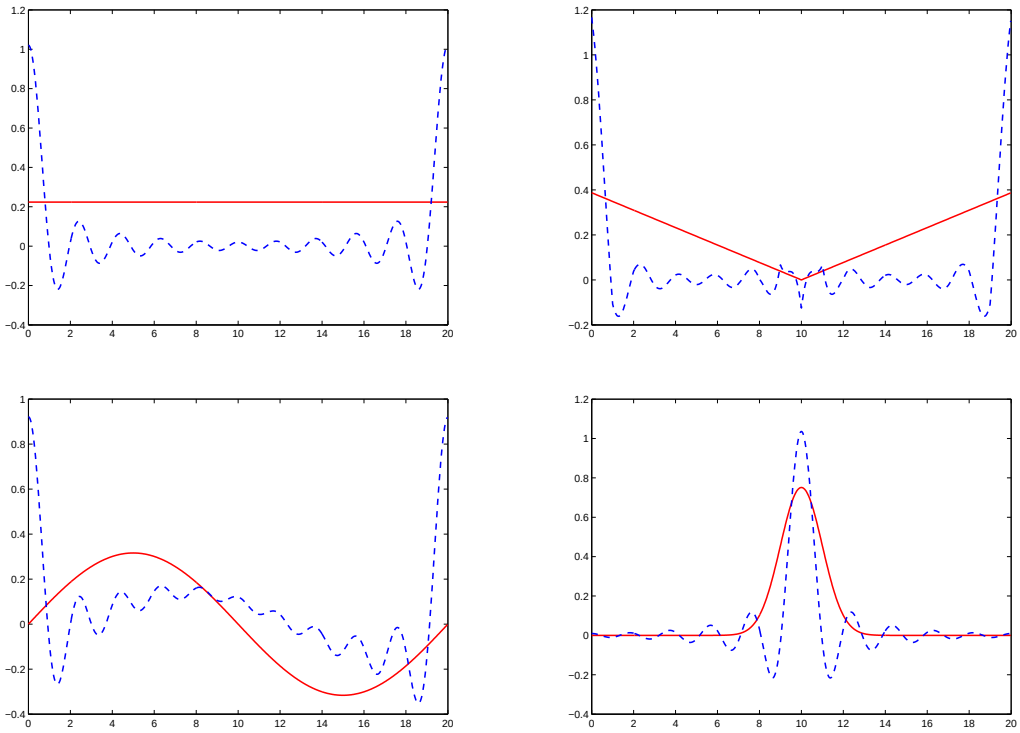

Figure 5.1: Application of Algorithm 3 to find Πg (dashed line), projection of function g (solid line) to the set of shift orthogonal functions. In these examples, function g is constant, absolute value, Gaussian, and sine function with unit L^2 norm. In these computations, SOPWs defined in Section 5.3 are used as the set of SOBFs basis. Here, $L = 20$, $w = 1$ (the length of the shifts) and $N = 6$; that is we are using total of 120 SOPWs to represent functions.

using $O(N_1 N_2)$. Therefore, Algorithm 1 can be performed using

$$N_1 N_2 O(L_1 L_2 \log(L_1 L_2)) + L_1 L_2 O(N_1 N_2) + N_1 N_2 O(L_1 L_2 \log(L_1 L_2))$$

operations, which leads to computational complexity of

$$O(M \log(L_1 L_2)) < O(M \log(M)).$$

Furthermore, note that each of the "for" loops in Algorithms 1 can be done in parallel. This enhances the speed of the algorithm even further and makes it suitable for inputs with large dimensions.

Another nice property of Algorithm 1 is that for real valued input vector $\vec{b}$, it outputs a real valued vector $Proj_{\mathcal{SSO}(N_1N_2)}(\vec{b})$ (i.e. recall Remark 5.1.5). The importance of this property is that if function g and SOBFs $\{\xi_{j_1}^{i_1}(x_1)\xi_{j_2}^{i_2}(x_2)\}$ are real valued then $\mathbf{SOBF}(g)$ would be a real valued vector, and therefore using Algorithm 1, the projected Πg would also be real valued.

5.3 Optimal SOBFs: Shift Orthogonal Plane Waves

This section provides an example of real valued SOBFs with certain nice properties called Shift Orthogonal Plane Waves (SOPWs). As it will be seen shortly, SOPWs are suitable for numerical computation because there exists an exact prescription of them in terms of Fourier basis. Therefore, functions can be expanded in terms of SOPWs very efficiently using FFT and its inverse.

Consider 1D domain $\Omega = [0, L]$ with periodic boundary and by scaling (i.e. replacing L by L/w) assume that $w = 1$. Furthermore, suppose that L is even. This assumption is made so that the formulas provided in this section are easier to express. Nevertheless, the assumption that L is even is not very restrictive as the parity of L is not significant in many applications.

Recall that functions

$$\phi_n(x) = \frac{1}{\sqrt{L}} e^{i2\pi nx/L} \qquad \text{for } n \in \mathbb{Z}, \tag{5.19}$$

as well as

$$\{\frac{1}{\sqrt{L}}, \frac{2}{\sqrt{2L}} \cos(2\pi nx/L), \frac{2}{\sqrt{2L}} \sin(2\pi nx/L)\}_{n=1}^{\infty} \tag{5.20}$$

form orthonormal bases for $L^2(\Omega)$. Denote the L-th roots of unity by

$$\omega_j = e^{i2\pi j/L} \qquad \text{for} \qquad j = 0, \ldots, L-1.$$

Shift Orthogonal Plane Waves (SOPWs) are denoted by

$$\{\theta_j^i(x)\}_{i=1,j=0}^{i=\infty,j=L-1},$$

and defined in the following way: set

$$\theta_j^1(x) = \frac{1}{\sqrt{L}} \sum_{|n|<\frac{L}{2}} \omega_j^{-n} \phi_n(x) + \frac{1}{\sqrt{2L}} \sum_{|n|=\frac{L}{2}} \omega_j^{-n} \phi_n(x) \tag{5.21}$$

$$\theta_j^k(x) = \frac{1}{\sqrt{L}} \sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1} \omega_j^{-n} \phi_n(x) + \\ + \frac{1}{\sqrt{2L}} \sum_{|n|=\frac{(k-1)L}{2}, \frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1} \omega_j^{-n} \phi_n(x) \tag{5.22}$$

Figure 5.2 plots SOPWs given by equations (5.21) and (5.22) with depth index ranging from 1 to 6 and shift index equal to $L/2$ for $L = 20$.

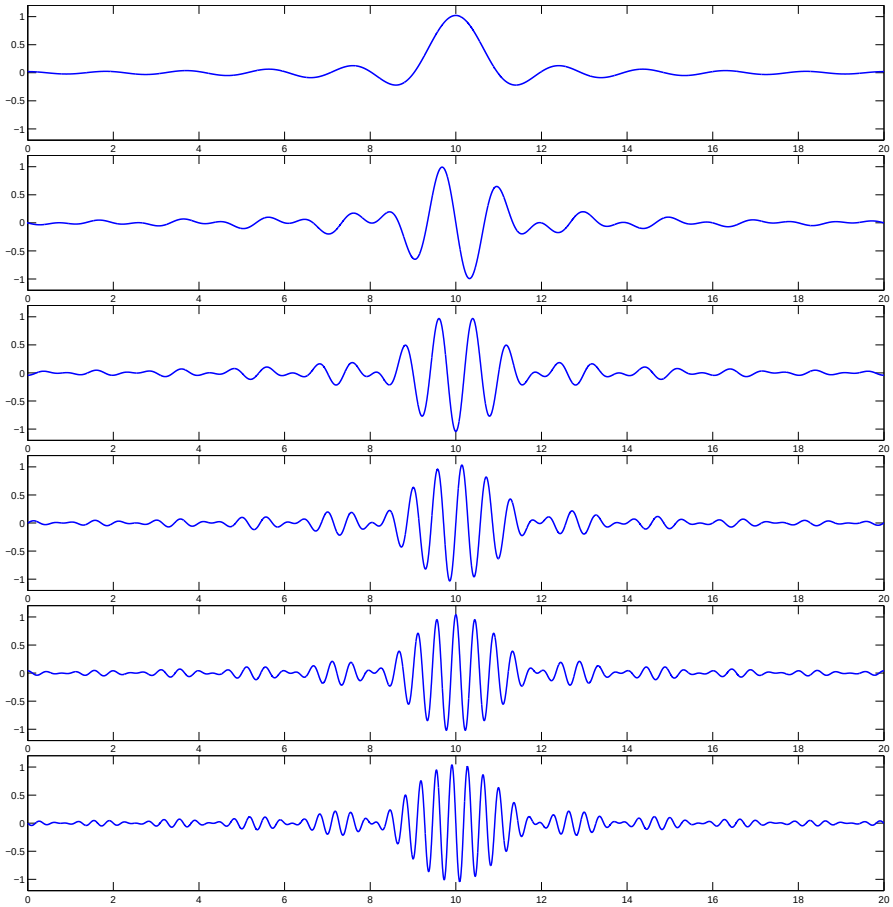

Figure 5.2: From top to bottom, the first 6 SOPWs given by equations (5.21) and (5.22) with distinct depth index and shift index equal to $L/2$ for $L = 20$.

Using the expression

$$\frac{1}{L}\sum_{n=k}^{k+L-1}\omega_j^n = \delta_{j0}, \tag{5.23}$$

and after some calculations one can verify that for $n \geq 1$,

$$\phi_0(x) = \frac{1}{\sqrt{L}}\sum_{j=0}^{L-1}\theta_j^1(x), \tag{5.24}$$

$$\phi_n(x) = \frac{(-\mathrm{sgn}(n)i)^{k-1}}{\sqrt{L}}\sum_{j=0}^{L-1}\omega_j^n\theta_j^k(x) \quad \text{for} \quad \frac{(k-1)L}{2} < |n| < \frac{kL}{2}, \tag{5.25}$$

$$\phi_n(x) = \frac{(-\mathrm{sgn}(n)i)^{k-1}}{\sqrt{2L}}\left(\sum_{j=0}^{L-1}\omega_j^n\theta_j^k(x) - \mathrm{sgn}(n)i\sum_{j=0}^{L-1}\omega_j^n\theta_j^{k+1}(x)\right) \text{ for } |n| = \frac{kL}{2}. \tag{5.26}$$

First note that $\theta_j^i(x) = \theta_0^i(x-j)$. Moreover, it is straightforward using identity (5.23) to verify that $\{\theta_j^i\}_{i=1,j=0}^{i=\infty,j=L-1}$ form an orthonormal set. Finally, $\{\theta_j^i\}_{i=1,j=0}^{i=\infty,j=L-1}$ is complete in $L^2(\Omega)$ because of relations (5.24), (5.25), (5.26) and completeness of $\{\phi_n\}_{n=-\infty}^{\infty}$. Hence, the set of SOPWs defined by (5.21) and (5.22) is an example of SOBFs.

Equations (5.21) and (5.22) can be re-written to represent SOPWs $\{\theta_j^i\}_{i=1,j=0}^{i=\infty,j=L-1}$ in terms of (5.20):

$$\theta_j^1(x) = \frac{1}{L} + \sum_{n=1}^{L/2-1}\frac{2}{L}\cos(\frac{2\pi n(x-j)}{L}) + \frac{\sqrt{2}}{L}\cos(\frac{2\pi(L/2)(x-j)}{L}),$$

for k even,

$$\theta_j^k(x) = \frac{2}{L}\sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}}(-1)^{\frac{k}{2}}\sin(\frac{2\pi n(x-j)}{L}) + \frac{\sqrt{2}}{L}\sum_{n=\frac{(k-1)L}{2},\frac{kL}{2}}(-1)^{\frac{k}{2}}\sin(\frac{2\pi n(x-j)}{L}),$$

for k odd (and $k > 1$),

$$\theta_j^k(x) = \frac{2}{L}\sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}}(-1)^{\frac{k-1}{2}}\cos(\frac{2\pi n(x-j)}{L}) + \frac{\sqrt{2}}{L}\sum_{n=\frac{(k-1)L}{2},\frac{kL}{2}}(-1)^{\frac{k-1}{2}}\cos(\frac{2\pi n(x-j)}{L}),$$

and in closed form:

$$\theta_j^1(x) = \frac{1}{L}\frac{\sin(2\pi(\frac{L-1}{2})(x-j)/L)}{\sin(\pi(x-j)/L)} + \frac{\sqrt{2}}{L}\cos(\frac{2\pi(L/2)(x-j)}{L}),$$

for k even,

$$\theta_j^k(x) = \frac{2}{L}(-1)^{\frac{k}{2}} \sin\left(\frac{2\pi(\frac{kL}{2} - \frac{L}{4})(x-j)}{L}\right)\left[\frac{\sin(\pi(\frac{L}{2}-1)(x-j)/L)}{\sin(\pi(x-j)/L)} + \sqrt{2}\cos(\frac{\pi(x-j)}{2})\right],$$

for k odd (and $k > 1$),

$$\theta_j^k(x) = \frac{2}{L}(-1)^{\frac{k-1}{2}} \cos\left(\frac{2\pi(\frac{kL}{2} - \frac{L}{4})(x-j)}{L}\right)\left[\frac{\sin(\pi(\frac{L}{2}-1)(x-j)/L)}{\sin(\pi(x-j)/L)} + \sqrt{2}\cos(\frac{\pi(x-j)}{2})\right].$$

Equations (5.21), (5.22), (5.24), (5.25), and (5.26) suggest that FFT can be used to switch between Fourier basis and SOPWs efficiently and easily. This is important for computational purposes as it provides an efficient method to represent functions in terms of SOPWs.

Another important property of the SOPWs is that θ_j^i are the solutions to a specific variational problem. This is shown in Appendix C. The final important property of θ_j^i is that for any i and j

$$\partial_{xx}\theta_j^i \in \text{span}\{\theta_k^i\}_{k=0}^{k=L-1}.$$

This can be seen by direct computation (see Appendix E) or using Euler-Lagrange equations for the variational problem (see Appendix D).

A disadvantage that SOPWs have, in comparison to Fourier basis (5.19), is that SOPWs are not eigenfunctions of the derivative operator. Nevertheless, it is shown in Appendix E that for any i and j,

$$\partial_x\theta_j^i \in \text{span}\{\theta_k^{i-1}, \theta_k^i, \theta_k^{i+1}\}_{k=0}^{L-1}.$$

5.4 Application to Solving CPWs

This section outlines how the projection algorithm described in Section 5.2 is used to compute Compressed Plain Waves (CPWs) (i.e., see [51]). The shift orthogonality constraints in the construction of CPWs makes their computation chal-

lenging and numerically inefficient. However, applying the projection algorithm circumvents these difficulties.

Basic compressed plane waves $\{\psi^n\}_{n=1}^{\infty}$ are defined by:

$$\begin{aligned}\psi^1(\mathbf{x}) = \underset{\psi}{\operatorname{argmin}} \frac{1}{\mu}\int_\Omega |\psi(\mathbf{x})|\ \mathrm{d}\mathbf{x} + \int_\Omega \psi(\mathbf{x})\hat{H}_0\psi(\mathbf{x})\ \mathrm{d}\mathbf{x} \\ \text{s.t.} \quad \int_\Omega \psi(\mathbf{x})\psi(\mathbf{x}-\mathbf{j}\mathbf{w})\ \mathrm{d}\mathbf{x} = \delta_{\mathbf{j}0}, \quad \mathbf{j}\in\mathbb{Z}^d,\end{aligned} \tag{5.27}$$

where $\hat{H}_0 = -\frac{1}{2}\Delta$. The higher modes can be recursively defined as:

$$\begin{aligned}\psi^{n+1}(\mathbf{x}) = \underset{\psi}{\operatorname{argmin}} \frac{1}{\mu}\int_\Omega |\psi(\mathbf{x})|\ \mathrm{d}\mathbf{x} + \int_\Omega \psi(\mathbf{x})\hat{H}_0\psi(\mathbf{x})\ \mathrm{d}\mathbf{x} \\ \text{s.t.} \quad \begin{cases} \displaystyle\int_\Omega \psi(\mathbf{x})\psi(\mathbf{x}-\mathbf{j}\mathbf{w})\ \mathrm{d}\mathbf{x} = \delta_{\mathbf{j}0}, & \mathbf{j}\in\mathbb{Z}^d \\ \displaystyle\int_\Omega \psi(\mathbf{x})\psi^i(\mathbf{x}-\mathbf{j}\mathbf{w})\ \mathrm{d}\mathbf{x} = 0, & i = 1,\cdots,n. \end{cases}\end{aligned} \tag{5.28}$$

To simplify our discussion, we only consider $\Omega = [0, L_1]\times[0, L_2]$, in 2D with periodic boundary conditions; the algorithms below can be straightforwardly extended to other dimensions. We use SOPWs given by equations (5.21) and (5.22) as the SOBFs used in Section 5.2. In particular, we expand the given function g in terms of SOPWs:

$$g(x_1, x_2) = \sum_{i_1,i_2=1}^{N_1,N_2}\ \sum_{j_1=0,j_2=0}^{L_1-1,L_2-1} b_{j_1,j_2}^{i_1,i_2}\theta_{j_1}^{i_1}(x_1)\theta_{j_2}^{i_2}(x_2).$$

Operators **SOPW** and **ISOPW** are defined in the similar way to definitions (5.6) and (5.7):

$$\mathbf{SOPW}(g)(i_1, i_2; j_1, j_2) := b_{j_1,j_2}^{i_1,i_2},$$

and

$$\mathbf{ISOPW}(b) := \sum_{i_1,i_2=1}^{N_1,N_2}\ \sum_{j_1=0,j_2=0}^{L_1-1,L_2-1} b_{j_1,j_2}^{i_1,i_2}\theta_{j_1}^{i_1}(x_1)\theta_{j_2}^{i_2}(x_2).$$

Other examples of SOBFs could also be used . For this application, SOPWs were chosen mainly due to the efficiency in calculating the result of operators **SOPW** and **ISOPW** from Fourier coefficient (i.e. recall from Section 5.3 that

FFT and its inverse provide an efficient procedure to switch between representation of a function in Fourier basis ϕ_n's and its representation in SOPWs θ_j^i's).

By introducing an auxiliary variable $u = \psi, v = \psi$, the constrained optimization problem is equivalent to the following problem:

$$\begin{aligned} \psi^1 &= \underset{\psi,u}{\operatorname{argmin}} \frac{1}{\mu} \int |u(\mathbf{x})| \, \mathrm{d}\mathbf{x} + \int \psi \hat{H}_0 \psi \, \mathrm{d}\mathbf{x} \\ \text{s.t.} \quad u = \psi, v = \psi \quad \& \quad & \int v(\mathbf{x}) v(\mathbf{x} - \mathbf{jw}) \, \mathrm{d}\mathbf{x} = \delta_{\mathbf{j}0}, \quad j \in \mathbb{Z}^2, \end{aligned} \tag{5.29}$$

which can be solved by an algorithm based on the Bregman iteration (i.e. see [48, 71, 33]).

Algorithm 4: Solving the first CPW using the projection algorithm

Initialize $u^0 = v^0 = \psi^{1,0}, D^0 = B^0 = 0$.

while *"not converged"* **do**

- $\psi^{1,k} = \underset{\psi}{\operatorname{argmin}} \int \psi \hat{H}_0 \psi \, \mathrm{d}\mathbf{x} + \frac{\lambda}{2} \int (\psi - u^{k-1} + D^{k-1})^2 \, \mathrm{d}\mathbf{x} + \frac{r}{2} \int (\psi - v^{k-1} + B^{k-1})^2 \, \mathrm{d}\mathbf{x}$;
- $v^k = \underset{v}{\operatorname{argmin}} \frac{r}{2} \int (\psi^{1,k} - v + B^{k-1})^2 \, \mathrm{d}\mathbf{x}$, s.t. $\int v(\mathbf{x}) v(\mathbf{x} - \mathbf{jw}) \, \mathrm{d}\mathbf{x} = \delta_{\mathbf{j}0}, \quad \mathbf{j} \in \mathbb{Z}^2$;
- $u^k = \underset{u}{\operatorname{argmin}} \frac{1}{\mu} \int |u| \, \mathrm{d}\mathbf{x} + \frac{\lambda}{2} \int (\psi^{1,k} - u + D^{k-1})^2 \, \mathrm{d}\mathbf{x}$;
- $D^k = D^{k-1} + \psi^{1,k} - u^k$;
- $B^k = B^{k-1} + \psi^{1,k} - v^k$.

All the above sub-optmization problems can be efficiently solved as follows:

$$\begin{aligned} &(2\hat{H}_0 + \lambda + r)\psi^{1,k} = \lambda(u^{k-1} - D^{k-1}) + r(v^{k-1} - B^{k-1}), \\ &v^k = \mathbf{ISOPW}(Proj_{\mathcal{SSO}(N_1 N_2)}(\mathbf{SOPW}(\psi^{1,k} + B^{k-1}))), \\ &u^k = \operatorname{sgn}(\psi^{1,k} + D^{k-1}) \max(0, |\psi^{1,k} + D^{k-1}| - \frac{1}{\lambda\mu}). \end{aligned}$$

Similarly, ψ^{n+1} is obtained by solving the optimization problem (5.28) efficiently. Suppose that the first n levels $\Psi^n = \{\psi^1, \cdots, \psi^n\}$ are already constructed

and let $a^m = \mathbf{SOPW}(\psi^m), m = 1, \cdots, n$. In this case, the goal is to find v^k satisfying

$$v^k = \underset{v}{\operatorname{argmin}} \int (\psi^{n+1,k} - v + B^{k-1})^2 \, \mathrm{d}\mathbf{x},$$
$$\text{s.t.} \begin{cases} \int v(\mathbf{x}) v(\mathbf{x} - \mathbf{jw}) \, \mathrm{d}\mathbf{x} = \delta_{\mathbf{j}0}, & \mathbf{j} \in \mathbb{Z}^2 \\ \int v(\mathbf{x}) \psi^m(\mathbf{x} - \mathbf{jw}) \, \mathrm{d}\mathbf{x} = 0, & m = 1, \cdots, n. \end{cases}$$

Define

$$\mathcal{S}(\Psi^n) = \operatorname{span}\{S(s_1, s_2) a^m\}_{s_1=0, s_2=0, m=1}^{s_1=L_1-1, s_2=L_2-1, m=n}.$$

Using the SOPWs basis, the above problem is equivalent to solving the following problem in SOPWs frequency space:

$$Proj_{\mathcal{SSO}(N_1 N_2) \cap \mathcal{S}(\Psi^n)^\perp}(b) := \operatorname{argmin} \|b - v\|_2 \quad \text{s.t.} \quad v \in \mathcal{SSO}(N_1 N_2) \cap \mathcal{S}(\Psi^n)^\perp. \tag{5.30}$$

In 5.30, the vector b is given, and the objective is to find vector v closest to b that is shift orthogonal and perpendicular to a^1 to a^n.

Lemmas 5.1.2 and 5.1.3 show that in order to solve problem (5.30), for each j_1 and j_2 one needs to find vector z_{j_1,j_2} that is closest to $\mathcal{B}(b)(:,:;j_1,j_2)$, perpendicular to $\mathcal{B}(a^m)(:,:;j_1,j_2)$ for $m = 1, \ldots, n$, and lies on the unit sphere. Note that by Lemmas 5.1.2 and 5.1.3, $\{\mathcal{B}(a^m)(:,:;j_1,j_2)\}_{m=1}^{m=n}$ form an orthonormal set of vectors for each j_1 and j_2, because elements of Ψ^n are constructed such that they are shift orthogonal and orthogonal to shift span of each other. Hence, z_{j_1,j_2} can be computed in two steps:

- $z_{j_1,j_2} = \mathcal{B}(b)(:,:;j_1,j_2) - \sum_{m=1}^{n} \langle \mathcal{B}(a^m)(:,:;j_1,j_2), \mathcal{B}(b)(:,:;j_1,j_2) \rangle \mathcal{B}(a^m)(:,:;j_1,j_2)$,
- $z_{j_1,j_2} = z_{j_1,j_2} / ||z_{j_1,j_2}||_2 \qquad (\text{if } \|z_{j_1,j_2}\|_2 \neq 0)$.

In summary:

Algorithm 5: Projection to $\mathcal{SSO}(N_1N_2) \cap \mathcal{S}(\Psi^n)^\perp$

Input: $b, a^1, \ldots, a^n$

Output: $v = Proj_{\mathcal{SSO}(N_1N_2)\cap\mathcal{S}(\Psi^n)^\perp}(b)$

for $i_1 = 1, \ldots, N_1$ *and* $i_2 = 1, \ldots, N_2$ **do**

  $\mathcal{B}(b)(i_1, i_2; \ : \ , \ : \ ) = L_1L_2\mathcal{F}_{2D}^{-1}(b(i_1, i_2; \ : \ , \ : \ ))$

for $j_1 = 0, \ldots, L_1 - 1$ *and* $j_2 = 0, \ldots, L_2 - 1$ **do**

  $z_{j_1,j_2} = \mathcal{B}(b)(:, :; j_1, j_2) - \sum_{m=1}^{n}\langle\mathcal{B}(a^m)(:, :; j_1, j_2), \mathcal{B}(b)(:, :; j_1, j_2)\rangle\mathcal{B}(a^m)(:, :; j_1, j_2)$ **if** $\|z_{j_1,j_2}\|_2 \neq 0$ **then**

    $\mathcal{B}(v)(:, :; j_1, j_2) = z_{j_1,j_2}/\|z_{j_1,j_2}\|_2$

  else

    $\mathcal{B}(v)(:, :; j_1, j_2)$=a unit real vector orthogonal to $\{\mathcal{B}(a)(:, :; j_1, j_2)\}_{m=1}^{m=n}$.

for $i_1 = 1, \ldots, N_1$ *and* $i_2 = 1, \ldots, N_2$ **do**

  $v(i_1, i_2; \ : \ , \ : \ ) = \frac{1}{L_1L_2}\mathcal{F}_{2D}(\mathcal{B}(v)(i_1, i_2; \ : \ , \ : \ ))$

Therefore, the following algorithm is used to solve for ψ^{n+1}:

Algorithm 6: Solving the n+1-th BCPW using the projection algorithm

Initialize $u^0 = v^0 = \psi^{n+1,0}, D^0 = B^0 = 0$.

while *"not converged"* **do**

  $\psi^{n+1,k}$ solves $(2\hat{H}_0 + \lambda + r)\psi^{n+1,k} = \lambda(u^{k-1} - D^{k-1}) + r(v^{k-1} - B^{k-1})$;

  $v^k = \mathbf{ISOPW}(Proj_{\mathcal{SSO}(N_1N_2)\cap\mathcal{S}(\Psi^n)^\perp}(\mathbf{SOPW}(\psi^{n+1,k} + B^{k-1})))$;

  $u^k = \text{sgn}(\psi^{1,k} + D^{k-1})\max(0, |\psi^{n+1,k} + D^{k-1}| - \dfrac{1}{\lambda\mu})$;

  $D^k = D^{k-1} + \psi^{n+1,k} - u^k$;

  $B^k = B^{k-1} + \psi^{n+1,k} - v^k$.

Figure 5.3 plots the first four BCPWs in 1D using the proposed algorithms. These results are consistent with the results in [51]. Table 5.1, highlights the

computational speed gained by using the new procedure outlined in this section.

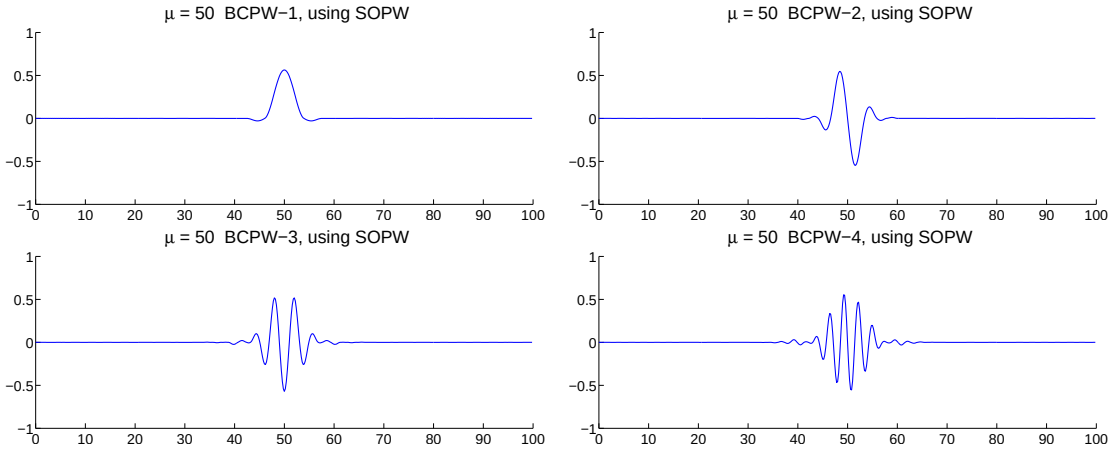

Figure 5.3: The first 4 one-dimensional BCPWs obtained by using Algorithms 4–6.

# of points	Algorithm proposed in [51]					The new procedure				
	ψ^1	ψ^2	ψ^3	ψ^4	total	ψ^1	ψ^2	ψ^3	ψ^4	total
500	23.05	29.39	21.40	9.42	83.26	0.30	1.47	0.44	0.28	2.49
1000	53.11	116.30	81.28	29.99	280.68	0.71	3.48	1.23	0.66	6.08

Table 5.1: CPU time consumption (seconds) for computing the first 4 one-dimensional Basic CPWs using SOPWs and FFT with the same accuracy. For these computations, $\mu = 50$, $L = 100$, and $w = 5$.

5.5 Conclusions

This chapter presented a theoretical analysis of Shift Orthogonal Basis Functions (SOBFs). An example of SOBFs that has several nice properties, called Shift Orthogonal Plane Waves (SOPWs), was given.

We also provided a fast algorithm for finding a closest shift orthogonal function to a given function. The algorithm can be easily implemented using FFT and has computational complexity bounded by $M \log(M)$, where M is the number

of coefficients used to store the input function. The algorithm described here is very useful for problems with shift orthogonality constraints. As an example, the algorithm is applied to computation of Compressed Plain Waves (CPWs).

APPENDIX A

Duality Formulation

Here, it is shown that the Lagrangian dual of (3.10) and (3.11) is of the form (3.12). These results are well known; they are included here for completeness.

Consider the following optimization problem that is of the form (3.10):

$$\min_{y} \|y\|_1 + \frac{1}{2\mu}\|y\|_2^2 + \frac{1}{2t}\|Ay - b\|_2^2. \tag{A.1}$$

Note that the above problem is equivalent to

$$\min_{y,z} \|y\|_1 + \frac{1}{2\mu}\|y\|_2^2 + \frac{1}{2t}\|z\|_2^2 \qquad \text{s.t.} \qquad Ay - b = z. \tag{A.2}$$

The Lagrangian associated with problem (A.2) is

$$\mathcal{L}(y, z; s) = \|y\|_1 + \frac{1}{2\mu}\|y\|_2^2 + \frac{1}{2t}\|z\|_2^2 - s^T(Ay - b - z).$$

To find the dual formulation, we fix s and compute

$$d(s) = \min_{y,z} \mathcal{L}(y, z; s). \tag{A.3}$$

Note that when we fix s, $\mathcal{L}(y, z; s)$ becomes a separable expression coordinate-wise. The corresponding minimization problem for the i-th coordinate is

$$\min_{y_i, z_i} |y_i| + \frac{1}{2\mu}y_i^2 + \frac{1}{2t}z_i^2 - r_i y_i + s_i z_i,$$

where $r = s^T A$. The above problem reduces to two easy one-dimensional minimization problems

$$\min_{y_i}\left(|y_i| + \frac{1}{2\mu}y_i^2 - r_i y_i\right) = \frac{-\mu}{2}\mathsf{shrink}(r_i)^2 \quad \text{and} \quad \min_{z_i}\left(\frac{1}{2t}z_i^2 + s_i z_i\right) = \frac{-t}{2}s_i^2,$$

with optimal solutions:

$$y_i^* = \mu \cdot \mathsf{shrink}(r_i) \qquad \text{and} \qquad z_i^* = -ts_i.$$

Hence,

$$d(s) = s^T b - \frac{\mu}{2}\|\mathsf{shrink}(A^T s)\|_2^2 - \frac{t}{2}\|s\|_2^2,$$

which is of the form (3.12) with $H(x) = x^2/2$.

Next consider the following optimization problem that is of the form (3.11):

$$\min_y \|y\|_1 + \frac{1}{2\mu}\|y\|_2^2 - \sqrt{t^2 - |Ay - b|^2}. \tag{A.4}$$

Note that the above problem is equivalent to

$$\min_{y,z} \|y\|_1 + \frac{1}{2\mu}\|y\|_2^2 - \sqrt{t^2 - \|z\|_2^2} \qquad \text{s.t.} \qquad Ay - b = z. \tag{A.5}$$

The Lagrangian associated with problem (A.2) is

$$\mathcal{L}(y, z; s) = \|y\|_1 + \frac{1}{2\mu}\|y\|_2^2 - \sqrt{t^2 - \|z\|_2^2} - s^T(Ay - b - z).$$

To find the dual formulation, we fix s and compute

$$d(s) = \min_{y,z} \mathcal{L}(y, z; s). \tag{A.6}$$

Note that when we fix s, $\mathcal{L}(y, z; s)$ becomes separable expressions in terms of y and z. Therefore, it suffices to solve the following separate subproblem:

$$\min_y \left(\|y\|_1 + \frac{1}{2\mu}\|y\|_2^2 - (s^T A)y \right), \tag{A.7}$$

$$\min_z \left(-\sqrt{t^2 - \|z\|_2^2} + s^T z \right) \qquad \text{s.t.} \qquad \|z\| \le t. \tag{A.8}$$

The solution to subproblem (A.7) is $y^* = \mu \cdot \mathsf{shrink}(A^T s)$. To solve subproblem (A.8), we take its gradient and set it equal to zero:

$$\frac{z_0}{\sqrt{t^2 - z_0^2}} + s = 0.$$

Rearranging gives,

$$z_0 = -s\sqrt{t^2 - \|z_0\|_2^2}.$$

Taking Euclidean norm from both side and simplifying yields that

$$|z_0\|_2^2 = \frac{t^2\|s\|_2^2}{1 + \|s\|_2^2}.$$

Substituting back into the above equation, implies that

$$z_0 = \frac{-ts}{\sqrt{1 + \|s\|_2^2}}.$$

On the other hand, the minimum value of the objective function (A.8) on the boundary $\|z\|_2 = t$ is $-t\|s\|_2$, which is achieved at $z = -st/\|s\|_2$. Since the value of the objective function in (A.8) at point z_0 is equal to $-t\sqrt{\|s\|_2^2 + 1}$, which is smaller than the minimum value achieve at the boundary, we conclude that the solution to problem (A.8) is indeed:

$$z^* = \frac{-ts}{\sqrt{1 + \|s\|_2^2}}.$$

Hence,

$$d(s) = s^T b - \frac{\mu}{2}\|\mathsf{shrink}(A^T s)\|_2^2 - t\sqrt{\|s\|_2^2 + 1},$$

which is of the form (3.12) with $H(x) = \sqrt{1 + x^2}$.

APPENDIX B

Pseudo-code for the TCCS Algorithm

Here, we provide a pseudo-code for the TCCS algorithm when $H(|\mathbf{s}|) = |\mathbf{s}|^2/2$ and $t \geq 0$, Algorithm 7, and for $t = 0$, Algorithm 8. Indeed, Algorithm 8 is an easy adaptation of Algorithm 7. The output of Algorithms 7 and 8 yields approximate solution to the LASSO problem (3.9) and the Compressed Sensing problem (3.6), respectively.

Some remarks regarding the implementation of the pseudo-codes are in order. In each iteration of the algorithms we need to compute α, which is the projection of b on the space $\{u : A_J^T u = \overrightarrow{\mathbf{1}}\}$. There are several ways to do this. One way is to use formula $b - \tilde{A}_J(\tilde{A}_J^T\tilde{A}_J)^\dagger\tilde{A}_J^T b$ and find the pseudo-inverse via the SVD decomposition. Another way is to use a QR decomposition to compute the least square solution. For example, we can use the expression $\alpha = b - A_J((A_J^T A_J)\backslash(A_J^T b))$ in Matlab for this purpose (i.e. Note that $A_J^T A_J$ is a square matrix, and therefore, backslash operator in Matlab yields the least square solution).

Also observe that in the pseudo-codes, we introduced a tolerance parameter ϵ. Due to numerical machine precision, we need to use lines

$$\|\alpha\|_\infty < \epsilon \qquad \text{and} \qquad J = \{i : |a_i^T s - 1| < \epsilon\},$$

in place of lines

$$\alpha == 0 \qquad \text{and} \qquad J = \{i : a_i^T s == 1\}.$$

The pseudo-codes provided here are meant to clarify the ideas and they do not necessarily implement the TCCS algorithm in the most efficient way.

Algorithm 7: The TCCS algorithm when $H(|\mathbf{s}|) = |\mathbf{s}|^2/2$ and $t \geq 0$. The output y_{opt} of this pseudo-code yields approximate solution to problem (3.9). Also, s_{opt} is the approximate solution to (3.4). Note that ϵ is not a real parameter of the algorithm, it is rather a tolerance parameter that is due numerical machine precision.

Input: A, t, b, and ϵ.

First part: $\tilde{A} = [A \mid -A]$

Choose τ_0 large enough so that $\tilde{A}^T(b/\tau_0) \leq 1$.

$J = \emptyset$, $s = b/\tau_0$

while $r = 0, \ldots$ **do**

  $\alpha =$ projection of b on the space $\{u : A_J^T u = \overrightarrow{\mathbf{1}}\}$.

  if $\|\alpha\|_\infty < \epsilon$ **then**

    // s has reached an extreme point of polytope at τ_r and will not move thereafter.

    $s_{opt} = s$

    break

  end

  $L = \{1, \ldots, n\} \setminus J_r$

  $v = (\overrightarrow{\mathbf{1}} - \tilde{A}_L^T s)./\tilde{A}_L^T \alpha$ // entry-wise division.

  $k =$ minimum positive entry of v.

    // k is the smallest positive number such that $a_i^T(s + k\alpha) = 1$ for some $i \in L$.

  Find τ_{r+1} such that $\frac{1}{\tau_{r+1}} - \frac{1}{\tau_r} = k$.

  if $\tau_{r+1} < t$ **then**

    $\tilde{k} = \frac{1}{t} - \frac{1}{\tau_r}$

    $s_{opt} = s + \tilde{k}\alpha$

    break

  end

  $s = s + k\alpha$

  $J = \{i : |a_i^T s - 1| < \epsilon\}$

end

Second part: $\tilde{y}_{opt} = \vec{0}$

$\tilde{y}_{opt}(J) = \tilde{A}_J^\dagger b - t\tilde{A}_J^\dagger (\tilde{A}_J^T)^\dagger \overrightarrow{\mathbf{1}}$ // set entries of $\tilde{y}_{opt}$ that belong to J.

$y_{opt} = [I \mid -I]\tilde{y}_{opt}$

return y_{opt}

Algorithm 8: The TCCS Algorithm for Compressed Sensing problems. This algorithm is an adaptation of Algorithm 7 when $t = 0$. The output of this pseudo-code yields approximate solution to problem (3.6). Also, s_{opt} is the approximate solution to (3.39). Note that ϵ is not a real parameter of the algorithm, it is rather a tolerance parameter that is due numerical machine precision.

Input: A, b, and ϵ.

First part: $\tilde{A} = [A \mid -A]$

Choose τ_0 large enough so that $\tilde{A}^T(b/\tau_0) \leq 1$.

$J = \emptyset$, $s = b/\tau_0$

while $r = 0, \ldots$ **do**

  $\alpha =$ projection of b on the space $\{u : A_J^T u = \overrightarrow{\mathbf{1}}\}$.

  if $\|\alpha\|_\infty < \epsilon$ **then**

    // s has reached an extreme point of polytope at τ_r and will not move thereafter.

    $s_{opt} = s$

    break

  end

  $L = \{1, \ldots, n\} \setminus J_r$

  $v = (\overrightarrow{\mathbf{1}} - \tilde{A}_L^T s)./\tilde{A}_L^T \alpha$ // entry-wise division.

  $k =$ minimum positive entry of v.

    // k is the smallest positive number such that $a_i^T(s + k\alpha) = 1$ for some $i \in L$.

  Find τ_{r+1} such that $\frac{1}{\tau_{r+1}} - \frac{1}{\tau_r} = k$.

  $s = s + k\alpha$

  $J = \{i : |a_i^T s - 1| < \epsilon\}$

end

Second part: $\tilde{y}_{opt} = \vec{0}$

$\tilde{y}_{opt}(J) = \tilde{A}_J^\dagger b$ // set entries of $\tilde{y}_{opt}$ that belong to J.

$y_{opt} = [I \mid -I]\tilde{y}_{opt}$

return y_{opt}

APPENDIX C

Variational Origin of SOPWs

Here, it is shown that in $1D$, if there is no L^1 term in the definition of CPWs (i.e. $\mu = \infty$) then they are essentially the same as SOPWs given by (5.21) and (5.22). Let $\Omega = [0, L]$ (where L is even) and by scaling assume that $w = 1$. For any function ψ define

$$\mathcal{J}_\infty(\psi) := \int_\Omega \psi \hat{H}_0 \psi \, \mathrm{d}\mathbf{x}. \tag{C.1}$$

As before, $\hat{H}_0 = -\frac{1}{2}\partial_{xx}$. It is clear that $\hat{H}_0$ has eigenfunctions $\phi_n(x) = \frac{1}{\sqrt{L}} e^{i2\pi nx/L}$ with corresponding eigenvalue $\lambda_n = 2(\pi n/L)^2$, $n = 0, \pm 1, \pm 2, \ldots$. Note that Basic CPWs $\{\theta^i\}_{i=1}^{i=\infty}$ when there is no L^1 term (i.e. $\mu = \infty$) are defined in the following way:

$$\begin{aligned}
&\psi^1 = \underset{\psi}{\operatorname{argmin}} \, \mathcal{J}_\infty(\psi) \quad \text{s.t.} \quad \int \psi(x)\psi(x-j)dx = \delta_{j0} \quad \text{for } j = 1, \ldots, L-1, \\
&\psi^k = \underset{\psi}{\operatorname{argmin}} \, \mathcal{J}_\infty(\psi) \quad \text{s.t.} \quad \begin{cases} \int \psi(x)\psi(x-j)dx = \delta_{j0}, \\ \int \psi(x)\psi^i(x-j)dx = 0 \quad \text{for } i = 1, \ldots, k-1. \end{cases}
\end{aligned} \tag{C.2}$$

The main result of this section is the following theorem, which implies $\{\psi^i\}_{i=1}^{i=\infty}$ are indeed SOPWs $\{\theta^i\}_{i=1}^{i=\infty}$:

Theorem C.0.1 *One set of solutions to problem* (C.2) *are*

$$\theta^1(x) = \frac{1}{\sqrt{L}} \sum_{|n|<\frac{L}{2}} \phi_n(x) + \frac{1}{\sqrt{2L}} \sum_{|n|=\frac{L}{2}} \phi_n(x),$$

and, for $k > 1$

$$\theta^k(x) = \frac{1}{\sqrt{L}} \sum_{\frac{(k-1)L}{2} < |n| < \frac{kL}{2}} (sgn(n)i)^{k-1} \phi_n(x) + \frac{1}{\sqrt{2L}} \sum_{|n| = \frac{(k-1)L}{2}, \frac{kL}{2}} (sgn(n)i)^{k-1} \phi_n(x)$$

Proof: Since ϕ_n's form complete set, for any function $\theta(x) \in L^2$ we can write

$$\theta(x) = \sum_{n=-\infty}^{\infty} a(n)\phi_n(x). \tag{C.3}$$

Consequently,

$$\mathcal{J}_\infty(\theta) = \sum_{n=-\infty}^{\infty} |a(n)|^2 \lambda_n = \sum_{n=1}^{\infty} (|a(n)|^2 + |a(-n)|^2)\lambda_n, \tag{C.4}$$

and shift orthogonality constraints yield that

$$\begin{aligned} \delta_{j0} = \int \theta^*(x)\theta(x-j)dx = \sum_{n=-\infty}^{\infty} |a(n)|^2 e^{-i2\pi jn/L} = \\ = |a(0)|^2 + \sum_{n=1}^{\infty} (|a(n)|^2 + |a(-n)|^2)\cos(2\pi jn/L) \\ - i\sum_{n=1}^{\infty} (|a(n)|^2 - |a(-n)|^2)\sin(2\pi jn/L). \end{aligned} \tag{C.5}$$

Therefore, for $\theta(x)$ to be feasible (i.e satisfy shift orthogonality constraints), it must be the case that $|a(n)| = |a(-n)|$ for all $n \geq 1$. Note that changing the phase value of $a(n)$ and $a(-n)$ does not change the value of the objective function (C.4). Thus, the phase factor of $a(n)$ and $a(-n)$ can be chosen in such a way that

$$a(-n) = a^*(n), \qquad \text{for } n = 1, 2, \ldots. \tag{C.6}$$

The above conditions guarantee that $\theta(x)$ is real valued. Hence, there always exist a real valued minimizers for variational problem (C.2). In view of (C.6), equations (C.4) and (C.5) can be re-written: objective function becomes

$$\mathcal{J}_\infty(\theta) = 2\sum_{n=1}^{\infty} |a(n)|^2 \lambda_n, \tag{C.7}$$

and shift orthogonality constraints yield that for $j = 0, 1, \ldots, L-1$,

$$\delta_{j0} = |a(0)|^2 + 2\sum_{n=1}^{\infty} |a(n)|^2 \cos(2\pi jn/L). \tag{C.8}$$

Lets first find θ^1. To that end, the goal is to find $\{a(n)\}_{n=0}^{\infty}$ that minimizes (C.7) and satisfies (C.8) for $j = 0, \ldots, L-1$. Set

$$\begin{cases} c(0) = |a(0)|^2 \\ c(n) = 2|a(n)|^2 \qquad \text{for } n = 1, 2, \ldots. \end{cases} \tag{C.9}$$

Let M be the $L \times L$ matrix whose (j, n)-th entry is $\cos(2\pi jn/L)$ for $j, n = 0, \ldots, L-1$. Let A be the infinite dimensional matrix whose (j, n)-th entry is $\cos(2\pi jn/L)$ for $j = 0, \ldots, L-1$ and $n \in \{0\} \cup \mathbb{N}$. Because

$$\cos(2\pi jn/L) = \cos(2\pi jn'/L) \qquad \text{if } n = n' \pmod{L},$$

matrix A is formed by concatenating infinitely many copies of M side by side, that is

$$A = [M|M|\cdots].$$

Therefore, to find θ^1, one needs to solve the following optimization problem:

$$\underset{c}{\operatorname{argmin}}\, \lambda^T c \qquad \text{s.t.} \qquad Ac = b, c \geq 0, \tag{C.10}$$

where

$$\lambda^T = [\lambda_0, \lambda_1, \cdots], \quad c^T = [c(0), c(1), \cdots], \quad \text{and} \quad b^T = [1, \underbrace{0, \cdots, 0}_{L-1}].$$

Observe that matrix M is not invertible. However, it can be partitioned in the following way: Let $\vec{1} = M(:, 1)$, $M_L = M(:, 2 : L/2-1)$, $\vec{e} = M(:, L/2)$ and $M_R = M(:, L/2+1 : L-1)$. Then

$$A = [\vec{1}|M_L|\vec{e}|M_R|\vec{1}|M_L|\vec{e}|M_R|\cdots].$$

The strategy is to guess the solution to problem (C.10) and then verify (i.e. using the dual formulation of (C.10)) that it is indeed the optimal solution. For this purpose we first prove the following three lemmas:

Lemma C.0.2 *The $L \times (L/2-1)$ matrices $[\vec{1}|M_L|\vec{e}]$ and $[\vec{e}|M_R|\vec{1}]$ have full rank.*

Proof: First observe that matrix $[\vec{e}|M_R|\vec{1}]$ is formed from matrix $[\vec{1}|M_L|\vec{e}]$ if the columns are arranged in the opposite order. Thus it suffices to only show $[\vec{1}|M_L|\vec{e}]$ has full rank. For contrary assume the opposite that $[\vec{1}|M_L|\vec{e}]$ is not full rank, then there exist nontrivial set of constants $\{k_n\}_{n=0}^{n=L/2}$ such that

$$\sum_{n=0}^{L/2} k_n \cos(2\pi jn/L) = 0 \qquad \text{for } j = 0, \ldots, L-1.$$

The above system of equations implies that

$$\sum_{n=0}^{L-1} k'_n e^{-i2\pi jn/L} = 0 \qquad \text{for } j = 0, \ldots, L-1, \tag{C.11}$$

where

$$k'_n = \begin{cases} k_n & \text{for } n = 0 \text{ and } L/2, \\ k_n/2 & \text{for } 0 < n < L/2, \\ k_{L-n}/2 & \text{for } L/2 < n \leq L-1. \end{cases}$$

However, system of equations (C.11) implies that the columns of the $L \times L$ Discrete Fourier Transform matrix are linearly dependent; which contradicts invertibility of the DFT matrix. ■

Lemma C.0.3 *For $k \geq 1$ and $j = 0, \ldots, L-1$:*

$$\frac{1}{L}\cos(\pi j(k-1)) + \frac{2}{L} \sum_{\frac{(k-1)L}{2} < n < \frac{kL}{2}} \cos(2\pi jn/L) + \frac{1}{L}\cos(\pi jk) = \delta_{j0}.$$

Proof: If $j = 0$ the result is clear. For $j = 1, \ldots, L-1$:

$$\begin{aligned}
&\frac{1}{L}\cos(\pi j(k-1)) + \frac{2}{L}\sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}} \cos(2\pi jn/L) + \frac{1}{L}\cos(\pi jk) = \\
=&\frac{1}{L}\cos(\pi j(k-1)) + \frac{2}{L}\sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}} \frac{1}{2}\left[\cos(2\pi jn/L) + \cos(2\pi j(kL-n)/L)\right] + \frac{1}{L}\cos(\pi jk) = \\
=&\frac{1}{L}\sum_{\frac{(k-1)L}{2}\le n<\frac{(k+1)L}{2}} \cos(2\pi jn/L) = \frac{1}{L}Re\left\{\sum_{n=(k-1)L/2}^{(k+1)L/2-1} e^{i2\pi jn/L}\right\} = \\
=&\frac{1}{L}Re\left\{e^{i\pi j(k-1)}\left(\frac{1-e^{i2\pi jL/L}}{1-e^{i2\pi j/L}}\right)\right\} = 0.
\end{aligned}$$

The result follows. ∎

Lemma C.0.4 *Assume that there exist vector $y \in \mathbb{R}^L$ and infinite dimensional vector s such that*

$$A^T y + s = \lambda, \qquad s \ge 0.$$

If c is feasible for problem (C.10)*, then*

$$\lambda^T c \ge y^T b.$$

Furtheremore, if additionally s and c satisfy complementary slackness property $s^T c = 0$, then c is the solution to problem (C.10)*.*

Proof: Observe that for any feasible c in problem (C.10),

$$\lambda^T c = (A^T y + s)^T c = y^T Ac + s^T c = y^T b + s^T c \ge y^T b,$$

where the last inequality is from nonnegativity of s and c. Hence, the minimum value of the objective function in problem (C.10) is $y^T b$. Moreover, the minimum is achieved if c is feasible and satisfies $s^T c = 0$. ∎

Now returning to optimization problem (C.10), set

$$c^T = [\frac{1}{L}, \underbrace{\frac{2}{L}, \ldots, \frac{2}{L}}_{L/2-1}, \frac{1}{L}, 0, 0, \ldots],$$

and find y that satisfies

$$[\vec{1}|M_L|\vec{e}]^T y = [\lambda_0, \ldots, \lambda_{L/2}]^T.$$

Such y exist because by Lemma C.0.2 matrix $[\vec{1}|M_L|\vec{e}]$ is full rank. Finally, set

$$s = \lambda - A^T y.$$

Lemma C.0.3 implies that c is feasible for problem (C.10). Moreover, it is straightforward to verify that

$$s^T = [\underbrace{0, \ldots, 0}_{L/2+1}, \lambda_{\frac{L}{2}+1} - \lambda_{\frac{L}{2}-1}, \lambda_{\frac{L}{2}+2} - \lambda_{\frac{L}{2}-2}, \ldots, \lambda_L - \lambda_0, \lambda_{L+1} - \lambda_1, \ldots$$
$$\ldots, \lambda_{\frac{3L}{2}+1} - \lambda_{\frac{L}{2}-1}, \lambda_{\frac{3L}{2}+2} - \lambda_{\frac{L}{2}-2}, \ldots, \lambda_{2L} - \lambda_0, \lambda_{2L+1} - \lambda_1, \ldots] \geq 0.$$

Thus, by Lemma C.0.4, c is the solution of problem (C.10). Hence, in view of (C.9),

$$\begin{cases} |a(n)| = 1/\sqrt{L} & \text{for } n = \pm 1, \ldots, \pm(\frac{L}{2} - 1), \\ |a(\pm\frac{L}{2})| = 1/\sqrt{2L} & \\ |a(n)| = 0 & \text{otherwise.} \end{cases}$$

Note that any phase values for $a(n)$ as long as (C.6) holds is acceptable. If all the phase factors are set to equal to 1, then

$$\theta^1(x) = \frac{1}{\sqrt{L}} \sum_{|n|<\frac{L}{2}} \phi_n(x) + \frac{1}{\sqrt{2L}} \sum_{|n|=\frac{L}{2}} \phi_n(x).$$

Next, find θ^2. For $j = 0, \ldots, L-1$, define

$$\theta^1_j(x) := \theta^1(x - j).$$

Observe that from relationships (5.24) and (5.25) (i.e. with $k = 1$),

$$\{\phi_n\}_{n=-L/2+1}^{n=L/2-1} \subset \text{span}\{\theta^1_j\}_{j=0}^{j=L-1}.$$

In particular, orthogonality to previous CPWs constraints in problem (C.2), imply that if θ^2 is expanded in the form (C.3), it is necessary (but not sufficient) that $a(0) = \cdots = a(\pm(\frac{L}{2} - 1)) = 0$. Let θ^* be the solution of problem

$$\min_{\theta} \mathcal{J}_\infty(\theta) \quad \text{s.t.} \quad \int \theta(x)\theta(x-j)dx = \delta_{j0}, \tag{C.12}$$

with an additional constraint that if θ^* is expanded in the form (C.3), then $a(n) = 0$ for $|n| < L/2$. Using the same arguments as before, one concludes that to find a candidate for θ^* it is required to solve an optimization problem similar to (C.10); however, this time

$$\lambda^T = [\lambda_{\frac{L}{2}}, \lambda_{\frac{L}{2}+1}, \cdots], \quad c^T = [c(\frac{L}{2}), c(\frac{L}{2}+1), \cdots], \quad \text{and} \quad M = [\vec{e}|M_R|\vec{1}|M_L].$$

Repeating the same line of logic as before, the optimal solution is still

$$c^T = [\frac{1}{L}, \underbrace{\frac{2}{L}, \ldots, \frac{2}{L}}_{L/2-1}, \frac{1}{L}, 0, 0, \ldots].$$

Hence, for θ^*,

$$\begin{cases} |a(\pm\frac{L}{2})| = |a(\pm L)| = 1/\sqrt{2L} & \\ |a(n)| = 1/\sqrt{L} & \text{for } n = \pm(\frac{L}{2}+1), \ldots, \pm(L-1), \\ |a(n)| = 0 & \text{otherwise.} \end{cases}$$

Now observe that θ^* is not necessarily the same as θ^2, as θ^2 satisfies stricter constraints (i.e. orthogonality to $\{\theta_j^1\}_{j=0}^{j=L-1}$) than θ^*. Nevertheless, there is a particular choice of phases for $a(n)$'s for which

$$\theta^*(x) = \frac{1}{\sqrt{L}} \sum_{\frac{L}{2}<|n|<L} (\text{sgn}(n)i)\phi_n(x) + \frac{1}{\sqrt{2L}} \sum_{|n|=\frac{L}{2},L} (\text{sgn}(n)i)\phi_n(x).$$

It is easy to verify that the above function is indeed orthogonal to set $\{\theta_j^1\}_{j=0}^{j=L-1}$. Therefore,

$$\theta^2(x) = \frac{1}{\sqrt{L}} \sum_{\frac{L}{2}<|n|<L} (\text{sgn}(n)i)\phi_n(x) + \frac{1}{\sqrt{2L}} \sum_{|n|=\frac{L}{2},L} (\text{sgn}(n)i)\phi_n(x).$$

Continue the above procedure to find the subsequent BCPWs: for example if θ^3 is expanded in the form (C.3), it is necessary (but not sufficient) that $a(n) = 0$, for $|n| < L$. For from relationships (5.24), (5.25) (i.e. with $k = 1, 2$) and (5.26) (i.e. with $k = 1$):

$$\{\phi_n\}_{n=-(L-1)}^{n=L-1} \subset \text{span}\{\theta_j^1, \theta_j^2\}_{j=0}^{j=L-1},$$

and from the orthogonality to previous CPWs constraints in problem (C.2).

Let θ^* be the solution of problem (C.12) with an additional constraint that if it is expanded in the form (C.3), then $a(n) = 0$ for $|n| < L$. We conclude that for θ^*,

$$\begin{cases} |a(\pm L)| = |a(\pm\frac{3L}{2})| = 1/\sqrt{2L} & \\ |a(n)| = 1/\sqrt{L} & \text{for } n = \pm(L+1), \ldots, \pm(\frac{3L}{2} - 1), \\ |a(n)| = 0 & \text{otherwise.} \end{cases}$$

Note that θ^* is already orthogonal to the space spanned by all shifts of function θ^1. To make θ^* orthogonal to the space spanned by all shifts of function θ^2 (and therefore, derive a formula for θ^3), a particular choice of phase factors for $a(n)$'s are chosen. Consequently,

$$\theta^3(x) = \frac{1}{\sqrt{L}} \sum_{L<|n|<\frac{3L}{2}} (-1)\phi_n(x) + \frac{1}{\sqrt{2L}} \sum_{|n|=L,\frac{3L}{2}} (-1)\phi_n(x).$$

APPENDIX D

Laplacian of the SOPWs

Here, it is shown that for any set of solutions to the variational problem (C.2):

$$\partial_{xx}\theta_j^k \in \text{span}\{\theta_\ell^k\}_{\ell=0}^{\ell=L-1},$$

where $\theta_j^k(x) := \theta^k(x-j)$. Observe that it suffices to show that

$$\partial_{xx}\theta^k \in \text{span}\{\theta_\ell^k\}_{\ell=0}^{\ell=L-1}.$$

From the theory of variational calculus with constraints (i.e. see for example [26, Chapter 8]) at the k-th step (i.e. when θ_j^i for $i = 1, \ldots, k-1$ are already determined), if θ^k is the solution to the variational problem (C.2), then it is the weak solution of the Euler-Lagrange equation

$$\Delta\theta^k = \sum_{i=1}^{k}\sum_{j=0}^{L-1}\lambda_j^i\theta_j^i, \tag{D.1}$$

where constants λ_j^i are the Lagrange multipliers corresponding to the orthonormality constraints:

$$\int \theta^k(x)\theta_j^k(x)\,\mathrm{d}x = \delta_{j0} \quad \text{and} \int \theta^k(x)\theta_j^i(x)\,\mathrm{d}x = 0 \quad \text{for } i = 1, \ldots, k-1.$$

It remains to show that $\lambda_j^i = 0$ for all $i < k$. Fix $n < k$ and $\ell \in \{0, \ldots, L-1\}$. Multiply both sides of (D.1) by θ_ℓ^n, integrate over the domain $[0, L]$, and use orthonormality of $\{\theta_j^i\}_{i=1,j=0}^{i=k,j=L-1}$ and integration by parts to conclude that

$$\lambda_\ell^n = \int \Delta\theta^k(x)\theta_\ell^n(x)\,\mathrm{d}x = \int \theta^k(x)\Delta\theta_\ell^n(x)\,\mathrm{d}x = \int \theta_{L-\ell}^k(x)\Delta\theta^n(x)\,\mathrm{d}x. \tag{D.2}$$

Next, observe that θ^n must satisfy a similar equation to (D.1); that is,

$$\Delta\theta^n = \sum_{i=1}^{n}\sum_{j=0}^{L-1}\gamma_j^i\theta_j^i.$$

From definition of θ^k in (C.2) and because $n < k$, one concludes that $\theta_{L-\ell}^k$ is orthogonal to $\{\theta_j^i\}_{i=1,j=0}^{i=n,j=L-1}$. Therefore, multiplying the above equation by $\theta_{L-\ell}^k$ and integrating over the domain, yields that

$$\int \theta_{L-\ell}^k \Delta\theta^n(x)\ \mathrm{d}x = 0.$$

The above equation and equation (D.2) imply that $\lambda_\ell^n = 0$ as was to be shown.

APPENDIX E

First and Second Derivative of SOPWs

Here, formulas for the first and second derivatives of SOPWs, defined by (5.21) and (5.22), are presented. These results are important in determining the matrix elements of the derivative and the Laplacian operator when SOPWs basis are used.

The first derivative of of SOPWs are given by the following theorem:

Theorem E.0.5 (First Derivatives) *For $k = 1, 2, \ldots$ and $\ell = 0, \ldots, L-1$,*

$$\partial_x \theta_\ell^k = \frac{\pi}{L}\left[-(k-1)\sum_j (-1)^{(k-1)(j-\ell)}\theta_j^{k-1} + \sum_j a(j-\ell)\theta_j^k + k\sum_j (-1)^{k(j-\ell)}\theta_j^{k+1}\right],$$

where

$$a(j-\ell) = \begin{cases} 0 & \text{if } j-\ell = 0, \\ (-1)^k(2k-1)\cot(\pi(j-\ell)/L) & \text{if } j-\ell \text{ is odd}, \\ \cot(\pi(j-\ell)/L) & \text{otherwise}, \end{cases}$$

and dummy variable j takes its values values from $\{0, 1, \ldots, L-1\}$.

The second derivative of SOPWs are given by the following theorem:

Theorem E.0.6 (Second Derivatives) *For $k = 1, 2, \ldots$ and $\ell = 0, \ldots, L-1$,*

$$\partial_{xx}\theta_\ell^k = \frac{-\pi^2}{L^2}\sum_j b(j-\ell)\theta_j^k$$

where

$$b(j-\ell) = \begin{cases} (k^2-k+1/3)L^2+2/3 & \text{if } j-\ell=0, \\ (-1)^k(4k-2)\csc^2(\pi(j-\ell)/L) & \text{if } j-\ell \text{ is odd}, \\ 2\csc^2(\pi(j-\ell)/L) & \text{otherwise}, \end{cases}$$

and dummy variable j takes its values from $\{0,1,\ldots,L-1\}$.

Recall that $\omega_j = e^{i2\pi j/L}$. The following lemma is essential in the proof of the above theorems:

Lemma E.0.7 *For positive integer k, even L and $j = 0,\ldots,L-1$:*

$$\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} n\omega_j^n = \begin{cases} 0 & \text{if } j=0, \\ \frac{-i}{2}L(-1)^k(2k-1)\cot(\pi j/L) & \text{if } j \text{ is odd}, \\ \frac{-i}{2}L\cot(\pi j/L) & \text{otherwise}, \end{cases}$$

and

$$\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} n^2\omega_j^n = \begin{cases} \frac{1}{12}(L-2)L[(3k^2-3k+1)L-1] & \text{if } j=0, \\ \frac{L}{2}(-1)^k(2k-1)\csc^2(\pi j/L) - (-1)^k(2k-1)\frac{L^2}{4} & \text{if } j \text{ is odd}, \\ \frac{L}{2}\csc^2(\pi j/L) - (k^2+(k-1)^2)\frac{L^2}{4} & \text{otherwise}, \end{cases}$$

Proof: It is easy to verify the case $j=0$, so we assume $j \neq 0$. Let

$$c(x) = \sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}} \cos(nx).$$

Then

$$\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} n\omega_j^n = \sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}} n(\omega_j^n - \bar{\omega}_j^n) = 2i \sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}} n\sin(\frac{2\pi nj}{L}) = (-2i)c'(\frac{2\pi j}{L}),$$

and

$$\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} n^2\omega_j^n = \sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}} n^2(\omega_j^n + \bar{\omega}_j^n) = 2 \sum_{\frac{(k-1)L}{2}<n<\frac{kL}{2}} n^2\cos(\frac{2\pi nj}{L}) = (-2)c''(\frac{2\pi j}{L}).$$

It is well known (i.e. for example see [36, page 290]) that

$$c(x) = \frac{\sin((\frac{L}{2}-1)x/2)}{\sin(x/2)} \cos\left(\frac{(2k-1)}{4} Lx\right).$$

The rest of the proof follows from straightforward but tedious calculations: one finds close formulas for $c'(x)$ and $c''(x)$, substitutes $x = 2\pi j/L$ and simplifies. In particular, Table E.1 is helpful in simplifying. ∎

$j \mod 4$	$\cos\left(\frac{\pi j(2k-1)}{2}\right)$	$\sin\left(\frac{\pi j(2k-1)}{2}\right)$	$\cos\left(\frac{\pi j(L/2-1)}{L}\right)$	$\sin\left(\frac{\pi j(L/2-1)}{L}\right)$
0	1	0	$\cos(\pi j/L)$	$-\sin(\pi j/L)$
1	0	$(-1)^{k+1}$	$\sin(\pi j/L)$	$\cos(\pi j/L)$
2	-1	0	$-\cos(\pi j/L)$	$\sin(\pi j/L)$
3	0	$(-1)^k$	$-\sin(\pi j/L)$	$-\cos(\pi j/L)$

Table E.1: Trigonometry identities for integers k, j and even positive number L.

The proof of Theorems E.0.5 and E.0.6 are very similar. The idea of the proof is simple: write SOPWs basis in terms of Fourier basis using formulas (5.21) and (5.22), take appropriate number of derivatives, and then use formulas (5.25) and (5.26) to write back the result in terms of the SOPWs basis.

Proof of Theorem E.0.5: First observe that because $\theta^k_\ell(x) = \theta^k_0(x-\ell)$, it suffices to find $\partial_x\theta^k_0$ and then by shifting, the corresponding formulas for SOPWs with other shift indices follow easily. Now using formulas (5.21) and (5.22),

$$\begin{aligned}
&\partial_x\theta^k_0 \\
=&\frac{1}{\sqrt{L}} \sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1}\partial_x\phi_n + \frac{1}{\sqrt{2L}} \sum_{|n|=\frac{(k-1)L}{2},\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1}\partial_x\phi_n \\
=&\frac{1}{\sqrt{L}} \sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1}(\frac{i2\pi n}{L})\phi_n + \frac{1}{\sqrt{2L}} \sum_{|n|=\frac{(k-1)L}{2},\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1}(\frac{i2\pi n}{L})\phi_n.
\end{aligned} \tag{E.1}$$

Now from equations (5.25) and using Lemma E.0.7,

$$\begin{aligned}
&\frac{1}{\sqrt{L}} \sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1}(\frac{i2\pi n}{L})\phi_n \\
=&\frac{1}{\sqrt{L}} \sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1}(\frac{i2\pi n}{L})\frac{(-\mathrm{sgn}(n)i)^{k-1}}{\sqrt{L}} \sum_{j=0}^{L-1} \omega_j^n \theta_i^k \\
=&\frac{i2\pi}{L^2} \sum_{j=0}^{L-1} \left(\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}} n\omega_j^n \right) \theta_j^k = \frac{\pi}{L} \sum_{j=0}^{L-1} a(j)\theta_j^k,
\end{aligned} \tag{E.2}$$

where

$$a(j) = \begin{cases} 0 & \text{if } j = 0, \\ (-1)^k(2k-1)\cot(\pi j/L) & \text{if } j \text{ is odd}, \\ \cot(\pi j/L) & \text{otherwise.} \end{cases}$$

On the other hand, equation (5.26) implies that for $|n| = \frac{kL}{2}$,

$$\phi_n(x) = \frac{(-\mathrm{sgn}(n)i)^{k-1}}{\sqrt{2L}} \left(\sum_{j=0}^{L-1} (-1)^{kj}\theta_j^k(x) - \mathrm{sgn}(n)i \sum_{j=0}^{L-1} (-1)^{kj}\theta_j^{k+1}(x) \right). \tag{E.3}$$

Therefore,

$$\begin{aligned}
&\frac{1}{\sqrt{2L}} \sum_{|n|=\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1}(\frac{i2\pi n}{L})\phi_n \\
=&\frac{1}{2L}(\frac{i2\pi}{L}) \sum_{|n|=\frac{kL}{2}} \left(n \sum_{j=0}^{L-1} (-1)^{kj}\theta_j^k - |n|i \sum_{j=0}^{L-1} (-1)^{kj}\theta_j^{k+1} \right) \\
=&\frac{\pi}{L} k \sum_{j=0}^{L-1} (-1)^{kj}\theta_j^{k+1}.
\end{aligned} \tag{E.4}$$

Again, equation (5.26) implies that for $|n| = \frac{(k-1)L}{2}$,

$$\phi_n(x) = \frac{(-\mathrm{sgn}(n)i)^{k-2}}{\sqrt{2L}} \left(\sum_{j=0}^{L-1} (-1)^{(k-1)j}\theta_j^{k-1}(x) - \mathrm{sgn}(n)i \sum_{j=0}^{L-1} (-1)^{(k-1)j}\theta_j^k(x) \right). \tag{E.5}$$

Therefore,

$$\begin{aligned}
&\frac{1}{\sqrt{2L}}\sum_{|n|=\frac{(k-1)L}{2}}(\mathrm{sgn}(n)i)^{k-1}(\frac{i2\pi n}{L})\phi_n\\
=&\frac{1}{2L}(\frac{i2\pi}{L})\sum_{|n|=\frac{(k-1)L}{2}}\left(|n|i\sum_{j=0}^{L-1}(-1)^{(k-1)j}\theta_j^{k-1}+n\sum_{j=0}^{L-1}(-1)^{(k-1)j}\theta_j^k\right)\\
=&-\frac{\pi}{L}(k-1)\sum_{j=0}^{L-1}(-1)^{(k-1)j}\theta_j^{k-1}. \qquad \text{(E.6)}
\end{aligned}$$

Substituting (E.2), (E.4) and (E.6) into equation (E.1) yields that

$$\partial_x\theta_0^k=\frac{\pi}{L}\left[-(k-1)\sum_{j=0}^{L-1}(-1)^{(k-1)j}\theta_j^{k-1}+\sum_{j=0}^{L-1}a(j)\theta_j^k+k\sum_{j=0}^{L-1}(-1)^{kj}\theta_j^{k+1}\right].$$

This completest the proof. ∎

Proof of Theorem E.0.6: Again observe that because $\theta_\ell^k(x)=\theta_0^k(x-\ell)$, it suffices to find $\partial_{xx}\theta_0^k$ and then by shifting, the corresponding formulas for SOPWs with other shift indices follow easily. Now using formulas (5.21) and (5.22),

$$\begin{aligned}
&\partial_{xx}\theta_0^k\\
=&\frac{1}{\sqrt{L}}\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}}(\mathrm{sgn}(n)i)^{k-1}\partial_{xx}\phi_n+\frac{1}{\sqrt{2L}}\sum_{|n|=\frac{(k-1)L}{2},\frac{kL}{2}}(\mathrm{sgn}(n)i)^{k-1}\partial_{xx}\phi_n\\
=&\frac{1}{\sqrt{L}}\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}}(\mathrm{sgn}(n)i)^{k-1}(\frac{-4\pi^2n^2}{L^2})\phi_n+\frac{1}{\sqrt{2L}}\sum_{|n|=\frac{(k-1)L}{2},\frac{kL}{2}}(\mathrm{sgn}(n)i)^{k-1}(\frac{-4\pi^2n^2}{L^2})\phi_n.
\end{aligned} \qquad \text{(E.7)}$$

Now from equations (5.25) and using Lemma E.0.7,

$$\begin{aligned}
&\frac{1}{\sqrt{L}}\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}}(\mathrm{sgn}(n)i)^{k-1}(\frac{-4\pi^2n^2}{L^2})\phi_n\\
=&\frac{1}{\sqrt{L}}\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}}(\mathrm{sgn}(n)i)^{k-1}(\frac{-4\pi^2n^2}{L^2})\frac{(-\mathrm{sgn}(n)i)^{k-1}}{\sqrt{L}}\sum_{j=0}^{L-1}\omega_j^n\theta_i^k\\
=&\frac{-4\pi^2}{L^3}\sum_{j=0}^{L-1}\left(\sum_{\frac{(k-1)L}{2}<|n|<\frac{kL}{2}}n^2\omega_j^n\right)\theta_j^k\\
=&\frac{-\pi^2}{L^2}\sum_{j=0}^{L-1}\tilde{b}(j)\theta_j^k, \qquad \text{(E.8)}
\end{aligned}$$

where

$$\tilde{b}(j) = \begin{cases} (L-2)[(3k^2-3k+1)L-1]/3 & \text{if } j=0, \\ (-1)^k(4k-2)\csc^2(\pi j/L) - (-1)^k(2k-1)L & \text{if } j \text{ is odd}, \\ 2\csc^2(\pi j/L) - (k^2+(k-1)^2)L & \text{otherwise.} \end{cases}$$

On the other hand, from (E.3),

$$\begin{aligned} &\frac{1}{\sqrt{2L}} \sum_{|n|=\frac{kL}{2}} (\mathrm{sgn}(n)i)^{k-1}(\frac{-4\pi^2 n^2}{L^2})\phi_n \\ =&\frac{1}{2L}(\frac{-4\pi^2}{L^2}) \sum_{|n|=\frac{kL}{2}} \left(n^2 \sum_{j=0}^{L-1} (-1)^{kj}\theta_j^k - n^2 \mathrm{sgn}(n) i \sum_{j=0}^{L-1} (-1)^{kj}\theta_j^{k+1} \right) \\ =&\frac{-\pi^2}{L^2} k^2 L \sum_{j=0}^{L-1} (-1)^{kj}\theta_j^k. \end{aligned} \tag{E.9}$$

Also from (E.5),

$$\begin{aligned} &\frac{1}{\sqrt{2L}} \sum_{|n|=\frac{(k-1)L}{2}} (\mathrm{sgn}(n)i)^{k-1}(\frac{-4\pi^2 n^2}{L^2})\phi_n \\ =&\frac{1}{2L}(\frac{-4\pi^2}{L^2}) \sum_{|n|=\frac{(k-1)L}{2}} \left(n^2 \mathrm{sgn}(n) i \sum_{j=0}^{L-1} (-1)^{(k-1)j}\theta_j^{k-1} + n^2 \sum_{j=0}^{L-1} (-1)^{(k-1)j}\theta_j^k \right) \\ =&\frac{-\pi^2}{L^2}(k-1)^2 L \sum_{j=0}^{L-1} (-1)^{(k-1)j}\theta_j^k. \end{aligned} \tag{E.10}$$

Substituting (E.8), (E.9) and (E.10) into equation (E.7) and simplifying yields that

$$\partial_{xx}\theta_0^k = \frac{-\pi^2}{L^2} \sum_{j=0}^{L-1} b(j)\theta_j^k.$$

This completest the proof. ■

References

[1] P. Auscher, *Remarks on the local Fourier bases*, In J.J. Benedetto and M. Frazier (eds.), Wavelets: Mathematics and Applications, CRC Press, Boca Raton, (1994), pp. 203–218.

[2] F. Barekat, *On the consistency of compressed modes for variational problems*, arXiv preprint, arXiv:1310.4552, 2013.

[3] F. Barekat, R. Caflisch, and S. Osher, *On the Support of Compressed Modes*, UCLA CAM Reports:14–14, 2014.

[4] J. Barzilai and J. M. Borwein, *Two-point step size gradient methods*, IMA J. Numer. Anal., 8 (1988), pp. 141–148.

[5] G. A. Bird, *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*, Claredon, Oxford, 1994.

[6] A.B. Bortz, M.H. Kalos, and J.L. Lebowitz, *A new algorithm for Monte Carlo simulation of Ising spin systems*, J. Comp. Phys., 17 (1975), pp. 10–18.

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations and Trends in Machine Learning, 3 (2011), pp. 1–122.

[8] H. Brezis, *Solutions with compact support of variational inequalities*, dedicated to the memory of I.G. Petrovski, Uspekhi Mat. Nauk, 29 (1974), pp. 103–108.

[9] H. Brezis and A. Friedman, *Estimates on the support of solutions of parabolic variational inequalities*, Ill. J. Math., 20 (1976), pp. 82–97.

[10] M. Burger, M. Moller, M. Benning, and S. Osher, *An adaptive inverse scale space method for compressed sensing*, Mathematics of Computation, 82 (2013), pp. 269–299.

[11] R.E. Caflisch, *Monte Carlo and Quasi-Monte Carlo Methods*, Acta Numerica (1998), pp. 1–49.

[12] R.E. Caflisch, S.J. Osher, H. Schaeffer, and G. Tran, *PDEs with Compressed Solutions*, arXiv preprint, arXiv:1311.5850

[13] E.J. Candes and T. Tao, *Decoding by linear programming*, IEEE Transactions on Information Theory, 51 (2005), pp. 4203–4215.

[14] E.J. Candès, Y.C. Eldar, T. Strohmer, and V. Voroninski, *Phase retrieval via matrix completion*, SIAM Journal on Imaging Sciences, 6 (2013), pp. 199–225.

[15] E. J. Candès, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Transactions on Information Theory, 52 (2006), pp.489–509.

[16] Y. Cao, H. Li, L.R. Petzold, *Efficient formulation of the stochastic simulation algorithm for chemically reacting systems*, J. Chem. Phys., 121 (2004), pp. 4059–4067.

[17] S. Chen and D. Donoho, *Basis Pursuit*, Conference on Signals, Systems and Computers. 1 (1994), pp. 41–44.

[18] M.T. Chu, R.J. Plemmons, *Real-valued, low rank, circulant approximation*, SIAM J. Matrix Anal. Appl., 24 (2003), pp. 645–659.

[19] S. Chen, D. Donoho, and M. Saunders, *Atomic decomposition by basis pursuit*, SIAM Rev., 43 (2001), pp. 129–159.

[20] J. Darbon, *On convex finite-dimensional variational methods in imaging sciences and Hamilton-Jacobi equations*, UCLA CAM report:13-59, 2013.

[21] J. Darbon and S. Osher, *Initial Value Problems for Hamilton-Jacobi equations and Sparsity for Linear Systems via ℓ^1 related optimization*, preprint, 2013.

[22] I. Daubechies, S. Jaffard, J.L. Journe, *A simple Wilson orthonormal basis with exponential decay*, SIAM J. Math. Anal., 22 (1991), pp. 554–572.

[23] I. Deak, *An Economical Method for Random Number Generation and a Normal Generator*, Computing 27 (1981), pp. 113–121.

[24] D.L. Donoho, *Compressed sensing*, IEEE Transactions on Information Theory, 52 (2006), pp. 1289–1306.

[25] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, *Least Angle Regression*, Annals of Statistics, 32 (2004), pp. 407–499.

[26] L.C. Evans, *Partial Differential Equations*, American Mathematical Society, 1998.

[27] M.P. Friedlander and P. Tseng, *Exact Regularization of Convex Problems*, SIAM J. Optim, 18 (2007), pp. 1326–1350 .

[28] M.A. Gibson and J. Bruck, *Exact stochastic simulation of chemical systems with many species and many channel*, J. Phys. Chem., 105 (2000), pp. 1876–1889.

[29] W. R. Gilks, N. G. Best, and K. K. C. Tan, *Adaptive rejection Metropolis sampling*, Applied Statistics, 44 (1995), pp. 455–472.

[30] W. R. Gilks and P. Wild, *Adaptive rejection sampling for Gibbs sampling*, *Applied Statistics*, 41 (1992), pp. 337–348.

[31] D.T. Gillespie, *Stochastic Simulation of Chemical Kinetics*, Annu. Rev. Phys. Chem., 58 (2007), pp. 35–55.

[32] D.T. Gillespie, *A general method for numerically simulating the stochastic time evolution of coupled chemical reactions*, J. Comput. Phys., 22 (1976), pp. 403–434.

[33] T. Goldstein and S. Osher. *The split Bregman method for ℓ_1-regularized problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 323–343.

[34] G.H. Golub, C.F. Van Loan, *Matrix Computations*, Johns Hopkins, 1966.

[35] M.R. Hestenes, *Multiplier and gradient methods*, Journal of Optimization Theory and Applications, 4 (1969), pp. 303–320.

[36] H.S. Hall and S.R. Knight, *Elementary Trigonometry*, MacMillan and Co., Ltd., 1952.

[37] E. Laeng, *Une base orthonormale de $L^2(\mathbb{R})$ dont les éléments sont bien localisés dans l'espace de phase et leurs supports adaptés à toute partition symétrique de l'espace des fréquences*, C. R. Acad. Sci. Paris, 311 (1990), pp. 677–680.

[38] M.J. Lai and W. Yin, *Augmented ℓ^1 and Nuclear-Norm Models with a Globally Linearly Convergent Algorithm*, SIAM J. Imaging Sciences, 6 (2013), pp. 1059–1091.

[39] E.H. Lieb and M. Loss, *Analysis*. AMS Graduate Studies in Mathematics, Vol. 14, 2001.

[40] S.G. Mallat and Z. Zhang, *Matching pursuits with time-frequency dictionaries*, IEEE Transactions on Signal Processing, 12 (1993), pp. 3397–3415.

[41] G. Marsaglia, W.W. Tsang, and J. Wang, *Fast Generation of Discrete Random Variables*, Journal of Statistical Software, 11 (2004), pp. 1–11.

[42] G. Marsaglia and W. W. Tsang, *A fast, easily implemented method for sampling from decreasing or symmetric unimodal density functions*, SIAM Journ. Scient. and Statis. Computing, 5 (1984), pp. 349–359.

[43] G. Marsaglia, *Xorshift RNGs*, Journal of Statistical Software, 8 (2003), pp. 1–6.

[44] N. Marzari and D. Vanderbilt, *Maximally localized generalized Wannier functions for composite energy bands*, Physical Review B, 56 (1997), pp. 12847–12865.

[45] J.M. McCollum, G.D. Peterson, C.D. Cox, M.L. Simpson, and N.F. Samatova, *The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior*, Comput. Bio. Chem., 30 (2006), pp. 39–49.

[46] L.J. Nelson, G. Hart, F. Zhou, and V. Ozoliņš, *Compressive sensing as a paradigm for building physics models*, Physical Review B, 87 (2013), pp. 1–12.

[47] Y. E. Nesterov, *A method of solving a convex programming problem with convergence rate* $O(1/k^2)$, Soviet Math. Dokl. v27 (1983), pp. 372–376.

[48] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, *An iterative regularization method for total variation-based image restoration*, Multiscale Model. Simul., 4 (2005), pp.460–489.

[49] J.T. Oxenius, *Kinetic Theory of Particles and Photons*, Springer-Verlag, Berlin, 1986.

[50] V. Ozoliņš, Rongjie Lai, R.E. Caflisch, and S.J. Osher, *Compressed Modes for Variational Problems in Mathematics and Physics*, Proceedings of the National Academy of Sciences, 110 (2013), pp. 18368–18373.

[51] V. Ozoliņš, Rongjie Lai, R.E. Caflisch, and S.J. Osher, *Compressed plane waves–compactly supported multiresolution basis for the Laplace operator*, Proceedings of the National Academy of Sciences, 111 (2014), pp. 1691–1696.

[52] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, *Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition*, Proceedings of the 27 th Annual Asilomar Conference on Signals, Systems, and Computers, (1993), pp. 40–44.

[53] M.J.D. Powell edited by R. Fletcher, *A method for nonlinear constraints in minimization problems*, in Optimization, (1969), pp. 283–298.

[54] R. Ramaswamy, N. Gonzalez-Segredo, and I.F. Sbalzarini, *A new class of highly efficient exact stochastic simulation algorithms for chemical reaction networks*, J. Chem. Phys., 130 (2009), pp.1–13 .

[55] R. Ramaswamy and I.F. Sbalzarini, *A partial-propensity variant of the composition-rejection stochastic simulation algorithm for chemical reaction networks*, J. Chem. Phys., 132 (2010), pp. 1–6.

[56] R. Ramaswamy and I.F. Sbalzarini, *A partial-propensity formulation of the stochastic simulation algorithm for chemical reaction networks with delays*, J. Chem. Phys., 134 (2011), pp. 1–8.

[57] B. Recht, M. Fazel, and P. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, SIAM Review, 52 (2010), pp. 471–501.

[58] H. Schaeffer, R. Caflisch, C.D. Hauck, and S. Osher, *Sparse dynamics for partial differential equations*, Proceedings of the National Academy of Sciences, 110 (2013), pp. 6634–6639.

[59] D.J. Sullivan, J.J. Rehr, J.W. Wilkins, K.G. Wilson, *Phase space Wannier functions in electronic structure calculations*, Research Report, Cornell University, 1987.

[60] R. Tibshirani, *Regression shrinkage and selection via the Lasso*, J. Roy. Stat. Soc. B (Method.), 58 (1996), pp. 267–288.

[61] J.A. Tropp and A.C. Gilbert, *Signal recovery from random measurements via orthogonal matching pursuit*, IEEE Transactions on Information Theory, 53 (2007), pp. 4655–4666.

[62] M.D. Vose, *A Linear Algorithm For Generating Random Numbers With a Given Distribution*, IEEE Transaction and Software Engineering, 17 (1991), pp. 972–975.

[63] A.J. Walker, *An efficient method for generating discrete random variables with general distributions*, ACM TOMS, 3 (1977), 253–256.

[64] G. H. Wannier, *The structure of electronic excitation levels in insulating crystals*, Physical Review, 52 (1937), pp. 0191–0197.

[65] L.T. Watson, *Theory of globally convergent probability-one homotopies for nonlinear programming*, SIAM J. Optim., 11 (2000), pp. 761–780.

[66] L.T. Watson and R.T. Haftka, *Modern homotopy methods in optimization*, Comput. Methods Appl. Mech. Engrg., 74 (1989), pp. 289–304.

[67] K.G. Wilson, *Generalized Wannier functions*, preprint, Cornell University, 1987.

[68] Y. Yang, M. Moller, and S. Osher, *A dual split Bregman method for fast ℓ^1 minimization*, Mathematics of computation, 82 (2013), pp. 2061-2085.

[69] K. Yin and S.J. Osher, *On the completeness of the compressed modes in the eigenspace*, UCLA CAM Reports:13–62, 2013.

[70] W. Yin, *Analysis and Generalizations of the Linearized Bregman Method*, SIAM J. on Imaging Sciences, 3 (2010), pp. 856–877.

[71] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. *Bregman iterative algorithms for l1-minimization with applications to compressed sensing*, SIAM Journal on Imaging Sciences, 1 (2008), pp. 143–168.

[72] Y.B. Zeldovich and Y.P. Raizer, *Physics of Shock Waves and High-Temperature Hydrodynamic Phenomena*, Dover, Mineola, NY, 2002.