

Search-Based Methods for the Sparse Signal Recovery Problem in Compressed Sensing

by

Nazım Burak Karahanođlu

Submitted to
the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

SABANCI UNIVERSITY

January 2013

SEARCH-BASED METHODS FOR THE SPARSE SIGNAL RECOVERY
PROBLEM IN COMPRESSED SENSING

APPROVED BY

Assist. Prof. Dr. Hakan ERDOĞAN
(Thesis Supervisor)

Assoc. Prof. Dr. Müjdat ÇETİN

Assoc. Prof. Dr. Cem GÜNERİ

Prof. Dr. Ş. İlker BİRBİL

Assist. Prof. Dr. İlker BAYRAM

DATE OF APPROVAL:

©Nazm Burak Karahanoglu 2013
All Rights Reserved

To my beloved wife, Akmer...

Acknowledgements

There are a lot of people to whom I am grateful because of their contributions to this dissertation. The completion of this work would not be possible without their generous support and encouragement.

First of all, I would like to express my deep and sincere gratitude my thesis supervisor Hakan Erdoğan for his invaluable guidance, tolerance, positiveness, support and encouragement throughout my doctoral studies.

I am grateful to Müjdat Çetin, Cem Güneri, Ş. İlker Birbil, İlker Bayram and Aytül Erçil for taking the time to read this dissertation and making valuable comments which have led to significant contributions to this work. In addition, I would like to thank Ş. İlker Birbil also for his support and contributions to the seventh chapter of this dissertation, and Müjdat Çetin and Cem Güneri for their positive attitudes and support in the progress meetings.

Hearty thanks are due to all my colleagues at the Information Technologies Institute of TÜBİTAK-BİLGEM for all their support during my doctoral studies. Without the permission of TÜBİTAK-BİLGEM, it would not be possible for me to finish the course work for this doctoral study.

I can not forget to send my heartfelt thanks to my lovely sister, Işık, who has always been by me despite the distance between us in the last years.

My deepest gratitude goes to my parents for their unflagging love and support throughout my life.

Finally, I would like to dedicate this work to my beloved wife, Akmer. Without her unconditional love and tireless support, completion of my doctoral studies would never be possible.

SEARCH-BASED METHODS FOR THE SPARSE SIGNAL RECOVERY
PROBLEM IN COMPRESSED SENSING

NAZIM BURAK KARAHANOĞLU

EE, Ph.D. Thesis, 2013

Thesis Supervisor: Hakan Erdoğan

Keywords: Compressed sensing, sparse signal recovery, best-first tree search, forward-backward search, mixed integer linear programming.

Abstract

The sparse signal recovery, which appears not only in compressed sensing but also in other related problems such as sparse overcomplete representations, denoising, sparse learning, etc. has drawn a large attraction in the last decade. The literature contains a vast number of recovery methods, which have been analysed in theoretical and empirical aspects.

This dissertation presents novel search-based sparse signal recovery methods. First, we discuss theoretical analysis of the orthogonal matching pursuit algorithm with more iterations than the number of nonzero elements of the underlying sparse signal. Second, best-first tree search is incorporated for sparse recovery by a novel method, whose tractability follows from the properly defined cost models and pruning techniques. The proposed method is evaluated by both theoretical and empirical analyses, which clearly emphasize the improvements in the recovery accuracy. Next, we introduce an iterative two stage thresholding algorithm, where the forward step adds a larger number of nonzero elements to the sparse representation than the backward one removes. The presented simulation results reveal not only the recovery abilities of the proposed method, but also illustrate optimal choices for the step sizes. Finally, we propose a new mixed integer linear programming formulation for sparse recovery. Due to the equivalency of this formulation to the original problem, the solution is guaranteed to be correct when it can be solved in reasonable time. The simulation results indicate that the solution can be found easily under some reasonable assumptions, especially for signals with constant amplitude nonzero elements.

SIKIŞTIRMALI ALGILAMA SEYREK İŞARET GERİ ÇATMA PROBLEMİ İÇİN ARAMA TABANLI YÖNTEMLER

NAZIM BURAK KARAHANÖĞLU

EE, Doktora Tezi, 2013

Tez Danışmanı: Hakan Erdoğan

Anahtar Kelimeler: sıkıştırımlı algılama, seyrek işaretlerin geri çatılması, ilk en iyiyle ağaç araması, ileri-geri arama, karışık tam sayılı doğrusal programlama

Özet

Sıkıştırımlı algılamanın yanı sıra seyrek tamüstü gösterimler, gürültü giderme ve seyrek öğrenme gibi alanlarda da rastlanan seyrek işaretlerin geri çatılması problemi, son yıllarda büyük ilgi çekmektedir. Literatürde, performansları teorik ve deneysel olarak analiz edilmiş çok sayıda geri çatma yöntemi bulunmaktadır.

Bu tezde, arama tabanlı yeni seyrek işaret geri çatma yöntemleri tartışılmaktadır. İlk olarak, dikgen eşleştirme arayışı algoritmasının geri çatılacak sinyalin sıfır olmayan elemanlarından daha fazla sayıda iterasyona izin verecek şekilde teorik bir analizi gerçekleştirilecektir. İkinci olarak, ilk en iyiyle arama yöntemini kullanan yeni bir seyrek işaret geri çatma algoritması tartışılacaktır. Önerilen yöntemde, ağaç aramasının çözülebilir olması için yeni maliyet fonksiyonları ve budama teknikleri kullanılacaktır. Bu yöntem, geri çatma doğruluğundaki iyileşmeleri açıkça ortaya koyan teorik ve deneysel analizler ile incelenecektir. Daha sonra, ileri adımın, seyrek gösterime geri adımın çıkardığından daha fazla sayıda sıfır olmayan eleman eklediği yeni bir iki aşamalı döngüsel algoritma tanımlanacaktır. Sunulan simülasyon sonuçları ile, önerilen yöntemin geri çatma becerisinin yanı sıra uygun adım uzunluğu seçimi konusu da irdelenecektir. Son olarak, seyrek geri çatma için yeni bir karışık tam sayılı doğrusal programlama formülasyonu önerilecektir. Bu formülasyonun asıl probleme denk olması, problemin makul sürelerde çözülebildiği durumlarda, bulunan sonucun doğruluğunu garanti etmektedir. Simülasyon sonuçları, bu problemin özellikle eşit büyüklükteki sıfır olmayan elemanlardan oluşan işaretler için bazı makul varsayımlar altında kolaylıkla çözülebildiğini ortaya koymaktadır.

Contents

Acknowledgements	iv
Abstract	v
Özet	vi
List of Figures	xi
List of Tables	xiii
Abbreviations	xiv
1 Introduction	1
1.1 Motivation	1
1.2 Contributions and Outline	3
2 An Overview of the Sparse Signal Recovery Problem and the Main-stream Recovery Approaches	7
2.1 Introduction	7
2.2 The Sparse Signal Recovery Problem	8
2.3 The Restricted Isometry Property	11
2.4 Sparse Signal Recovery Algorithms	13
2.4.1 Greedy Pursuit Approaches	13
2.4.2 Convex Optimization	18
2.4.3 Nonconvex Optimization	19
2.4.4 Bayesian Methods	20
2.4.5 Iterative Reweighted Methods	22
3 Theoretical and Empirical Analyses of Orthogonal Matching Pursuit with Different Termination Criteria	24
3.1 Introduction	24
3.1.1 Outline and Contributions	25
3.1.2 Notation	26
3.2 Orthogonal Matching Pursuit	27
3.3 Recent Developments on the Theoretical Analysis of OMP	28

3.4	Theoretical Analysis of OMP	30
3.4.1	Preliminaries	30
3.4.2	Success Condition for a Single OMPe Iteration	32
3.4.3	Online Recovery Guarantees for OMPe	35
3.4.4	On the Validity of the Online Guarantees	36
3.5	Empirical Analysis	39
3.5.1	Phase Transitions	39
3.5.2	Empirical Success and Failure Rates of OMPe Iterations	42
3.6	Discussion and Future Work	44
4	A* Orthogonal Matching Pursuit: : Best-First Search for Compressed Sensing	46
4.1	Introduction	46
4.1.1	Outline	48
4.2	A* Search	48
4.2.1	Fundamentals of the A* Search	48
4.2.2	Different Auxiliary Function Structures	51
4.3	Sparse Signal Reconstruction using A* Search	52
4.3.1	Notation	53
4.3.2	Initialization of the Search Tree	54
4.3.3	Expansion of the Selected Partial Path	54
4.3.3.1	Extensions per Path Pruning	54
4.3.3.2	Tree Size Pruning	55
4.3.3.3	Equivalent Path Pruning	57
4.3.4	Selection of the Most Promising Path	58
4.3.4.1	The Additive Cost Model	59
4.3.4.2	The Adaptive Cost Model	60
4.3.4.3	The Multiplicative Cost Model	60
4.3.5	A* Orthogonal Matching Pursuit	61
4.3.6	Complexity vs. Accuracy	63
4.4	AStarOMP Software for Fast Recovery with A* Search	64
4.4.1	Trie Structure in AStarOMP	65
4.4.2	Computation of the Orthogonal Projection in AStarOMP	68
4.4.3	Reusability of the Developed Code	69
4.5	Empirical Analyses	70
4.5.1	Reconstruction of Synthetically Generated 1D Data	70
4.5.1.1	Different Coefficient Distributions	71
4.5.1.2	Performance over Different Observation Lengths	76
4.5.1.3	Comparison of Different Search Parameters	77
4.5.1.4	Reconstruction from Noisy Observations	79
4.5.2	Reconstruction of Images	80
4.6	Summary	83
5	Theoretical and Empirical Analyses of A*OMP With a Novel Adaptive Cost Model	85
5.1	Introduction	85
5.1.1	Definitions	86

5.1.2	Outline	86
5.2	Theoretical Analysis of A*OMP	87
5.2.1	Preliminaries	87
5.2.2	Success Condition of an A*OMP Iteration	89
5.2.3	Exact Recovery Conditions for A*OMPK	91
5.2.4	Exact Recovery with A*OMPe	93
5.2.5	Theoretical Comparison of the Two Different Termination Criteria	96
5.3	The Adaptive-Multiplicative Cost Model	97
5.4	Empirical Analyses of A*OMPe	99
5.4.1	Experimental Setup	100
5.4.2	Exact Recovery Rates and Reconstruction Error	100
5.4.3	Phase Transitions	103
5.4.4	Recovery from Noisy Observations	106
5.4.5	A Hybrid Approach for Faster Practical Recovery	106
5.4.6	Image Recovery Examples	108
5.4.6.1	Demonstration on a Sparse Image	109
5.4.6.2	Demonstration on a Compressible Image	109
5.5	Summary	111
6	Forward-Backward Pursuit: A Novel Two Stage Algorithm for Sparse Signal Recovery	113
6.1	Introduction	113
6.1.1	Outline	114
6.2	The Forward-Backward Pursuit Algorithm	115
6.2.1	Notation	116
6.2.2	The Proposed Method	116
6.3	Relations to Other Greedy Algorithms	118
6.3.1	Two Stage Thresholding Algorithms	118
6.3.2	Forward Greedy Algorithms	120
6.4	Empirical Analyses	120
6.4.1	Exact Recovery Rates and Reconstruction Error	121
6.4.2	Phase Transitions	124
6.4.3	Recovery from Noisy Observations	129
6.4.4	Image Recovery Examples	130
6.4.4.1	Demonstration on a Sparse Image	131
6.4.4.2	Demonstration on a Compressible Image	132
6.5	Summary	133
7	A Mixed Integer Linear Programming Approach for Sparse Signal Recovery	136
7.1	Introduction	136
7.2	The Equivalent MILP Formulation of the Sparse Signal Recovery Problem	137
7.2.1	Problem Formulation	137
7.2.2	Implementation of the MILP Formulation	138
7.2.3	Practical Issues	139
7.3	Simulations	141
7.4	Conclusions and Future Work	145

8	Summary and Future Work	147
8.1	Summary	147
8.2	Suggestions for Future Work	149

	Bibliography	152
--	---------------------	------------

List of Figures

2.1	Observation model for the sparse recovery problem.	8
2.2	Sparse recovery problem illustrated via selection of nonzero indices.	9
2.3	Growth of the search tree during recovery with TB-OMP and FBMP.	17
3.1	Empirical phase transitions of OMP_e , OMP_K , BP, and SP for the recovery of Gaussian, uniform, and CARS sparse signals from noise-free observations.	40
3.2	Histograms of failed OMP_e iterations over 200 perfectly recovered Gaussian sparse vectors	43
4.1	Evaluation of the search tree during A*OMP algorithm	53
4.2	Evaluation of the search tree during a single iteration of the A*OMP algorithm	56
4.3	Path Equivalency	57
4.4	Comparison of equivalent path detection with tree (left) and trie (right) structures.	67
4.5	Reconstruction results over sparsity for the uniform sparse signals employing Gaussian observation matrices.	71
4.6	Probability density estimates of the ANMSE values for $K = 30$	72
4.7	Number of misidentified entries per test sample for $K = 30$	72
4.8	Reconstruction results over sparsity for the Gaussian sparse signals using Gaussian observation matrices.	75
4.9	Reconstruction results over sparsity for the Gaussian sparse signals using Bernoulli observation matrices.	75
4.10	Reconstruction results over sparsity for the binary sparse signals using Gaussian observation matrices.	76
4.11	Reconstruction results over observation length for the uniform sparse signals where $K = 25$ using a single Gaussian observation matrix for each M	77
4.12	Reconstruction results over α for the uniform sparse signals using Gaussian observation matrices.	78
4.13	Reconstruction results for the sparse binary signals for $B = 2$ and $B = 3$ using Gaussian observation matrices.	78
4.14	Average distortion ratio over SNR for reconstruction of sparse signals from noisy observations using Gaussian observation matrices.	80
4.15	Reconstructions of the image “Lena” using BP, SP, and Mul-A*OMP with $B = 3$	82
4.16	Reconstruction error per pixel of image “Lena” for Mul-A*OMP with $B = 3$ and BP.	83

5.1	Recovery results over sparsity for the Gaussian sparse signals.	102
5.2	Recovery results over sparsity for the uniform sparse signals.	102
5.3	Average run time of A*OMP per vector with the AStarOMP software. . .	103
5.4	Recovery results over sparsity for the binary sparse signals.	103
5.5	Phase transitions of AMul-A*OMP _e , BP, SP, OMP, ISD, and SL0 for the Gaussian, uniform, and CARS sparse signals.	105
5.6	Average distortion over SNR in noisy recovery scenario	107
5.7	Average run time per vector of A*OMP in the noisy recovery scenario using the AStarOMP software.	107
5.8	Performance of the hybrid scheme for the Gaussian sparse vectors. . . .	108
5.9	Recovery of the image “bridge” using BP and AMul-A*OMP _e	110
5.10	Recovery of the image “butterfly” using BP and AMul-A*OMP _e . BP and AMul-A*OMP _e yield 27.8 and 27.5 dB PSNR, respectively.	111
6.1	Reconstruction results over sparsity for the Gaussian sparse vectors. For FBP, $\beta = \alpha - 1$	122
6.2	Reconstruction results over sparsity for the Gaussian sparse vectors. For FBP, $\alpha = 20$	122
6.3	Phase transitions of FBP with different forward and backward step sizes for the uniform, Gaussian, and CARS sparse signals.	126
6.4	Phase transitions of FBP, BP, SP, OMP, and AMul-A*OMP _e for the Gaussian, uniform, and CARS sparse signals.	128
6.5	Average recovery distortion over SNR in case of noise contaminated ob- servations	130
6.6	Average run time per sample in case of noise contaminated observations .	130
6.7	Recovery of the image “bridge” using BP and FBP	132
6.8	Recovery of the image “butterfly” using BP and FBP. BP and FBP yield 27.8 and 27.4 dB PSNR, respectively.	133
7.1	Average recovery results for the binary and CARS sparse signals	143
7.2	Average recovery results for the uniform and Gaussian sparse signals . . .	144

List of Tables

4.1	Average A*OMP iterations per vector for the uniform sparse signals . . .	73
4.2	Average equivalent paths per vector for the uniform sparse signals	73
4.3	Average run time in sec. per vector for the uniform sparse signals	74
4.4	Average Mul-A*OMP iterations per vector with respect to α and P for the uniform sparse signals with $K = 35$	79
4.5	Average Mul-A*OMP iterations with respect to B and P per vector in the sparse binary problem	79
4.6	PSNR values for images reconstructed using different algorithms	83
7.1	Average run time in seconds per sparse vector	145

Abbreviations

ANMSE	A verage N ormalized M ean- S quared- E rror
A*OMP	A* O rthogonal M atching P ursuit
BP	B asis P ursuit
CARS	C onstant A mplitude R andom S ign
CoSaMP	C ompressive S ampling M atching P ursuit
CS	C ompressed S ensing
FBMP	F ast B ayesian M atching P ursuit
FBP	F orward B ackward P ursuit
FoBa	F orward- B ackward Greedy Algorithm
GOMP	G eneralized O rthogonal M atching P ursuit
IHT	I terative H ard T hresholding
ISD	I terative S upport D etection
LP	L inear P rogramming
MILP	M ixed I nteger L inear P rogramming
MP	M atching P ursuit
OMP	O rthogonal M atching P ursuit
PSNR	P eak S ignal-to- N oise R atio
RIC	R estricted I sometry C onstant
RIP	R estricted I sometry P roperty
ROMP	R egularized O rthogonal M atching P ursuit
SBL	S parse B ayesian L earning
SL0	S moothed ℓ_0
SNR	S ignal-to- N oise R atio
SP	S ubspace P ursuit
StOMP	S tagewise O rthogonal M atching P ursuit

TB-OMP Tree search **B**ased **O**rthogonal **M**atching **P**ursuit
TST Two **S**tage **T**hresholding

Chapter 1

Introduction

1.1 Motivation

Compression has always been one of the most important and most deeply investigated topics in the signal processing and communication communities. Traditionally, data compression has been considered as a completely independent process from data acquisition. In this conventional understanding, the signals should be captured at the Shannon-Nyquist rate before compression could be applied. The most popular means for compression is the transform coding, where the captured signal is first converted by some efficient transform technique such as the Discrete Cosine Transform or Discrete Wavelet Transform into an appropriate domain in which it may be represented by a limited number of significantly large transform coefficients. Compression can only then be performed by thresholding which keeps only the largest magnitude transform coefficients. Examples of commonly used compression standards include the Moving Picture Experts Group (MPEG) standards for video, MPEG-1 Audio Layer III (MP3), and Advanced Audio Coding (AAC) techniques for audio, the Joint Photographic Experts Group (JPEG) standards for image, etc.

On the other hand, the emerging compressed sensing (CS) field aims at combining the compression process with the data acquisition in contrast to the conventional compression techniques. In the CS acquisition model, compression is implicitly performed while signals are captured by a number of linear observations¹ below the Shannon-Nyquist rate.

¹This observation process is usually modelled via the so-called observation matrix, which maps the signal of interest onto the observation domain which has less dimensions than the signal itself.

This data reduction rate leads to the most fundamental question of the CS theory: Can a reduced number of observations, which are below the Shannon-Nyquist rate, contain enough information for exact reconstruction of signals? At first, this might seem quite unnatural, however the CS literature [1–4] states that it is indeed possible under some conditions. *Sparse*² signals can be exactly recovered if the observation matrix satisfies some necessary conditions, such as the well-known restricted isometry property (RIP) [1]. Similarly, *compressible*³ signals can also be approximated with small error under RIP. Although it is hard to show that the RIP holds for a fixed matrix, some families of random matrices such as those with independent and identically distributed entries from the Gaussian or Bernoulli distributions, or random selections from the discrete Fourier transform are known to satisfy the RIP with high probabilities [4]. In addition, most real world signals are compressible or sparse in some appropriate transform domain, such as the wavelet domain or the discrete cosine transform basis for natural images. Combining the compressibility of real world signals with the RIP of random matrices, it is possible to unite the signal acquisition with compression via compressed sensing techniques.

As a natural consequence of acquisition by a compressed set of linear measurements, the necessity arises for reconstruction of the acquired signals. Due to the dimensionality reduction during the acquisition of the signals, this problem is analytically ill-posed: There exists infinitely many signals which lead to the same set of measurements. Therefore, the problem should be cast as an optimization problem which seeks the sparsest one among these possible solutions. Though compressibility allows for this sparsity promoting formulation, direct solution of the resultant optimization problem still necessitates an intractable combinatorial search. Consequently, a vast number of alternative recovery methods, which exploit different properties of the underlying recovery problem, have recently been proposed. [5] presents an insightful overview of the sparse signal recovery literature, where the existing methods are classified into five categories as the convex optimization, greedy pursuits, Bayesian methods, nonconvex optimization, and brute-force methods.

²A signal is called sparse if most of its elements are zero. Similarly, a K -sparse signal has at most K nonzeros.

³A signal is called compressible if its sorted coefficient magnitudes exhibit a power law decay in some appropriate transform domain.

The dissertation at hand concentrates on the sparse signal recovery problem, and presents a number of novel techniques for this purpose. Due to the search-like structures employed by the presented methods, we refer to these in common as search-based methods. The presented methods are investigated regarding both theoretical and empirical aspects. Their empirical recovery performances are demonstrated by various compressed sensing simulations. RIP-based theoretical analyses are also presented whenever possible. In addition, we present theoretical and empirical analyses of the orthogonal matching pursuit (OMP) method [6] which is a simple, yet well-acknowledged greedy sparse signal recovery algorithm in the CS community.

Though the sparse signal recovery problem is mostly referred to as “compressed sensing” in the literature, CS itself is not the only application domain of the sparse signal recovery methods. There also exists other closely related problems in the literature, such as sparse overcomplete representations, dictionary learning, error correction, denoising, sparse learning, subset selection, etc. Although this dissertation examines the empirical performance with CS simulations, the proposed sparse signal recovery algorithms can also be trivially applied for other closely related problems as well.

1.2 Contributions and Outline

In this work, we focus on novel algorithms for the sparse signal recovery problem, regarding both theoretical and empirical aspects. First, we devote Chapter 2 to an overview of the existing sparse signal recovery algorithms in the literature. Before the new recovery techniques are introduced, Chapter 3 concentrates on theoretical and empirical analyses of the well-acknowledged OMP algorithm, which may be seen as a greedy search method. The A* orthogonal matching pursuit (A*OMP) method, presented in Chapter 4, is based on a semi-greedy best-first tree search, while the forward-backward pursuit (FBP) of Chapter 6 performs a greedy search by the addition and removal of nonzero elements during the forward and backward stages. In addition to these, Chapter 7 proposes solving a reformulation of the original sparse signal recovery problem by mixed integer linear programming (MILP) techniques including the powerful branch-and-bound methods, which obtain the optimal solution via an exhaustive search on a solution tree. Although the proposed methods employ different routines for solving the sparse signal recovery problem, they all incorporate search-based structures. Due to this

similarity, we find it convenient to present them under the general class of search-based methods.

In Chapter 3, we present RIP-based theoretical analysis of the OMP algorithm for recovery of sparse signals from noise-free measurements. Our analysis follow a strategy similar to [7], which analyses OMP recovery in K iterations where K denotes the number of nonzero elements of the underlying sparse signal. In particular, we extend this analysis to allow for more than K OMP iterations. This leads to online recovery conditions depending on the internal state of OMP, i.e., the number of correct and false detections in an intermediate step. Due to this dependency, we cannot convert our results into exact recovery guarantees for all K -sparse signals. However, the presented analysis still states that OMP can exactly recover a K -sparse signal within $\frac{3}{2}K$ iterations if an intermediate step satisfies some conditions on the number of correct and false detections in addition to the online recovery condition. In contrast, the state-of-the-art exact recovery guarantees, such as [8, 9] and [10], necessitate $6K$ to $30K$ iterations, which is impractical in many situations. In addition to the theoretical analysis, we also provide an empirical demonstration of the OMP recovery performance for different types of sparse signals in comparison to some mainstream sparse signal recovery algorithms in the literature.

Chapter 4 introduces the A*OMP algorithm, which utilizes an efficient tree search for solving the sparse signal recovery problem. The proposed method employs the A* search [11–15], which is a best-first tree search technique frequently used in problems such as path finding, graph traversal, and speech recognition. A*OMP possesses not only appropriate cost models which provide means for both simultaneous handling of paths with different lengths throughout the search and reduction of the computational burden, but also pruning techniques which reduce the tree size effectively. Proper definitions of these two are very important for the tractability of the sophisticated tree search as proposed. Addressing this issue, A*OMP provides means for complexity-accuracy trade-off by proper adjustment of the cost model and pruning parameters as demonstrated in Chapter 4. In addition, Chapter 4 also discusses the AStarOMP software, which is developed as an efficient implementation of the algorithm for the purpose of demonstrating the recovery abilities in practice. The simulations in Chapter 4 illustrate the recovery performance of A*OMP in comparison to some other mainstream algorithms for a variety of scenarios including both synthetically generated sparse data and images. These

results reveal that the best-first search can significantly improve the recovery performance with proper definition of the cost model and appropriate selection of the pruning parameters.

In addition to the empirical evaluation of A*OMP in Chapter 4, the theoretical aspects of the sparse signal recovery via A*OMP are discussed in Chapter 5. These analyses not only state RIP-based exact recovery guarantees for A*OMP, but also provide a theoretical comparison of the recovery performance with different termination criteria. As expected by the promising empirical recovery results which are obtained after the incorporation of the best-first search in Chapter 4, A*OMP is shown to possess stronger exact recovery guarantees than the OMP algorithm. Moreover, our theoretical results also indicate the optimality of the termination criterion which is based on the residue of the measurement vector. In addition to these theoretical findings, we also develop a novel cost model, which significantly accelerates the algorithm in practice. Finally, Chapter 5 contains a variety of simulations which illustrate the improvements in both the recovery accuracy and speed of the algorithm with the proposed modifications. The results of these simulations clearly support the theoretical findings of Chapter 5.

We introduce another novel search-based technique for sparse signal recovery in Chapter 6. This technique, called the forward-backward pursuit, is a novel iterative scheme where each iteration consists of two stages. Let us define the term *support* as the set of indices corresponding to the locations of nonzero elements in the underlying sparse signal. The first one of the two stages in each FBP iteration is the forward stage, which expands the support estimate by addition of α new indices. The latter is the backward stage, which removes β indices from the support estimate, where $\beta < \alpha$. This constitutes a greedy algorithm which resembles two stage thresholding (TST) [16–18], while the expansion of the support estimate by $\alpha - \beta$ atoms⁴ per iteration presents a novel extension over the TST schemes in the literature. The recovery simulations in Chapter 6 illustrate the recovery accuracy via a variety of scenarios including both synthetic 1D and real 2D data. In addition, an empirical strategy for choosing optimal step sizes is also demonstrated.

Chapter 7 proposes a new MILP formulation of the sparse signal recovery problem. This MILP formulation is obtained by the introduction of an auxiliary binary vector where the

⁴Atoms refer to the columns of the observation matrix.

recovered nonzero elements are located by ones. Joint optimization for finding this binary auxiliary vector together with the sparse vector of interest leads to an MILP problem. By addition of a few appropriate constraints in order to reduce the size of the feasible solution space, this problem can be solved by MILP techniques. This new formulation has an important advantage over the mainstream sparse signal recovery methods: It is not an approximation, but it is equivalent to the underlying sparse optimization problem. Therefore, the solution becomes exactly equal to the optimal solution of the original sparse signal recovery problem, once it can be found in reasonable time. We demonstrate tractability of the solution by recovery simulations involving different sparse signal types. The proposed scheme improves recovery over the mainstream recovery methods especially when the underlying sparse signals have constant amplitude nonzero elements.

Chapter 2

An Overview of the Sparse Signal Recovery Problem and the Mainstream Recovery Approaches

2.1 Introduction

In contradiction to the conventional acquisition process, where a signal is captured as a whole before the dimensionality reduction can be applied via some transform coding, the rapidly emerging compressed sensing (CS) field targets acquisition of *sparse* or *compressible* signals directly in reduced dimensions. The dimensionality reduction is achieved by capturing a set of linear measurements instead of the signal itself, where the number of the measurements, M , is less than the signal dimension, N . As a result, the underlying signal has to be recovered from the observations, which is ill-posed due to the dimensionality reduction.

Despite the fact that the recovery problem is analytically ill-posed, CS literature [1–4] states that it is indeed possible to recover the underlying sparse signal from observations below the Shannon-Nyquist rate under appropriate conditions such as the restricted isometry property (RIP). The literature contains a broad range of methods which have

$$\underbrace{\begin{bmatrix} y_1 \\ \vdots \\ y_M \end{bmatrix}}_{\mathbf{y} \in \mathbb{R}^{M \times 1}} = \underbrace{\begin{bmatrix} \phi_1 & \cdots & \cdots & \phi_M \end{bmatrix}}_{\mathbf{\Phi} \in \mathbb{R}^{M \times N}} \underbrace{\begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_N \end{bmatrix}}_{\mathbf{x} \in \mathbb{R}^{N \times 1}}$$

FIGURE 2.1: Observation model for the sparse recovery problem.

been proposed for solving this ill-posed recovery problem. These methods can be categorized into a number of algorithmic families with respect to their varying approaches to the problem. An overview and categorization of the mainstream sparse signal recovery methods can be found in [5].

This chapter serves as a literature survey which summarizes the current state of the art in the CS field. First, we provide a definition of the sparse signal recovery problem in Section 2.2. The restricted isometry property, which provides an important means for theoretical justification of sparse signal recovery approaches, is introduced in Section 2.3. Finally, Section 2.4 is devoted to the discussion of the major sparse signal recovery algorithms.

2.2 The Sparse Signal Recovery Problem

As mentioned in the introduction, the fundamental problem of CS is to recover a sparse or compressible signal from some reduced set of observations. Let $\mathbf{x} \in \mathbb{R}^N$ be a sparse signal with $K \ll N$ nonzero entries. We refer to such a signal as K -sparse. Under noise-free conditions, the “*compressed*” linear measurements of the K -sparse \mathbf{x} are modelled using the observation matrix $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ as

$$\mathbf{y} = \mathbf{\Phi} \mathbf{x} \tag{2.1}$$

where $\mathbf{y} \in \mathbb{R}^M$ and $K < M < N$. The matrix $\mathbf{\Phi}$ is often called the dictionary, acknowledging its role during the recovery. This observation model is illustrated in Figure 2.1.

Since the number of measurements is less than the signal dimension, the recovery of \mathbf{x} from \mathbf{y} is analytically ill-posed. That is, there exists multiple solutions of (2.1), which

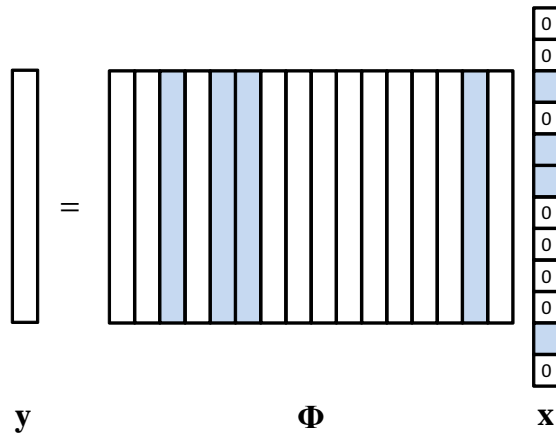


FIGURE 2.2: Sparse recovery problem illustrated via selection of nonzero indices where $K = 4$. Color-filled entries denote the nonzero elements of \mathbf{x} and the corresponding columns of Φ which contribute to \mathbf{y} .

are shifted versions of the desired solution \mathbf{x} in the null-space of the dictionary Φ . One of them is, for example, the minimum ℓ_2 norm solution, which can be obtained using the pseudo-inverse of Φ . Though this solution satisfies the observation model (2.1), there is no guarantee that it is the desired sparsest solution, and generally it is not.

On the other hand, we may simply rewrite (2.1) as

$$\mathbf{y} = \sum_{i=1}^N x_i \phi_i, \quad (2.2)$$

where x_i is the i th element of \mathbf{x} and ϕ_i , which is sometimes referred to as an atom, denotes the i th column vector of Φ . Since only K of the x_i 's are nonzero due to the sparsity of \mathbf{x} , we observe that the problem is reduced to finding the K nonzero indices of \mathbf{x} corresponding to the atoms which best explain \mathbf{y} . Figure 2.2 illustrates the sparse recovery problem as selection of nonzero indices, which are marked as the color-filled elements of \mathbf{x} . Exploiting this basic observation, the sparse signal recovery is cast into an optimization problem in the CS theory as

$$\mathbf{x} = \arg \min \|\mathbf{x}\|_0 \quad \text{subject to} \quad \mathbf{y} = \Phi \mathbf{x}, \quad (2.3)$$

where $\|\mathbf{x}\|_0$ denotes the number of nonzero elements in \mathbf{x} . Note that, although it does not actually satisfy the requirements of a proper norm, $\|\cdot\|_0$ is often called the ℓ_0 norm in the CS literature by abuse of the terminology.

In addition to (2.1), other similar sparse signal recovery formulations also appear in the literature for slightly modified problems. One of them is the case where \mathbf{x} is not exactly sparse, but compressible, i.e., most of its energy is concentrated in a few elements. Another example is encountered when the observation process is noise contaminated or not exact. In this case, the observation model is modified as

$$\mathbf{y} = \Phi \mathbf{x} + \mathbf{n}, \quad (2.4)$$

where \mathbf{n} denotes some additive noise component or observation error. In these situations, the problem might be cast as a sparse signal approximation problem:

$$\mathbf{x} = \arg \min \|\mathbf{x}\|_0 \quad \text{subject to} \quad \|\mathbf{y} - \Phi \mathbf{x}\|_2 \leq \varepsilon, \quad (2.5)$$

where ε is defined as a measure for how close the sparse approximation should satisfy the observation constraints. Finally, we can also write a mixed formulation [5], where the regularization parameter τ governs the sparsity of the solution:

$$\mathbf{x} = \arg \min \frac{1}{2} \|\mathbf{y} - \Phi \mathbf{x}\|_2 + \tau \|\mathbf{x}\|_0. \quad (2.6)$$

Direct solutions of these ℓ_0 norm minimization problems above, however, are all computationally very expensive as they require exhaustive combinatorial search over all subsets of the columns of Φ [2, 19]. Thus, direct solution is not feasible even for signals that are moderate in size. As a consequence, sparse signal recovery techniques in the literature mostly concentrate on indirect means to obtain an approximation of \mathbf{x} .

Either referred to as sparse signal recovery or approximation, formulations similar to (2.3), (2.5), and (2.6) appear not only for CS [3, 4, 16, 16–18, 20–65], but also for related problems such as sparse overcomplete representations [6, 66–72], error correction [1, 73], denoising [74–79], sparse learning [80–88], etc. Note that, in the literature, it is quite common to use the term compressed sensing for referring to the sparse optimization formulations, even though they are also encountered in a wide range of related problems.

2.3 The Restricted Isometry Property

Theoretical analysis of sparse signal recovery algorithms has been an important topic in the compressed sensing community. For this purpose, researchers have concentrated on notions such as the null space property [67, 89], coherence [67, 90–93], probabilistic analysis [20, 94], restricted isometries [1, 4, 7, 17, 18, 95], etc.

In the last decade, RIP [1] has been acknowledged as an important means for obtaining theoretical guarantees of the proposed algorithms. To get an understanding of the RIP, we can count on two important requirements for exact recovery of \mathbf{x} from the observation $\mathbf{y} = \Phi\mathbf{x}$ [5]:

- **Uniqueness:** The uniqueness of a K -sparse representation \mathbf{x} for each \mathbf{y} implies an algebraic condition on submatrices of Φ . Assume that there exists some \mathbf{z} such that $\mathbf{y} = \Phi\mathbf{x} = \Phi\mathbf{z}$. Then, $\Phi(\mathbf{x} - \mathbf{z}) = \mathbf{0}$. To ensure that \mathbf{x} is unique, we need $\|\mathbf{z}\|_0 > K$ for any possible \mathbf{z} . That is, all subsets of Φ containing at most $2K$ columns should be linearly independent.
- **Stability:** In addition to uniqueness, tractability of the sparse representation necessitates that each signal should be stably determined. That is, perturbations in the sparse coefficients should lead to similar perturbations in the measurements, i.e., $\|\Delta\mathbf{x}\|_2$ and $\|\Phi(\Delta\mathbf{x})\|_2$ should be comparable.

A common means for imposing these two requirements is the restricted isometry property [1] which plays an important role in the theory of compressed sensing:

Theorem 2.1 (Restricted isometry property). *A matrix Φ is said to satisfy the L -RIP for any positive integer L if there exists a restricted isometry constant (RIC) $\delta_L \in (0, 1)$ satisfying*

$$(1 - \delta_L)\|\mathbf{x}\|_2^2 \leq \|\Phi\mathbf{x}\|_2^2 \leq (1 + \delta_L)\|\mathbf{x}\|_2^2, \quad (2.7)$$

for all \mathbf{x} where $\|\mathbf{x}\|_0 \leq L$.

The nature of the RIP can be better understood by the following proposition which can be seen as a natural extension of the uniqueness and stability requirements: A system satisfying the RIP for some constant acts almost like an orthonormal system for sparse

linear combinations of its columns [1]. In other words, Φ approximately preserves the distance between any two K -sparse vectors if it satisfies $2K$ -RIP. In particular, the lower bound in (2.7) implies the uniqueness condition, and the upper bound represents the stability, which is especially important for robust recovery from perturbed measurements. Since RIP represents these two properties together, it provides means for developing exact recovery guarantees of sparse signals from lower dimensional observations.

Analysis in [1, 4] state that certain random matrices satisfy the RIP with high probabilities, when some specific conditions hold for K based on M and N . Random matrices with independent and identically distributed entries that follow Gaussian or Bernoulli distributions are stated to satisfy the K -RIP with high probabilities if

$$M \geq cK \log \left(\frac{N}{K} \right), \quad (2.8)$$

where c is a function of the restricted isometry constant δ_K . On the other hand, in case the columns of the observation matrix are selected randomly among the columns of the discrete Fourier transform matrix, the number of necessary measurements can be obtained as

$$M \geq cK \log^6 N. \quad (2.9)$$

Some improvements on these bounds have also been reported in the literature (see for example [96–99]). Motivated by the fact that they satisfy RIP with high probabilities when these bounds hold, random observation matrices are frequently utilized in compressed sensing in order to provide more compact representations of sparse signals.

Utilization of matrices satisfying RIP with high probabilities allows for theoretical analysis of the recovery algorithms via development of upper bounds on the RIC to guarantee exact recovery of sparse signals. That is, exact recovery guarantees of algorithms may be stated in terms of specific upper bounds on RIC. Via the conditions on the number of measurements for satisfying RIP, these bounds may be related to the number of necessary measurements chosen from specific random ensembles as well. Relaxing the upper bound, i.e. allowing for a larger RIC, can be interpreted as reducing the number of necessary measurements for exact recovery. As a consequence, RIP-based theoretical exact recovery guarantees have been stated for many sparse signal recovery algorithms such as [1, 4, 7, 17, 18, 28, 30, 100, 101] in terms of upper bounds on RIC.

Before concluding the discussion of RIP, we also would like to mention the following simple, yet important property of RIC:

Lemma 2.1 (Monotonicity of RIC). *Assume that the matrix Φ satisfies L -RIP with δ_L . Then, for any positive integer $C > L$, we have*

$$\delta_C \geq \delta_L.$$

This states that RIC increases monotonically with the number of nonzero indices allowed. We exploit this property later in the following chapters while developing exact recovery guarantees for recovery algorithms.

2.4 Sparse Signal Recovery Algorithms

In the literature, there is a vast number of sparse signal recovery methods which attack the problem from different perspectives. In this section, we provide a brief review of these methods in five categories. Note that this specific categorization is chosen in order to provide a structured review of algorithms, while other categorizations are also obviously possible¹. In addition, some methods do not strictly fall into one of the categories we present below. Especially some algorithms which we list among the greedy pursuits can also be grouped into different classes. However, we choose a broad categorization, and review such algorithms in the class which they are most similar to.

2.4.1 Greedy Pursuit Approaches

The greedy pursuit methods are fundamentally based on search mechanisms which iteratively expand or refine a sparse estimate. This family includes algorithms which select one coefficient per iteration as well as algorithms which select or modify multiple coefficients per iteration, where each iteration may be followed by a pruning (or thresholding) step. In addition, we also cover some techniques with tree search structures in this category, as such structures also resemble the greedy algorithms.

¹A partially overlapping categorization and overview of the mainstream sparse signal recovery methods can be found in a recent publication of Tropp and Wright [5]

Historically, matching pursuit (MP) [68] is the first greedy pursuit algorithm. MP starts with an empty support estimate for \mathbf{x} . At each iteration, it expands the support estimate by addition of the index which corresponds to the dictionary column having the highest magnitude inner product with the residue², while the corresponding entry of \mathbf{x} is set equal to this inner product. The residue is also updated accordingly. The iterations are run until either a predefined number (i.e., K) of atoms is selected, or the residue is small enough. A major drawback of MP is that it does not take into account the nonorthogonality of the dictionary columns. Due to this nonorthogonality, setting the value of the selected entry equal to the corresponding inner product at each iteration is a suboptimal choice. MP tries to address this issue by refining the nonzero elements of the recovered vector using the orthogonal projection coefficients of the observation vector onto the selected support after the termination of the algorithm. This choice, however, is also not optimal, since the stagewise selection of indices is still suboptimal due to the fact that the residue is not orthogonal to the selected support set at intermediate iterations.

The orthogonal matching pursuit (OMP) algorithm [6] is one of the most acknowledged greedy algorithms, due to its simplicity and empirically competitive recovery performance. OMP extends the MP algorithm by a stagewise orthogonality condition which addresses the nonorthogonality of the columns of Φ . In order to avoid suboptimal selection of indices, OMP performs the orthogonal projection of the observation vector onto the selected support set after each iteration. By this way, the residue is assured to be orthogonal to the set of selected columns, increasing the reconstruction accuracy. As for the recovery guarantees, theoretical analyses of OMP have been first performed either using a coherence parameter [91] or via probabilistic analysis [20, 94]. Recently, RIP has also been utilized for theoretical analysis of OMP both with only K steps [7, 95] and with more than K steps [8–10]. We further visit OMP in the next chapter, which discusses its theoretical and empirical performance in detail.

More sophisticated pursuit methods, which select multiple columns per iteration, have also been of interest to the CS researchers. For example, stagewise OMP (StOMP) [102] selects at each step all of the dictionary columns whose absolute inner products with the residue are higher than an adaptive threshold depending on the ℓ_2 norm of

²The residue of the i th iteration is the vector $\mathbf{r}_i = \mathbf{y} - \Phi \hat{\mathbf{x}}_i$, where $\hat{\mathbf{x}}_i$ is the estimate of \mathbf{x} after the i th iteration.

the residue. Alternatively, regularized OMP (ROMP) [27, 101] groups the atoms with similar magnitude inner products together at each iteration, and selects the group with maximum energy. Due to this regularization, the ROMP algorithm is equipped with theoretical performance guarantees based on a RIP bound. A recent proposal, the generalized OMP algorithm (GOMP) [28, 103] extends OMP by selecting a fixed number of nonzero elements at each iteration with respect to the highest inner product with the residue. GOMP is also supported by RIP-based theoretical exact recovery guarantees.

Another set of greedy pursuit algorithms including compressive sampling matching pursuit (CoSaMP) [18] and subspace pursuit (SP) [17] combine selection of multiple nonzero elements per iteration with a pruning step. These algorithms keep a support estimate of K indices throughout the iterations. At each iteration, they first expand the support estimate by the αK indices ($\alpha = 1$ for SP and $\alpha = 2$ for CoSaMP), corresponding to the dictionary atoms having the maximum absolute inner product with the residue. Following this expansion, they compute the coefficients for indices in the support estimate by orthogonal projection of \mathbf{y} onto the subspace defined by the support estimate. Before going for the next iteration, they finally prune the support estimate to retain only indices corresponding to the K largest coefficients. CoSaMP and SP are provided with theoretical guarantees, showing that these two stage schemes achieve exact reconstruction when the dictionary satisfies some RIP condition.

Recently, Maleki and Donoho have presented an algorithmic framework called two stage thresholding (TST) [16], into which algorithms such as SP and CoSaMP fall. As the name suggests, this framework involves algorithms that employ two stage iterative schemes. The first stage is similar to the simple iterative thresholding algorithms: The sparse estimate is first updated in the direction opposite to the gradient of the residue³, which is followed by thresholding in order to get a new sparse estimate. The optimal values of the nonzero elements are then computed by the orthogonal projection of \mathbf{y} onto the selected support set. Finally, a second thresholding operator is applied on these coefficients. This second thresholding which further imposes sparsity on the support estimate, yields the support estimate of the corresponding iteration. The evaluation of TST-type algorithms in [16] announces an optimum TST version, which turns out to be a modified SP algorithm with pre-computed optimum step sizes.

³This becomes equivalent to choosing the indices having highest magnitude inner products with the residue when the consequent thresholding operation is configured to keep a fixed number of largest elements. With this specific setting, CoSaMP and SP can be obtained as TST-type algorithms.

One other family of the greedy pursuits is the iterative thresholding algorithms [29–34]. The iterative hard thresholding (IHT) [29, 30] algorithm typically first updates the approximation of \mathbf{x} in the direction opposite to the gradient of the residual error at each iteration. The sparsity constraint is then imposed by pruning the approximation either by thresholding or keeping only a predefined number of the largest entries. [30] establishes that IHT algorithms enjoy RIP-based theoretical exact recovery guarantees similar to those of CoSaMP and SP. [34] presents the accelerated IHT algorithm, which utilizes acceleration methods to improve the convergence speed, while the performance guarantees of the original IHT method are preserved. In [31], Cevher proposes the Nesterov Iterative hard thresholding method, which incorporates the Nesterov’s proximal gradient method [104] to update the approximation of \mathbf{x} . This method provides no *a priori* performance guarantee, but still an online performance guarantee.

Gradient pursuits algorithms [35, 36] attempt at obtaining a fast approximation of the OMP algorithm by applying gradient-based acceleration techniques for the orthogonal projection step. Gradient pursuit [35] employs a gradient step to modify the sparse approximation at each iteration instead of the orthogonal projection. The more effective approximate conjugate gradient pursuit, which employs an approximate conjugate gradient step at each iteration, performs close to OMP with reduced computational complexity and storage requirements. Recently, an extension of the idea, stagewise weak gradient pursuits [36] incorporate selection of multiple columns per iteration, based on a threshold directly related to the maximum inner product among the dictionary columns and the residue.

CS literature also contains a number of unsophisticated tree search based methods which may also be counted among greedy (or, to be exact, semi-greedy) methods. The tree search based OMP (TB-OMP) [38] employs a tree search that opens B children per leaf node at each iteration. A rather flexible version of this is the flexible TB-OMP [39], where the branching factor B is decreased at each level in order to reduce the tree size. Another straightforward tree search appears in the fast Bayesian matching pursuit (FBMP) [40], where each iteration first opens all children of the leaf nodes, and then retains the best D among all opened nodes with respect to their posterior probabilities. These tree search based structures can be seen as rather straightforward applications of tree search in CS. Though some of these methods employ simple techniques for reducing the tree size, their applications are limited to small-scale problems due to large tree

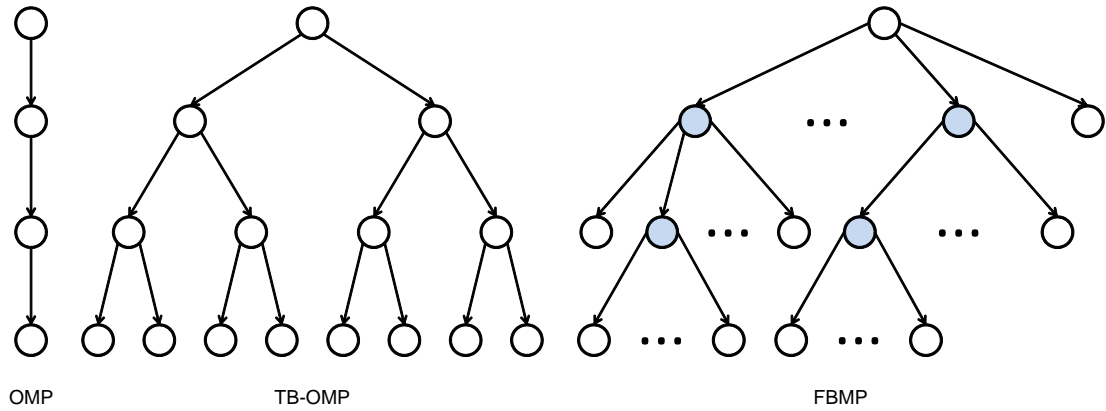


FIGURE 2.3: Growth of the search tree during recovery with TB-OMP and FBMP in comparison to the single-path OMP algorithm. Each node in the graph represents a chosen index of \mathbf{x} at different iterations of an algorithm. Tree-based methods consider and evaluate multiple indices at each level of the tree. In this particular example, TB-OMP explores $B = 2$ children per leaf node at each iteration, while FBMP explores all children of the best $D = 2$ nodes at each level. The best nodes at each level of FBMP are marked as color-filled.

sizes that appear in practice. Figure 2.3 illustrates the growth of the search tree during the recovery with TB-OMP and FBMP methods in comparison to the single-path OMP algorithm.

The randomized OMP algorithm [79], which aims at improving the OMP recovery from noisy measurements, yields an estimate of the minimum mean-squared error solution by averaging multiple sparse representations which are obtained by running a randomized version of OMP several times. At each run, the indices in the support estimate are selected at random with probabilities depending on their inner products with the residue. The final estimate is then obtained by combination of the multiple representations with an appropriate weighting scheme.

There has also been efforts to accelerate matching pursuit type algorithms by reducing the complexity of the inner product computation. One example is the tree-based pursuit [37], which provides a mechanism for clustering the vectors in the dictionary in a tree structure. In the proposed tree structure, each inner node is a common representation of its child nodes, while the leaf nodes are themselves the dictionary atoms. With this structure, the selection of the best candidate dictionary atom can be performed iteratively from the root of the tree to the best tree leaf by following the best candidate node at each level. This leads to a reduction in the complexity of the search for the best atom at the expense of a slight reduction in the performance of MP. [37] applies

this idea to MP only, while incorporation of the clustered tree structure into other MP variants is also trivial.

2.4.2 Convex Optimization

This important class of sparse signal recovery algorithms is based on relaxation of the ℓ_0 norm minimization in the sparse signal recovery problem with ℓ_1 norm minimization. The motivation for this replacement is that the ℓ_1 norm minimization provides the closest convex approximation to the ℓ_0 norm minimization problem. This translation of the problem makes the solution possible via computationally tractable convex optimization algorithms.

In the noise-free case, the convex form of (2.3) is written as:

$$\mathbf{x} = \arg \min \|\mathbf{x}\|_1 \quad \text{subject to} \quad \mathbf{y} = \Phi \mathbf{x}. \quad (2.10)$$

Similarly, the mixed formulation may also be put into convex form as

$$\mathbf{x} = \arg \min \frac{1}{2} \|\mathbf{y} - \Phi \mathbf{x}\|_2 + \tau \|\mathbf{x}\|_1, \quad (2.11)$$

where larger τ values imply solutions with smaller ℓ_1 norm. Among other convex formulations, the LASSO [105] formulation, which also takes the observation noise into account, can be written as

$$\mathbf{x} = \arg \min \|\mathbf{y} - \Phi \mathbf{x}\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}\|_1 \leq \beta. \quad (2.12)$$

Another common formulation for the noisy case parameterizes the error norm explicitly:

$$\mathbf{x} = \arg \min \|\mathbf{x}\|_1 \quad \text{s.t.} \quad \|\mathbf{y} - \Phi \mathbf{x}\|_2 \leq \varepsilon. \quad (2.13)$$

Historically, ℓ_1 norm minimization for sparse approximation has first appeared in basis pursuit (BP) [66]. This method is based on solving the convex optimization problem in (2.10) by linear programming (LP) techniques. Employing well known LP techniques, this problem can be solved in polynomial time. The LP-equivalent of (2.10), discussion

of simplex and interior point methods for BP, and details of the algorithm can be found in [66].

In addition to BP, [106] also proposes a similar primal-dual interior-point framework for solving the ℓ_1 norm minimization problem. More recently, [107] applies a primal log-barrier approach for a quadratic reformulation of the ℓ_1 norm minimization problem in the mixed form. Implementations of the standard primal-dual and log-barrier methods are available in the ℓ_1 -magic software package [108].

Pivoting algorithms have also been utilized for solving the ℓ_1 norm minimization problem. Homotopy method of [109] is proposed for solving a noisy overdetermined ℓ_1 -penalized least squares problem. A similar approach is applied to the noiseless underdetermined ℓ_1 norm minimization problem by Donoho and Tsaig in [110].

The restricted isometry property plays an important role for the applicability of the ℓ_1 norm minimization instead of the original ℓ_0 norm minimization problem. Extensive analyses of the necessary RIP conditions for the ℓ_1 relaxation, convergence issues, and bounds on the number of necessary measurements can be found in the literature [1, 2, 4, 100, 111]. These analyses show that the ℓ_0 and ℓ_1 norm minimization problems lead to the same K -sparse representation if the observation matrix satisfies RIP with $\delta_{2K} < \sqrt{2} - 1$.

2.4.3 Nonconvex Optimization

Though employing nonconvex optimization techniques for sparse signal recovery has not been as popular as the convex or greedy methods, there still exists some nonconvex sparse signal recovery approaches which we would like to pronounce here.

A nonconvex formulation of the CS reconstruction problem can be obtained via l_p norm relaxation of (2.3) [41]

$$\min \|\mathbf{x}\|_p^p \quad \text{subject to} \quad \mathbf{y} = \Phi \mathbf{x}. \quad (2.14)$$

for $0 < p < 1$. [41] develops RIP-based theoretical results for the nonconvex relaxation, which show improvements over the convex relaxation.

Chartrand has suggested a number of different techniques for solving (2.14). An iterative method based on the lagged-diffusivity algorithm is developed in [41]. [42] adopts

the projected gradient descent algorithm with some regularization for finding the global minimum of (2.3). The nonconvex minimization scheme is applied to the reconstruction of magnetic resonance images in [43] where Fourier-based algorithms for convex minimization are extended to the nonconvex case. An iterative reweighted nonconvex minimization method is also developed in [44].

Another nonconvex method is the smoothed ℓ_0 (SL0) [72], which is based on minimizing a smoothed nonconvex approximation $F_\rho(\mathbf{x})$ instead of $\|\mathbf{x}\|_0$. The minimization of $F_\rho(\mathbf{x})$, where the parameter ρ controls the quality of the approximation, is performed using the graduated nonconvexity principle [112]. The algorithm starts with the minimization of a coarse approximation, and improves the quality of the approximation at each step by modifying ρ . At each iteration, the result from the previous iteration is used as the starting point in order to avoid falling into the local minima of $F_\rho(\mathbf{x})$. A number of different approximations to the ℓ_0 norm are pronounced in [72], while the algorithm is demonstrated for only one of them using the steepest descent method for solving the minimization problem at each iteration. The method is also employed for the error correction problem in [113].

2.4.4 Bayesian Methods

Another family of sparse signal recovery algorithms is the Bayesian methods, which follow the Bayesian inference for solving the recovery problem. Before we provide a short summary of such methods, note that, in a general perspective, the convex formulation of the sparse signal reconstruction problem can also be obtained by Bayesian techniques as a maximum a posteriori estimation problem which utilizes a Laplacian prior for the entries of the unknown sparse signal. However, the motivation behind the most convex methods is replacing the ℓ_0 norm with its closest convex approximation, the ℓ_1 norm, and not maximum a posteriori estimation. Therefore, we find it more convenient to differentiate the convex methods as a different class of algorithms than the Bayesian ones.

The sparse Bayesian learning (SBL) [80] of Tipping provides a framework for exploiting sparsity in the regression and classification problems. In [80], the solution to the SBL problem is obtained via relevance vector machines, which resemble the well-known support-vector machines. For CS purposes, the regression case is of greater interest: For

sparse Bayesian regression, a Bayesian model is developed by incorporating hyperparameters into a hierarchical Gaussian prior of weights. These hyperparameters promote sparsity by favoring vanishing weights in the corresponding hyperpriors (which incorporate some uninformative fixed parameters). The regression problem is then solved by maximizing the marginal likelihood of the observations via the maximum likelihood method. A fast iterative algorithm for solving the relevance vector machine problem is also introduced in [85]. In [48], Ji *et.al.* employ this fast algorithm for the CS reconstruction scenario. This Bayesian compressive sensing framework provides full posterior density function for the underlying sparse signal, from which not only the sparse signal but also the “error bar”, *i.e.* reliability of the reconstruction, can be estimated. In addition, the authors also propose to use this framework for adaptive optimization of the compressed sensing measurements.

Another sparsity-promoting Bayesian approach for the regression and classification problems is provided in [88]. This technique employs a Laplace prior of weights, which is realized by an equivalent hierarchical Bayesian model utilizing zero-mean Gaussian priors with independent and exponentially distributed variances. The dependency on the exponential distribution is then further simplified by the adoption of a Jeffreys noninformative hyperprior.

Using the Laplace prior to model the sparsity of signals is also investigated in [49]. In this case, the Laplace prior is imposed by a three-stage hierarchical model: The first two stages consist of zero-mean Gaussian weight priors with independent and exponentially distributed variances, while the third stage models the parameter of the exponential distribution by a Gamma hyperprior. The authors develop a mechanism that estimates all the incorporated hyperparameters from the model via the maximum likelihood method. They also provide a practical fast greedy algorithm which has tractable computational complexity.

An insightful analysis of the SBL framework is provided in a series of publications of Wipf *et.al.* [82, 83, 86]. In [86], they provide an analysis of the local and global minima of SBL, showing that the global minimum of SBL is the maximally sparse solution. However, convergence errors are introduced when the algorithm finds some other sparse solutions occurring at the local minima. The nonseparable weight prior of SBL is analyzed in [83] in comparison to the general sparse signal recovery formulation which imposes a

separable weight prior. The nonseparable weight prior is shown to reduce the number of local minima effectively. In addition, Wipf *et.al.* also provide iterative reweighted ℓ_1 and ℓ_2 norm minimization methods for solving the SBL problem [81, 82].

2.4.5 Iterative Reweighted Methods

This section outlines a number of sparse signal reconstruction algorithms which employ iterative reweighted structures. In fact, the methods we group into this category start with different formulations of the problem, such as the Bayesian approach or convex minimization, while they end up with a common iterative reweighted scheme which is based on stagewise refining of the sparse estimate via consequent weighted ℓ_p norm minimizations where the weights are chosen adaptively throughout the iterations. Therefore, it is also possible to categorize these methods into other classes with respect to their initial formulations. However we find it more appropriate to classify them into a common family because of the similar iterative reweighted structures they end up with.

A reweighted ℓ_1 norm minimization scheme is proposed in [45] by Candes *et.al.*. This approach is based on iterative refining of the solution of the unweighted ℓ_1 norm minimization problem. Each iteration of this approach solves a reweighted ℓ_1 norm minimization problem, where the weight for each coefficient is selected inversely proportional to the magnitude of the coefficient obtained after the previous iteration. In other words, larger coefficients get smaller weights, and vice versa (with some regularization for small coefficients to avoid dividing by zero). Thus, these weights decrease the difference between the ℓ_0 norm, which penalizes all nonzero coefficients equally, and the ℓ_1 norm, where larger coefficients get larger penalties.

Iterative support detection (ISD) [46] is another iterative scheme based on reweighted ℓ_1 norm minimization. Similar to the method proposed by Candes in [45], ISD also starts with the solution of the unweighted ℓ_1 norm minimization problem. At each iteration, ISD first identifies a support estimate for the underlying sparse signal by applying an adaptive threshold on the estimate of the previous iteration. The weights of the ℓ_1 norm minimization problem are then selected such that only the indices out of the detected support estimate are penalized⁴. As the detected support estimate contains

⁴That is, the weights of the indices which are already in the detected support set are set as 0. Hence, this reweighted ℓ_1 norm is only computed over the indices out of the detected support set.

larger magnitude elements, this reweighting scheme avoids large contributions of these elements to the ℓ_1 norm, making the resultant weighted ℓ_1 norm minimization problem more sensitive to smaller nonzero elements.

In [47], Daubechies *et.al.* concentrate on an iterative reweighted ℓ_2 norm minimization scheme. The proposed algorithm applies iterative reweighted least squares minimization where the weights are selected inversely proportional to the magnitudes of the coefficient estimates from the previous iteration with some quadratic regularization. Each iteration of this IRLS minimization yields the smallest weighted ℓ_2 norm solution of the sparse signal recovery problem. The final solution is obtained as a limiting case by adaptively decreasing the regularization term.

The sparse Bayesian method of Wipf et.al [81–83] also employs iterative reweighted schemes to solve the SBL problem with nonseparable priors. An iterative reweighted ℓ_1 norm minimization scheme is developed in [81] to solve the SBL problem with nonseparable priors, while [82] derives an iterative reweighted ℓ_2 norm minimization approach for the same purpose. [82] and [83] evaluate the performance of these methods in comparison to the ones with separable priors in the literature.

Iterative reweighting has also been applied for the nonconvex formulation of the sparse signal recovery problem. [44] provides an iterative reweighted nonconvex minimization procedure by appropriate selection of the weights of IRLS such that the weighted minimization problem resembles ℓ_p norm with decreasing regularization for $0 < p < 1$. The algorithm converges to the minimum ℓ_p norm solution in the limit as the regularization term vanishes.

Chapter 3

Theoretical and Empirical Analyses of Orthogonal Matching Pursuit with Different Termination Criteria

3.1 Introduction

Orthogonal matching pursuit (OMP) [6] is one of the most widely recognized greedy algorithms for the sparse signal recovery and approximation problems. OMP aims at iterative detection of the support of the underlying sparse signal by identifying the best match to the residue among the dictionary atoms at each iteration. Due to its simplicity and empirically competitive performance, OMP and its variants have been frequently used in sparse problems such as [3, 20, 28, 35, 91, 114].

Theoretical analysis of OMP has been of interest to the CS community since the introduction of the algorithm. Initially, theoretical analyses of OMP have been performed either using a coherence parameter [91] or via probabilistic analysis [20, 94]. Recently, the restricted isometry property (RIP) has been demonstrated to provide a straightforward K -step analysis of OMP [95]. The obtained RIP condition has been later improved in [7].

On the other hand, OMP is known to provide better empirical recovery performance when it is allowed to run for more than K iterations. To get an intuitive understanding, let us consider that the dictionary satisfies L -RIP with some $0 < \delta_L < 1$ where $M > L > K$. Then, selecting L indices in the support estimate improves the recovery, as soon as the correct support is a subset of the selected indices¹. Motivated by this observation, exact recovery of OMP with more than K iterations has also been recently analysed [8–10]. These studies state RIP-based guarantees for exact recovery of all K -sparse signals via OMP within $6K$ to $30K$ iterations.

In this chapter, we aim at providing a recovery analysis of the OMP algorithm regarding both theoretical and empirical aspects. For this purpose, we extend the theoretical analysis in [7] to cover for more than K iterations, and then demonstrate OMP recovery with phase transitions in comparison to some other mainstream recovery algorithms. In particular, we concentrate on the residue-based termination rule, which terminates when the residue of the observed vector gets small enough, in contrast to the sparsity-based termination, which limits the number of iterations by K . To avoid ambiguity, we use the term OMP_K to indicate the sparsity-based termination rule, and OMP_e for the residue-based termination.

3.1.1 Outline and Contributions

Before presenting our theoretical analyses, we find it important to discuss the OMP algorithm in short, and summarize the recent theoretical developments about it. For this purpose, we first provide a brief overview of the OMP algorithm in Section 3.2. In addition, Section 3.3 outlines the recent developments on the RIP-based theoretical analysis of OMP.

As for the theoretical analyses, we develop a model by extending the findings of [7] to cover more than K iterations in Section 3.4. In Theorem 3.2, we derive RIP-based online guarantees for the success of an OMP_e iteration. Next, we present online recovery guarantees for OMP_e in Theorem 3.3, which is obtained by generalizing Theorem 3.2 for all consequent iterations. Since both Theorem 3.2 and Theorem 3.3 depend on the number of correct and false indices in a particular support estimate, generalization

¹In this case, the null space of the selected support set contains only the null vector due to the L -RIP. Therefore, the solution of the corresponding projection problem is unique.

of these results for all K -sparse signals necessitates assuring the existence of support estimates with sufficiently large number of correct detections. Unfortunately, we cannot provide such guarantees. However, OMP_e obviously enjoys all theoretical guarantees of OMP_K for the noise-free case². Furthermore, Section 3.4.4, which deals with the validity of the developed online guarantees in practice, states that Theorem 3.3 becomes less restrictive than Theorem 3.1 when the number of correct and false detections in the support estimate satisfy some conditions. Under these conditions, it becomes possible to satisfy Theorem 3.3 although Theorem 3.1 fails. If satisfied under these conditions, Theorem 3.3 provides online exact recovery guarantees for K -sparse signals within $\frac{3}{2}K$ iterations. This number is clearly less than the $6K$ to $30K$ iterations, which are necessary for the state-of-the-art exact recovery guarantees of [8], [9], and [10].

Finally, we present empirical phase transition curves for three different types of sparse signals in order to demonstrate the recovery performance of OMP in comparison to some other mainstream algorithms. In addition, we provide histograms of the number of false indices after successful OMP_e termination in Section 3.5.2. This demonstrate that the upper bound on the number of false indices which the online guarantees require is loose in practice.

3.1.2 Notation

Before proceeding further, we present the notation we use throughout this chapter. First, let $\mathbf{x} \in \mathbb{R}^N$ denote the K -sparse signal of interest, and $\tilde{\mathbf{x}}^i$ be the recovery of \mathbf{x} after the i th iteration of OMP. M represents the number of observations, where $K < M < N$. We define the dictionary as $\Phi = [\phi_1 \ \phi_2 \ \dots \ \phi_N]$, where $\phi_i \in \mathbb{R}^M$ is the i th column vector in Φ . The observation vector is referred to as $\mathbf{y} \in \mathbb{R}^M$, where $\mathbf{y} = \Phi\mathbf{x}$. \mathcal{T} denotes the correct support of \mathbf{x} . $\mathcal{T}^l = \{t_1, t_2, \dots, t_l\}$ is the support estimate for \mathbf{x} after the l th iteration of OMP, where t_i is the index selected at the i th iteration. n_c and n_f are the number of correct and false indices in \mathcal{T}^l , respectively, i.e., $|\mathcal{T} \cap \mathcal{T}^l| = n_c$ and $|\mathcal{T}^l - \mathcal{T}| = n_f$, where $|\mathcal{A}|$ denotes the number of elements in the set \mathcal{A} . $\Phi_{\mathcal{J}}$ denotes the matrix consisting of the columns of Φ indexed by the set \mathcal{J} . Similarly, $\mathbf{x}_{\mathcal{J}}$ is the vector consisting of the elements of \mathbf{x} indexed by the set \mathcal{J} . Finally, \mathbf{r}^l is the residue after the orthogonal projection of \mathbf{y} onto $\Phi_{\mathcal{T}^l}$ by the end of the l th iteration.

²It is obvious that the first K steps of both variants are identical. In parallel, Theorem 3.1 is a special case of Theorem 3.3. This theoretically guarantees this intuitive fact.

3.2 Orthogonal Matching Pursuit

OMP is a simple forward greedy algorithm that searches for the support of \mathbf{x} by identifying one element per iteration. It starts with an empty support estimate, i.e., $\mathcal{T}^0 = \emptyset$, $\tilde{\mathbf{x}}^0 = \mathbf{0}$ and $\mathbf{r}^0 = \mathbf{y}$. At each iteration l , OMP first selects the index of the dictionary atom that best matches the residue \mathbf{r}^{l-1} of the previous iteration via

$$t_l = \arg \max_n |\langle \phi_n, \mathbf{r}^{l-1} \rangle|. \quad (3.1)$$

The support estimate is then expanded by the addition of the selected index as

$$\mathcal{T}^l = \mathcal{T}^{l-1} \cup \{t_l\}. \quad (3.2)$$

Following the expansion of the support, the sparse estimate is updated by the projection of \mathbf{y} onto the subspace defined by \mathcal{T}^l , which is obtained by solving the least-squares problem

$$\tilde{\mathbf{x}}_{\mathcal{T}}^l = \arg \min_{\alpha} \|\mathbf{y} - \Phi_{\mathcal{T}^l} \alpha\|_2. \quad (3.3)$$

Next, the residue is updated as

$$\mathbf{r}^l = \mathbf{y} - \Phi \tilde{\mathbf{x}}^l. \quad (3.4)$$

The OMP algorithm repeats the steps above until the specified termination criterion is fulfilled. After termination, $\tilde{\mathbf{x}}^l$ yields the recovered sparse vector.

The last two steps, (3.3) and (3.4), ensure the orthogonality of the residue to the subspace defined by the selected support estimate, which is very important since the dictionary columns are not orthogonal to each other. Dealing with the nonorthogonality of the dictionary, the orthogonal projection step leads to optimal selection of indices at each iteration.

It is possible to employ a variety of termination criteria for OMP. For example, the algorithm may be run until the residue does not decrease anymore, or $\|\Phi^* \mathbf{r}^l\|_{\infty}$ gets smaller than a predefined threshold. However, in this chapter, we concentrate on the sparsity-based and residue-based termination criteria. For this purpose, we define a combined criterion that can represent both of these simultaneously. That is, the algorithm terminates if

$$(l \geq K_{\max}) \vee (\|\mathbf{r}^l\|_2 \leq \varepsilon \|\mathbf{y}\|_2) \quad (3.5)$$

Algorithm 3.1 ORTHOGONAL MATCHING PURSUIT

input: Φ , \mathbf{y} , termination criterion
initialize: $\mathcal{T}^0 = \emptyset$, $\tilde{\mathbf{x}}^0 = \mathbf{0}$, $\mathbf{r}^0 = \mathbf{y}$, $l = 0$
while not converged **do**
 $l = l + 1$
 $t_l = \arg \max_n |\langle \phi_n, \mathbf{r}^{l-1} \rangle|$
 $\mathcal{T}^l = \mathcal{T}^{l-1} \cup \{t_l\}$
 $\tilde{\mathbf{x}}_{\mathcal{T}^l}^l = \arg \min_{\alpha} \|\mathbf{y} - \Phi_{\mathcal{T}^l} \alpha\|_2$
 $\mathbf{r}^l = \mathbf{y} - \Phi_{\mathcal{T}^l} \tilde{\mathbf{x}}_{\mathcal{T}^l}^l$
end while
return $\tilde{\mathbf{x}}^l$

With this definition, the sparsity-based criterion can be employed by setting $K_{\max} = K$ and $\varepsilon = 0$. Note that this criterion requires an *a priori* estimate of K , which is not available in many practical situations. On the other hand, residue-based termination can be imposed by choosing K_{\max} high enough (i.e., $K_{\max} \gg K$), and ε small enough with respect to the noise level or measurement errors.

The pseudo-code of the entire OMP algorithm is given in Algorithm 3.1. The procedure is very simple, and can be implemented with a few lines of code in MATLAB. In addition, the empirical performance of OMP is quite competitive in practice. These two facts have brought OMP a wide reputation as a well-acknowledged greedy sparse approximation algorithm.

3.3 Recent Developments on the Theoretical Analysis of OMP

Explaining the empirically competitive performance of OMP via extensive theoretical analyses has been of interest to the CS community since the introduction of the algorithm. First contributions on the theoretical analyses of OMP have concentrated either on a coherence parameter or probabilistic analysis. [91] presents an OMP analysis based on the coherence parameter $\mu = \max_{i,j} |\langle \phi_i, \phi_j \rangle|$. This work states that OMP will recover any K -sparse signal via Φ if $\mu < \frac{1}{2K-1}$. An alternative to the coherence based

analysis involves probabilistic measures. [20] states that $M = O(K \log N)$ random measurements are sufficient for exact recovery of \mathbf{x} with high probability if Φ is drawn from a suitable random distribution. This result is further improved in [94] by showing that a lower number of measurements is sufficient for asymptotic recovery. However, these probabilistic analyses do not guarantee that any such fixed matrix will provide exact recovery of all sparse instances.

Recently, Davenport and Wakin have presented a very straightforward K -step analysis of OMP based on RIP [95]. Their work states that OMP guarantees exact recovery of any K -sparse signal from noise-free measurements in K iterations if Φ fulfills RIP with a restricted isometry constant (RIC) satisfying $\delta_{K+1} < \frac{1}{3\sqrt{K}}$. Lately, this result has been further improved by Wang and Shim in [7] which provides a less restrictive RIP bound for OMP. According to these RIP bounds, OMP requires $M = O(K^2 \log(N))$ measurements for exact recovery in K iterations.

As our analysis is based on extending the findings of Wang and Shim, we present their result formally in the following theorem:

Theorem 3.1 (Exact recovery condition for OMP [7]). *OMP perfectly recovers any K -sparse signal from noise-free measurements in K iterations if the observation matrix Φ satisfies RIP with*

$$\delta_{K+1} < \frac{1}{\sqrt{K} + 1}. \quad (3.6)$$

In [7], Theorem 3.1 is proven by induction. It can be shown that (3.6) guarantees the success of the first iteration. Then, this result can be generalized to all of the following iterations, guaranteeing exact recovery of any K -sparse signal in exactly K iterations. Note that Theorem 3.1 represents a special case of Theorem 3.3, which is introduced below.

Due to the intuitive improvements in the OMP recovery accuracy with more than K iterations, theoretical analyses have also been performed for developing more general exact recovery guarantees. Zhang has shown that OMP can exactly recover all K -sparse signals within $30K$ iterations when the observation matrix satisfies RIP with $\delta_{31K} \leq \frac{1}{3}$ [8]. In addition, his work also involves error bounds for recovery from noisy observations. In [9], Foucart has reduced the number of iterations necessary for exact recovery to $12K$ with a RIP condition based on $\delta_{22K} \leq \frac{1}{6}$. Recently, the number of

necessary iterations has been further reduced by Wang and Shim in [10], which derives exact recovery guarantees within $6K$ OMP iterations for observation matrices satisfying RIP with $\delta_{\lfloor 8.93K \rfloor} < 0.03248$. According to these analyses, the number of measurements OMP requires for exact recovery reduces to $O(K \log(N))$ when more than K iterations are allowed.

3.4 Theoretical Analysis of OMP

3.4.1 Preliminaries

The analyses we present in this chapter are based on a number of preliminary results. Below, we present these preliminary results including a number of observations which are well-known in the CS community as well as some results which we derive in this work for our purposes. Specifically, Lemma 3.1 presents a direct consequence of RIP, while Lemma 3.2 and Corollary 3.1 are taken from [17] and [18], respectively. Lemma 3.3 follows from Corollary 3.1 by some simple derivation, and Remark 3.1 is a direct consequence of Lemma 3.3. Finally, we derive Lemma 3.4, which we will later exploit for comparing the RIP bound of Theorem 3.1 with our result. The proofs are omitted either when they are very trivial, or when they are already presented in the corresponding references. In addition, note that, the results below hold when the observation matrix Φ satisfies RIP with the given values of RIC. This dependency is omitted in the text below for the sake of the clearness.

Lemma 3.1 (Direct Consequence of RIP). *Let $I \subset \{1, 2, \dots, N\}$. For any arbitrary vector $\mathbf{z} \in \mathbb{R}^{|I|}$*

$$(1 - \delta_{|I|})\|\mathbf{z}\|_2 \leq \|\Phi_I^* \Phi_I \mathbf{z}\|_2 \leq (1 + \delta_{|I|})\|\mathbf{z}\|_2.$$

Lemma 3.2 (Lemma 1 in [17]). *Let $I, J \subset \{1, 2, \dots, N\}$ such that $I \cap J = \emptyset$. For any arbitrary vector $\mathbf{z} \in \mathbb{R}^{|J|}$*

$$\|\Phi_I^* \Phi_J \mathbf{z}\|_2 \leq \delta_{|I|+|J|}\|\mathbf{z}\|_2.$$

Corollary 3.1 (Corollary 3.4 in [18]). *For every positive integer c and r*

$$\delta_{cr} < c\delta_{2r}.$$

Lemma 3.3. For any positive integer K

$$\delta_{K+1} > \frac{\delta_3 \lceil \frac{K}{2} \rceil}{3},$$

where $\lceil z \rceil$ denotes the ceiling of z , i.e., the smallest integer greater than or equal to z .

Proof. Lemma 3.3 is a consequence of Corollary 3.1. We first replace $c = 3$ and $r = \lceil \frac{K}{2} \rceil$ into (3.7). By rearranging terms, we get

$$\delta_2 \lceil \frac{K}{2} \rceil > \frac{\delta_3 \lceil \frac{K}{2} \rceil}{3}.$$

$K + 1 \geq 2 \lceil \frac{K}{2} \rceil$ holds by the definition of the ceiling operator. Then, we obtain $\delta_{K+1} \geq \delta_2 \lceil \frac{K}{2} \rceil$ due to the monotonicity of RIC. Hence, we can write

$$\begin{aligned} \delta_{K+1} &\geq \delta_2 \lceil \frac{K}{2} \rceil \\ &> \frac{\delta_3 \lceil \frac{K}{2} \rceil}{3}. \end{aligned}$$

This completes the proof. □

Remark 3.1 (Direct consequence of Lemma 3.3). *Theorem 3.1 is violated if*

$$\delta_3 \lceil \frac{K}{2} \rceil \geq \frac{3}{\sqrt{K} + 1}. \quad (3.7)$$

Proof. Combining (3.7) with Lemma 3.3, we get $\delta_{K+1} > \frac{1}{\sqrt{K} + 1}$. This clearly contradicts Theorem 3.1. □

Lemma 3.4. Assume $K \geq 25$. There exists at least one positive integer $n_c < K$ that satisfies

$$\frac{3}{\sqrt{K} + 1} \leq \frac{1}{\sqrt{K - n_c} + 1}. \quad (3.8)$$

Moreover, such values of n_c are bounded by

$$K > n_c \geq \frac{8K + 4\sqrt{K} - 4}{9}. \quad (3.9)$$

Proof. Set $K - n_c = sK$ where $0 < s < 1$. Replacing s into (3.8), we get

$$\frac{3}{\sqrt{K} + 1} \leq \frac{1}{\sqrt{sK} + 1}.$$

Arranging the terms, we obtain the following bound for s :

$$s \leq \left(\frac{\sqrt{K} - 2}{3\sqrt{K}} \right)^2.$$

Then, the lower bound for n_c is obtained as

$$\begin{aligned} n_c &= (1 - s)K \\ &\geq \frac{8K + 4\sqrt{K} - 4}{9}. \end{aligned} \quad (3.10)$$

On the other hand, $n_c < K$ requires $sK = K - n_c \geq 1$. Hence, K should satisfy

$$\begin{aligned} K &\geq \frac{1}{s} \\ &\geq \left(\frac{3\sqrt{K}}{\sqrt{K} - 2} \right)^2. \end{aligned}$$

Rearranging terms we get

$$K \geq 5\sqrt{K},$$

which is satisfied when $K \geq 25$. Combining this with (3.10), we conclude that (3.8) is satisfied if

$$K > n_c \geq \frac{8K + 4\sqrt{K} - 4}{9}$$

for $K \geq 25$.

□

3.4.2 Success Condition for a Single OMP_e Iteration

Having presented the necessary preliminary results, we can now move on to the analysis of OMP_e. We start with the success of a single iteration, for which the theorem below states a sufficient condition depending on the number of correct and false indices in the corresponding support estimate.

Theorem 3.2. *Let $|\mathcal{T}^l \cap \mathcal{T}| = n_c$ and $|\mathcal{T}^l - \mathcal{T}| = n_f$ after the l th iteration. Then the iteration $l + 1$ will be successful, i.e., $t_{l+1} \in \mathcal{T} - \mathcal{T}^l$, if Φ satisfies RIP with*

$$\delta_{K+n_f+1} < \frac{1}{\sqrt{K - n_c + 1}}. \quad (3.11)$$

Proof. As \mathbf{r}^l is the projection error of \mathbf{y} onto $\Phi_{\mathcal{T}^l}$, we have $\mathbf{r}^l \perp \Phi_{\mathcal{T}^l}$. Therefore,

$$\langle \phi_i, \mathbf{r}^l \rangle = 0, \quad \forall i \in \mathcal{T}^l. \quad (3.12)$$

Then, we can write

$$\begin{aligned} \|\Phi_{\mathcal{T} \cup \mathcal{T}^l}^* \mathbf{r}^l\|_2^2 &= \sum_{i \in \mathcal{T} \cup \mathcal{T}^l} \langle \phi_i, \mathbf{r}^l \rangle^2 \\ &= \sum_{i \in \mathcal{T} - \mathcal{T}^l} \langle \phi_i, \mathbf{r}^l \rangle^2, \end{aligned} \quad (3.13)$$

where the righthand side of (3.13) contains only $K - n_c$ nonzero terms. Combining (3.13), and the norm inequality, we obtain

$$\|\Phi_{\mathcal{T} \cup \mathcal{T}^l}^* \mathbf{r}^l\|_\infty \geq \frac{1}{\sqrt{K - n_c}} \|\Phi_{\mathcal{T} \cup \mathcal{T}^l}^* \mathbf{r}^l\|_2. \quad (3.14)$$

Next, \mathbf{r}^l can be written as

$$\begin{aligned} \mathbf{r}^l &= y - \Phi_{\mathcal{T}^l} \tilde{\mathbf{x}}_{\mathcal{T}^l}^l \\ &= \Phi_{\mathcal{T}} \mathbf{x}_{\mathcal{T}} - \Phi_{\mathcal{T}^l} \tilde{\mathbf{x}}_{\mathcal{T}^l}^l \\ &= \Phi_{\mathcal{T} \cup \mathcal{T}^l} \mathbf{z}, \end{aligned}$$

where \mathbf{z} is a vector of length $K + n_f$. By Lemma 3.1, we obtain

$$\begin{aligned} \|\Phi_{\mathcal{T} \cup \mathcal{T}^l}^* \mathbf{r}^l\|_2 &= \|\Phi_{\mathcal{T} \cup \mathcal{T}^l}^* \Phi_{\mathcal{T} \cup \mathcal{T}^l} \mathbf{z}\|_2 \\ &\geq (1 - \delta_{K+n_f}) \|\mathbf{z}\|_2. \end{aligned} \quad (3.15)$$

Replacing (3.15) into (3.14) yields

$$\|\Phi_{\mathcal{T} \cup \mathcal{T}^l}^* \mathbf{r}^l\|_\infty \geq \frac{1 - \delta_{K+n_f}}{\sqrt{K - n_c}} \|\mathbf{z}\|_2. \quad (3.16)$$

Remember that the selection rule for the index t_{l+1} at iteration $l + 1$ is defined as

$$t_{l+1} = \arg \max_i \left| \langle \phi_i, \mathbf{r}^l \rangle \right|. \quad (3.17)$$

Combining this definition with (3.16), we obtain

$$\begin{aligned} |\langle \phi_{t_{l+1}}, \mathbf{r}^l \rangle| &= \|\Phi^* \mathbf{r}^l\|_\infty \\ &\geq \|\Phi_{\mathcal{T} \cup \mathcal{T}^l}^* \mathbf{r}^l\|_\infty \\ &\geq \frac{1 - \delta_{K+n_f}}{\sqrt{K - n_c}} \|\mathbf{z}\|_2. \end{aligned}$$

Now, suppose that iteration $l + 1$ fails, i.e., $t_{l+1} \notin \mathcal{T} \cup \mathcal{T}^l$. Then, we can write

$$\begin{aligned} |\langle \phi_{t_{l+1}}, \mathbf{r}^l \rangle| &= \|\phi_{t_{l+1}}^* \Phi_{\mathcal{T} \cup \mathcal{T}^l} \mathbf{z}\|_2 \\ &\leq \delta_{K+n_f+1} \|\mathbf{z}\|_2. \end{aligned}$$

by Lemma 3.2. Clearly, this never occurs if

$$\frac{1 - \delta_{K+n_f}}{\sqrt{K - n_c}} \|\mathbf{z}\|_2 > \delta_{K+n_f+1} \|\mathbf{z}\|_2$$

or equivalently

$$\sqrt{K - n_c} \delta_{K+n_f+1} + \delta_{K+n_f} < 1 \quad (3.18)$$

Following the monotonicity of RIC, we know that $\delta_{K+n_f+1} \geq \delta_{K+n_f}$. Hence, (3.18) is guaranteed when

$$\sqrt{K - n_c} \delta_{K+n_f+1} + \delta_{K+n_f+1} < 1,$$

which is equivalent to

$$\delta_{K+n_f+1} < \frac{1}{\sqrt{K - n_c} + 1}. \quad (3.19)$$

Hence, $t_{l+1} \in \mathcal{T} \cup \mathcal{T}^l$ when (3.19) holds. We also know that $\langle \phi_i, \mathbf{r}^l \rangle = 0$ for all $i \in \mathcal{T}^l$. Therefore, a selected index cannot be selected again in the following iterations, i.e., $t_{l+1} \notin \mathcal{T}^l$. Combination of $t_{l+1} \in \mathcal{T} \cup \mathcal{T}^l$ and $t_{l+1} \notin \mathcal{T}^l$ finally leads to $t_{l+1} \in \mathcal{T} - \mathcal{T}^l$. To conclude, given n_c and n_f , (3.19) guarantees that iteration $l + 1$ will be successful. \square

Theorem 3.2 and Theorem 3.1 are naturally related. Theorem 3.1 is based on the fact that the RIP condition in (3.6) guarantees exact recovery of an iteration, provided that all previous iterations have been successful. The dependency on the success of all previous iterations is necessary for exact recovery in K iterations³. In contrast,

³Note that the success condition of an OMP_K iteration corresponds to the case $n_f = 0$ in (3.11). The proof of Theorem 3.1 presented in [7] is based on this restricted condition.

Theorem 3.2 removes the dependency of the success condition of an OMP iteration on the success of all previous iterations in order to allow for false detections in the support estimate. This generalizes the success condition of a single iteration to a broader extent which can handle failures among previous iterations. However, as a trade-off, we end up with an online guarantee that depends on the number of correct and incorrect indices in the support estimate of a specific iteration.

3.4.3 Online Recovery Guarantees for OMP_e

Online recovery guarantees for OMP_e can be obtained by generalization of Theorem 3.2 to all the following iterations until the successful termination of the algorithm. That is, the conditions in Theorem 3.2 do guarantee the success of not only a particular iteration, but also all the following ones. This is stated in the following theorem:

Theorem 3.3. *Let $|\mathcal{T}^l \cap \mathcal{T}| = n_c$ and $|\mathcal{T}^l - \mathcal{T}| = n_f$ after iteration l . Then, OMP_e perfectly recovers a K -sparse signal in a total of $K + n_f$ iterations if Φ satisfies RIP with*

$$\delta_{K+n_f+1} < \frac{1}{\sqrt{K - n_c + 1}}. \quad (3.20)$$

Proof. We prove Theorem 3.3 by induction. According to Theorem 3.2, (3.20) already guarantees success of the iteration $l + 1$. As a result of this, $t_{l+1} \in \mathcal{T} - \mathcal{T}^l$, and \mathcal{T}_{l+1} will contain $n_c + 1$ correct indices. Next, the right hand side of (3.20) increases monotonically with the number of correct indices in the support estimate:

$$\frac{1}{\sqrt{K - n_c + 1}} < \frac{1}{\sqrt{K - n_c - 1 + 1}}.$$

Hence, the iteration $l + 2$ requires a less restrictive RIP condition than the iteration $l + 1$ does. Therefore, (3.20) also guarantees the success of the iteration $l + 2$ in addition to the iteration $l + 1$. By induction, this applies to all of the following iterations, as each of them requires a less restrictive RIP condition. Consequently, after $K - n_c$ additional iterations, the support estimate will contain K correct indices, i.e., $\mathcal{T} \subset \mathcal{T}_{K+n_f}$, where the number of total iterations becomes $l + K - n_c = K + n_f$. (3.20) finally guarantees that the orthogonal projection coefficients of \mathbf{y} onto \mathcal{T}_{K+n_f} yield exactly \mathbf{x} . \square

Being an extension of Theorem 3.2, Theorem 3.3 also depends on the number of correct and incorrect indices in a particular support estimate. This allows online recovery guarantees which cover more than K iterations. Yet, this also prevents us from generalizing our results as exact recovery guarantees for all K -sparse signals, since the existence of intermediate steps with enough correct indices in addition to a small number of false indices is hard to guarantee. We cannot provide a proof of this for the time being, leaving it as a future work. However, we investigate the possibility of the existence of such support estimates in the next section for some particular conditions. In addition, we also would like to refer the reader to Section 3.5.2, where we investigate the number of incorrect indices empirically by histograms. These histograms demonstrate that n_f is indeed bounded in practice.

Note that the equivalency of Theorem 3.1 with Theorem 3.3 when $n_f = n_c = 0$ is a natural consequence. From a general perspective, Theorem 3.3 is a generalization of Theorem 3.1 to cover for more than K iterations. That is, it imposes exact recovery guarantees for OMP in K iterations if (3.20) is satisfied with $n_f = n_c = 0$. Otherwise, it provides an online recovery condition which allows for more than K iterations.

3.4.4 On the Validity of the Online Guarantees

In order for the online recovery condition in Theorem 3.3 to be meaningful, it should also be shown that this condition can be satisfied online at some intermediate iteration in case the K -step recovery condition of Theorem 3.1, fails. For this purpose, we provide below a comparison of the RIP conditions in Theorem 3.3 and Theorem 3.1. This comparison proves that Theorem 3.3 requires a less restrictive bound on the RIC than Theorem 3.1 does when n_c and n_f are large and small enough, respectively.

In order to state that (3.20) implies a less restrictive condition than (3.6) at least for some particular cases, we need to compare the corresponding bounds:

$$\delta_{K+1} < \frac{1}{\sqrt{K} + 1} \iff \delta_{K+n_f+1} < \frac{1}{\sqrt{K - n_c} + 1}$$

Unfortunately, the right and left-hand sides of the two constraints are related in the same direction:

$$\begin{aligned} \delta_{K+n_f+1} &\geq \delta_{K+1}, \\ \frac{1}{\sqrt{K-n_c+1}} &\geq \frac{1}{\sqrt{K+1}}. \end{aligned}$$

Hence, it is not possible to compare these two conditions directly. Intuitively, when n_f is small, and n_c is large, we expect Theorem 3.3 to be less restrictive. To illustrate, consider $n_f = 1$ and $n_c \gg n_f$. In this case, Theorem 3.3 requires an RIP condition based on δ_{K+2} instead of δ_{K+1} of Theorem 3.1. That is, the two RIC's are practically very close to each other. However, the upper bound in (3.20) is significantly larger than the one in (3.6) because of n_c being large. Hence, (3.20) becomes practically less restrictive in this situation.

Despite the intuitive reasoning, exact mathematical comparison of these two conditions is tricky, since it is not easy to obtain a tight bound on $\frac{\delta_{K+n_f+1}}{\delta_{K+1}}$ for all n_f . However, even by employing a loose bound on $\frac{\delta_{K+n_f+1}}{\delta_{K+1}}$, we can show that (3.20) becomes less restrictive than (3.6) for some particular cases:

Theorem 3.4. *Assume that $K \geq 25$, $1 \leq n_f < \lceil \frac{K}{2} \rceil$, and n_c satisfies*

$$K > n_c \geq \frac{8K + 4\sqrt{K} - 4}{9} \tag{3.21}$$

at iteration l . Then, (3.20) becomes less restrictive than (3.6) at iteration l . In such a case, the online recovery guarantees of Theorem 3.3 might be satisfied, even though K -step recovery cannot be guaranteed. Moreover, if Theorem 3.3 is satisfied under these conditions, OMP_e is guaranteed to provide exact recovery within $\frac{3}{2}K$ iterations.

Proof. Assume that

$$\delta_{K+n_f+1} \geq \frac{3}{\sqrt{K+1}}. \tag{3.22}$$

Since $n_f < \lceil \frac{K}{2} \rceil$, we observe that $3 \lceil \frac{K}{2} \rceil \geq K + n_f + 1$. Following the monotonicity of RIC, we obtain

$$\delta_{3 \lceil \frac{K}{2} \rceil} \geq \frac{3}{\sqrt{K+1}}. \tag{3.23}$$

Remark 3.1 guarantees failure of (3.6) for this case⁴.

⁴This accmplies with the OMP_K failure following the assumption $n_f \geq 1$.

On the other hand, Lemma 3.4 leads to

$$\frac{3}{\sqrt{K} + 1} \leq \frac{1}{\sqrt{K - n_c} + 1}$$

when (3.21) is satisfied, and $K \geq 25$. Hence, there exists some δ_{K+n_f+1} where

$$\frac{3}{\sqrt{K} + 1} \leq \delta_{K+n_f+1} \leq \frac{1}{\sqrt{K - n_c} + 1}.$$

Clearly, δ_{K+n_f+1} values in this range satisfy (3.20).

To conclude, when the parameters K , n_f , and n_c satisfy the assumptions, there exists some δ_{K+n_f+1} which fulfill (3.20), though (3.6) does not hold for δ_{K+1} . Then, (3.20) becomes less restrictive than (3.6), and the online recovery guarantees of Theorem 3.3 might still be satisfied even though K -step recovery cannot be guaranteed for this range of parameters. In such a case, Theorem 3.3 guarantees exact recovery within $\frac{3}{2}K$ iterations since $n_f < \lceil \frac{K}{2} \rceil$ and all the following iterations are guaranteed to be successful. \square

Theorem 3.4 states one particular case where the online guarantees of Theorem 3.3 turn into a less restrictive condition than the K -step exact recovery guarantees. Although we cannot yet generalize them, the presented online recovery guarantees can explain recovery of at least some particular sparse instances by OMP_e in practice. Moreover, when the conditions of Theorem 3.4 are satisfied, exact recovery is possible within $\frac{3}{2}K$ iterations. This number is clearly much less than the $6K$ iterations which are needed for exact recovery of all K -sparse signals with OMP_e .

Note that the assumptions $K \geq 25$ and (3.21) in Theorem 3.4 rely on $n_f < \lceil \frac{K}{2} \rceil$. This upper bound is chosen specifically in order to be able to establish (3.23). In other words, both $K \geq 25$ and (3.21) actually apply for the boundary condition $n_f = \lceil \frac{K}{2} \rceil - 1$. These conditions are necessary to prove Theorem 3.4. However, we believe that these bounds are loose. We intuitively expect that Theorem 3.4 also holds for smaller lower bounds on K and n_c . That is, the online recovery guarantees are expected to turn into less restrictive conditions for smaller K and n_c values as well. Moreover, these bounds may be further improved with a tighter upper bound on n_f . Unfortunately, we cannot yet prove these, since the proof requires a tighter upper bound on $\frac{\delta_{K+n_f+1}}{\delta_{K+1}}$, which we are not able to incorporate into the analysis. Nonetheless, we analyse n_f for successful OMP_e

recoveries via histograms in the next section. These indicate that the bound $n_f < \lceil \frac{K}{2} \rceil$ is usually loose in practice.

3.5 Empirical Analysis

3.5.1 Phase Transitions

In this section, we compare the empirical recovery performances of OMP_e and OMP_K with basis pursuit (BP) [66] and subspace pursuit (SP) [17] via phase transitions. The simulations include three different distributions for the nonzero elements of the sparse test vectors. The nonzero elements of the so-called Gaussian sparse signals are drawn from the standard Gaussian distribution, while those of the uniform sparse signals are distributed uniformly in $[-1, 1]$. The last ensemble involved is the constant amplitude random sign (CARS) sparse signals (following the definition in [16]) where the nonzero elements have unit magnitude with random signs. For OMP_e , the termination parameter is selected as $\varepsilon = 10^{-6}$ and the number of maximum allowable iterations is $K_{\max} = M$.

We compute the empirical phase transitions in order to provide an extensive evaluation over a wide range of K and M . For this purpose, let's first define the normalized measures $\lambda = \frac{M}{N}$ and $\rho = \frac{K}{M}$ for the number of observations and for the sparsity level, respectively. We fix $N = 250$, and alter M and K to sample the $\{\lambda, \rho\}$ space for $\lambda \in [0.1, 0.9]$ and $\rho \in (0, 1]^5$. We randomly generate 200 sparse instances for each $\{\lambda, \rho\}$ tuple. Next, we draw a random Gaussian observation matrix for each test instance and run each of the candidate recovery algorithms. Specifying the exact recovery criterion as $\|\mathbf{x} - \tilde{\mathbf{x}}\|_2 \leq 10^{-2}\|\mathbf{x}\|_2$,⁶ where $\tilde{\mathbf{x}}$ denotes the recovery of \mathbf{x} , we count the number of exactly recovered samples in each test. Then, we compute the phase transitions using the methodology described in [16]. This methodology uses a generalized linear model with logistic link to describe the exact recovery curve over ρ for each λ . Using this model, we detect the ρ value which yields 50% exact recovery probability. The empirical phase transition curve is finally obtained by combining the detected ρ values over the

⁵The λ axis is sampled with a resolution of 0.1, while the corresponding ρ values are chosen densely around the phase transition region for a specific λ in order to obtain a fine modelling of the transition region.

⁶This exact recovery condition is the same as the one in [16]. This choice has been made for the compatibility of the computed phase transitions with [16].

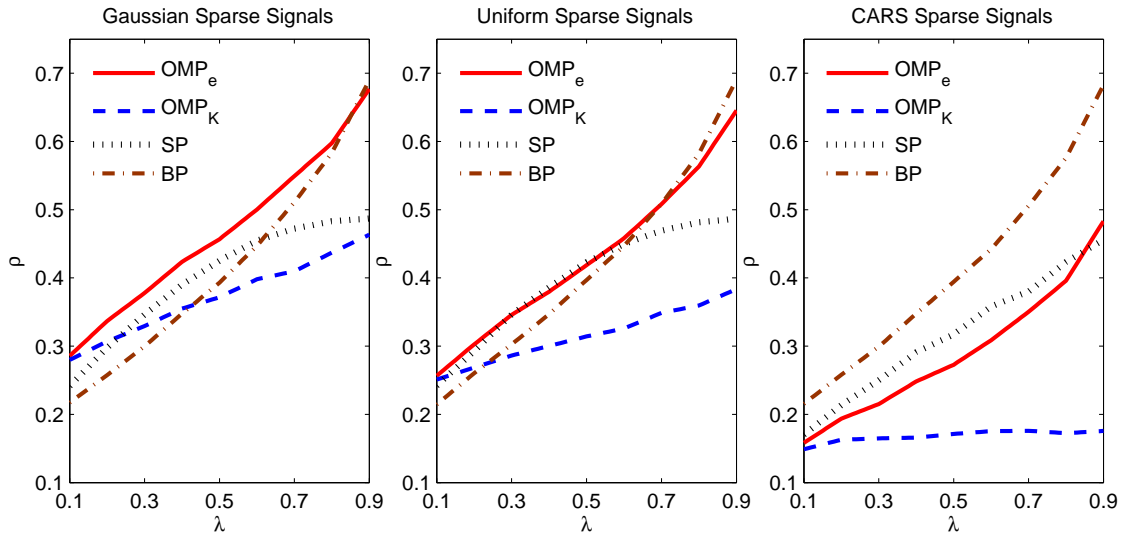


FIGURE 3.1: Empirical phase transitions of OMP_e , OMP_K , BP, and SP for the recovery of Gaussian, uniform, and CARS sparse signals from noise-free observations. The entries of the observation matrices are selected as independent and identically distributed Gaussian random variables. The results are obtained over 200 trials. The axes labels are defined as $\rho = \frac{K}{M}$ and $\lambda = \frac{M}{N}$ where $N = 250$.

whole λ range⁷. This procedure is repeated for the Gaussian, uniform, and CARS sparse signals to reveal the effect of nonzero element distribution.

Figure 3.1 depicts the phase transition curves of OMP_e , OMP_K , BP, and SP for the Gaussian, uniform, and CARS sparse signals. OMP_e yields better phase transitions than OMP_K does for all distributions, as we intuitively expect. On the other hand, the recovery performance of OMP highly depends on the coefficient distribution, while BP is robust to it, and SP shows less variation than OMP does. At one end stands the Gaussian sparse signals, where OMP_e outperforms BP and SP. For the uniform sparse signals, OMP_e might also be considered as the optimal algorithm among the candidates when the whole λ range is taken into account. In contradiction to these, the performance of OMP degrades severely for the CARS ensemble, which is indeed referred to as the most challenging case for the greedy algorithms in the literature [16, 17].

These results clearly indicate the dependency of the OMP recovery performance on the coefficient distribution. When the nonzero values cover a wide range, such as for the Gaussian distribution, the performance of OMP is boosted. In contrast, nonzero values of equal magnitude constitute the most difficult recovery problem for OMP. In fact, this

⁷Note that, due to narrow phase transition regions [16], the region below the phase transition curve promises exact recovery with high probability for the corresponding recovery method.

dependency can be better explained by some basic analytical observations on $\|\Phi_{\mathcal{T}}^* \mathbf{y}\|_{\infty}$. Assuming that the columns of Φ are normalized, we can write the upper bound on $\|\Phi_{\mathcal{T}}^* \mathbf{y}\|_{\infty}$ as

$$\begin{aligned}
\|\Phi_{\mathcal{T}}^* \mathbf{y}\|_{\infty} &= \max_{t \in \mathcal{T}} |\phi_t^* \Phi_{\mathcal{T}} \mathbf{x}_{\mathcal{T}}| \\
&= \max_{t \in \mathcal{T}} |\phi_t^* \Phi_{\mathcal{T}-t} \mathbf{x}_{\mathcal{T}-t} + \phi_t^* \phi_t \mathbf{x}_t| \\
&\leq \max_{t \in \mathcal{T}} |\phi_t^* \Phi_{\mathcal{T}-t} \mathbf{x}_{\mathcal{T}-t}| + |\phi_t^* \phi_t \mathbf{x}_t| \\
&\leq \max_{t \in \mathcal{T}} \delta_K \|\mathbf{x}_{\mathcal{T}-t}\|_2 + |\mathbf{x}_t|.
\end{aligned} \tag{3.24}$$

First, we consider the case where there are no restrictions on the nonzero values of \mathbf{x} . The Gaussian sparse signals can be seen an example of this case. For simplicity, let us set $a = \|\mathbf{x}_{\mathcal{T}-t}\|_2$. Clearly, $0 \leq a \leq \|\mathbf{x}\|_2$ in this setting. Hence, the upper bound on $\|\Phi_{\mathcal{T}}^* \mathbf{y}\|_{\infty}$ is given by

$$\max_{0 \leq a \leq \|\mathbf{x}\|_2} a \delta_K + \sqrt{\|\mathbf{x}\|_2^2 - a^2}. \tag{3.25}$$

We simply take the derivative of (3.25) with respect to a , and set it equal to zero:

$$\delta_K - \frac{a}{\sqrt{\|\mathbf{x}\|_2^2 - a^2}} = 0. \tag{3.26}$$

Then, the a value that maximizes (3.25) is found as

$$a = \frac{\delta_K \|\mathbf{x}\|_2}{\sqrt{1 + \delta_K^2}}. \tag{3.27}$$

Replacing this into (3.25), we obtain

$$\delta_K \|\mathbf{x}_{\mathcal{T}-t}\|_2 + |\mathbf{x}_t| \leq \sqrt{1 + \delta_K^2} \|\mathbf{x}\|_2. \tag{3.28}$$

Consequently, the upper bound on $\|\Phi_{\mathcal{T}}^* \mathbf{y}\|_{\infty}$ is obtained as

$$\|\Phi_{\mathcal{T}}^* \mathbf{y}\|_{\infty} \leq \sqrt{1 + \delta_K^2} \|\mathbf{x}\|_2 \tag{3.29}$$

when there are no restrictions on the nonzero values of \mathbf{x} . Note that this upper bound defines the range which the values of the correlation vector at correct indices span during the first iteration.

Now, let's consider the CARS case, where $|\mathbf{x}_t| = 1$ and $\|\mathbf{x}_{\mathcal{T}-t}\|_2 = \sqrt{K-1}$ for every $t \in \mathcal{T}$. In this case, the upper bound on $\|\Phi_{\mathcal{T}}^* \mathbf{y}\|_{\infty}$ is given by

$$|\Phi_{\mathcal{T}}^* \mathbf{y}|_{\infty} \leq 1 + \delta_K \sqrt{K-1}. \quad (3.30)$$

The upper bound in (3.30) is obviously much smaller than the one in (3.29) in practice. (In order to compare them, fix the energy of \mathbf{x} , i.e., replace $\|\mathbf{x}\|_2 = \sqrt{K}$ into (3.29).) This constitutes no problems if Theorem 3.1 is satisfied. Consider, however, that Theorem 3.1 fails: In that case, the elements of $\Phi^* \mathbf{y}$ at indices out of \mathcal{T} are more likely to exceed $|\Phi_{\mathcal{T}}^* \mathbf{y}|_{\infty}$ if \mathbf{x} is a CARS sparse signal, since $\|\Phi_{\mathcal{T}}^* \mathbf{y}\|_{\infty}$ is typically smaller for this kind of signals. Hence, the probability of failure at the first iteration becomes higher for the CARS sparse signals⁸. In other words, the maximum element of the correlation vector is less likely to be in the correct support for the CARS sparse signals, i.e., the correlation maximization step fails with higher probability. As a result of this, it is natural that the failure rates of OMP-type algorithms increase when the range which is spanned by the absolute values of the nonzero elements of the underlying sparse signals decreases. The CARS signals have the smallest range of span, hence the worst performance of OMP-type algorithms naturally appears for these signals. Note that this behaviour can be expected in common for all algorithms which employ a similar correlation maximization step. For example, Figure 3.1 indicates that the performance of SP, which employs a similar correlation maximization step, also decreases for sparse signals with constant amplitude nonzero elements.

3.5.2 Empirical Success and Failure Rates of OMP_e Iterations

Theorem 3.4 is based on the assumption $n_f < \lceil \frac{K}{2} \rceil$, which leads to the other constraints on K and n_c , i.e., $K \geq 25$ and (3.21). Hence, satisfying the limit on the number of failed iterations is critical for Theorem 3.4. On the other hand, the bound $n_f < \lceil \frac{K}{2} \rceil$ may also be loose for many practical examples, making these constraints too restrictive in practice. Therefore, it is worth to investigate the number of failed iterations until the termination in order to validate these constraints.

⁸Note that, though we skip it here, a similar analogy might be carried out to the following iterations as well.

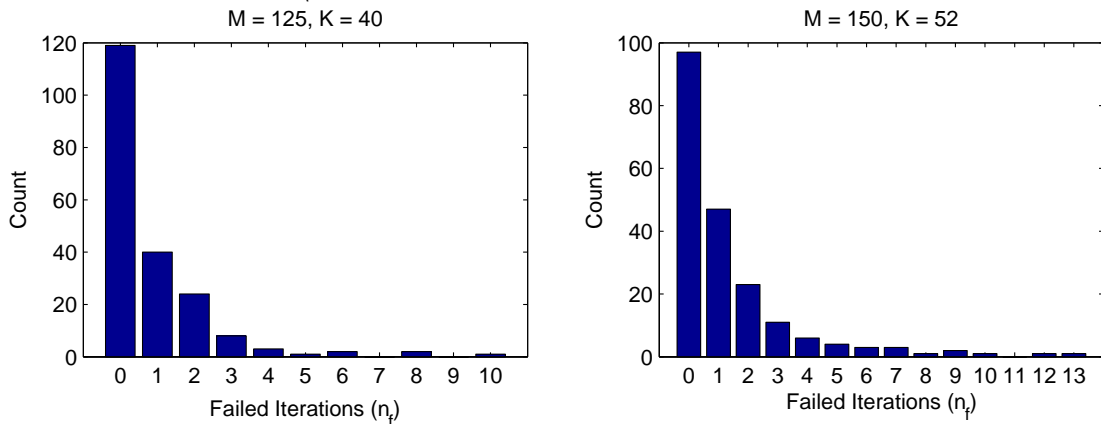


FIGURE 3.2: Histograms of failed OMP_e iterations (n_f) over 200 perfectly recovered Gaussian sparse vectors. $n_f = 0$ corresponds to the samples which are successfully recovered by OMP_K . OMP_K perfectly recovers 119 out of 200 samples when $M = 125$, $N = 40$ ($\lambda = 0.5$, $\rho = 0.32$), and 97 out of 200 samples for $M = 150$, $N = 52$ ($\lambda = 0.6$, $\rho = 0.347$). OMP_e recovers all samples perfectly in both cases. The number of failed OMP_e iterations do not exceed $K/4$ for both cases.

For this purpose, we choose two examples from the recovery simulations above, and investigate the histograms of n_f until the successful termination of OMP_e . The successful termination criterion is important here, as OMP_e may run until it reaches the maximum number of iterations (M) in case of a failure, which makes the resultant histograms noninformative. Therefore, we consider two cases where OMP_e perfectly recovers all of the test instances, while OMP_K cannot, namely $M = 125$, $K = 40$ ($\lambda = 0.5$, $\rho = 0.32$) and $M = 150$, $K = 52$ ($\lambda = 0.6$, $\rho = 0.347$). The histograms of failed iterations are depicted in Figure 3.2. OMP_K can only recover 119 out of 200 test instances perfectly for the first case, and 97 for the latter. For these instances, OMP_e also provides perfect recovery with no failed iterations, hence these correspond to the region $n_f = 0$ in the plots. On the other hand, OMP_e takes a number of wrong steps before finally finding the correct solution in the rest of the recovery problems where OMP_K fails. We observe that the number of these wrong steps is smaller than the upper bound $\lceil \frac{K}{2} \rceil - 1$. Actually, in both of the tests OMP_e never takes more than $K/4$ wrong steps. Hence, the assumption $n_f < \lceil \frac{K}{2} \rceil$ turns out to be empirically loose at least for these two cases.

3.6 Discussion and Future Work

In this chapter, we have discussed theoretical and empirical analyses of the OMP recovery from noise-free observations with the termination criterion based on the residual power. This type of termination criterion presents a more suitable objective than setting the number of iterations equal to K when the aim is finding an exact K -sparse representation, rather than obtaining the best K -sparse approximation.

The theoretical analyses in Section 3.4 state an online recovery condition for OMP_e based on the number of correct and false indices in the support estimate of an intermediate iteration. Though we cannot cast this condition into exact recovery guarantees for all K -sparse signals due to the lack of a proof for the existence of such support estimates, we still state that it may be satisfied online if n_c and n_f satisfy some bounds where OMP_K recovery already fails.

On the other hand, the state-of-the-art results necessitate $6K$ to $30K$ iterations for exact recovery of all K -sparse signals with OMP_e [8–10]. Although these guarantees are valid for all K -sparse signals, the number of iterations needed for obtaining them is mostly beyond the practical limits. In addition, the exact recovery condition that can be guaranteed in $6K$ iterations necessitates RIP with $\delta_{\lceil 8.93K \rceil}$, which clearly requires that $\lceil 8.93K \rceil \leq M^9$. Hence, even if OMP would run for $6K$ steps, this condition still necessitates $M \geq \lceil 8.93K \rceil$. In many practical applications, M will be chosen less than $6K$ or $\lceil 8.93K \rceil$, in which case these exact recovery guarantees cannot be valid anymore. In contrast, according to Theorem 3.4, the conditions presented in this chapter may be imposed to provide online guarantees for recovery within possibly less than $\frac{3}{2}K$ iterations. This number is clearly well below the number of iterations required for the state-of-the-art exact recovery guarantees, such as [8], [9], and [10].

We have also demonstrated the recovery performance of OMP_K and OMP_e via simulations involving sparse signals with different nonzero coefficient distributions. The phase transitions presented in Section 3.5.1 reveal that OMP_e is capable of providing better recovery rates than BP and SP when the nonzero elements follow the Gaussian or uniform distributions. Finally, we have presented histograms of the number of failed

⁹Otherwise, it would not be possible to satisfy $\lceil 8.93K \rceil$ -RIP for any $\delta_{\lceil 8.93K \rceil}$.

iterations in order to test the validity of the upper bound $n_f < \lceil \frac{K}{2} \rceil$. These histograms indicate that this upper bound is not only valid, but also loose in practice.

The results developed in this chapter provide a basis for the theoretical analysis of the A*OMP algorithm in Chapter 5. Furthermore, future work may be conducted on theoretical guarantees for the existence of support estimates satisfying the necessary conditions in order to generalize the developed online conditions as exact recovery guarantees for all K -sparse signals. Moreover, these conditions may be further improved by incorporating a tighter bound on either $\frac{\delta_{K+n_f+1}}{\delta_{K+1}}$ or n_f as future work. To conclude, we believe that these findings will provide a basis for improving the theoretical analyses of OMP and its variants as part of future work in the field.

Chapter 4

A^{*} Orthogonal Matching Pursuit: Best-First Search for Compressed Sensing

4.1 Introduction

Tree search techniques have been occasionally utilized in straightforward manners for sparse signal recovery in the CS literature. For example, the tree search based orthogonal matching pursuit (TB-OMP) [38] employs a tree search that opens the best B children of each leaf node at each iteration. A rather flexible version of this method is the flexible TB-OMP [39], where the branching factor B is decreased at each level in order to reduce the tree size. Another straightforward tree search also appears for the fast Bayesian matching pursuit [40], which iteratively opens all children of the leaf nodes, and retains the best D among all opened nodes with respect to their posterior probabilities. These methods, however, can be seen as trivial applications of the tree search for sparse signal recovery purposes. The common practice shared by these methods can be simply outlined as opening a fixed number of children of all nodes at the deepest level, pruning these leaves, and then proceeding to the next level. Such applications of the tree search lack a number of important features which increase the efficiency and performance of the search, such as selection and extension of the best path on-the-fly, allowance of paths with different lengths in the tree, and appropriate pruning schemes. As a result of

such deficiencies, not only these rather unsophisticated tree search based techniques are suboptimal, but also their application remains limited only to a few dimensional sparse signal recovery problems due to large tree sizes appearing in practice.

The A* search [11–15] is an efficient best-first search technique which is frequently utilized in problems such as path finding, graph traversal, and speech recognition. One of the most important features of the A* search is its flexibility to allow for the existence of paths with different lengths in the tree. Via an appropriate auxiliary cost function, A* search makes comparison of such paths possible, allowing for iterative selection and expansion of the best path in the tree. This property promotes A* search as an efficient technique for utilization of the tree search in the sparse signal recovery problem.

This chapter introduces a semi-greedy sparse signal recovery algorithm based on the A* search technique. This algorithm, called A* orthogonal matching pursuit (A*OMP), utilizes the A* search to find the optimal solution of the sparse signal recovery problem on a search tree where the most promising path is iteratively expanded in a way similar to the well-known orthogonal matching pursuit (OMP) algorithm. Utilization of the best-first search allows for the efficient evaluation of multiple paths during the search, and hence promises improvements over the OMP-like algorithms, which can be considered as single path search techniques.

This combination of A* search and OMP is not straightforward: It necessitates appropriately defined cost models which enable the A* search to perform the stage-wise residue minimization in an intelligent manner, in addition to the effective pruning techniques which make the algorithm tractable in practice. We address the former by the introduction of three cost models, which allow for the comparison of paths with different lengths. These include two novel dynamic structures, which better comply with our needs, in addition to the conventional additive one. As for the pruning capability, we provide a number of strategies which, together with the cost model parameters, enable a complexity-accuracy trade-off. The effectiveness of the proposed pruning techniques in addition to the dynamic cost models is demonstrated via a number of reconstruction simulations. These simulations, including different nonzero coefficient distributions, Gaussian and Bernoulli type random observation matrices, noise contaminated measurements, and images, demonstrate that utilization of the best-first search is able to improve the reconstruction accuracy from compressed measurements.

To avoid some possible misunderstanding, we would like to note that the tree search in the context of the A*OMP algorithm is completely general to all kinds of sparse signals. That is, A*OMP is neither specific for the tree-sparse signals nor does it make use of a tree-structured overcomplete basis as for the tree-based OMP algorithm [51]. The algorithm is not specific for any other structured sparse signals as well. Furthermore, A*OMP aims at finding a closer approximation to the true solution with the minimum ℓ_0 norm, thus the objective is to improve reconstruction quality not to decrease computational complexity of finding a greedy solution, such as in the list decoding [115].

The findings of this chapter have been partially published in the Digital Signal Processing journal [116]. A preliminary version of this work has also been presented at the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2011) [117].

4.1.1 Outline

This chapter is organized as follows: We first summarize the A* search technique in Section 4.2. Section 4.3 is devoted to the discussion of the A*OMP algorithm, together with the novel cost functions and pruning methods. In Section 4.4, we describe the AStarOMP software, which we have developed for the purpose of fast and robust recovery via A*OMP, and discuss the related implementation issues. We finally demonstrate the reconstruction performance of A*OMP in comparison to the basis pursuit (BP) [66], subspace pursuit (SP) [17], and OMP [6] algorithms in Section 4.5, before concluding the chapter with a short summary of our findings.

4.2 A* Search

4.2.1 Fundamentals of the A* Search

A* search [11–15] is an iterative tree search technique which is based on finding the tree path that minimizes or maximizes some evaluation function. The A* search mechanism iteratively selects the most promising path in the tree, i.e., the one with the optimum value of the evaluation function, and expands this path by opening some of its children.

These steps are repeated until the chosen path fulfills some termination criterion, for example a specific number of nodes, or a specific value of the evaluation function.

The presentation of the A* search in this section relies on some particular choices regarding our particular problem. One of them is the choice of the evaluation function, which, in general, may involve minimization, maximization or any other meaningful optimality criterion that allows for distinguishing the best path among the others. Having said that, we concentrate merely on the minimization problem in the rest of this section, since our problem can be represented as a minimization over some evaluation function based on the residue (see the next section for a detailed formulation). Furthermore, this chapter utilizes a sparsity-based termination criterion for A*OMP, hence the discussion below is limited to the specific selection of the A* search termination criterion as finding an optimal path of length K , which denotes the sparsity level of the underlying signal. That is, we discuss the case where the A* search terminates when a path of length K turns out to be the one with minimum cost.

First, we would like to define some notation and concepts which ease the introduction of the A* search technique. Let $\mathcal{T}_i^i = \{t_1^i, t_2^i, \dots, t_{l^i}^i\}$ denote the set of nodes on the i th tree path, where l^i denotes the number of nodes and t_j^i is the j th node on the i th path, sorted by the order of selection. In addition, define $g(\mathcal{T}_i^i)$ as the evaluation function for the path \mathcal{T}_i^i . Next, we introduce two definitions:

Definition 4.1 (Complete path). A path is called *complete* if the number of nodes along that path is equal to the maximum number of possible nodes¹.

Definition 4.2 (Partial path). A path is called *partial* if the number of nodes along that path is less than the maximum number of possible nodes.

Mathematically, we are interested in finding the optimal complete path $\tilde{\mathcal{T}}_K$ that leads to the minimum value of the evaluation function, i.e.,

$$\tilde{\mathcal{T}}_K = \arg \min_{\mathcal{T}_K^i} g(\mathcal{T}_K^i). \quad (4.1)$$

A* search aims at finding this optimal complete path by selection and expansion of only one path per iteration. This process leads to a search tree which contains paths

¹A path is restricted to K nodes in this chapter, hence completeness of path i translates as $l^i = K$ for now. However, this number is extended to $K_{\max} > K$ in the next chapter in order to impose the residue-based termination criterion, hence we prefer stating a general definition here.

with different lengths. This is in fact one of the most important flexibilities of the A* search technique. However, it also introduces a fundamental problem: Paths with different lengths cannot be effectively compared using an evaluation function which directly depends on the number of nodes on a path, since such an evaluation function would unfairly favor longer paths, leading to suboptimal choices. In order to deal with this difficulty, A* search owns a correction mechanism that utilizes an auxiliary function to compensate for the unexplored nodes along a path.

For a path $\mathcal{T}_{l^i}^i$ of length $l^i \leq K$, the auxiliary function $d(\mathcal{T}_{l^i}^i)$ should be defined such that $d(\mathcal{T}_K^i) = 0$ and

$$d(\mathcal{T}_{l^i}^i) \geq g(\mathcal{T}_{l^i}^i) - g(\mathcal{T}_{l^i}^i \cup \mathcal{Z}_{K-l^i}), \quad \forall \mathcal{Z}_{K-l^i}, \quad (4.2)$$

where \cup denotes concatenation of two paths, and \mathcal{Z}_{K-l^i} is a sequence of $K - l^i$ nodes whose concatenation with $\mathcal{T}_{l^i}^i$ results in a complete path. With this definition, the auxiliary function $d(\mathcal{T}_{l^i}^i)$ is larger than or equal to the decrement in the evaluation function $g(\cdot)$ that any complete extension of the path $\mathcal{T}_{l^i}^i$ could yield.

Now, we define the cost function as

$$f(\mathcal{T}_{l^i}^i) = g(\mathcal{T}_{l^i}^i) - d(\mathcal{T}_{l^i}^i). \quad (4.3)$$

Let us consider a complete path \mathcal{T}_K^1 and a partial path $\mathcal{T}_{l^2}^2$ of length $l^2 < K$. Combining (4.2) and (4.3),

$$g(\mathcal{T}_K^1) \leq g(\mathcal{T}_{l^2}^2 \cup \mathcal{Z}_{K-l^2}), \quad \forall \mathcal{Z}_{K-l^2} \quad (4.4)$$

is guaranteed if

$$f(\mathcal{T}_K^1) \leq f(\mathcal{T}_{l^2}^2). \quad (4.5)$$

That is, (4.5) is sufficient to assure that \mathcal{T}_K^1 has a lower cost than all of the possible complete extensions of $\mathcal{T}_{l^2}^2$. Therefore, selection of the most promising path can be accomplished by minimizing the cost function $f(\mathcal{T}_{l^i}^i)$, once the auxiliary function is defined appropriately to satisfy (4.2).

We can now outline the A* search as follows: We start with an initial tree which consists of all possible paths with single nodes. At any iteration of the search, the tree path with the minimum cost is chosen for expansion. All children of this path are explored by adding the corresponding leaf nodes to the tree. This selection and expansion process is

iterated until the chosen path turns out to be complete². This path is returned as the final decision.

Before concluding this section, we would like to point out two important issues. First, satisfying (4.2) may be either impossible or unpractical. In that case, the algorithm should employ suboptimal cost models. This issue is discussed further in Section 4.3.4 in relation to the different A*OMP cost models. In addition, exploring all children of the most promising path at each iteration may also be intractable in practice. Therefore, we modify A* search in Section 4.3.3 by introducing pruning techniques such as limiting the number of explored paths per iteration and limiting the total number of paths in the tree.

4.2.2 Different Auxiliary Function Structures

The auxiliary function plays an important role in the performance of the A* search algorithm, since an inappropriate choice would lead to selection of suboptimal paths, and possibly to failure of the search. In a minimization problem, a reasonable selection methodology for the auxiliary function is to mimic the decay of the cost function over the unexplored nodes. For this purpose, we find it beneficial to consider different structures for the auxiliary function, which exploit different properties of the decay in the cost of a path.

The typical choice for the auxiliary function, which has been introduced above, employs an additive structure following [13]. Based on this structure, it is possible to define auxiliary functions with different forms. In Section 4.3.4, we derive two different cost models starting from this structure. For the first one, which we call additive cost model, we assume that each node is expected to decrease the cost by a constant value, i.e., the auxiliary function becomes equal to a constant times the number of unexplored nodes in the path. Second, we introduce an adaptive cost model, where the expected decrease in the cost is determined adaptively with respect to the decrease that has occurred during the addition of the previous node to the path. The simulation results given in Section 4.5 state that this adaptive model improves the performance of the A* search in the sparse signal recovery problem.

²As before, the discussion is restricted to the sparsity-based termination assumption. Any other termination criterion is obviously possible, such as the residue-based criterion of the next chapter.

Addition is only one of the possible structures for the auxiliary function. The auxiliary function may also be selected in other forms depending on the structure of the underlying problem. Below, we propose a second structure which employs a multiplicative auxiliary function model. The resultant form, which is based on the assumption that each node reduces the cost by some constant rate $0 < \alpha < 1$, is called the multiplicative cost model:

$$f_{\text{Mul}}(\mathcal{T}_l^i) = \alpha^{K-l^i} g(\mathcal{T}_l^i). \quad (4.6)$$

These two structures will form the basis for the introduction of the A*OMP cost models in Section 4.3.4. As details of these structures make much sense in the context of A*OMP, the related discussion will be extended in Section 4.3.4, in combination with the path selection mechanism of A*OMP.

4.3 Sparse Signal Reconstruction using A* Search

The A*OMP algorithm casts the sparse signal recovery problem as a tree search for the correct support of the K -sparse \mathbf{x} among a number of dynamically evolving candidate subsets. These candidate subsets are stored as paths from the root node to leaf nodes of the search tree, where each node represents an index in the support of \mathbf{x} , or equivalently a column of the dictionary Φ . The search tree is built up and evaluated iteratively by the A* search. The search starts with candidate subsets of single elements. At each iteration, the tree is expanded by appending new dictionary atoms to the most promising path, which is selected to minimize a cost function based on the residue. In this way, A*OMP performs a multi-path search for the best one among all possible K -element subsets of Φ . Though the A*OMP search tree actually restricts the search to a set of iteratively generated candidate subsets, it is general with the capability of representing all possible K -element subsets of Φ . Figure 4.1 illustrates evaluation of a sample search tree.

Incorporation of a multi-path search strategy is motivated by the expectation that it would improve the reconstruction accuracy especially where a single-path algorithm such as OMP fails because of the linear dependency of the dictionary atoms. In cases where the search over a single path yields a wrong representation, the correct one will mostly be in the set of closely related candidate representations. This issue is discussed

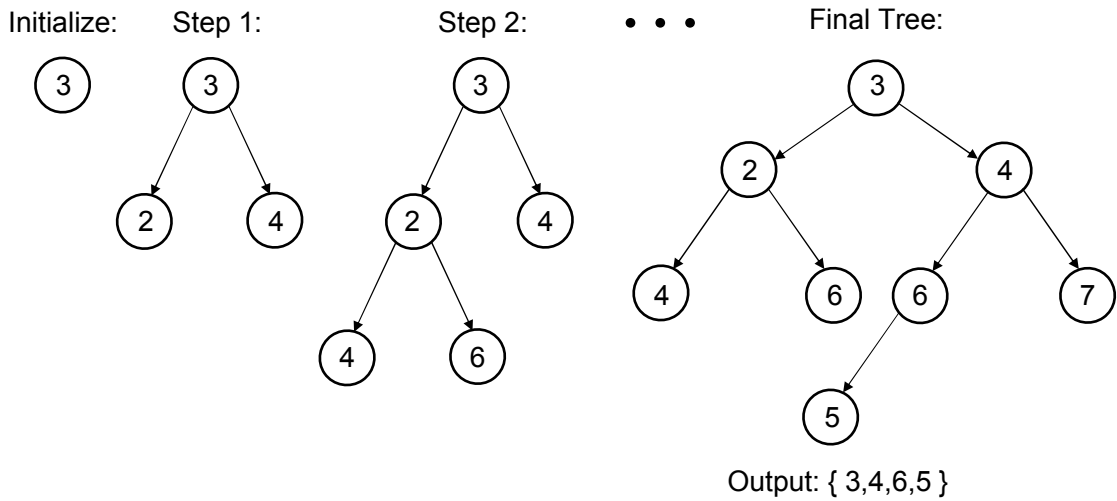


FIGURE 4.1: Evaluation of the search tree during A*OMP algorithm. The search is started with a single initial path. Two children of the best path is opened per iteration.

theoretically in the next chapter, where we state an RIP condition to guarantee that at least one of the nodes explored at an iteration is among the correct support of \mathbf{x} . This condition turns out to be less restrictive than the RIP condition for the success of OMP, which can be seen as a search where only a single node is explored. By a properly configured multi-path search, i.e., by proper selection of the cost model and pruning parameters as discussed below, correct paths can be distinguished from the other candidates. In such a case, the proposed multi-path strategy increases the recovery performance especially when too few measurements are provided.

Below, we first define the notation we use for the rest of this chapter. Then, we describe utilization of the tree search for A*OMP in three main steps: Initialization of the search tree, selection of the most promising, or the best, path, and expansion of the selected partial path. Next, we state a complete definition of A*OMP before discussing the complexity-accuracy trade-off via the provided search parameters.

4.3.1 Notation

We denote the K -sparse signal of interest by $\mathbf{x} \in \mathbb{R}^N$. M represents the number of observations. $\phi_i \in \mathbb{R}^M$ is the i th column of the dictionary, i.e., $\Phi = [\phi_1 \ \phi_2 \ \dots \ \phi_N]$. The observation vector is referred to as $\mathbf{y} \in \mathbb{R}^M$, where $\mathbf{y} = \Phi \mathbf{x}$. We represent the tree paths by $\mathcal{T}_i^i = \{t_1, t_2, \dots, t_i\}$, as defined in Section 4.2.1. Note that \mathcal{T}_i^i also denotes the support estimate of the corresponding path. \mathbf{r}^i is the residue of path i , and $\hat{\mathbf{x}}^i$ is the

estimate of \mathbf{x} by the i th path. The best path at a certain time is referred to as b . Let $\Phi_{\mathcal{J}}$ denote the matrix of the columns of Φ indexed by \mathcal{J} , and $\mathbf{x}_{\mathcal{J}}$ denote the vector of the elements of \mathbf{x} indexed by \mathcal{J} . Finally, \mathcal{S} refers to the set of all paths in the search tree.

4.3.2 Initialization of the Search Tree

For an N -dimensional signal, A* search originally initializes the search tree by N paths with length one. This is unpractical in most cases since N is usually large. In fact, only $K \ll N$ dictionary atoms are relevant to \mathbf{y} . Moreover, each iteration expands the tree with multiple children of the selected partial path, which introduces repetitions among the candidate paths. Hence, the search might be started with less paths than N to address the tractability issues. As a consequence, we limit the initial search tree to the $I \ll K$ subsets, each of which contains one of the I atoms having the highest absolute inner product with \mathbf{y} . Note that another possibility would be selecting a dynamic range of atoms whose absolute inner products with \mathbf{y} are greater than a certain threshold.

4.3.3 Expansion of the Selected Partial Path

In the typical A* search, all children of the most promising partial path are added to the search tree at each iteration. In practice, this results in too many search paths because of the high number of possible children: To illustrate, let the length of the selected partial path be l . The leaf node of this path has $N - l \approx N$ children since $l < K \ll N$. Hence, each iteration considers approximately N new paths. Consequently, given $K \ll N$, the upper bound on the number of paths involved overall in the search is obtained as $I \times N^{K-1}$. Since a search tree with that many nodes is not tractable, we employ three pruning strategies in order to limit the tree size:

4.3.3.1 Extensions per Path Pruning

For our purposes, the specific ordering of nodes along a path is unimportant. At each step, we require only to add one of the K correct atoms to the representation, and not a specific one of them. In addition, most of the dictionary atoms are irrelevant to \mathbf{y} since we typically have $K \ll N$. Moreover, the tree search visits repetitions of

the candidate paths at different branches by its nature. Combining all these reasons, expanding only a few children of the selected partial path becomes a reasonable sacrifice. At each A*OMP iteration, we expand the search tree only by the B children which have the highest absolute inner product with the residue of the selected path. This strategy is similar to the branching factor of the TB-OMP. Note that another reasonable choice would be considering only the children whose absolute inner products with the residue are higher than a threshold.

Extensions per path pruning decreases the upper bound on the number of paths from $I \times N^{K-1}$ to $I \times B^{K-1}$. Practically, I and B are chosen much smaller than N . Consequently, the number of paths explored throughout the search decrease drastically. Finally, we would like to note that $I \times B^{K-1}$ is only a very loose upper bound. The number of explored paths turns out to be much smaller than this bound in practice, as demonstrated by the recovery simulations below.

4.3.3.2 Tree Size Pruning

Even though the number of extensions per path is limited to B , addition of new paths at each iteration still increases the memory requirements, as the corresponding residues should also be stored. To reduce the memory requirements, we adopt a strategy similar to the “beam search”, and limit the maximum number of paths in the tree by the beam width P . When this limit is exceeded, the worst paths, i.e., the ones with maximum cost, are removed from the tree until P paths remain.

Figure 4.2 illustrates the extensions per path pruning and the tree size pruning rules where $P = 4$ and $B = 3$. Figure 4.2a depicts a search tree with four paths at the beginning of an iteration. The cost of each path is indicated as C_i . The path 4, which has the minimum cost, is selected as the best path. Let the best B children of the path 4 be the nodes 2, 8, and 9, ordered with descending correlation to the residue. In Figure 4.2b, the best child 2 is directly appended to the path 4. Note that appending the best child to the chosen path is safe since this does not increase the number of paths in the tree. Figure 4.2c depicts addition of the second child 8, after which there appear five paths on the tree. As the tree size is limited to $P = 4$, the path 2, which has the maximum cost, is removed by the tree size pruning rule. Finally, we consider the node 9 in Figure 4.2d for exploration. Observe that the resultant path has higher cost than

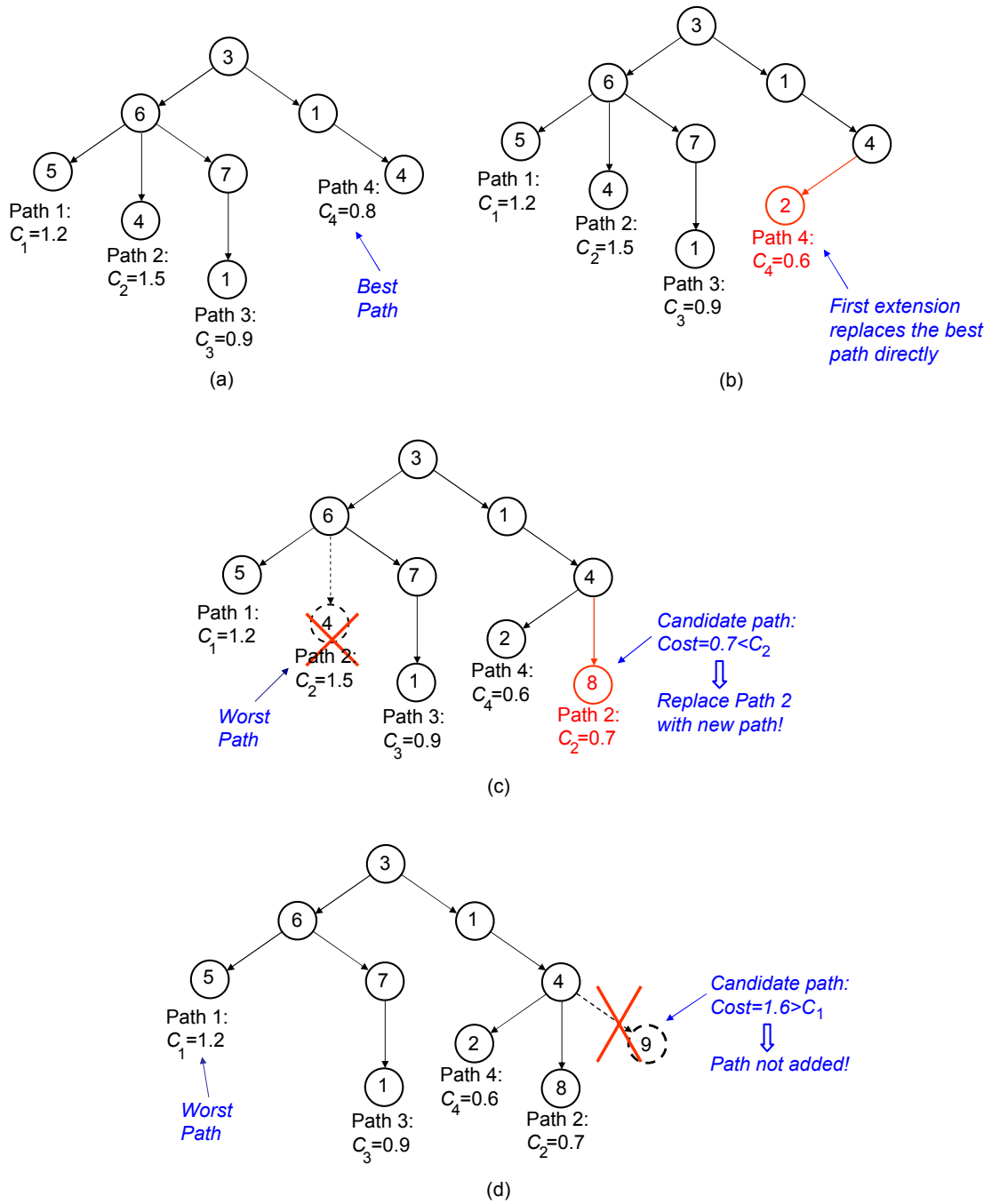


FIGURE 4.2: Evaluation of the search tree during a single iteration of the A*OMP algorithm

the other four paths in the tree. In this case, the tree size pruning rule does not allow addition of this path to the search tree.

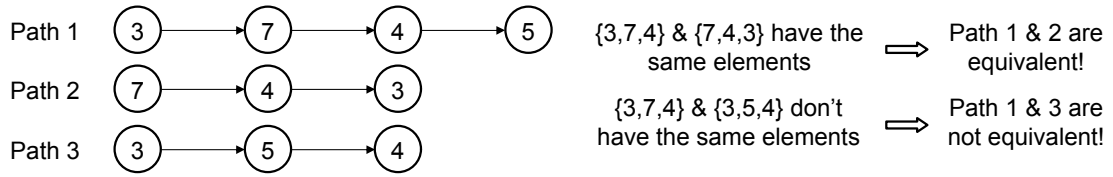


FIGURE 4.3: Path Equivalency: The paths 1 and 2 are equivalent as the first three nodes of the path 1 contain only the nodes in the path 2. The path 3 is not equivalent to the path 1 as the node 5 is not an element of the first three nodes of the path 1. Orthogonal projection ensures the path 2 to select node 5 as the next one, while there is no guarantee that the path 3 will select the node 7 next.

4.3.3.3 Equivalent Path Pruning

Avoiding insertion of equivalent paths to the search tree is also important for improving the search performance. For this purpose, we define the following path equivalency notion which covers not only the permutations of nodes along a path, but also paths with different lengths:

Definition 4.3 (Equivalent path). Let $\mathcal{T}_{l^1}^1$ and $\mathcal{T}_{l^2}^2$ be two paths with $l^1 \geq l^2$. Let us define $\mathcal{T}_{l^2}^{1,p}$ as the partial path that consists of the first l^2 nodes in $\mathcal{T}_{l^1}^1$, i.e., $\mathcal{T}_{l^2}^{1,p} = t_1^1, t_2^1, \dots, t_{l^2}^1$. $\mathcal{T}_{l^1}^1$ and $\mathcal{T}_{l^2}^2$ are equivalent if and only if $\mathcal{T}_{l^2}^{1,p}$ and $\mathcal{T}_{l^2}^2$ share the same set of indices. In this case, orthogonality of the residue to the selected support ensures that $\mathcal{T}_{l^2}^{1,p}$ and $\mathcal{T}_{l^2}^2$ represent exactly the same path. Consequently, insertion of $\mathcal{T}_{l^2}^2$ into the tree is unnecessary, as $\mathcal{T}_{l^2}^{1,p}$ has already been expanded by one of the previous iterations.

Figure 4.3 illustrates the path equivalency. The path 2 and the first three nodes of path 1 share the same set of indices, which makes the paths 1 and 2 equivalent. Orthogonal projection ensures that the node 5 will be among the best children of the path 2. On the contrary, the paths 1 and 3 are not equivalent as the first three nodes of the path 1 and the path 3 do not share the same set of nodes. There exists no guarantee that node 7 will be among the best children of the path 3.

Let us now summarize the extension of a selected partial path \mathcal{T}^b with these three pruning rules: First, the best B children of \mathcal{T}^b are chosen as the dictionary atoms having highest magnitude inner product with the residue. Next, we obtain B new candidate paths, each of which is formed by appending one of these B children to \mathcal{T}^b . We apply the equivalent path pruning rule by eliminating the candidates which are equivalent to the already explored paths in the tree. For each of the remaining candidate paths, we first compute the residue via the orthogonal projection of \mathbf{y} onto the subspace obtained

by extension of \mathcal{T}^b with the new index, and then the cost as discussed below. Then, we remove \mathcal{T}^b from the tree, and add the candidate paths to the tree. Finally, we prune the tree if the number of paths exceeds P .

4.3.4 Selection of the Most Promising Path

In the sparse signal recovery problem, a natural criterion for choosing the most promising path is the minimum residual error. Consequently, the evaluation function for the path \mathcal{T}_l^i can be written as

$$g(\mathcal{T}_l^i) = \|\mathbf{r}^i\|_2 = \|\mathbf{y} - \Phi \hat{\mathbf{x}}^i\|_2. \quad (4.7)$$

where $\hat{\mathbf{x}}^i$ is obtained as orthogonal projection coefficients of the residue onto the subspace defined by the set \mathcal{T}_l^i .

As discussed in Section 4.2, A* search employs an auxiliary function to compensate for the different path lengths appearing in the search tree. Definition of the auxiliary function is important for the simultaneous comparison of the multiple paths in the search tree. By proper evaluation of these paths, though any single one of them is limited to the RIP condition of OMP algorithm alone, A*OMP can relax the required RIP condition for exact recovery, increasing the probability of finding a final path that is not altered by the linear dependency of the atoms in the dictionary.

Ideally, the auxiliary function should mimic the decay of the residue along a path. In addition, it should also satisfy (4.2) to guarantee the optimality of the most promising path selection. Unfortunately, both of these are not practical. The former is impossible since the decay in the residue cannot be predicted precisely. On the other hand, and more vitally, the latter is not tractable for most cases, since it necessitates too many paths to be explored during the search unless some very specific assumptions are made on the nonzero elements of \mathbf{x} . Therefore, we need some reasonable assumptions in order to obtain cost models which lead to high recovery rates. Below, we suggest three different cost models which exploit different assumptions about the residue.

4.3.4.1 The Additive Cost Model

The additive cost model assumes that the K columns of Φ corresponding to the nonzero indices of \mathbf{x} make on the average equal contributions to $\|\mathbf{y}\|_2$. That is, we assume that the average contribution of a dictionary atom is $\delta_e = \frac{\|\mathbf{y}\|_2}{K}$. Then, the unopened $K - l$ nodes for the path \mathcal{T}_l^i are expected to reduce $\|\mathbf{r}^i\|_2$ by $(K - l)\delta_e$. Combining this with (4.2), the additive auxiliary function should satisfy

$$d_{\text{Add}}(\mathcal{T}_l^i) \geq (K - l^i) \frac{\|\mathbf{y}\|_2}{K}. \quad (4.8)$$

Consequently, we define the additive auxiliary function as

$$d_{\text{Add}}(\mathcal{T}_l^i) = \beta(K - l^i) \frac{\|\mathbf{y}\|_2}{K}, \quad (4.9)$$

where β is a constant greater than 1. Finally, we obtain the additive cost function as

$$f_{\text{Add}}(\mathcal{T}_l^i) = \|\mathbf{r}^i\|_2 - \beta \frac{(K - l^i)}{K} \|\mathbf{y}\|_2. \quad (4.10)$$

Here, β acts as a regularization constant. If it is large, shorter paths are favored, making the search explore more candidates. When it becomes smaller, the search prefers longer paths. Note that favoring shorter paths increases the number of paths opened throughout the search, which improves the recovery accuracy at the expense of increased complexity. Hence, β should be chosen to balance the available computational power or time restrictions and the recovery performance.

Note that $\delta_e = \frac{\|\mathbf{y}\|_2}{K}$ may not hold for each particular path. However, we observe that (4.8) requires this assumption only on the average in order to satisfy (4.2). Observe that the auxiliary function is mostly computed over a group of unexplored nodes instead of a single node. In such a case, it is practically sufficient when this group of unopened nodes make equal contributions on the average. Moreover, we intuitively expect the search to first select larger nonzero elements of \mathbf{x} . Hence, vectors with smaller magnitude coefficients are generally left to the deeper levels of the search tree, where satisfying (4.2) becomes more critical. Since such vectors have smaller contributions to $\|\mathbf{y}\|_2$, the additive auxiliary function satisfies (4.2) with higher probabilities at this more critical stages of the tree.

4.3.4.2 The Adaptive Cost Model

The auxiliary function can also be chosen dynamically by modifying the expectation on the average contribution of an unopened node on-the-fly as

$$\delta_e = \|\mathbf{r}_{l^i-1}^i\|_2 - \|\mathbf{r}_{l^i}^i\|_2, \quad (4.11)$$

where $\mathbf{r}_{l^i}^i$ denotes the residue obtained after the first l nodes of the i th path. As for the additive case, we incorporate $\beta > 1$ to obtain the adaptive auxiliary function as

$$d_{\text{Adap}}(\mathcal{T}_{l^i}^i) = \beta(\|\mathbf{r}_{l^i-1}^i\|_2 - \|\mathbf{r}_{l^i}^i\|_2)(K - l^i). \quad (4.12)$$

The adaptive cost function can then be written as follows:

$$f_{\text{Adap}}(\mathcal{T}_{l^i}^i) = \|\mathbf{r}_{l^i}^i\|_2 - \beta(\|\mathbf{r}_{l^i-1}^i\|_2 - \|\mathbf{r}_{l^i}^i\|_2)(K - l^i), \quad (4.13)$$

where the role of the regularization constant β is very similar to the additive case.

Empirical justification of the adaptive cost model is very similar to the additive case. As above, the decrement of the residual norm in (4.11) may not be satisfied for each particular node. However, the adaptive cost model mostly necessitates that the adaptive auxiliary function in (4.12) satisfies (4.2) on the average over a group of unexplored nodes. Hence, as for the additive case, it is practically sufficient when the adaptive auxiliary model is valid on the average.

4.3.4.3 The Multiplicative Cost Model

As defined in Section 4.2.2, the multiplicative cost model employs a weighting function which depends on the assumption that each node reduces the cost by a constant ratio, α . For our purposes, the multiplicative cost function is defined as

$$f_{\text{Mul}}(\mathcal{T}_{l^i}^i) = \alpha^{K-l^i} \|\mathbf{r}_{l^i}^i\|_2. \quad (4.14)$$

where α should be chosen between 0 and 1. The role of α is very close to that of β for the additive structure. When α is close to 0, short paths are assigned very small costs,

making the search to prefer them. On the contrary, if we choose α close to 1, weighting is hardly effective on the cost function, hence longer paths will be favored.

Similar to the additive structures above, the multiplicative cost model also needs to be valid only on the average. That is, a particular node may violate the underlying assumption, but it is empirically enough if a group of unexplored nodes satisfy it, which is more likely to occur. In addition, since the A* search is configured to select first the vectors with higher contributions to \mathbf{y} , the residue is expected to decrease slower among the deeper levels of the search tree, which are better covered by the multiplicative model.

In contrast to the additive one, the adaptive and multiplicative cost models adjust the expected decay in \mathbf{r}_i dynamically throughout the search. These dynamic structures are expected to provide a better modeling of the decay in $\|\mathbf{r}_i\|$, and hence improve the recovery accuracy. The simulation results in Section 4.5 indicate that they indeed yield higher recovery rates than the additive model.

4.3.5 A* Orthogonal Matching Pursuit

Having discussed the fundamental stages of the A*OMP algorithm, we can now complete the discussion by defining the termination criterion. In this chapter, we employ a sparsity-based termination criterion, which stops the search when the best path is complete. We will extend this criterion in the next chapter in order to get a more powerful A*OMP version.

We can now outline A*OMP: I out of the P paths, which are kept in a stack, are initialized as the I vectors which have the highest absolute inner product with \mathbf{y} , and the remaining $P - I$ paths are left empty. The cost for the empty paths is $\|\mathbf{y}\|_2$, hence they will be removed first. In each iteration, first, we select the path with minimum cost. We, then, expand the best B children of the selected path applying the pruning rules discussed in Section 4.3.3. The search continues to select and expand the best path iteratively until the selected path has length K .

The pseudo-code for the algorithm is given in Algorithm 4.1. Note that, for generality purposes, the termination criterion is specified as $(l^b \geq K_{\max}) \vee (\|\mathbf{r}^b\|_2 \leq \varepsilon\|\mathbf{y}\|_2)$. The sparsity-based criterion is imposed by setting $K_{\max} = K$, and $\varepsilon = 0$, while this structure also allows for other termination criteria via appropriate selection of these parameters.

Algorithm 4.1 A* ORTHOGONAL MATCHING PURSUIT

define: $P, I, B, \varepsilon, K_{\max}, f(\cdot), (\alpha \text{ or } \beta)$

input: Φ, \mathbf{y}

initialize:

$\mathcal{T}^i = \emptyset, \mathbf{r}^i = \mathbf{y}, \forall i = 1, 2, \dots, P$ ▷ empty paths

$\Delta\mathcal{T} = \arg \max_{\mathcal{J}, |\mathcal{J}|=I} \sum_{j \in \mathcal{J}} |\langle \phi_j, \mathbf{y} \rangle|$

for $i \leftarrow 1$ to I **do** ▷ I paths of length 1

$n = i$ th index in $\Delta\mathcal{T}$

$\mathcal{T}^i = \{n\}$

$\mathbf{r}^i = \mathbf{y} - \langle \mathbf{y}, \phi_n \rangle \phi_n$

end for

$b = 1$ ▷ initial best path

iterate:

while ($l^b < K_{\max}$) & ($\|\mathbf{r}^b\|_2 > \varepsilon \|\mathbf{y}\|_2$) **do**

$\Delta\mathcal{T} = \arg \max_{\mathcal{J}, |\mathcal{J}|=B} \sum_{j \in \mathcal{J}} |\langle \phi_j, \mathbf{r}^b \rangle|$ ▷ child nodes to be explored

$\tilde{\mathcal{T}} = \mathcal{T}^b$ ▷ store best path

$w = b$ ▷ first replace the best node

for $i \leftarrow 1$ to B **do**

$n = i$ th index in $\Delta\mathcal{T}$

$\hat{\mathcal{T}} = \tilde{\mathcal{T}} \cup \{n\}$ ▷ candidate path

$\mathbf{z} = \arg \min \|\mathbf{y} - \Phi_{\hat{\mathcal{T}}}\hat{\mathbf{z}}\|_2$ ▷ orthogonal projection

$\hat{\mathbf{r}} = \mathbf{y} - \Phi_{\hat{\mathcal{T}}}\hat{\mathbf{z}}$ ▷ update residue

if ($f(\hat{\mathcal{T}}) < f(\mathcal{T}^w)$) & ▷ tree size pruning

 ($\hat{\mathcal{T}} \notin \mathcal{S}$) **then** ▷ equivalent path pruning

$\mathcal{T}^w = \hat{\mathcal{T}}, \mathbf{r}^w = \hat{\mathbf{r}}$

end if

$w = \arg \max_{i \in \{1, 2, \dots, P\}} f(\mathcal{T}^i)$ ▷ replace worst path

end for

$b = \arg \min_{i \in \{1, 2, \dots, P\}} f(\mathcal{T}^i)$ ▷ best path

end while

return \mathcal{T}^b

4.3.6 Complexity vs. Accuracy

The complexity of A*OMP approach mainly arises from two time consuming operations: The inner product checks between the residue and the dictionary atoms, which necessitates both calculation and sorting of the inner products, and the orthogonal projection of the residue onto the subspace defined by the selected support set. The number of inner product checks is directly equal to the number of iterations. The orthogonal projection, on the other hand, is necessary for each path, except the I initial paths and the paths which are pruned by the equivalent path pruning. Let n_i and n_{eq} denote the number of iterations and the number of detected equivalent paths, respectively. Then, the number of orthogonal projections computed becomes $B(n_i - 1) - n_{eq}$. Consequently, the important factors that govern the complexity of A*OMP are, first, the number of iterations and, second, the number of equivalent paths detected. However, it is not possible to find some reasonable approximations of these. The only approximation to the number of paths is the upper bound which trivially assumes that every possible node on the tree is explored. Such an upper bound is obviously far away from being realistic. In order to provide an insight to this issue, we investigate n_i and n_{eq} experimentally in Section 4.5.1.1.

On the other hand, the pruning strategies introduced in Section 4.3.3 can be seen as a trade-off between the accuracy and complexity of A*OMP. If we set $I = N$, $B = N$, and $P = \infty$, the algorithm will perform an exhaustive search, which is prohibitively complex. On the other hand, setting $I = 1$ and $B = 1$ yields the simple OMP algorithm. A choice between the accuracy and complexity of the search can be made by adjusting the pruning parameters in between the two ends. The accuracy is expected to increase with these parameters, as demonstrated in Section 4.5.1.3. In practice, these parameters, of course, may not be increased after some point because of tractability issues. In addition, with respect to the results in Section 4.5.1.3, it is also questionable whether any further performance improvement can be achieved after some point.

The cost model is also extremely important in the complexity-accuracy trade-off. An appropriate modeling of the decay in the residue improves the ability to predict the branches on which the solution lies. Therefore, the auxiliary function is important for both choosing the best path and pruning. With an appropriate choice, the trade-off between the complexity and accuracy is boosted in the favor of accuracy, such as for the

dynamic cost functions which improve the reconstruction ability (see the first example in Section 4.5).

In addition, the auxiliary function parameters α and β also affect the complexity-accuracy trade-off. Choosing $\beta \gg 1$ or $0 < \alpha \ll 1$ makes the search favor shorter paths, leading to improvements in the recovery accuracy with longer search times. On the contrary, when β and α are close to 1, the algorithm performs similar to OMP. These improvements are, of course, also expected to have some limits, for example, decreasing α does not improve the performance after some point, as demonstrated in Section 4.5.1.3.

In order to get the best out of the search parameters, they should better be considered together. For example, reducing α increases the number of paths explored throughout the search. Consequently, a lower α value should be accompanied by an increment in the beam width P in order to improve the recovery accuracy. This also holds when β or B are increased, which similarly increase the number of paths involved in the search. The results of the recovery simulations in Section 4.5.1.3 further illustrate this issue.

4.4 AStarOMP Software for Fast Recovery with A* Search

In order to provide means for fast and robust sparse signal recovery with the A*OMP algorithm, we have developed the AStarOMP software package³ [118]. The AStarOMP software is designed for efficient handling of the implementation-related problems of the A*OMP algorithm, such as the storage and management of the search tree, the computation of the orthogonal projection, etc. AStarOMP incorporates a trie⁴ structure to implement the A* search tree in an efficient way. This trie structure allows for compact storage of the search tree in addition to efficient handling of tree operations such as addition of nodes and equivalent path detection, as discussed below. On the other hand, the orthogonal projection is performed using the QR factorization technique. In addition, the code for AStarOMP is developed in a flexible manner so that the corresponding A* search implementation is easily portable to other similar subset search

³The AStarOMP software package, the code, documentation, and the related MATLAB implementation are available at <http://myweb.sabanciuniv.edu/karahanoglu/research/>.

⁴A trie, or a prefix tree is defined as an ordered tree structure that is used to store a dynamic set in computer science.

problems. Below, we discuss these issues, while the reader is referred to the AStarOMP documentation for further implementation details.

4.4.1 Trie Structure in AStarOMP

The AStarOMP software incorporates a trie structure which implements the A* search tree in an efficient way. The trie structure of AStarOMP provides an efficient storage of the A* search tree where the nodes of each path are sorted with respect to decreasing priorities. Before the A* search starts, the priorities are calculated for each dictionary atom with respect to its similarity with the observation. The atoms which have higher absolute inner product to the observation get higher priorities. This priority order based on the correlation with the observation promises a number of important advantages. First, it reduces the tree size and the storage requirements by exploiting common nodes between paths. Second, it not only speeds up both the equivalent path detection and new path addition processes, but also allows for combination of them as a single operation.

By the nature of the tree search, there exists a high number of common nodes between the paths in the search tree. This, however, does not necessarily translate into the efficiency of the tree representation, as the ordering of these nodes is also important to reduce the tree size. In fact, the common nodes should appear at the same levels of different paths in order to reduce the storage needs. By the proposed priority-based ordering, AStarOMP aims at exploiting the common nodes as much as possible. As a consequence of this ordering, any specific set of nodes should appear in the same priority order, which is independent from the order of selection. Moreover, we intuitively expect that the atoms which are highly correlated to the residue are subject to be selected by more paths during the search. By the proposed priority ordering, such atoms are placed at lower levels of the trie. Combining these two advantages, the proposed priority-based ordering increases the number of shared nodes. This decreases the tree size, and hence the storage requirements.

Another advantage of using the proposed ordered structure is the ease of checking for equivalent paths. Without this ordering, the complexity of checking for equivalent paths would be very high since the nodes are free to appear at any level along any path. However, when the nodes in each path are sorted in the predefined order of decreasing priority, we can easily identify the location where a candidate node may exist in the trie

by traversing that particular path from leaf to the root. That is, for each path extension, we can easily identify the potential parent of the prospective node, and consequently, the level at which this prospective node should lie if an equivalent path is to exist in the tree. To illustrate, let's assume that we would like to add the node n to the path \mathcal{T} . In addition, let $\tilde{\mathcal{T}}$ represent the prospective path which this insertion would lead to. Via traversal of \mathcal{T} from its leaf node towards the root, we can easily find the potential parent node of n . We observe that the potential parent node of n , call it node p_n , breaks the path \mathcal{T} into two subpaths: Let $\mathcal{T}^{s,1}$ be the part of \mathcal{T} consisting of the node p_n and its ancestors, and $\mathcal{T}^{s,2}$ be the part of \mathcal{T} below the node p_n , i.e., the set of all of the descendants of p_n on the path \mathcal{T} . Similarly, define $\tilde{\mathcal{T}}^{s,1}$ as the part of $\tilde{\mathcal{T}}$ consisting of the node p_n and its ancestors, and $\tilde{\mathcal{T}}^{s,2}$ be the part of path $\tilde{\mathcal{T}}$ below the node p_n . By this definition, $\mathcal{T}^{s,1}$ and $\tilde{\mathcal{T}}^{s,1}$ share the same set of nodes, that is they are equivalent. Therefore, existence of $\tilde{\mathcal{T}}^{s,1}$ in the tree is obvious, and need not be checked. On the other hand, $\tilde{\mathcal{T}}^{s,2}$ is obtained as the ordered concatenation of n and $\mathcal{T}^{s,2}$. Consequently, if an equivalent of the prospective path $\tilde{\mathcal{T}}$ is to exist in the tree, $\tilde{\mathcal{T}}^{s,2}$ should already be below the node p_n . Moreover, the specific ordering of the nodes along $\tilde{\mathcal{T}}^{s,2}$ is already known as a by-product of the traversal of \mathcal{T} from the leaf node to the node p_n during the search for the parent node. Hence, the equivalent path detection is significantly simplified as a result of the priority-based ordering provided by the trie structure: To detect equivalent paths, we only need to check the existence of the ordered subpath $\tilde{\mathcal{T}}^{s,2}$ under the node p_n , whose location can be easily obtained. Note that both finding p_n and checking for the existence of $\tilde{\mathcal{T}}^{s,2}$ can be performed in linear time with the priority-based ordering. Otherwise, the detection of equivalent paths would require checking the equivalency of all tree paths to the prospective path, which is prohibitively complex. Equivalent path detections with and without the priority based ordering are compared in Figure 4.4.

Finally, the priority-based ordering provides another similar advantage for the addition of new paths to the tree. Observe that addition of the new path $\tilde{\mathcal{T}}$ is quite similar to the equivalent path detection for the same path: In fact, extension by the new node n is equivalent to the addition of the subpath $\tilde{\mathcal{T}}^{s,2}$ under the node p_n . Therefore, all the advantages listed above for the equivalent path detection are also valid for the addition of new paths. Moreover, the nature of the search provides another more significant advantage: Observe that the equivalent path detection should already be performed prior to each addition operation, since new extensions are only added to the tree if there

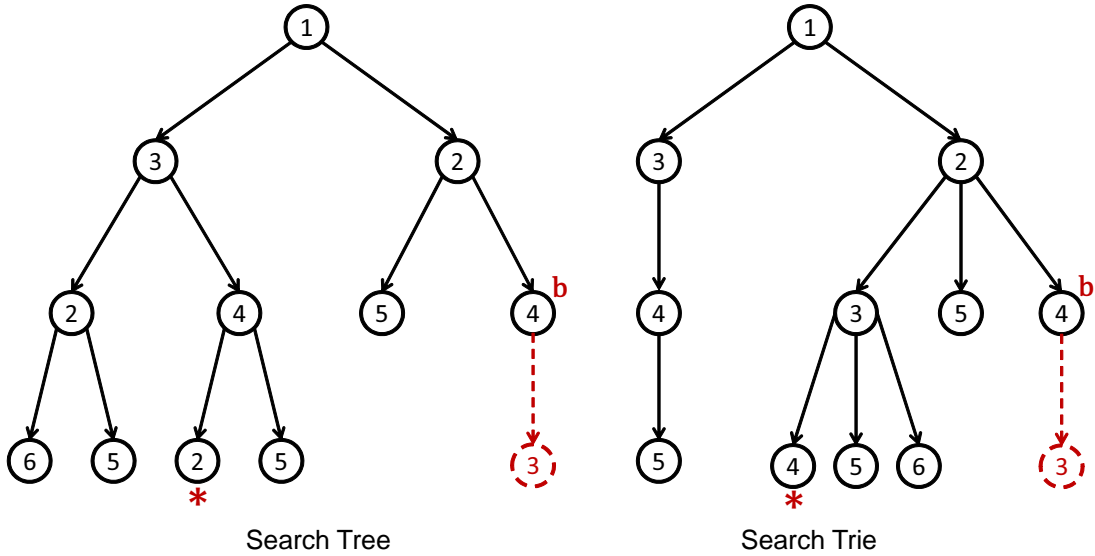


FIGURE 4.4: Comparison of equivalent path detection with tree (left) and trie (right) structures. b denotes the best path on both structures. Assume we expand b with node 3. The corresponding equivalent paths are marked by $*$ signs on both sides. On the left side, where no ordering is involved, the equivalent path appears on an arbitrary branch, and equivalent path detection requires a complex check over all paths in the tree. In contrast, the trie structure on the right side employs priority based ordering. For simplicity, we assume that the priority of each node is equivalent to its label, smaller labels appearing before the larger ones. As a result of this ordering, the equivalent path appears at a specific location. Observe that we can identify the potential parent of the new node 3 as the node 2. Then, we observe that nodes 3 and 4 exist as a subpath under the parent node 2, and simply detect the equivalent path. In addition, if the equivalent path would not exist, we would just identify the location of the new node 3 after the equivalent path check, and place it exactly at the location of the equivalent path.

exists no equivalent paths in the tree. Following the similarity of the path addition and the equivalent path detection, it is trivial to combine the two processes. That is, the equivalent path detection may be performed as a by-product of the path addition without any extra cost. Exploiting this fact, AStarOMP traverses a tree path only once per each addition operation. During this traversal, the subpath $\tilde{\mathcal{T}}^{s,2}$ and the parent node p_n are found. Then, it searches node by node for the existence of the ordered subpath $\tilde{\mathcal{T}}^{s,2}$ under p_n . If one of the nodes is not found in the specific order, then we conclude that there are no paths equivalent to the new one in the tree, and add this specific node with all of its children along $\tilde{\mathcal{T}}^{s,2}$ to the tree. Otherwise, if all nodes in $\tilde{\mathcal{T}}^{s,2}$ are found, then there exists an equivalent path in the tree, and no addition operation is performed.

4.4.2 Computation of the Orthogonal Projection in AStarOMP

As for solving the orthogonal projection, a number of least squares solvers can be employed. One of these alternatives is using the pseudo-inverse of the matrix which is composed of the selected columns from the dictionary. This method requires no additional storage per path except the residue, however is very slow due to the computation of the pseudo-inverse at each extension. Among other faster alternatives, we may count the Cholesky decomposition and the QR factorization. Among these two, the Cholesky decomposition is usually deemed for being slower than the QR factorization method, while it also requires less amount of storage. The QR factorization, on the other hand, is one of the fastest means for solving such least squares problems. Hence, AStarOMP utilizes the QR factorization method to solve the least squares problem for computing the orthogonal projection of the residue onto the subspace defined by the selected support estimate at each expansion.

With the QR factorization method, it is possible to obtain the projection residue without explicitly solving for the projection coefficients, which are not necessary for our purposes at intermediate iterations. We only need to solve for the \mathbf{Q} and \mathbf{R} matrices, with which the residue might be obtained. Hence, AStarOMP computes the projection coefficients only once to obtain the final estimate after the termination of the search. Moreover, in our case, the projection problem is incremental, that is, at each expansion, a new vector needs to be added to the projection basis. In this case, the QR factorization technique is extremely efficient since the \mathbf{Q} and \mathbf{R} matrices may be stored for further use. To exploit this advantage, each candidate path in the AStarOMP search tree stores a structure called the “side information”, which contains the related QR factorization data and the residue. When a path is to be expanded, this side information structure is extended by the selected basis vector, and stored for the new path.

Note that, not all paths in the tree, but only the P best ones should hold this side information, since the other ones have in fact been pruned, and will not be expanded. AStarOMP does not actually remove these pruned paths from the trie, but marks them as pruned. This is necessary to prevent the search from opening any paths equivalent to the pruned ones in the later stages. On the other hand, the leafs of the P candidate paths are kept in a separate list which is sorted by the cost of these paths. The side information structures are also kept in this list. This allows AStarOMP to store only

P side information structures. In addition, selection of the best and worst paths do not necessitate any additional sorting operation, since the list is already sorted. Finally, keeping a fixed number of side information structures is advantageous since this limits the storage requirements. The AStarOMP software has been implemented efficiently to allocate these side information structures just before the start of the search. The allocated side information structures are then dynamically assigned to the paths in the search stack. Hence, no additional memory allocation operation is necessary during the search.

As part of future work, the AStarOMP software package can be extended with the Cholesky decomposition, providing the user the possibility of choosing the appropriate approach depending on the problem size. The Cholesky decomposition might be useful especially when the recovery problem at hand is large, since it requires less memory than the QR factorization method.

4.4.3 Reusability of the Developed Code

As for the reusability of the developed code, AStarOMP has been implemented as a library consisting of two main components. One of these components consists of the A* search implementation, which is independent from any specific structures related to the sparse signal recovery problem. This structure allows for further reusability of the developed A* search implementation for other similar search problems, such as the subset search, as well. On the other hand, the problem specific path expansion and evaluation routines are collected in another class which is derived from a base OMP implementation. This separate class allows for easy implementation of any OMP-like greedy algorithm. These two components interact through an interface class, which allows for importing any similar search problem by simple modification of its function calls. Hence, the code can be easily ported for any similar search problem without modifying the actual A* search implementation.

As a result of this structure, both the A* search and OMP routines are available as separate packages which can easily be ported into any other similar problems. Therefore, the developed software provides means for not only the solution of other similar search problems by the modified A* search implementation provided but also efficient implementation of other OMP-like greedy sparse signal recovery approaches.

4.5 Empirical Analyses

In this section, we demonstrate sparse signal recovery via A*OMP in two different problems in comparison to the BP, SP, and OMP algorithms. The first of these problems is the recovery of synthetically generated 1D signals with different nonzero coefficient distributions from noise-free and noisy measurements. With this simulation set, the recovery performance of A*OMP is investigated thoroughly with respect to the sparsity level, the observation length, and the observation matrix type in addition to the demonstration of the complexity-accuracy trade-off via modification of the search parameters. The second problem, which involves recovery of sparse images, serves for the purpose of testing the proposed approach with realistic sparse coefficients.

As the A*OMP algorithm is run with a sparsity-based termination criterion in this chapter, OMP is also terminated after K iterations in the simulations below. Note that this corresponds to the OMP_K of the previous chapter.

The recovery simulations for A*OMP are performed using the AStarOMP software. The other algorithms are run using freely available tools such as the ℓ_1 -magic [108] and Sparsify [119] software packages.

4.5.1 Reconstruction of Synthetically Generated 1D Data

In this section, we evaluate three versions of A*OMP using the additive, adaptive, and multiplicative cost models. These are abbreviated as Add-A*OMP, Adap-A*OMP, and Mul-A*OMP, respectively. The experiments cover different nonzero coefficient distributions, including the uniform and Gaussian distributions as well as binary nonzero coefficients. We investigate reconstruction via Gaussian and Bernoulli observation matrices and compare different A*OMP parameters. Finally, we demonstrate A*OMP for sparse signal recovery from noisy observations.

All the simulations in this section are repeated over 500 randomly generated K -sparse samples. Reconstruction accuracies are given in terms of both the exact reconstruction rate and the average normalized mean-squared-error (ANMSE), which is defined as the average ratio of the square of the ℓ_2 norm of the reconstruction error to $\|\mathbf{x}\|_2^2$ over the 500 test samples. For the noisy scenarios, we give the reconstruction error in the

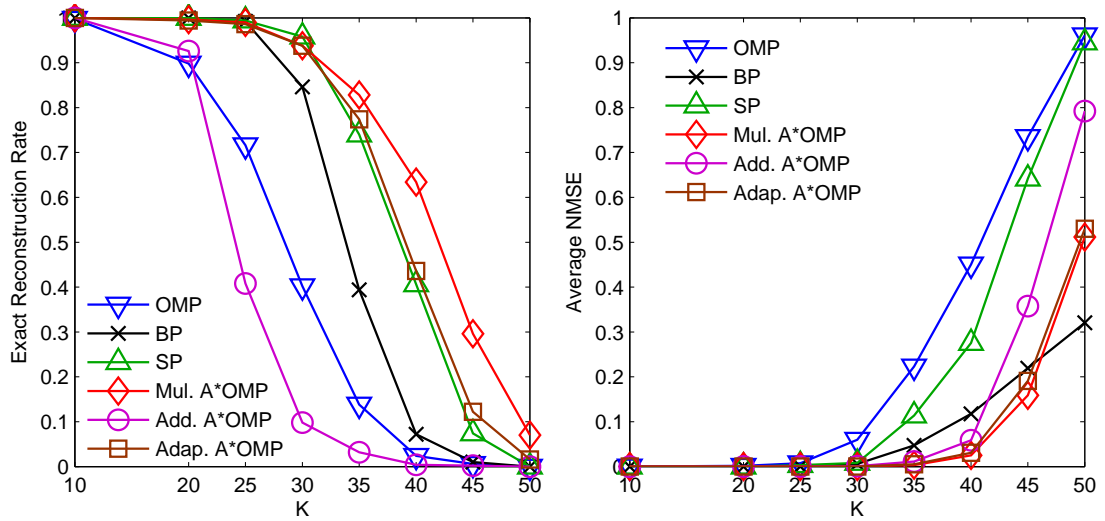
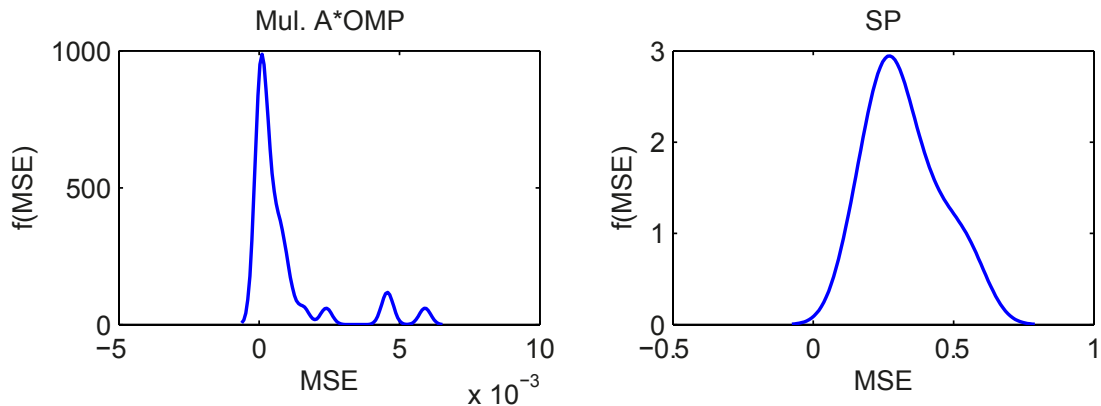
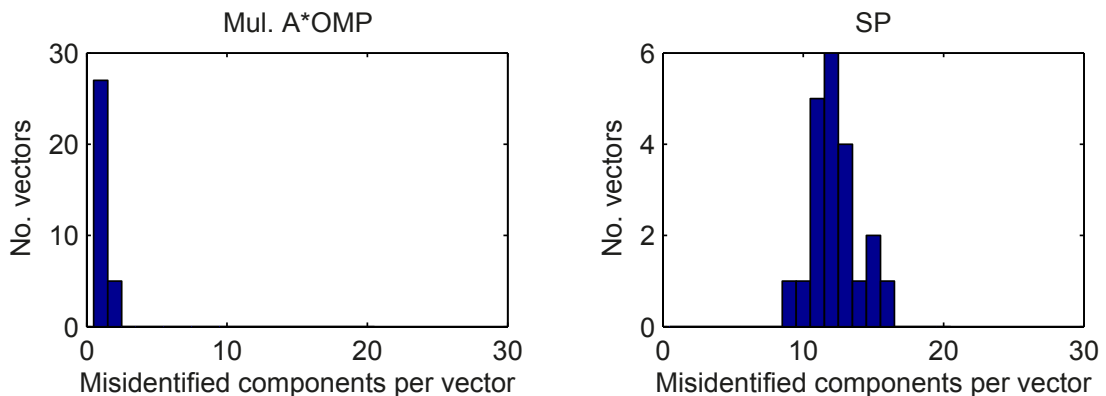


FIGURE 4.5: Reconstruction results over sparsity for the uniform sparse signals employing Gaussian observation matrices.

decibel scale, which we call the distortion ratio. Unless given explicitly, the following are common in all simulations: The A*OMP parameters were set as $I = 3$, $B = 2$, $P = 200$, $\beta = 1.25$, and $\alpha = 0.8$. Test samples have length $N = 256$ from which $M = 100$ random observations are taken. For each test sample, we employ an individual observation matrix Φ whose entries are drawn from the Gaussian distribution with mean 0 and standard deviation $1/N$.

4.5.1.1 Different Coefficient Distributions

The first set of simulations employ sparse signals with nonzero coefficients drawn from the uniform distribution in $[-1, 1]$. We refer to these signals as uniform sparse signals in the rest of this chapter. The results of these simulations are depicted in Figure 4.5 for $K \in [10, 50]$. In this test, Adap-A*OMP and Mul-A*OMP clearly provide lower ANMSE than BP, SP, and OMP, except for $K = 50$ where BP provides lower recovery error. Due to the sparsity-based termination, the ANMSE of OMP is the worst, while that of SP is only slightly better. BP provides lower error than SP and OMP, however it is still worse than A*OMP except for $K = 50$. Even the Add-A*OMP, which employs no dynamic cost model, yields lower error than BP up to $K = 40$. In addition to ANMSE, Mul-A*OMP, on general, yields higher exact recovery rates than the other candidates. Though SP yields high ANMSE, its exact recovery frequency competes with that of Mul-A*OMP up to $K = 30$, and even exceeds it slightly at $K = 30$. For Add-A*OMP,

FIGURE 4.6: Probability density estimates of the ANMSE values for $K = 30$.FIGURE 4.7: Number of misidentified entries per test sample for $K = 30$.

the situation is contrary: Despite the low ANMSE values, its exact reconstruction rate is even worse than that of the OMP algorithm. These results indicate that the static cost model of Add-A*OMP most of the time fails at smaller nonzero coefficients. The adaptive and multiplicative cost models, which dynamically adjust the expected decay in $\|\mathbf{r}\|_2$ individually for each path, are clearly more effective in compensating for the path length differences.

As for SP, the exact recovery rate is much better than the ANMSE values promise. This indicates that the amount of recovery error SP introduces per failure is much higher than that of the A*OMP algorithm. To visualize this fact, the probability density estimates of the ANMSE are depicted in Figure 4.6 for SP and Mul-A*OMP. These are computed using Gaussian kernels over ANMSE of the test vectors which cannot be exactly reconstructed for $K = 30$. The figures show that the ANMSE values are on the order of 10^{-3} for Mul-A*OMP, while they range up to 0.8, with mean about 0.3 for SP. This arises from the difference in the average number of misidentified elements per failure, which is shown in Figure 4.7 for $K = 30$. Mul-A*OMP has misidentified only

one or two of the 30 nonzero components, while SP has missed 9 to 16 components, and on the average about 12 per failure. These figures indicate that if the reconstruction is not exact, SP almost completely fails, however A*OMP can still reconstruct the sparse vector with a small amount of error, which is less than 1% of the signal norm for $K = 30$.

As discussed in Section 4.3.6, the two important factors for the complexity of A*OMP are the average number of A*OMP iterations per vector and the average number of equivalent paths detected per vector. Table 4.1 lists the average A*OMP iterations per vector in this scenario in comparison to the upper bound on the number of A*OMP iterations. This upper bound can easily be obtained as $I \times (2^{K-1} - 1)$ for $B = 2$ by assuming that all of the opened partial paths are selected one by one as the best path throughout the search. The actual number of iterations is incomparably lower than this upper bound. Moreover, though the upper bound increases exponentially with K , the actual number of iterations exhibit a much lower slope. The second important factor, the average number of equivalent paths per vector is given in Table 4.2. These numbers are comparable to the number of iterations, which states the effectiveness of the equivalent path pruning rule. These results indicate that pruning and proper selection of the cost model make it possible to run the search for cases where the upper bound becomes unpractically high.

TABLE 4.1: Average A*OMP iterations per vector for the uniform sparse signals

	K			
	10	20	30	40
Mul-A*OMP	13.8	164	1695	4177
Adap-A*OMP	19	167.4	2443	6109
Upper Bound	1533	1.57×10^6	1.61×10^9	1.65×10^{12}

TABLE 4.2: Average equivalent paths per vector for the uniform sparse signals

	K			
	10	20	30	40
Mul-A*OMP	4.4	114.1	975.2	1776
Adap-A*OMP	11.2	126.6	1355	1831

Finally, in order to provide an insight about the speed of the search, we list in Table 4.3 the average run times for Mul-A*OMP, Adap-A*OMP, and OMP on a modest Pentium Dual-Core CPU at 2.3GHz. These numbers are obtained using the AStarOMP software and a comparable OMP implementation developed using similar code pieces specially for obtaining comparable run times. Note that the indicated run times may be improved by appropriate means. For example, the structure of A*OMP makes parallel processing of the B candidates possible at each iteration. Moreover, the search can easily be modified to open more than one promising path per iteration in parallel. Hence, these run times can be significantly reduced by parallel programming techniques.

TABLE 4.3: Average run time in sec. per vector for the uniform sparse signals

	K			
	10	20	30	40
OMP	0.0012	0.0025	0.0036	0.0050
Mul-A*OMP	0.0022	0.0261	0.3158	0.8292
Adap-A*OMP	0.0032	0.0276	0.4601	1.1525

For the second set of simulations, we employ Gaussian sparse vectors, whose nonzero entries are drawn from the standard Gaussian distribution. Figure 4.8 depicts the ANMSE and exact reconstruction rates for this test. In this scenario, Mul-A*OMP provides clearly better reconstruction than BP, SP, and OMP. We observe that it provides both lower ANMSE and higher exact reconstruction rate than its competitors. SP yields the second best exact reconstruction rate, however, its ANMSE is the worst, as a consequence of the almost complete failure of an incorrect reconstruction.

In order to question the choice of the observation matrix, we repeat the last scenario with observation matrices whose nonzero entries are drawn from the Bernoulli distribution. The ANMSE values and the exact reconstruction rates for this test are illustrated in Figure 4.9. Comparing Figure 4.9 with Figure 4.8, we observe that the ANMSE values remain quite unaltered for Mul-A*OMP and BP, while those for SP increase. Mul-A*OMP leads to the least amount of error for Bernoulli-type observation matrices as well. As for the exact reconstruction, only BP keeps the same rates as above, while the rates of all others fall. However, Mul-A*OMP still provides the highest exact recovery rates in general. BP and SP compete with Mul-A*OMP until $K = 25$, where SP is

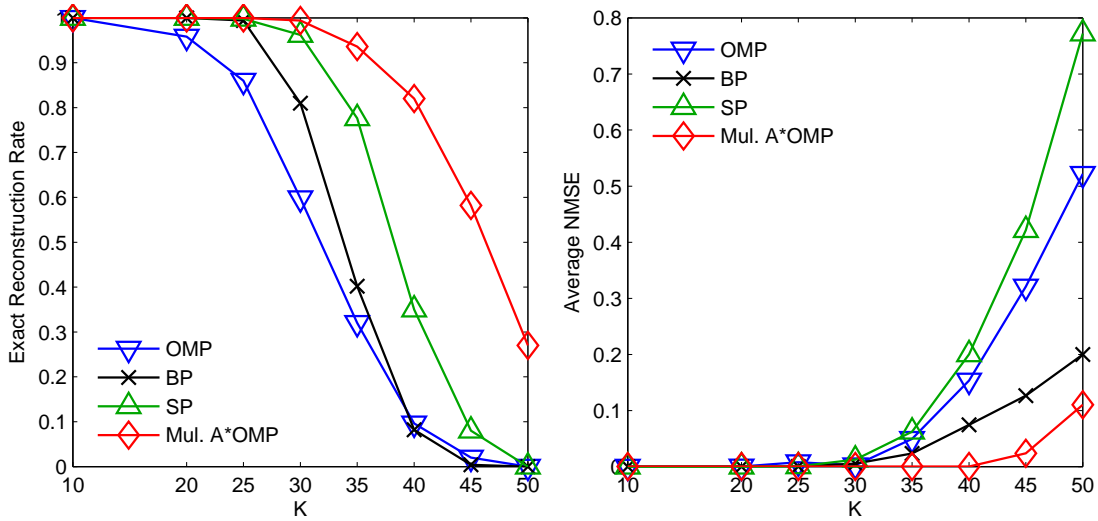


FIGURE 4.8: Reconstruction results over sparsity for the Gaussian sparse signals using Gaussian observation matrices.

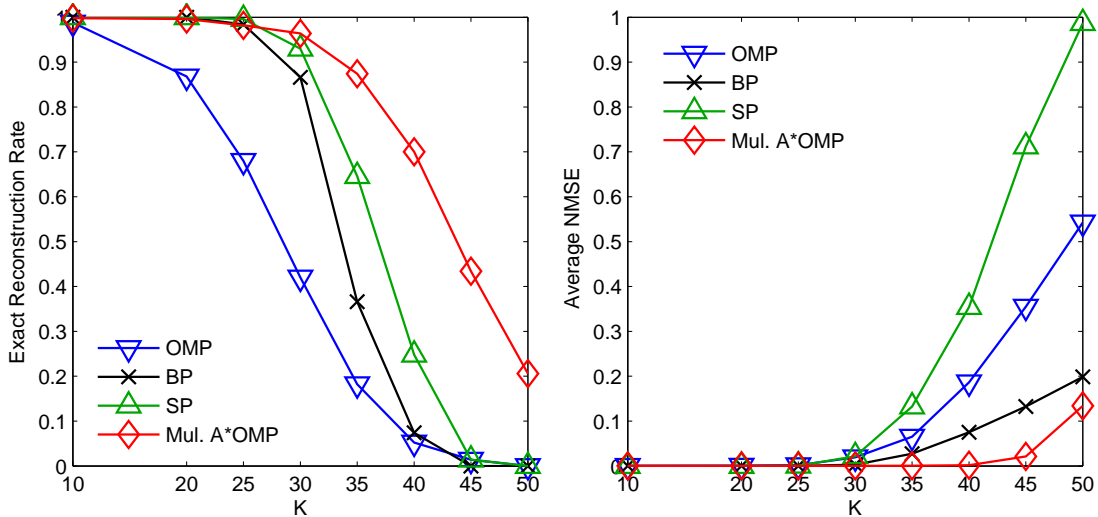


FIGURE 4.9: Reconstruction results over sparsity for the Gaussian sparse signals using Bernoulli observation matrices.

slightly better. When K further increases, Mul-A*OMP has the highest exact recovery frequency.

The next problem is the reconstruction of binary sparse signals, where the nonzero coefficients are selected as one. The results of this simulation set are shown in Figure 4.10. We observe that BP clearly yields better reconstruction than the others in this case. SP also performs better than A*OMP. The failure of A*OMP is related to the fact that this is a particularly challenging case for OMP-type of algorithms [17]. Consider the discussion in Section 3.5.1 about the narrow spread of the correlation vector between the observations and the dictionary atoms for OMP-type algorithms. As explained, this

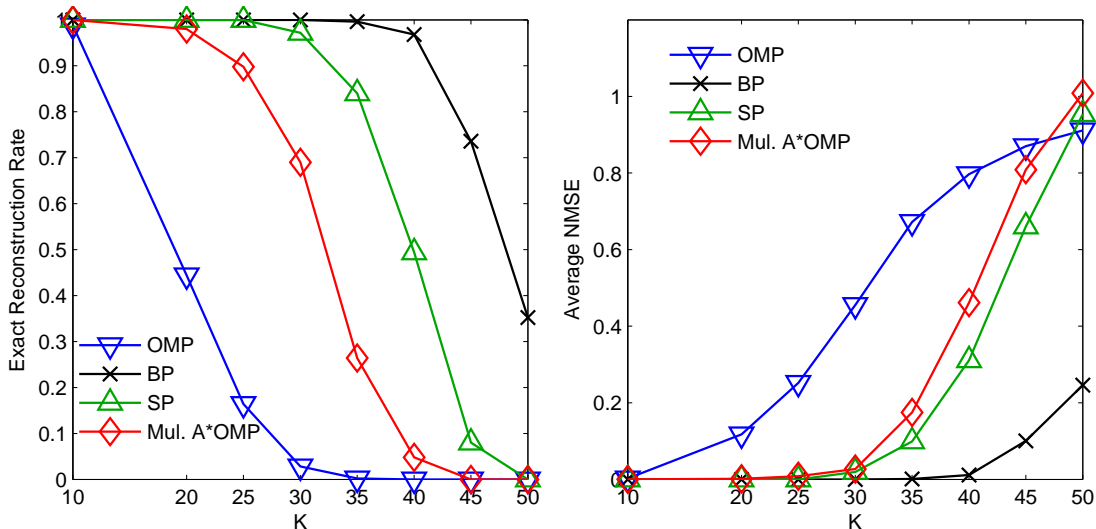


FIGURE 4.10: Reconstruction results over sparsity for the binary sparse signals using Gaussian observation matrices.

narrow range leads to the suboptimality of the correlation maximization process, and hence increases the failure rate of algorithms which involve a correlation maximization step. In contrast, for sparse binary signals, ℓ_0 norm of the correct solution is exactly equal to its ℓ_1 norm, which might be considered as an advantage for BP in this particular scenario.

4.5.1.2 Performance over Different Observation Lengths

Another interesting test case is the reconstruction ability over the observation length M . Figure 4.11 depicts the recovery performance over M for the uniform sparse signals where $K = 25$. For each M value, a single Gaussian observation matrix is employed to obtain observations from all signals. We observe that Mul-A*OMP is the best in terms of the exact recovery rates, while SP and BP compete it for $M \geq 90$ and $M \geq 100$, respectively. The ANMSE of Mul-A*OMP is also lower than the others except for the case of $M = 50$ where BP provides lower error than Mul-A*OMP. Note that 50 observations are already too few for exact recovery in this case since $2K = 50$. Therefore, even BP almost completely fails though it yields the lowest ANMSE for $M = 50$.

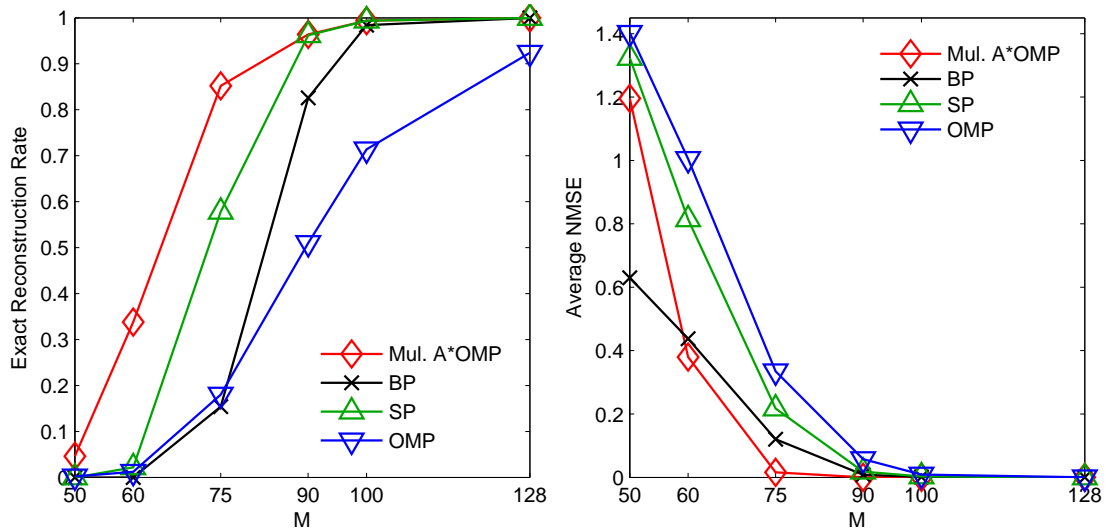


FIGURE 4.11: Reconstruction results over observation length for the uniform sparse signals where $K = 25$ using a single Gaussian observation matrix for each M .

4.5.1.3 Comparison of Different Search Parameters

Choosing the search parameters is an important issue for the A*OMP algorithm. This issue has been discussed above in Section 4.3.6, indicating two main points: The reconstruction performance of A*OMP might be increased by modifying the search parameters to explore more paths in the search at the expense of more iterations and longer search times. In order to demonstrate this, we consider two scenarios, where we first alter α , and next B together with P .

Figure 4.12 depicts the performance of Mul-A*OMP over α for the uniform sparse signals with $K = 30$ and $K = 35$. The dashed and solid lines indicate results for $P = 200$ and $P = 5000$, respectively. For $K = 30$, the reconstruction performance increases when α is reduced from 0.95 to about 0.8, whereas any further reduction of α does not significantly affect the performance. In addition, there is hardly any difference between selecting $P = 200$ and $P = 5000$. This suggests that setting $P = 200$ and $\alpha \approx 0.8$ seems to be enough for $K = 30$. When $K = 35$, however, more paths are involved in the search, and increasing P improves the reconstruction. When $P = 200$, reducing α below 0.9 does not improve the performance but slightly degrade it. On the contrary, if P is increased to 5000, the reconstruction is improved until α is reduced to 0.8, below which the reconstruction performance does not change anymore with α . Though not given in the figures, the authors have observed that setting $P > 5000$ has hardly any effect on the reconstruction. These results demonstrate that reducing α improves the reconstruction

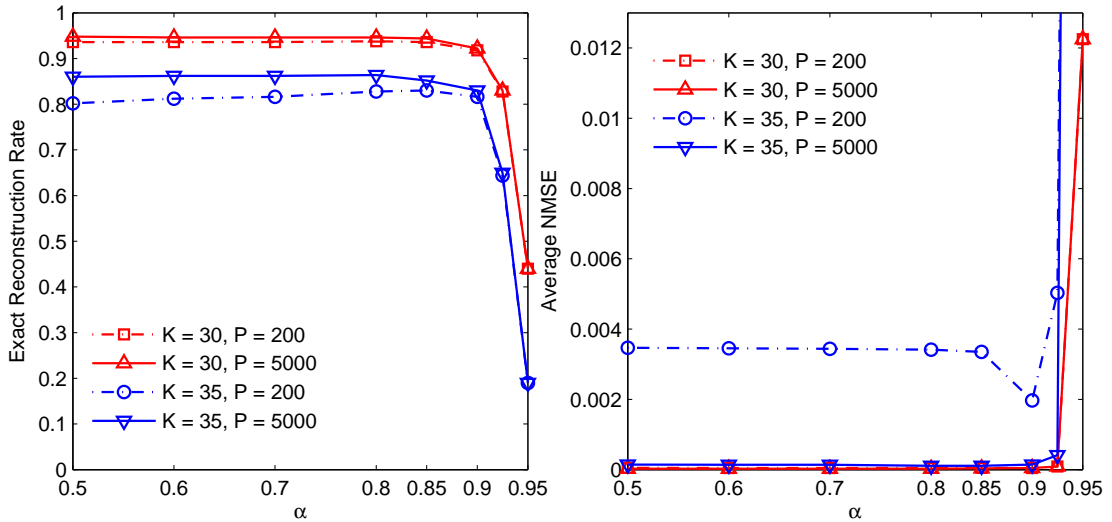


FIGURE 4.12: Reconstruction results over α for the uniform sparse signals using Gaussian observation matrices.

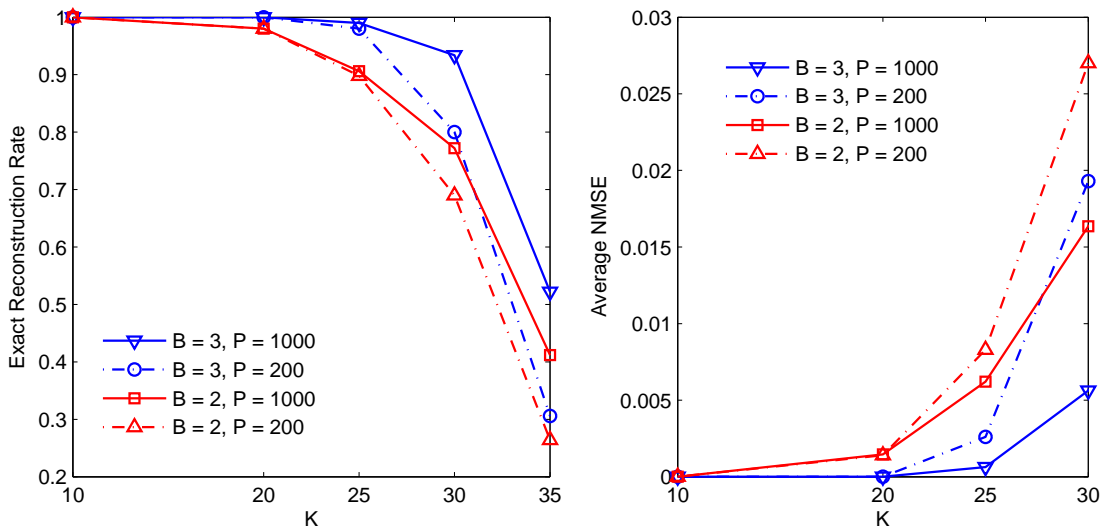


FIGURE 4.13: Reconstruction results for the sparse binary signals for $B = 2$ and $B = 3$ using Gaussian observation matrices.

until some convergence point. Table 4.4 lists the average number of search iterations while α and P are varied. We observe that decreasing α and increasing P increase the number of paths explored during the search. This clarifies the complexity-accuracy trade-off which leads to improved recovery performance at the expense of increased complexity.

Next, we illustrate the performance of Mul- A^* OMP with $B = 2$ and $B = 3$ for the sparse binary signals in Figure 4.13. The experiment is repeated for $P = 200$ and $P = 1000$, which are depicted by dashed and solid lines, respectively. We observe that increasing B from 2 to 3 improves the reconstruction. This improvement is further

TABLE 4.4: Average Mul-A*OMP iterations per vector with respect to α and P for the uniform sparse signals with $K = 35$

	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 0.8$	$\alpha = 0.9$
$P = 200$	4158	3927	3565	2932	1353
$P = 5000$	58204	51710	41781	25527	4026

enhanced by increasing P from 200 to 1000 when $K \geq 25$, where a larger search stack can better cover for the increased number of paths involved in the search. Table 4.5 lists the average number of search iterations, which increase with B and P . Hence, the improvement is obtained at the expense of complexity as above.

TABLE 4.5: Average Mul-A*OMP iterations with respect to B and P per vector in the sparse binary problem

	P = 200		P = 1000	
	B=2	B=3	B=2	B=3
$K = 10$	48	114	48	114
$K = 20$	1046	2095	1275	7159
$K = 30$	3424	4249	12278	18240

The results in this section explain how the performance of A*OMP can be adjusted by the search parameters. The mechanism behind is simple: Increasing the number of paths explored by the search improves the results until a convergence point, at the expense of increasing the complexity. According to the experimental results, one advantage is that even with modest settings such as $I = 2$, $P = 200$, and $\alpha = 0.8$ employed in the experiments, A*OMP can provide higher exact recovery rates and lower error than the other candidates for the uniform and Gaussian sparse signals. This indicates that the A*OMP recovery, at least in these cases, is quite robust against the choice of search parameters.

4.5.1.4 Reconstruction from Noisy Observations

In order to evaluate the recovery performance of A*OMP in noisy situations, we alter the observation model as

$$\mathbf{y} = \Phi \mathbf{x} + \mathbf{n} \quad (4.15)$$

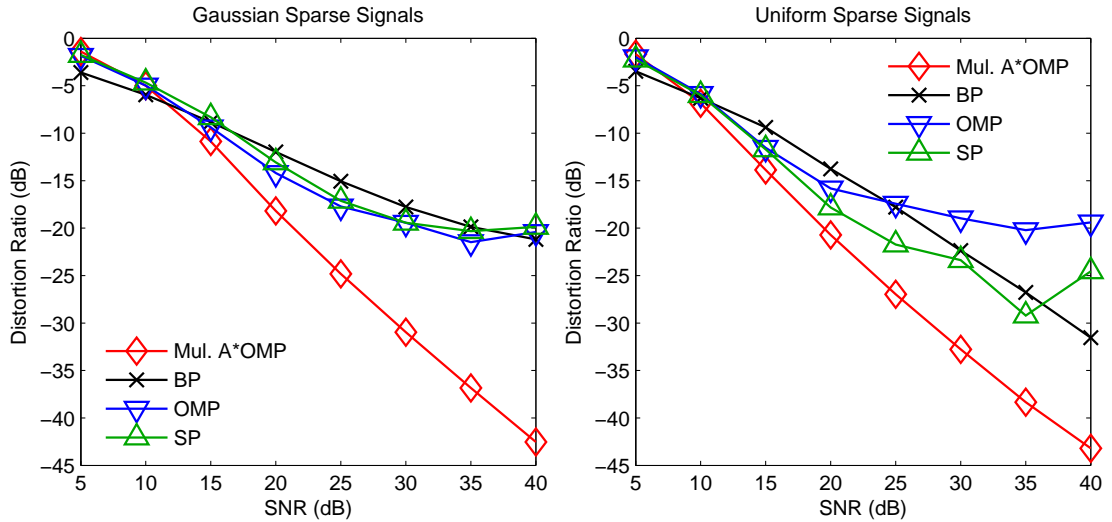


FIGURE 4.14: Average distortion ratio over SNR for reconstruction of sparse signals from noisy observations using Gaussian observation matrices.

where \mathbf{n} represents some additive noise component. We model \mathbf{n} as white Gaussian noise, and alter the signal-to-noise ratio (SNR), which is defined as $20 \log \frac{\|\mathbf{y}\|_2}{\|\mathbf{n}\|_2}$, for obtaining a general performance measure. Figure 4.14 illustrates the recovery results from these observation vectors which are contaminated by white Gaussian noise at different SNR levels. Here, K is 25 and 30 for Gaussian and uniform sparse signals, respectively. During these simulations, the regularization parameter of BP is adjusted proportional to the true SNR level. The results are shown in terms of the distortion ratio, which is in the decibel scale, for a better match with the SNR levels. We observe that Mul-A*OMP produces less distortion than BP, SP, and OMP for about 10 dB and higher SNR. When the SNR decreases, BP starts being slightly more effective than the other algorithms.

4.5.2 Reconstruction of Images

We finally simulate the reconstruction ability of A*OMP on some commonly used images including “Lena”, “Tracy”, “cameraman”, etc. In this experiment, the images are reconstructed in 8×8 blocks which provide important advantages for reducing both the complexity of the search and the memory requirements. First, without block-processing, the reconstruction problem requires searching among $N = 512^2 = 262144$ dictionary atoms. However, block-processing reduces the problem to 4096 subproblems with $N = 64$, which is more efficient as each subproblem requires a search in a 4096-fold reduced dimensionality. Second, block-processing reduces the total number of search paths drastically. To

illustrate, let's set $B = 2$. From Section 4.3.3, the number of search paths for each K -sparse block is upper bounded by $I \times 2^{(K-1)}$. Then, for the whole image, the upper bound becomes $4096 \times I \times 2^{(K-1)} = I \times 2^{(K+11)}$. If no block processing were involved, the upper bound would be $I \times 2^D$ where D , which denotes the sparsity level of the whole image, would clearly be much larger than $K + 11$. Finally, block-processing also reduces the length of the involved paths. Note that the block structure is shared by all of the involved recovery methods.

Despite the reduction in the complexity of the recovery process, the applicability of the block recovery as proposed above is still an important issue for practical compressed sensing of images. In fact, the use of this block recovery approach may be justified by the following intuitive observations: First, whenever it is possible to take the measurements of the whole image via some mask, measurements of an individual block may also be taken using a similar structure which may be obtained by setting the values of the mask corresponding to the other blocks as zero. That is, the mask may be configured such that only a single block of the image is observed through it at a time. On the other hand, future research may also facilitate the possibility of using individual masks for all image blocks in parallel, which would allow for taking simultaneous observations from all blocks. Such a strategy could even increase the efficiency of the observation process in addition to the simplification of the recovery problem, since each pixel would only contribute to a highly reduced number of local observations.

The simulations are performed with five 512×512 grayscale images using the 2D Haar Wavelet basis Ψ . Note that in this case, the dictionary is not Φ itself, but the holographic basis $\mathbf{V} = \Phi\Psi$. That is, \mathbf{x} denoting the sparse wavelet coefficient vector of interest, the image itself is obtained as $\Psi\mathbf{x}$ after the recovery of \mathbf{x} from the observation $\mathbf{y} = \Phi\Psi\mathbf{x}$. The images are first preprocessed such that each 8×8 block is K -sparse in the 2D Haar Wavelet basis where $K = 14$. A single observation matrix Φ of size $M \times N$, which is randomly drawn from the Gaussian distribution with mean 0 and standard deviation $1/N$, is employed to compute the measurements of length $M = 32$ from each block. Mul-A*OMP and Adap-A*OMP are run for both $B = 2$ and $B = 3$. We select $I = 3$ and $P = 200$. The cost function parameters are set as $\alpha = 0.5$ and $\beta = 1.25$. Here, α is reduced in order to compensate for the reduction in the auxiliary function due to the small K value.



FIGURE 4.15: Reconstructions of the image “Lena” using BP, SP, and Mul-A*OMP with $B = 3$

Table 4.6 lists the peak signal-to-noise ratio (PSNR) of the reconstructed images. A*OMP clearly yields better reconstruction than the other methods. Increasing B from 2 to 3 further improves the reconstruction performance. A*OMP improves PSNR up to 5.8 dB, and on the average 4.4 dB over BP. As an example, Figure 4.15 depicts reconstruction of “Lena” using SP, BP, and Mul-A*OMP with $B = 3$. That Mul-A*OMP reconstruction provides lower error can be observed better in Figure 4.16 which illustrates the absolute error per pixel for the BP and Mul-A*OMP reconstructions. For BP, errors are concentrated around the boundaries and detailed regions, while Mul-A*OMP clearly produces less distortion all around the image.

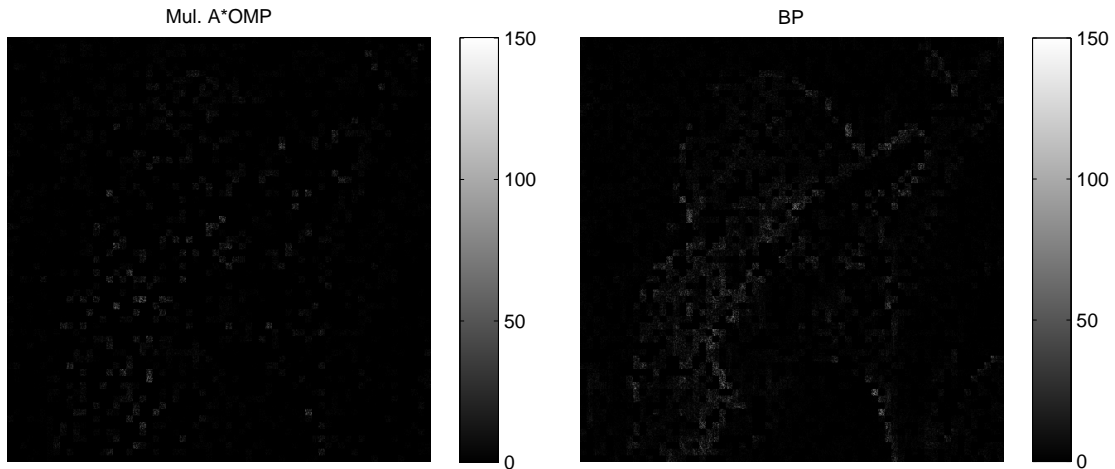


FIGURE 4.16: Reconstruction error per pixel of image “Lena” for Mul-A*OMP with $B = 3$ and BP.

TABLE 4.6: PSNR values for images reconstructed using different algorithms

	BP	OMP	SP	Mul-A*OMP		Adap-A*OMP	
				B=2	B=3	B=2	B=3
Lena	33.5	29.6	27.5	36.4	38.3	35.2	37
Tracy	40.6	36.8	33.9	44.8	46.4	44.5	45.5
Pirate	31.7	27.7	25.3	33.6	34.5	32.8	34.2
Cameraman	34.4	30.7	28.5	38.4	40.2	36.7	39.5
Mandrill	28.3	24.4	22.1	30.3	31.3	29.3	30.8

4.6 Summary

In this chapter, we have introduced a novel CS reconstruction approach, A*OMP, which is based on an effective combination of OMP with A* search. This semi-greedy method performs a tree search which favors the paths minimizing the cost function on-the-fly. In order to compare paths with different lengths, novel dynamic cost functions, which exhibit better recovery rates in the provided experiments, have been proposed. Pruning strategies have been introduced to limit the complexity, run time, and memory requirements of the search. A complexity-accuracy trade-off has also been provided via adjustment of the search parameters. In the presented demonstrations, A*OMP, with some modest settings, performs better reconstruction than BP and SP not only for the uniform and Gaussian sparse signals but also for sparse images. Moreover, it shows robust performance under the presence of noise for SNR values higher than 10 dB. To

conclude, the demonstrated reconstruction performance of A*OMP indicates that it is a promising approach which is capable of producing significant improvements in the recovery accuracy.

Chapter 5

Theoretical and Empirical Analyses of A*OMP With a Novel Adaptive Cost Model

5.1 Introduction

This chapter concentrates on the theoretical and empirical analyses of the A*OMP algorithm with a novel adaptive cost model. Our main contribution is the theoretical analysis of A*OMP, which includes not only exact recovery guarantees, but also theoretical comparison of two different termination criteria. The former states an RIP condition for exact recovery of sparse signals from noise-free measurements via A*OMP. The latter provides a theoretical understanding of the improvements in the recovery performance when the residue-based termination is employed instead of the sparsity-based one, which has been utilized in Chapter 4. As for the second important contribution of this chapter, we introduce a novel dynamic cost model, the adaptive-multiplicative cost model, which enhances the efficiency of the search by significant reduction of the run times. These claims are also supported by the extensive empirical recovery analyses provided. These analysis reveal two important aspects: First, the residue-based termination criterion improves both the accuracy and the speed of the A*OMP recovery. Second, the adaptive-multiplicative cost model effectively reduces the number of explored paths, which further accelerates the search.

A preliminary version of this chapter, including introduction of the adaptive-multiplicative cost model and some of the experiments, have been presented at the 2012 European Signal Processing Conference (EUSIPCO-2012) [120].

5.1.1 Definitions

Let us first clarify the notation used in this chapter: As before, we denote the K -sparse signal of interest by $\mathbf{x} \in \mathbb{R}^N$. M represents the number of observations. The observation matrix is defined as $\Phi = [\phi_1 \ \phi_2 \ \dots \ \phi_N]$, where $\phi_i \in \mathbb{R}^M$ denotes the i th column of Φ . The observation vector is referred to as $\mathbf{y} \in \mathbb{R}^M$, where $\mathbf{y} = \Phi \mathbf{x}$. We define \mathcal{T} as the correct support of \mathbf{x} . \mathcal{T}^i , \mathbf{r}^i , l^i , and $f(\mathcal{T}^i)$ denote the support estimate, residue, length, and cost of the i th path, respectively. $\hat{\mathbf{x}}^i$ is the estimate of \mathbf{x} by the i th path. The best path at a certain time is referred to as b . $\Delta\mathcal{T}$ represents the set of indices selected during the expansion of path b , i.e., the indices of B largest elements in $|\Phi^* \mathbf{r}^b|$. Finally, $\Phi_{\mathcal{J}}$ denotes the matrix of the columns of Φ indexed by \mathcal{J} , and $\mathbf{x}_{\mathcal{J}}$ is the vector composed of the elements of \mathbf{x} indexed by \mathcal{J} .

As discussed in Chapter 4, different termination criteria may be imposed by modification of the termination parameters K_{\max} and ε . The sparsity-based termination, utilized in Chapter 4 by setting $K_{\max} = K$ and $\varepsilon = 0$, searches for an exactly K -sparse representation. Below, we refer to this version of A*OMP as A*OMP $_K$. On the other hand, the residue-based termination is denoted by A*OMP $_e$, where $K_{\max} > K$ and problem-specific ε is selected very close to zero for noise-free observations, or small enough with respect to the noise level in the noisy case.

5.1.2 Outline

We present the main theoretical contributions of this chapter in Section 5.2, where we develop RIP-based guarantees for exact recovery of sparse signals from noise-free measurements via A*OMP. Our analysis method is similar to the OMP analyses presented in Chapter 3. We first develop conditions for the success of a single A*OMP iteration in Section 5.2.2. This result forms a basis for the rest of the theoretical analysis. In Section 5.2.3, we derive a RIP condition for exact recovery via A*OMP $_K$. This condition turns out to be less restrictive than the one developed in [7] for exact recovery with

OMP_K. In Section 5.2.4, we present conditions for exact recovery of a sparse signal via A*OMP_e. In addition, we show that exact recovery guarantees of A*OMP_K represent a special case of these conditions. Section 5.2.5 compares the exact recovery conditions of A*OMP_e and A*OMP_K. This clarifies that A*OMP_e possesses a less restrictive exact recovery condition than A*OMP_K. In a more general perspective, this result is parallel to the findings of Chapter 3, where OMP has been shown to have a less restrictive condition with the residue-based termination.

In Section 5.3, we discuss the adaptive-multiplicative cost model, which is an adaptive extension of the multiplicative cost model introduced in the previous chapter. This cost model allows for more flexibility when choosing the auxiliary function parameter α . That is, α may be chosen larger than the multiplicative cost model allows. This accelerates the search by reducing the number of explored nodes.

Finally, Section 5.4 presents extensive empirical analyses of A*OMP_e in comparison to basis pursuit (BP) [66], subspace pursuit (SP) [17], OMP [6], iterative hard thresholding (IHT) [30], iterative support detection (ISD) [46], and smoothed ℓ_0 (SL0) [72]. The most important ones among the presented results are the empirical phase transition graphs which are obtained by a set of computationally expensive experiments involving different signal characteristics. These clearly reveal the recovery abilities of A*OMP_e. In addition, we also investigate the recovery rates and average error for noisy and noise-free cases. Comparison of the run times illustrates the acceleration of the algorithm by both the adaptive-multiplicative cost function and the residue-based termination. Moreover, we test a hybrid of OMP and A*OMP_e for further speed up. Finally, A*OMP_e recovery is demonstrated on images.

5.2 Theoretical Analysis of A*OMP

5.2.1 Preliminaries

We first concentrate on two preliminary lemmas which are necessary for the analysis in the rest of this chapter. These statements are analogous to the ones presented prior to the theoretical discussion of OMP in Section 3.4. In particular, Lemma 5.1 follows from Lemma 3.3 by some simple derivation, and Lemma 5.2 is similar to the Lemma 3.4 of

Chapter 3, except for the introduction of B . In fact, Lemma 3.3 and Lemma 3.4 are special cases of Lemma 5.1 and Lemma 5.2 for $B = 1$.

Lemma 5.1 (Direct consequence of Lemma 3.3). *For positive integers K and B ,*

$$\delta_{K+B} > \frac{\delta_{3\lceil K/2 \rceil}}{3},$$

where $\lceil z \rceil$ denotes the ceiling of z , i.e., the smallest integer greater than or equal to z .

Proof. Lemma 3.3 already states that Lemma 5.1 holds for $B = 1$. By monotonicity of the RIC, $\delta_{K+B} \geq \delta_{K+1}$ when $B > 1$. Hence, Lemma 5.1 also holds for $B > 1$. \square

Lemma 5.2. *Assume $K \geq (3+2\sqrt{B})^2$. There exists at least one positive integer $n_c < K$ such that*

$$\frac{3\sqrt{B}}{\sqrt{K} + \sqrt{B}} \leq \frac{\sqrt{B}}{\sqrt{K - n_c} + \sqrt{B}}. \quad (5.1)$$

Moreover, n_c values which satisfy (5.1) are bounded by

$$K > n_c \geq \frac{8K + 4\sqrt{BK} - 4B}{9}. \quad (5.2)$$

Proof. Set $K - n_c = sK$ where $0 < s < 1$, and replace this into (5.1):

$$\frac{3\sqrt{B}}{\sqrt{K} + \sqrt{B}} \leq \frac{\sqrt{B}}{\sqrt{sK} + \sqrt{B}}.$$

It can trivially be shown that s is bounded by

$$0 < s \leq \left(\frac{\sqrt{K} - 2\sqrt{B}}{3\sqrt{K}} \right)^2.$$

Then, we obtain the lower bound for n_c as

$$\begin{aligned} n_c &= (1-s)K \\ &\geq \frac{8K + 4\sqrt{BK} - 4B}{9}. \end{aligned} \quad (5.3)$$

Now that $n_c < K$, we can write $sK = K - n_c \geq 1$. This translates as

$$\begin{aligned} K &\geq \frac{1}{s} \\ &\geq \left(\frac{3\sqrt{K}}{\sqrt{K} - 2\sqrt{B}} \right)^2, \end{aligned}$$

from which we deduce the assumption $K \geq (3 + 2\sqrt{B})^2$.

Combining this with (5.3), we conclude that the n_c values bounded by (5.2) satisfy (5.1) when $K \geq (3 + 2\sqrt{B})^2$. \square

5.2.2 Success Condition of an A*OMP Iteration

Let us define the success condition for an A*OMP iteration as $\Delta\mathcal{T}$ containing at least one correct index, i.e., $\Delta\mathcal{T} \cap \{\mathcal{T} - \mathcal{T}^b\} \neq \emptyset$. Theorem 5.1 establishes an RIP condition for the success of an iteration given the number of correct and incorrect indices in the support estimate of the best path \mathcal{T}^b :

Theorem 5.1. *Let $n_c = |\mathcal{T}^b \cap \mathcal{T}|$ and $n_f = |\mathcal{T}^b - \mathcal{T}|$. When path b is expanded, at least one index in $\Delta\mathcal{T}$ is in the correct support of \mathbf{x} , i.e., $\Delta\mathcal{T} \cap \{\mathcal{T} - \mathcal{T}^b\} \neq \emptyset$, if Φ satisfies RIP with*

$$\delta_{K+n_f+B} < \min \left(\frac{\sqrt{B}}{\sqrt{K} - n_c + \sqrt{B}}, \frac{1}{2} \right). \quad (5.4)$$

Proof. Remember that $\Delta\mathcal{T}$ is defined as

$$\Delta\mathcal{T} = \arg \max_{\mathcal{J}, |\mathcal{J}|=B} \sum_{j \in \mathcal{J}} |\langle \phi_j, \mathbf{r}^b \rangle|.$$

By some simple derivation, it can be shown that this is equivalent to

$$\Delta\mathcal{T} = \arg \max_{\mathcal{J}, |\mathcal{J}|=B} \|\Phi_{\mathcal{J}}^* \mathbf{r}^b\|_2. \quad (5.5)$$

Since \mathbf{r}^b is the orthogonal projection error of \mathbf{y} onto $\Phi_{\mathcal{T}^b}$, $\mathbf{r}^b \perp \Phi_{\mathcal{T}^b}$. Therefore,

$$\langle \phi_i, \mathbf{r}^b \rangle = 0, \quad \forall i \in \mathcal{T}^b.$$

Consequently, we write

$$\begin{aligned}\|\Phi_{\mathcal{T} \cup \mathcal{T}^b}^* \mathbf{r}^b\|_2^2 &= \sum_{i \in \mathcal{T} \cup \mathcal{T}^b} \langle \phi_i, \mathbf{r}^b \rangle^2 \\ &= \sum_{i \in T - \mathcal{T}^b} \langle \phi_i, \mathbf{r}^b \rangle^2.\end{aligned}\quad (5.6)$$

The right hand side of (5.6) contains only $K - n_c$ nonzero terms. Then, combining (5.5), (5.6), and the norm inequality, we get

$$\|\Phi_{\Delta \mathcal{T}}^* \mathbf{r}^b\|_2 = \max_{\mathcal{J}, |\mathcal{J}|=B} \|\Phi_{\mathcal{J}}^* \mathbf{r}^b\|_2 \geq c \|\Phi_{\mathcal{T} \cup \mathcal{T}^b}^* \mathbf{r}^b\|_2, \quad (5.7)$$

where c is defined as

$$c \triangleq \min \left(\sqrt{\frac{B}{K - n_c}}, 1 \right).$$

Next, the residue can be written as

$$\begin{aligned}\mathbf{r}^b &= y - \Phi_{\mathcal{T}^b} \hat{\mathbf{x}}_{\mathcal{T}^b}^b \\ &= \Phi_{\mathcal{T} \mathbf{x}_{\mathcal{T}}} - \Phi_{\mathcal{T}^b} \hat{\mathbf{x}}_{\mathcal{T}^b}^b \\ &= \Phi_{\mathcal{T} \cup \mathcal{T}^b} \mathbf{z},\end{aligned}\quad (5.8)$$

where $\mathbf{z} \in \mathbb{R}^{K+n_f}$. Using Lemma 3.1, (5.7), and (5.8), we write

$$\begin{aligned}\|\Phi_{\Delta \mathcal{T}}^* \mathbf{r}^b\|_2 &\geq c \|\Phi_{\mathcal{T} \cup \mathcal{T}^b}^* \Phi_{\mathcal{T} \cup \mathcal{T}^b} \mathbf{z}\|_2 \\ &\geq c(1 - \delta_{K+n_f}) \|\mathbf{z}\|_2.\end{aligned}\quad (5.9)$$

Now, suppose that the A*OMP iteration fails, i.e., $\Delta \mathcal{T} \cap T = \emptyset$. Then,

$$\|\Phi_{\Delta \mathcal{T}}^* \mathbf{r}^b\|_2 = \|\Phi_{\Delta \mathcal{T}}^* \Phi_{\mathcal{T} \cup \mathcal{T}^b} \mathbf{z}\|_2 \leq \delta_{K+n_f+B} \|\mathbf{z}\|_2$$

by Lemma 3.2. Clearly, this never occurs if

$$c(1 - \delta_{K+n_f}) \|\mathbf{z}\|_2 > \delta_{K+n_f+B} \|\mathbf{z}\|_2$$

or equivalently

$$\frac{\delta_{K+n_f+B}}{c} + \delta_{K+n_f} < 1. \quad (5.10)$$

Following the monotonicity of RIC, $\delta_{K+n_f+B} \geq \delta_{K+n_f}$. Hence, (5.10) is satisfied when $(\frac{1}{c} + 1) \delta_{K+n_f+B} < 1$, or equivalently

$$\begin{aligned} \delta_{K+n_f+B} &< \frac{c}{1+c} \\ &< \min\left(\frac{\sqrt{B}}{\sqrt{K-n_c} + \sqrt{B}}, \frac{1}{2}\right) \end{aligned}$$

by the definition of c . This condition guarantees that $\Delta\mathcal{T} \cap \mathcal{T} \neq \emptyset$. Moreover, since $\langle \phi_i, \mathbf{r}^b \rangle = 0$ for all $i \in \mathcal{T}^b$, we know that $\Delta\mathcal{T} \cap \mathcal{T}^b = \emptyset$. Hence, we conclude $\Delta\mathcal{T} \cap \{\mathcal{T} - \mathcal{T}^b\} \neq \emptyset$, that is the A*OMP iteration is successful when (5.4) holds. \square

Theorem 5.1 does not directly imply any exact recovery guarantees. However, it is used below as a basis for developing the exact recovery guarantees of A*OMP. Note that we assume that $\sqrt{B} \leq \sqrt{K-n_c}$, and skip the term $\frac{1}{2}$ in Theorem 5.1 for simplicity most of the time in the rest of this chapter. This assumption can be justified by the fact that B is chosen small, such as 2 or 3, in practice.

5.2.3 Exact Recovery Conditions for A*OMP_K

Exact recovery via A*OMP_K requires some conditions on the best path selection in addition to Theorem 5.1. For this purpose, we need to present some definitions. First, remember that the path i is defined as *complete* in Section 4.2.1 if $l^i = K_{\max}$. For A*OMP_K, this condition turns out to be $l^i = K$ since $K_{\max} = K$. In addition, we introduce the following definitions:

Definition 5.1 (Optimal path). Path i is called *optimal* if $\mathcal{T}^i \subset \mathcal{T}$.

Definition 5.2 (Optimal pruning). Pruning is defined as *optimal* if it does not remove all of the optimal paths from the search tree.

Definition 5.3 (Optimal cost condition). The *optimal cost condition* is defined as

$$F(\widehat{\mathcal{T}}^i) < F(\mathcal{T}^j), \quad \forall \widehat{\mathcal{T}}^i \in \mathcal{S}^{opt}, \quad \forall \mathcal{T}^j \in \{\mathcal{S}_K - \mathcal{S}^{opt}\}, \quad (5.11)$$

where \mathcal{S}_K and \mathcal{S}^{opt} denote the sets of all complete paths and all optimal paths, respectively.

In words, the optimal cost condition assures that the cost of an optimal path is lower than that of any nonoptimal complete path. Once satisfied, this guarantees that A*OMP_K either terminates at an optimal path or there are no optimal paths in the search tree.

Theorem 5.2 exploits these definitions to develop exact recovery guarantees for A*OMP_K:

Theorem 5.2. *Assume that the optimal cost condition holds and the pruning is optimal. Set $I \geq B$. Then, A*OMP_K perfectly recovers any K -sparse signal from noise-free measurements if the observation matrix Φ satisfies RIP with*

$$\delta_{K+B} < \frac{\sqrt{B}}{\sqrt{K} + \sqrt{B}}. \quad (5.12)$$

Proof. Let us start with the initialization. Replacing $n_c = n_f = 0$ into Theorem 5.1, (5.12) assures success of the initialization since $I \geq B$.

Next, assume that A*OMP_K selects an optimal path of length l , where $n_c = l$ and $n_f = 0$. By Theorem 5.1, expansion of this path is successful if

$$\delta_{K+B} < \frac{\sqrt{B}}{\sqrt{K-l} + \sqrt{B}}. \quad (5.13)$$

Since $l > 0$, (5.13) is less restrictive than (5.12). Hence, expansion of an optimal path is guaranteed to be successful when (5.12) holds.

Now, we have shown that there exists one or more optimal paths after the initialization when (5.12) holds. Moreover, we have also shown that expansion of an optimal path introduces at least one longer optimal path, and by assumption pruning cannot remove all of the optimal paths. Altogether, these guarantee existence of at least one optimal path in the tree at any iteration. Under these conditions, the optimal cost condition in (5.11) assures selection of optimal paths before termination. That is, the search cannot terminate at a suboptimal path. Instead, optimal paths are chosen for expansion, until the search terminates at an optimal complete path. Therefore, we conclude that A*OMP_K guarantees exact recovery of any K -sparse signal from noise-free measurements if (5.12) and the other assumptions are satisfied. \square

The assumptions in Theorem 5.2 may at first seem restrictive. However, intuitive reasoning states that these are not only quite reasonable, but also necessary in practice. First, since the ordering of nodes along a path is unimportant, there exists $K!$ possible

paths which represent the correct solution \mathbf{x} . That is, there exists a large number of optimal paths in the tree in case (5.12) holds, while it is sufficient that not all but only one of them satisfies the assumptions. In addition, optimal paths tend to have very small costs, which make them less likely to be pruned. Finally, obtaining theoretical guarantees for the optimal cost condition is hard, since this requires cost models which are neither trivial nor practical. The former follows from the fact that we cannot exactly predict the decay of the residue along unexplored nodes. Yet, the latter is more important: In practice, the cost model should be efficient, i.e., A*OMP should explore as few nodes as possible. Hence, cost models which explore too many nodes should be avoided. As a result, practical implementations have to deal with cost models which cannot guarantee (5.11), such as the ones we employ. Therefore, we build our analysis on the expectation that the optimal cost condition holds for some optimal paths, which is justified by intuitive discussion of the cost models (see the introduction of the cost models in Chapter 4) in addition to the empirical recovery performance A*OMP provides in the simulations.

We observe that the RIP condition $\delta_{K+1} < \frac{1}{\sqrt{K+1}}$ of Theorem 3.1, which guarantees exact recovery via OMP_K [7], can be obtained as a special case of Theorem 5.2 when $B = I = 1$. Moreover, when the exact recovery conditions of OMP_K and A*OMP_K are compared, (5.12) is clearly less restrictive, which explains the improved recovery accuracy of A*OMP $_K$ over OMP_K .

5.2.4 Exact Recovery with A*OMP $_e$

When $K_{\max} > K$, we need to extend the definitions of the previous section. First, note that path i is now complete if $l^i = K_{\max} > K$. In addition, we introduce the following terms:

Definition 5.4 (Potentially-optimal path). Path i is called *potentially-optimal* (*p-optimal*) if $K + n_f \leq K_{\max}$ and Φ satisfies RIP with

$$\delta_{K+n_f+B} < \frac{\sqrt{B}}{\sqrt{K-n_c} + \sqrt{B}}, \quad (5.14)$$

where n_c and n_f are the number of correct and false indices in the particular path i as before.

Definition 5.5 (Potentially-optimal pruning). Pruning is defined as *p-optimal* if it does not remove all of the p-optimal paths from the search tree.

Definition 5.6 (Potentially-optimal cost condition). The *p-optimal cost condition* is defined as

$$F(\widehat{\mathcal{T}}^i) < F(\mathcal{T}^j), \forall \widehat{\mathcal{T}}^i \in \mathcal{S}^{\text{p-opt}}, \forall \mathcal{T}^j \in \{\mathcal{S}_{K_{\max}} - \mathcal{S}^{\text{p-opt}}\},$$

where $\mathcal{S}_{K_{\max}}$ and $\mathcal{S}^{\text{p-opt}}$ denote the set of all complete paths and the set of all p-optimal paths, respectively.

Next, a recovery condition for A*OMP_e is stated in Theorem 5.3 using the lemma discussed below:

Lemma 5.3. *Let path i be p-optimal with n_c correct and n_f incorrect indices. Then, expansion of path i introduces at least one p-optimal path with $n_c + 1$ correct indices.*

Proof. By p-optimality, expansion of path i is successful and introduces at least one path, say j , with $n_c + 1$ correct and n_f incorrect indices. Moreover, the upper bounds which (5.4) imposes on the RIC for the paths i and j are related as

$$\frac{\sqrt{B}}{\sqrt{K - n_c} + \sqrt{B}} < \frac{\sqrt{B}}{\sqrt{K - n_c - 1} + \sqrt{B}},$$

where the left and right hand sides are the upper bounds for path i and j , respectively. Since the upper bound on the required RIC is larger for path j , and path i satisfies (5.4), path j also satisfies (5.4). In addition, $K + n_f \leq K_{\max}$ also holds for path j , as the successful expansion of path i does not alter n_f . Consequently, path j is p-optimal. Therefore, we conclude that expansion of path i introduces at least one p-optimal path with $n_c + 1$ correct indices. \square

Theorem 5.3. *Set $\varepsilon = 0$ and $K_{\max} \leq M - K$. Let Φ be full rank. Assume that the p-optimal cost condition holds and the pruning is p-optimal. Then, A*OMP_e perfectly recovers a K -sparse signal from noise-free measurements if the search, at any step, expands a path with $K + n_f \leq K_{\max}$ and RIC satisfying*

$$\delta_{K+n_f+B} < \min \left(\frac{\sqrt{B}}{\sqrt{K - n_c} + \sqrt{B}}, \frac{1}{2} \right). \quad (5.15)$$

Proof. First, by (5.15) and $K + n_f \leq K_{\max}$, the best path b at this certain iteration is clearly p-optimal. Combining Lemma 5.3 with the p-optimal pruning assumption, a single p-optimal path guarantees existence of p-optimal paths until the termination of the search. In addition, since each expansion of a p-optimal path introduces at least one correct index, iterative expansion of p-optimal paths leads to a superset of \mathcal{T} with maximum K_{\max} elements. The orthogonal projection of \mathbf{y} onto such a set gives the correct solution when (5.15) holds.

On the other hand, as $\varepsilon = 0$, termination of the search requires that the residue vanishes. Since $K_{\max} \leq M - K$ and Φ is full rank, the residue may vanish if and only if \mathcal{T} is a subset of the support estimate¹. Hence, the search terminates either when the recovery is successful with $\|\mathbf{r}^b\|_2 = 0$, or a complete path which is not p-optimal becomes the best path, where $\|\mathbf{r}^b\|_2 > 0$. By the p-optimal cost condition, the latter cannot happen when the tree contains p-optimal paths. Once they exist, the p-optimal paths must be chosen for expansion. Doing this iteratively, the search identifies a superset of \mathcal{T} , which yields the correct solution. \square

Since (5.15) and $K + n_f \leq K_{\max}$ lead to the p-optimality of the selected path, Theorem 5.3 may also be alternatively stated as an exact recovery guarantee based on the expansion of a p-optimal path at some intermediate iteration. However, for the sake of completeness of Theorem 5.3, we have chosen to state the conditions explicitly. In addition, the condition $\varepsilon = 0$ translates into a very small ε in practice to account for the numerical computation errors. However, to assure theoretical correctness, we state this condition as $\varepsilon = 0$ in Theorem 5.3.

As Theorem 5.3 depends on the existence of a p-optimal path, it does not directly guarantee exact recovery of all K -sparse signals. We unfortunately cannot provide theoretical guarantees for the existence of intermediate (i.e., neither complete nor empty) p-optimal paths. However, we can assure them in a special case: Observe that (5.15) and (5.12) are equal when $n_c = n_f = 0$. This states that the empty set is p-optimal when (5.12) is satisfied. Consequently, (5.12) guarantees the existence of at least one p-optimal path in the search tree for any K -sparse signal. Therefore, (5.12), together with the other assumptions in Theorem 5.3, implies guarantees for exact recovery of all

¹Due to Φ being full rank, linearly dependent subsets of Φ should contain at least $M + 1$ columns. Hence, any other solution of the recovery problem contains at least $M - K + 1$ nonzero entries.

K -sparse signals from noise-free measurements via A*OMP_e. Moreover, Theorem 5.2 can be obtained as a special case of Theorem 5.3 where $n_c = n_f = 0$.

5.2.5 Theoretical Comparison of the Two Different Termination Criteria

Though Theorem 5.2 is a special case of Theorem 5.3, we have not yet clarified if it is possible to satisfy Theorem 5.3 despite failure of Theorem 5.2. In other words, we question whether the search may find a p-optimal path even when (5.12) fails. We address this issue in the following theorem.

Theorem 5.4. *Assume $K \geq (3 + 2\sqrt{B})^2$. If $1 \leq n_f \leq \lceil K/2 \rceil - B$ and n_c satisfies*

$$K > n_c \geq \frac{8K + 4\sqrt{BK} - 4B}{9} \quad (5.16)$$

at some intermediate iteration, (5.15) becomes less restrictive than (5.12). Hence, it is possible to satisfy Theorem 5.3 though Theorem 5.2 is violated.

Proof. Assume that

$$\delta_{K+n_f+B} \geq \frac{3\sqrt{B}}{\sqrt{K} + \sqrt{B}}. \quad (5.17)$$

Since $n_f \leq \lceil K/2 \rceil - B$, we can write $3\lceil K/2 \rceil \geq K + n_f + B$. Following the monotonicity of RIC, we obtain

$$\delta_{3\lceil K/2 \rceil} \geq \frac{3\sqrt{B}}{\sqrt{K} + \sqrt{B}}. \quad (5.18)$$

Then, by Lemma 5.1

$$\delta_{K+B} > \frac{\sqrt{B}}{\sqrt{K} + \sqrt{B}},$$

which clearly contradicts (5.12).

On the other hand, Lemma 5.2 yields

$$\frac{3\sqrt{B}}{\sqrt{K} + \sqrt{B}} \leq \frac{\sqrt{B}}{\sqrt{K - n_c} + \sqrt{B}}$$

for n_c satisfying (5.16) and $K \geq (3 + 2\sqrt{B})^2$. That is, there exists some range of δ_{K+n_f+B} such that

$$\frac{3\sqrt{B}}{\sqrt{K} + \sqrt{B}} \leq \delta_{K+n_f+B} \leq \frac{\sqrt{B}}{\sqrt{K - n_c} + \sqrt{B}}.$$

This range satisfies (5.15). Therefore, when the parameters K , n_f , and n_c satisfy the necessary conditions, there exists some δ_{K+n_f+B} which fulfill (5.15), though (5.12) fails for δ_{K+B} . Hence, it is possible for the search to find a p-optimal path in the intermediate iterations and satisfy Theorem 5.3 even if Theorem 5.2 is violated. \square

Theorem 5.4 clarifies that A*OMP_e can perfectly recover a range of sparse signals, for which A*OMP_K possesses no exact recovery guarantees. In addition, remember that A*OMP_e also enjoys the exact recovery guarantees of A*OMP_K. Hence, we conclude that A*OMP_e provides means for exact recovery of a wider range of sparse signals than A*OMP_K. This reveals that the residue-based termination is more beneficial for the recovery of sparse signals from noise-free observations than its sparsity-based counterpart.

Before closing this section, it is also worth to discuss the assumption $n_f \leq \lceil K/2 \rceil - B$ in Theorem 5.4. Note that the OMP_e equivalent of this condition, which is obtained as $n_f < \lceil K/2 \rceil$ by setting $B = 1$, has been discussed in Section 3.4.4. Similar to the OMP_e case, the assumptions $K \geq (3 + 2\sqrt{B})^2$ and (5.16) in Theorem 5.4 also rely on $n_f \leq \lceil K/2 \rceil - B$, which is chosen specifically to establish the bound (5.18) on $\delta_{3\lceil K/2 \rceil}$. In fact, both $K \geq (3 + 2\sqrt{B})^2$ and (5.16) actually apply at the boundary condition $n_f = \lceil K/2 \rceil - B$. This condition is sufficient to prove Theorem 5.4, however, it is not really necessary in practice. Similar to its OMP_e analogue, Theorem 3.4, we intuitively expect Theorem 5.4 to be valid for smaller K and n_c values when n_f is also small. Hence, A*OMP_e is expected to improve the recovery accuracy not only for $K \geq (3 + 2\sqrt{B})^2$, but also for smaller K values. Moreover, the lower bound on n_c would also decrease if a bound on $\delta_{K+n_f+B}/\delta_{K+B}$ could be established for smaller n_f values. This translates as a smaller number of correct indices in the support estimate will be sufficient for exact recovery if the number of incorrect indices decrease. Unfortunately, we cannot extend Theorem 5.4 for a tighter bound on n_f , as we could neither do for Theorem 3.4. However, the empirical recovery analysis in Section 5.4 indicate that A*OMP_e improves the recovery accuracy even when $K < (3 + 2\sqrt{B})^2$.

5.3 The Adaptive-Multiplicative Cost Model

As discussed in the previous chapter, A*OMP requires properly defined cost functions for the simultaneous handling of paths with different lengths. The choice for the cost

function plays a major role in the performance of the algorithm, especially in terms of the complexity-accuracy trade-off. For this purpose, a number of cost structures have already been introduced in Section 4.3.

Among the proposed structures, the multiplicative cost model, which has been defined in (4.14), relies on the expectation that each unexplored node decreases the residue by a constant rate $\alpha_{\text{Mul}} \in (0, 1)$. In order to extend this model for paths longer than K , we simply replace K with K_{max} :

$$f_{\text{Mul}}(\mathcal{T}^i) = \alpha_{\text{Mul}}^{K_{\text{max}} - l^i} \|\mathbf{r}^i\|_2.$$

This definition of the multiplicative model can now be employed with the residue-based termination criterion. Note that the residue-based termination allows for a larger α_{Mul} than the sparsity-based termination. As a result, we utilize larger α_{Mul} values in this chapter than the ones in the previous chapter. This significantly improves the termination speed of the algorithm due to the reduction in the number of the nodes explored throughout the search.

Adaptive cost structures can adapt themselves to the actual decrement in the residue. Being motivated by the empirical improvements with the adaptive cost model in Chapter 4 over the additive one, we define an adaptive extension of the multiplicative model, which is called the adaptive-multiplicative cost model, as

$$f_{\text{AMul}}(\mathcal{T}^i) = \left(\alpha_{\text{AMul}} \frac{\|\mathbf{r}_{l^i}^i\|_2}{\|\mathbf{r}_{l^i-1}^i\|_2} \right)^{K_{\text{max}} - l^i} \|\mathbf{r}_{l^i}^i\|_2, \quad (5.19)$$

where $\mathbf{r}_{l^i}^i$ denotes the residue after the first l nodes of the path i , l^i is the length of path i , and $\alpha_{\text{AMul}} \in (0, 1]$ is the cost function parameter.

The adaptive-multiplicative cost model relies on the following assumption: Each unexplored node reduces the energy of the residue by a rate proportional to the decay occurred during the last expansion of the path. This rate is modeled by the auxiliary term $\alpha_{\text{AMul}} \|\mathbf{r}_{l^i}^i\|_2 / \|\mathbf{r}_{l^i-1}^i\|_2$, while the exponent $K_{\text{max}} - l^i$ extends the auxiliary function to all unexplored nodes along path i . The motivation behind this choice can be explained as follows: As the search is expected to select the nodes in the order of descending absolute inner products with \mathbf{y} , a node is more likely to produce less reduction in $\|\mathbf{r}^i\|_2$

than its ancestors do. This may obviously be violated for some particular nodes, similar to the other proposed cost models. However, the auxiliary term of the cost function is mostly computed over a group of nodes instead of a single one. Hence, it is practically sufficient when the decay in the residue obeys this assumption over a group of nodes. Moreover, the tree usually contains multiple optimal/p-optimal paths which can lead to the correct solution if chosen. Therefore, that some particular paths violate this assumption does not actually harm the recovery performance. Hence, similar to the other proposed cost models, the adaptive-multiplicative cost model also needs to be valid only on the average, i.e., any particular sequence of nodes may violate it, however, we expect it to hold in general and lead the search to the correct solution.

As for the cost model parameter α , the adaptive structure of the adaptive-multiplicative cost model allows for a larger choice than the multiplicative model does. This reduces the auxiliary term on the average and makes the search favor longer paths. Consequently, the search explores fewer nodes and terminates faster. This speed up is demonstrated in Section 5.4.

5.4 Empirical Analyses of A*OMP_e

In this section, we illustrate the recovery performance of A*OMP_e in comparison to A*OMP_K, BP, SP, OMP_e, IHT, ISD, and SL0 in various scenarios. In the simulations, we employ the adaptive-multiplicative and multiplicative cost models which are denoted as AMul-A*OMP and Mul-A*OMP, respectively. First, we evaluate Mul-A*OMP and AMul-A*OMP in terms of the exact recovery rates, average recovery error and run times. In order to generalize the results to a wide range of M and K , we provide the empirical phase transition curves which are obtained using the procedure in [16]. Then, we investigate recovery from noisy observations. We also demonstrate a hybrid of OMP_e and A*OMP_e which accelerates the recovery significantly. Finally, we test our proposal on images to illustrate the recovery performance for more realistic cases. Note that in the rest of this chapter, we skip the subscript in OMP_e, since this is the only version of OMP employed in the following simulations.

5.4.1 Experimental Setup

Unless given explicitly, the experimental setup is as follows: We set $I = 3$, $B = 2$, and $P = 200$. ε is chosen as 10^{-6} in the noiseless case, while it is selected with respect to the noise level in the noisy scenarios. Each test is repeated over a randomly generated set of sparse samples. For each sample, Φ is drawn from the Gaussian distribution with mean zero and standard deviation $1/N$. The average normalized mean-squared-error (ANMSE) is defined as

$$\text{ANMSE} = \frac{1}{L} \sum_{i=1}^L \frac{\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2}{\|\mathbf{x}_i\|_2^2} \quad (5.20)$$

where $\hat{\mathbf{x}}_i$ is the reconstruction of the i th test vector \mathbf{x}_i , and L is the number of test samples. For the noisy examples, we specify the distortion ratio as $10 \log_{10}(\text{ANMSE})$, in order to better relate the recovery distortion to the signal-to-noise ratio (SNR).

The cost model parameter is selected as $\alpha_{\text{Mul}} = 0.8$ for Mul-A*OMP_K . As the residue-based termination criterion allows for a larger value, it is relaxed to $\alpha_{\text{Mul}} = 0.9$ for Mul-A*OMP_e . Since the AMul model also allows for larger choices than the Mul model, we choose an even larger value, $\alpha_{\text{AMul}} = 0.97$, for AMul-A*OMP_e . As a result of these increments, the A*OMP algorithm is accelerated significantly.

The nonzero entries of the test samples are selected from four different random ensembles. The nonzero entries of the Gaussian sparse signals are drawn from the standard Gaussian distribution. Nonzero elements of the uniform sparse signals are distributed uniformly in $[-1, 1]$, while those of the binary sparse signals are set to 1. The constant amplitude random sign (CARS) sparse signals have nonzero elements with unit magnitude and random sign.

The recovery simulations for A*OMP are performed using the AStarOMP software package [118]. The other algorithms are run using freely available software such as ℓ_1 -magic [108], Sparsify [119], Threshold-ISD [121], and the MATLAB implementation of SL0 [122].

5.4.2 Exact Recovery Rates and Reconstruction Error

The first set of simulations involve the exact recovery rates and ANMSE for the Gaussian, uniform, and binary sparse signals. For this case, we set $N = 256$ and $M = 100$, whereas

$K \in [10, 45]$ and $K_{\max} = 55$. Each test set consists of 500 randomly generated sparse vectors.

The recovery results for the Gaussian and uniform sparse signals are depicted in Figure 5.1 and 5.2, respectively. We observe that the A*OMP variants yield similar ANMSE values, whereas the residue-based termination significantly improves the exact recovery rates. It is also evident that A*OMP_e is better than A*OMP_K at identifying smaller magnitude coefficients, which hardly change the ANMSE, however increase the exact recovery rates. In comparison to the other algorithms, the A*OMP variants perform significantly better recovery. At the best, A*OMP_e provides exact recovery until $K = 40$ and $K = 35$ for the Gaussian and uniform ensembles, respectively. These breakpoints are clearly far beyond the other algorithms.

To reveal the benefits of the adaptive-multiplicative cost model over the multiplicative one, we plot the average run time per vector in Figure 5.3. The figure is limited to the OMP and A*OMP algorithms, which are tested using the AStarOMP software. The other algorithms are ignored as they run in the MATLAB environment, because of which their run times are not comparable. We observe that both the residue-based termination and the adaptive-multiplicative cost model significantly accelerate A*OMP due to the relaxation of α to larger values. Since AMul-A*OMP_e can afford the largest α , it is significantly faster than the other A*OMP variants. These findings confirm the claim in Section 5.3 that increasing α reduces the number of explored nodes, and hence accelerates A*OMP².

In Figure 5.4, we illustrate the recovery performance for the binary sparse signals, which are known as the most challenging case for greedy algorithms [16, 17]. As expected, ℓ_1 norm minimization is the best performer in this simulation. As above, A*OMP recovery is significantly improved with the utilization of the residue-based termination criterion. Although A*OMP_K performs worse than SP, we observe that A*OMP_e outperforms all the greedy alternatives involved in the tests.

²Note that the complexity-accuracy trade-off, discussed in the previous chapter, is also valid for the adaptive-multiplicative cost model. That is, decreasing α_{AMul} would further improve the recovery, but also increase the run time of the search.

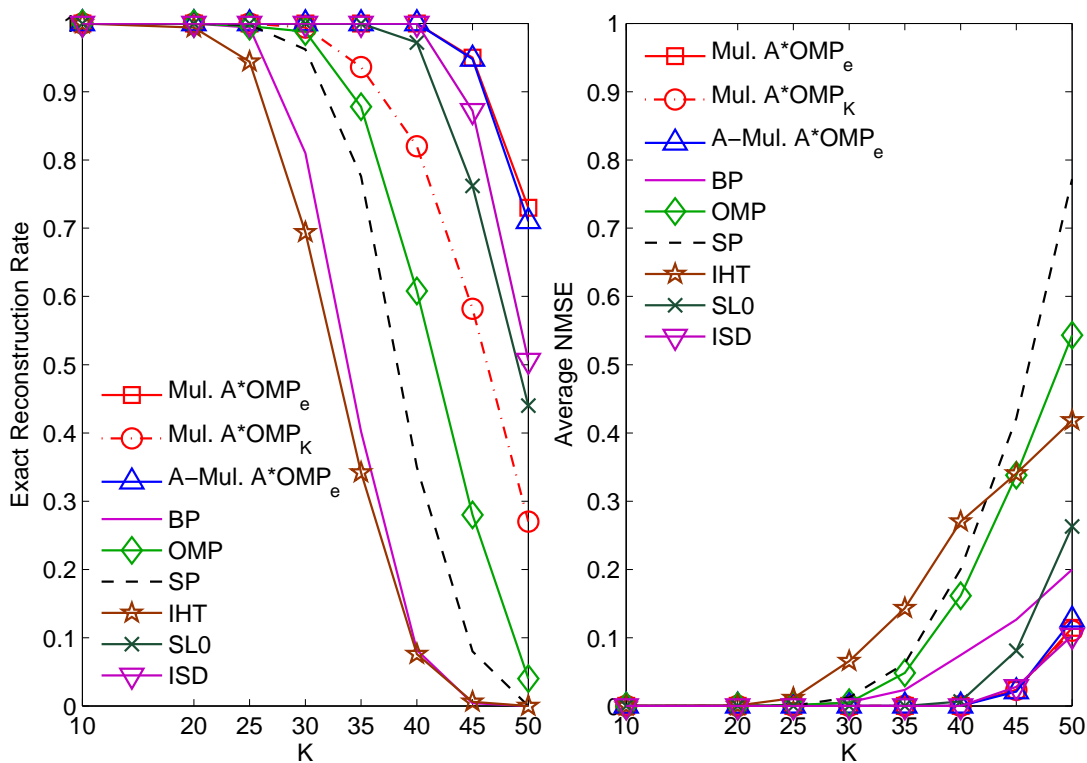


FIGURE 5.1: Recovery results over sparsity for the Gaussian sparse signals.

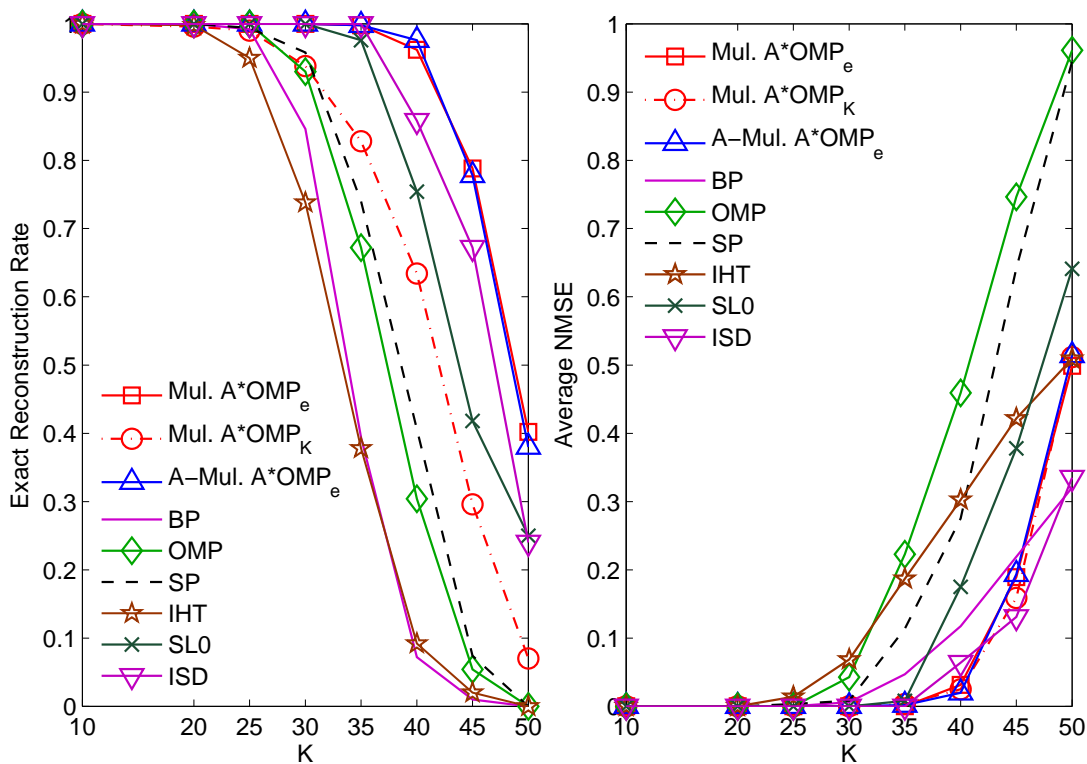


FIGURE 5.2: Recovery results over sparsity for the uniform sparse signals.

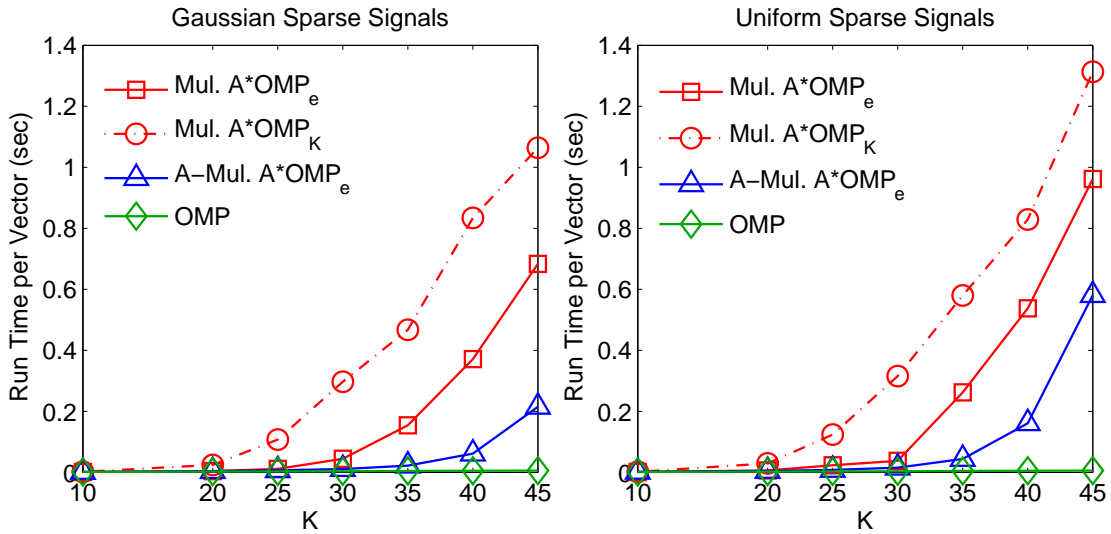


FIGURE 5.3: Average run time of A*OMP per vector with the AStarOMP software.

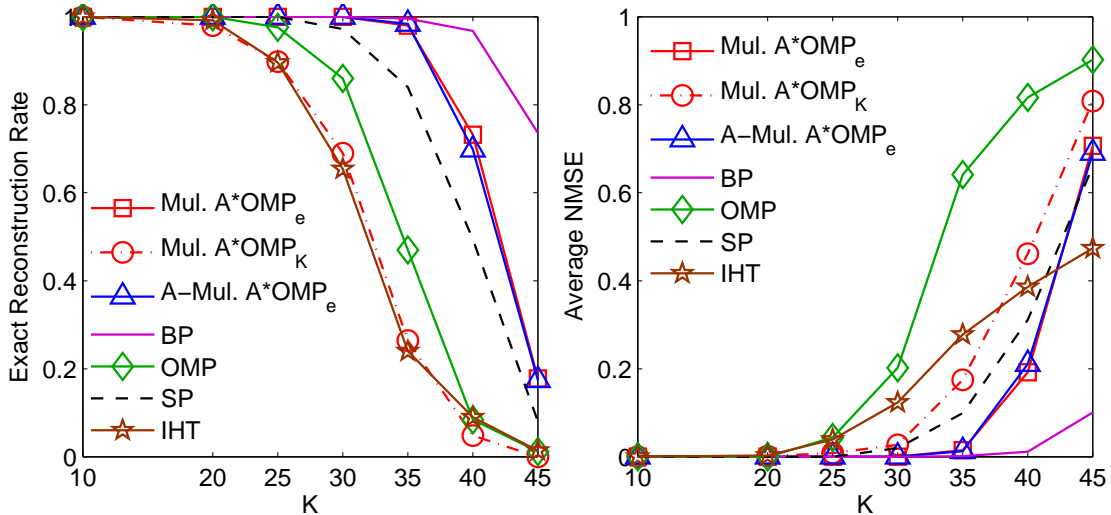


FIGURE 5.4: Recovery results over sparsity for the binary sparse signals.

5.4.3 Phase Transitions

Empirical phase transitions provide important means for practical evaluation of sparse signal recovery algorithms, since they reveal the recovery performance over the feasible range of M and K . Let us first define normalized measures for the observation length and sparsity level as $\lambda = \frac{M}{N}$ and $\rho = \frac{K}{M}$. As discussed in [16], the phase transition curve is mostly a function of λ . That is, it remains unaltered when N changes. Hence, phase transition curves allow for general characterization of the recovery performance.

In order to obtain the phase transition curves, we fix $N = 250$, and alter M and K

to sample the $\{\lambda, \rho\}$ space for $\lambda \in [0.1, 0.9]$ and $\rho \in (0, 1]^3$. For each $\{\lambda, \rho\}$ tuple, we randomly generate 200 sparse instances and run AMul-A*OMP_e, OMP, BP, SP, ISD and SL0 algorithms for the recovery. Setting the exact recovery criterion as $\|\mathbf{x} - \hat{\mathbf{x}}\|_2 \leq 10^{-2}\|\mathbf{x}\|_2$,⁴ where $\hat{\mathbf{x}}$ is the recovery of \mathbf{x} , we count the number of exactly recovered samples in each test. The phase transitions are then obtained using the methodology described in [16]. That is, for each λ , we employ a generalized linear model with logistic link to describe the exact recovery curve over ρ , and then find the ρ value which yields 50% exact recovery probability. Combining the results over the whole λ range, we end up with the empirical phase transition curve⁵. This procedure is repeated for the Gaussian, uniform, and CARS sparse signals to reveal the effect of nonzero element distribution⁶.

First, it is worth to discuss the optimal choice of K_{\max} . Consider the normalized measure $\rho_{\max} = K_{\max}/M$. This definition helps us to identify the optimal ρ_{\max} values over the whole λ range. Due to the robustness of the phase transitions with respect to N , we can then select $K_{\max} = \rho_{\max}M$ using the optimal ρ_{\max} value for a particular λ . In order to find an optimal formulation for ρ_{\max} as a function of λ , we have run a number of recovery simulations, where we have observed that the phase transition of AMul-A*OMP_e is quite robust to the choice of K_{\max} , with a perturbation up to %3 in the phase transition curve. Hence, the recovery accuracy is mostly independent of K_{\max} ⁷. Yet, based on our experience from these experiments, we propose to choose $\rho_{\max} = 0.5 + 0.5\lambda$ taking into account both the accuracy and the complexity of the search⁸. The phase transition curves below are obtained with this setting.

The resultant empirical phase transition curves are depicted in Figure 5.5. These indicate that AMul-A*OMP_e yields better phase transitions than the other algorithms for the Gaussian and uniform sparse signals. Contrarily, for the CARS case, BP and ISD perform better than the other algorithms involved, while AMul-A*OMP_e is the third best. As for the effect of the coefficient distribution on the recovery performance,

³The λ axis is sampled with a resolution of 0.1, while the corresponding ρ values are chosen densely around the phase transition region for a specific λ in order to obtain a fine modelling of the transition region.

⁴This exact recovery condition is the same as the one in [16]. This choice has been made for the compatibility of the computed phase transitions with [16].

⁵Note that, due to narrow phase transition regions [16], the region below the phase transition curve promises exact recovery with high probability for the corresponding recovery method.

⁶This procedure is the same as the computation of phase transitions in Chapter 3.

⁷Obviously, K_{\max} should be chosen large enough, i.e., larger than the underlying sparsity level.

⁸Observe that with this choice, K_{\max} is assured to be larger than the phase transition region, i.e., larger than the maximum sparsity level which AMul-A*OMP_e can exactly recover for all λ values and involved distributions.

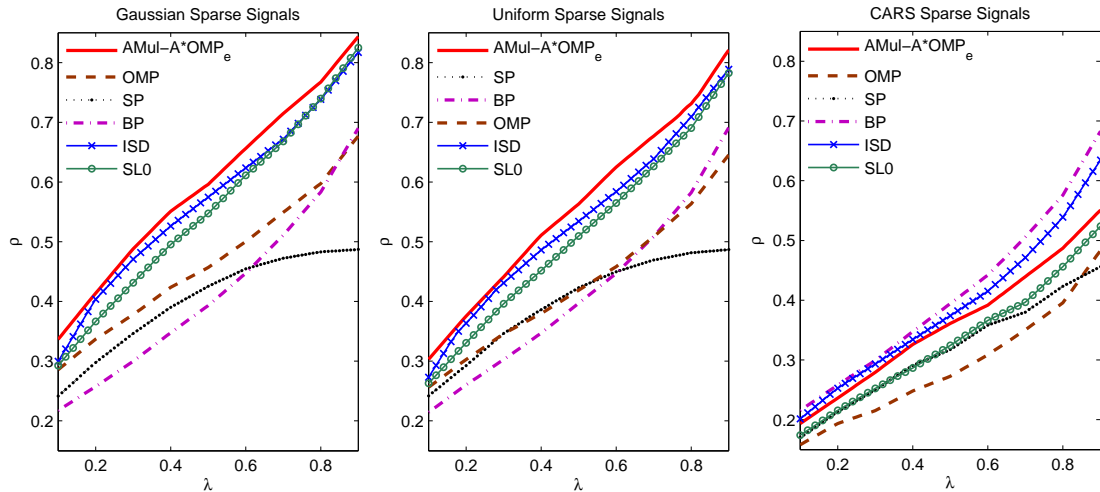


FIGURE 5.5: Phase transitions of $AMul-A^*OMP_e$, BP, SP, OMP, ISD, and SL0 for the Gaussian, uniform, and CARS sparse signals.

we observe that BP is robust, whereas the phase transition curves for $AMul-A^*OMP_e$ and OMP exhibit the highest variation among different nonzero element distributions. When the nonzero values cover a wide range, such as for the Gaussian distribution, the performances of A^*OMP_e and OMP are boosted. In contrast, nonzero values of equal magnitude expectedly turn out to be the most challenging case for these two. These observations indicate that OMP-type algorithms are more effective when the nonzero elements span a wide range of magnitudes. Remember that we have discussed this issue in Section 3.5.1 with a simple analytical reasoning for OMP, which states that the spread of the elements in the correlation vector between the observation and the dictionary atoms is narrower for CARS type signals. This increases the error rate of OMP-type algorithms, including A^*OMP , for such signals.

Comparison of the phase transition curves with the theoretical guarantees of A^*OMP leads to an important conclusion. According to Section 5.2, the exact recovery conditions of A^*OMP require an RIC which is inversely proportional to \sqrt{K} . In the literature, such conditions are acknowledged to necessitate $M \approx O(K^2 \log(N))$ measurements for exact recovery, i.e., $M \propto K^2$, [4, 123–125]. On the other hand, the empirical phase transitions in Figure 5.5 imply that the number of necessary measurements for exact recovery exhibit a much smaller slope with increasing K . In fact, we may deduce from these curves that approximately $M \propto K^{\frac{2}{3}}$ measurements are enough to exactly recover \mathbf{x} in practice⁹.

⁹Though we have not made extra efforts to find the best match with the empirical phase transitions, the curves obtained by setting $M \approx cK^{\frac{2}{3}}$, where c is a constant depending on N , turns out to be close

This suggests that the presented theoretical guarantees are loose especially for large K values. Consequently, the empirical exact recovery rates of A*OMP promise being, at least asymptotically, better than the rates which the presented theoretical analysis may guarantee.

5.4.4 Recovery from Noisy Observations

In order to evaluate the empirical recovery performance of A*OMP_e in noisy situations, we alter the observation model as

$$\mathbf{y} = \Phi\mathbf{x} + \mathbf{n} \quad (5.21)$$

where \mathbf{n} represents some additive observation noise. We model \mathbf{n} as white Gaussian noise, and alter SNR, which is defined as $20 \log \frac{\|\mathbf{y}\|_2}{\|\mathbf{n}\|_2}$, for the purpose of obtaining a general performance measure. Figure 5.6 illustrates the recovery performance over SNR where $K = 30$ and $K = 25$ for the Gaussian and uniform sparse signals, respectively. As mentioned before, the termination parameter ε is adjusted proportional to the true SNR level in these simulations. The regularization parameter of BP is also adjusted in a similar fashion. We observe that A*OMP is superior to the other algorithms except for 5dB SNR where BP is slightly better. In addition, A*OMP_e improves the recovery accuracy slightly over A*OMP_K for low SNR values. Figure 5.7 depicts the average A*OMP run times in this scenario. Similar to the previous examples, AMul-A*OMP_e is significantly faster than the other A*OMP variants.

5.4.5 A Hybrid Approach for Faster Practical Recovery

Based on the results above, it is possible to speed up the recovery from noise-free observations using a hybrid of OMP and AMul-A*OMP_e. First, OMP provides exact recovery up to some mid-sparsity range. Moreover, there are regions where AMul-A*OMP_e provides exact recovery while OMP also yields quite high recovery rates. In these regions, we can facilitate faster recovery without sacrificing the accuracy by a simple two stage hybrid scheme: We run OMP first, and then AMul-A*OMP_e only if OMP fails. This

is enough to the curves in Figure 5.5. Note that we do not intend to find a precise relation between M and K , and the exponent $\frac{2}{3}$ is only a rough indicator of the relation between the two quantities.

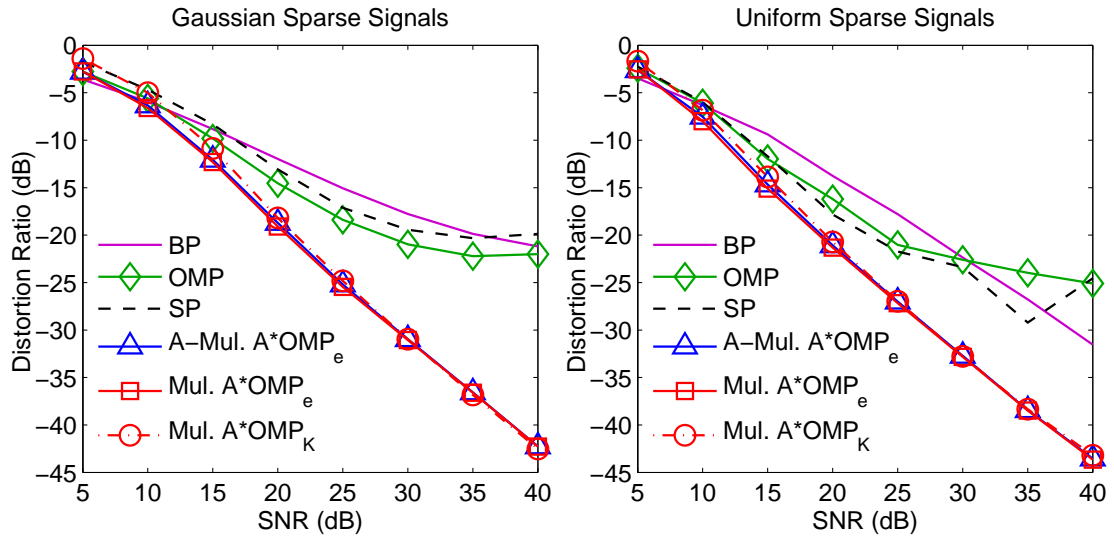


FIGURE 5.6: Average recovery distortion over SNR in the noisy recovery scenario. K is selected as 30 and 25 for the Gaussian and uniform sparse signals, respectively.

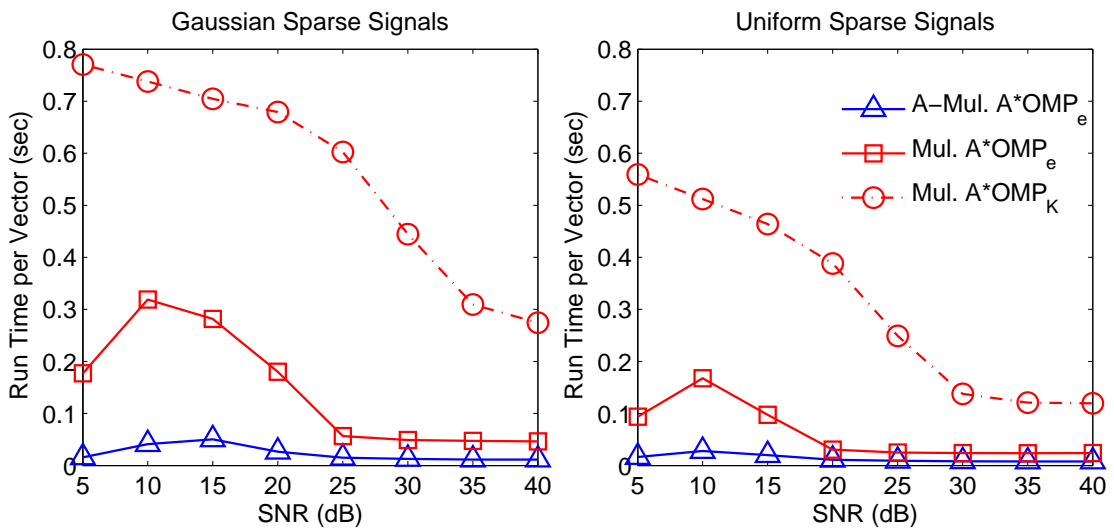


FIGURE 5.7: Average run time per vector of A*OMP in the noisy recovery scenario using the AStarOMP software. K is selected as 30 and 25 for the Gaussian and uniform sparse signals, respectively.

strategy reduces the number of AMul-A*OMP_e runs and accelerates the algorithm, if we can properly identify OMP failures. This is indeed not difficult: Assuming that $K+K_{\max}$ -RIP holds, OMP is successful when the residue vanishes. Consequently, the hybrid approach runs AMul-A*OMP_e only when $\|\mathbf{r}\|_2 > \varepsilon$ after OMP. Moreover, we use the order by which OMP chooses the vectors in consequent iterations in order to set the priorities of trie nodes in the AStarOMP software. That is, a vector OMP chooses first gets higher priority, and is placed at the lower levels of the search trie. This reduces not

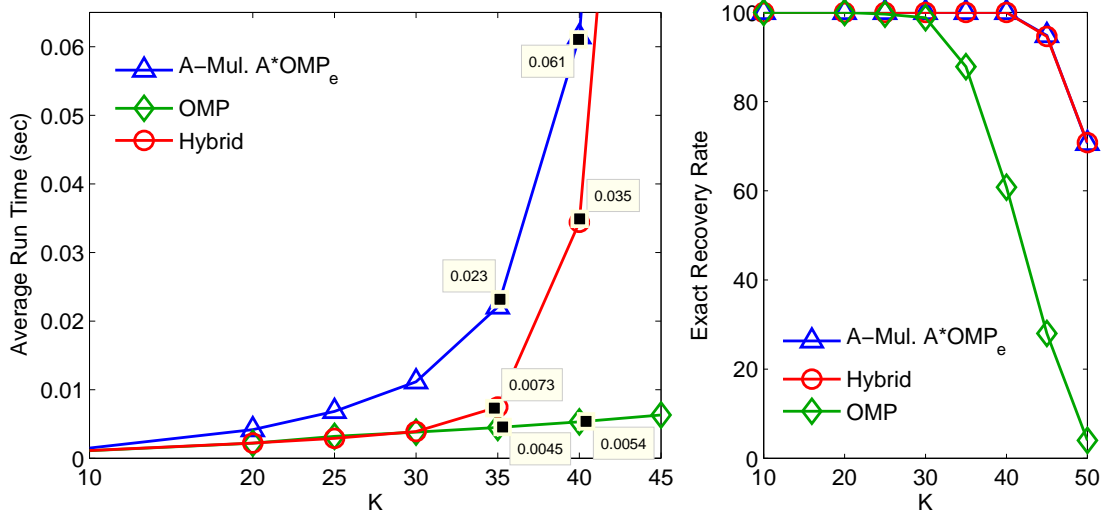


FIGURE 5.8: Performance of the hybrid scheme for the Gaussian sparse vectors.

only the trie size but also the cost of path additions.

The recovery results for the hybrid approach are depicted in Figure 5.8 in comparison to the OMP and AMul- A^*OMP_e algorithms. We observe that AMul- A^*OMP_e and the hybrid approach yield identical exact recovery rates, while the latter is significantly faster. This acceleration is proportional to the exact recovery rate of OMP. That is, the hybrid approach is faster when the exact recovery rate of OMP is higher. These results show that this hybrid approach is indeed able to detect the OMP failures, and run AMul- A^*OMP_e only for those instances.

5.4.6 Image Recovery Examples

As for a more realistic case, we demonstrate recovery of two 512×512 images below. The recovery of these images is performed in blocks of size 8×8 as in the previous chapter. The aim of this block processing is to break the recovery problem into a number of smaller and simpler subproblems. To exploit compressibility of the images, we perform the reconstruction in the 2D Haar Wavelet basis Ψ . Note that, in this case, the reconstruction dictionary is not the observation matrix Φ itself, but $\Phi\Psi$. That is, \mathbf{x} denoting the sparse wavelet coefficient vector of interest, the image itself is obtained as $\Psi\mathbf{x}$ after the recovery of \mathbf{x} from the observation $\mathbf{y} = \Phi\Psi\mathbf{x}$. In the first example, the image “bridge” is preprocessed prior to the recovery such that each 8×8 block is K -sparse

in the 2D Haar Wavelet basis, i.e., for each block only the K largest magnitude wavelet coefficients are kept. As for the second example, the image “butterfly” is recovered without any such preprocessing. From each block of the images, $M = 32$ observations are taken, where the entries of Φ are randomly drawn from the Gaussian distribution with mean zero and standard deviation $1/N$. The search parameters of AMul-A*OMP are selected as $I = 3$ and $P = 200$, while B varies in $\{2, 3\}$. Due to the different settings of the two problems, AMul-A*OMP is run with different K_{\max} and α_{AMul} values, as discussed below.

5.4.6.1 Demonstration on a Sparse Image

The first example is the recovery of the image “bridge”, which, as indicated above, is first preprocessed such that each 8×8 block is K -sparse in the 2D Haar Wavelet basis, where $K = 12$. For this sparse case, K_{\max} is selected as 20 while α_{AMul} is reduced to 0.85 in order to compensate for the decrement in K_{\max} , which decreases the auxiliary term in (5.19).

Recovery results for the preprocessed sparse image “bridge” are shown in Figure 5.9. The upper left panel of Figure 5.9 is the preprocessed image “bridge” itself, while BP and AMul-A*OMP_e recoveries are depicted in the other panels. We observe that BP provides a peak signal-to-noise ratio (PSNR) value of 29.9 dB, while AMul-A*OMP_e improves the recovery PSNR to 42.1 dB for $B = 2$ and to 49.3 dB for $B = 3$. A careful investigation of the recovered images yields that AMul-A*OMP_e improves the recovery especially at detailed regions and boundaries.

5.4.6.2 Demonstration on a Compressible Image

The second image recovery example in this section deals with a harder problem since the image “butterfly” is not sparse. In this case, we exploit the compressibility of this image in the transform domain, and aim at recovering the best K -sparse approximation to the image in this domain. Due to the different structure of the recovery problem than the one above, AMul-A*OMP should better be run with a different setting in this case. We set $K_{\max} = 12$, that is, we are interesting in recovering the largest 12 wavelet coefficients for each block. On the other hand, after working with a number of

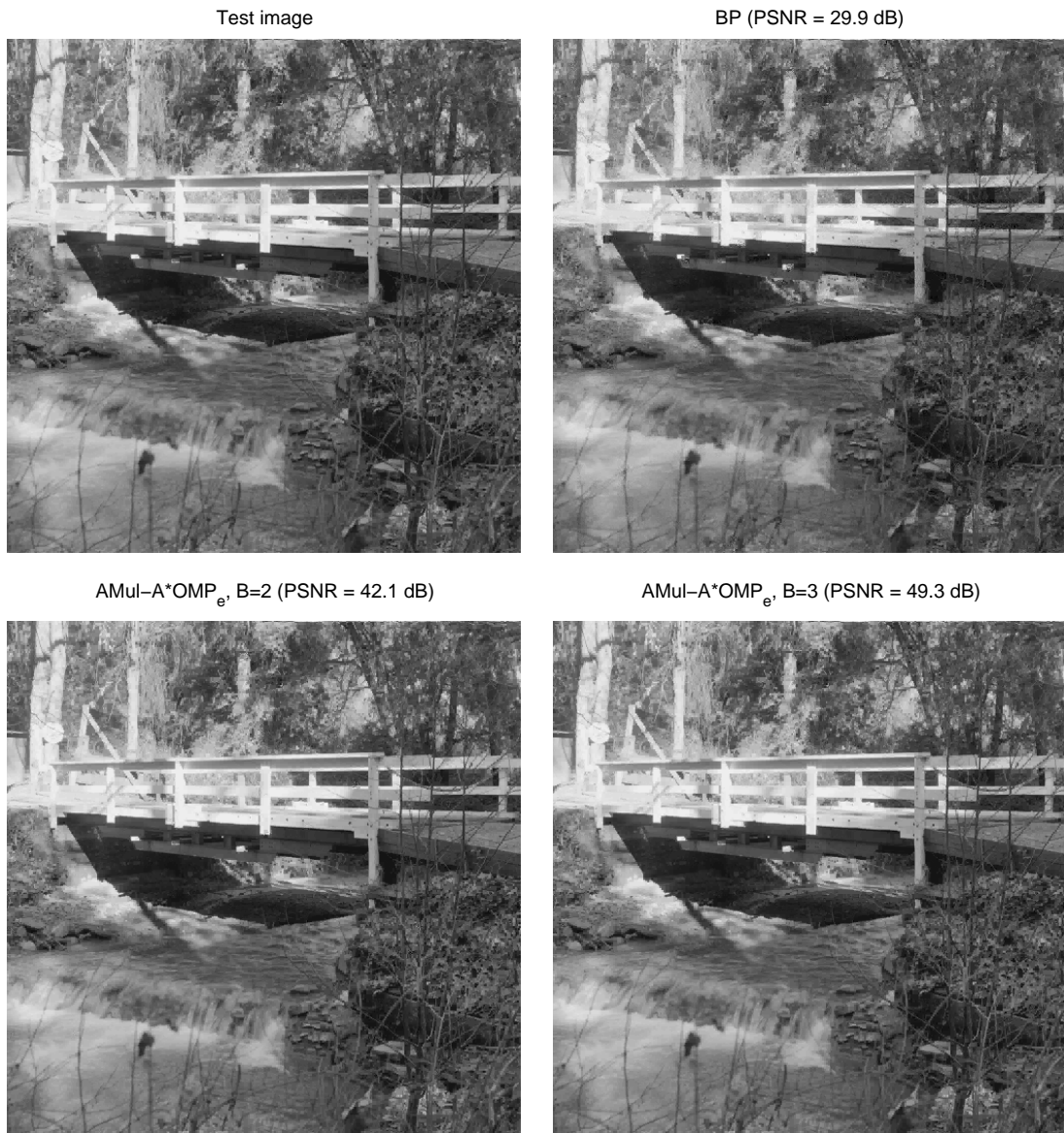


FIGURE 5.9: Recovery of the image “bridge” using BP and $AMul-A^*OMP_e$.

compressible images, we have observed that increasing the auxiliary term reduces the quality of the approximation. Therefore, α_{AMul} is selected as 0.97^{10} . For this case, we demonstrate $AMul-A^*OMP$ only with $B = 3^{11}$.

Figure 5.10 shows the recovery results for the image “butterfly”, which is depicted on the left panel of the figure. The middle and right panels of Figure 5.10 are the BP and $AMul-A^*OMP_e$ recoveries of the image, respectively. In this case, both BP and

¹⁰This set of parameters was observed to yield optimal $AMul-A^*OMP$ recovery accuracy over a set of other images as well.

¹¹Note that modifying B and I does not significantly alter the recovery performance.

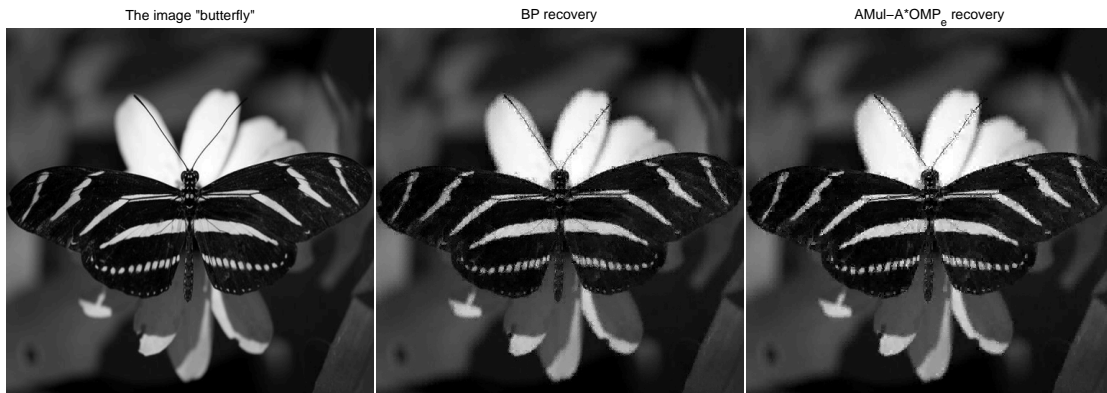


FIGURE 5.10: Recovery of the image “butterfly” using BP and $AMul-A^*OMP_e$. BP and $AMul-A^*OMP_e$ yield 27.8 and 27.5 dB PSNR, respectively.

$AMul-A^*OMP_e$ recoveries end up with very close PSNR values¹². In particular, BP and $AMul-A^*OMP_e$ yield 27.8 and 27.5 dB PSNR, respectively¹³. We observe that the two approximations are also close in terms of the perceptual quality, while each algorithm performs better than the other on some particular regions of the image.

5.5 Summary

In this chapter, our fundamental goal has been the theoretical analysis of the A^*OMP algorithm. For this purpose, we have first derived an RIP condition for the exact recovery of any K -sparse signal from noise-free measurements with A^*OMP_K . Next, we have extended this result to A^*OMP_e , which utilizes the residue-based termination criterion instead of the sparsity-based one. In particular, we have stated that a K -sparse signal can be recovered with A^*OMP_e if the search selects a p -optimal path for expansion where the notion of p -optimality is based on the number of correct and incorrect indices in the support estimate. Interestingly, the exact recovery guarantees of A^*OMP_K represent a special case of this condition. This has led to the conclusion that A^*OMP_e enjoys at least the same general exact recovery guarantees as A^*OMP_K . Further comparison of the two has also revealed that the recovery condition of A^*OMP_e represents a less restrictive requirement than that of A^*OMP_K . This result encourages utilising the

¹²We have observed similar behavior for a set of other images as well.

¹³Note that the PSNR value of the best 12-sparse representation of the image “butterfly”, i.e., the maximum PSNR value that can be obtained with any algorithm searching for the best 12-sparse representation, is 32.3 dB.

residue-based termination criterion instead of the sparsity-based one for recovery from noise-free observations.

In addition, we have also introduced the novel adaptive-multiplicative cost model, which extends the multiplicative model in an adaptive manner. This model allows for a larger choice of the auxiliary model parameter, which reduces the number of nodes explored throughout the search. As a result of this reduction, the AMul cost model accelerates the search without sacrificing the recovery accuracy.

Lastly, we have demonstrated the empirical recovery performance of AMul- A^*OMP_e by extensive simulations, including sparse signals with different characteristics in addition to noisy and noise-free observations. The results of these experiments support that AMul- A^*OMP_e possesses better recovery capabilities and shorter execution times than A^*OMP_K . A^*OMP variants perform better recovery than BP, SP, IHT, OMP, SL0, and ISD for the uniform and Gaussian sparse signals. With constant magnitude nonzero elements, such as for the binary and CARS sparse signals, AMul- A^*OMP_e still provides better recovery accuracy than the greedy alternatives involved, while BP yields the most accurate recovery among the candidates for such signals. Among the experiments, we have also presented a hybrid approach, which first applies OMP, and then AMul- A^*OMP_e only if OMP failure is detected. The experiments have shown that this hybrid approach accelerates the recovery without sacrificing the accuracy. Finally, we have demonstrated AMul- A^*OMP_e on images, where we have observed that both algorithms perform very close when the underlying image is compressible, whereas AMul- A^*OMP_e promises significant improvements in the recovery accuracy over BP for sparse images.

Chapter 6

Forward-Backward Pursuit: A Novel Two Stage Algorithm for Sparse Signal Recovery

6.1 Introduction

In this chapter, we introduce a novel two stage greedy algorithm, which is called forward-backward pursuit (FBP), for sparse signal recovery. As the name indicates, FBP employs forward selection and backward removal steps which iteratively expand and shrink the support estimate of the underlying sparse signal. With this structure, FBP falls into the general category of two stage thresholding (TST) algorithms [16], which present a framework for methods based on the iterative utilization of two stages with thresholding.

Though FBP can be seen close to the other TST-type algorithms such as subspace pursuit (SP) [17] and compressive sampling matching pursuit (CoSaMP) [18], it involves a fundamental difference from these: In contrast to SP and CoSaMP, the forward and backward step sizes of FBP are not the same. Utilization of a larger forward step than the backward one allows for the iterative expansion of the support estimate, which, to the best of our knowledge, appears for the first time in the context of TST algorithms.

Due to the utilization of different forward and backward step sizes, FBP possesses some important advantages over both the other TST algorithms and forward greedy schemes

such as the orthogonal matching pursuit (OMP) algorithm [6]. As for the TST schemes, CoSaMP and SP require *a priori* estimate of the sparsity level, K , which is most of the time not available in practice. On the contrary, FBP allows for the expansion of the support estimate from scratch until the residual error of the observation is either small enough with respect to the noise level or vanishes for the noiseless case. Hence, FBP does not require K *a priori* in contrast to SP and CoSaMP. Additionally, the backward step of FBP can remove some possibly misidentified atoms from the support estimate, which is an advantage over the forward greedy algorithms.

Another forward-backward greedy approach, namely the FoBa algorithm, has been investigated in [84] for the sparse learning problem. Though both FoBa and FBP consist of iterative forward and backward steps, they have some fundamental differences: First, the FoBa algorithm employs strict forward and backward step sizes of one. On the contrary, the forward step size of FBP is greater than 1, while the backward step size, which should be smaller than the forward step size, might also be chosen greater than one. By increasing the difference between the forward and backward step sizes, FBP terminates in less iterations. Second, FoBa takes a number of forward steps before it takes a backward step depending on an adaptive decision criterion, while FBP employs no criterion for the backward step¹, which immediately follows each forward step. Finally, FoBa has been applied for the sparse learning problem, whereas we propose and evaluate FBP for sparse signal recovery from compressed measurements.

The findings of this chapter have been presented at the 2012 European Signal Processing Conference (EUSIPCO-2012) [126] in a partial form.

6.1.1 Outline

This chapter is organized as follows: The FBP algorithm is introduced in Section 6.2. Section 6.3 discusses the relations of the FBP algorithm to the TST-type algorithms and the forward greedy methods. Section 6.4 is devoted to the analyses of the empirical recovery performance of FBP. The recovery abilities of FBP are demonstrated on sparse signals with different nonzero coefficient distributions in noiseless and noisy observation scenarios in addition to images in comparison to BP, SP, and OMP. These results show that FBP can perform better recovery than SP and BP in most scenarios. This indicates

¹Note that this is not trivial as in the FoBa case when the backward step size is greater than one.

that SP, which is announced as the globally optimum TST scheme in [16], is not necessarily optimal for all nonzero element distributions². Finally, we conclude this chapter with a brief summary of our findings in Section 6.5.

6.2 The Forward-Backward Pursuit Algorithm

Forward-backward pursuit is an iterative algorithm, which employs two stages at each iteration. The first stage of FBP, the forward step, is meant for expanding the support estimate by α indices, where $\alpha > 1$. We call α the forward step size. These α indices are chosen as the maximum magnitude elements of the correlation vector between the residue and the dictionary columns. The second stage of FBP is the backward step which prunes the support estimate by removing β indices where $\beta < \alpha$. Analogous to α , β is referred to as the backward step size. In order to decide which indices will be removed, the projection coefficients of the atoms in the support estimate are computed by orthogonal projection of the observation vector onto the subspace represented by the support estimate. Then, the indices corresponding to the smallest magnitude coefficients are pruned. The orthogonality of the residue to the subspace defined by the pruned support estimate is ensured by a second projection of the residue onto this subspace. These forward and backward steps are iterated until the energy of the residue either vanishes or is less than a threshold, which is proportional to the energy of the observed vector.

An important issue for the performance of FBP is the choice of the forward and backward step sizes. The forward step size α should be chosen larger than 1. It is possible to choose α as large as problem-specific constraints allow, while a reasonable approach would obviously be selecting it small in comparison to the observation length M in order to avoid linearly dependent subsets in the expanded support estimate after the forward step. As for the backward step, by the definition of FBP, β should be smaller than α . This choice is necessary for the support estimate to be enlarged by $\alpha - \beta$ indices at each iteration. As discussed below in relation to the other TST schemes, this leads to an advantageous mechanism in which no *a priori* estimate of the sparsity level K is

²In [16], the optimality is discussed in terms of the worst case performance of the greedy algorithms, which corresponds to the recovery of sparse signals with constant magnitude nonzero elements. In that case, the modified SP turns out to be the best TST scheme. However, our results indicate that this is not necessarily true for other distributions.

required. As for finding an empirically optimal rule for choosing α and β , we present phase transition curves of FBP with various α and β choices among the simulation results below. It turns out that choosing $\alpha \in [0.2M, 0.3M]$ and $\beta = \alpha - 1$ leads to the optimal recovery performance in practice, whereas the algorithm is also quite robust to other choices of α and β as well.

6.2.1 Notation

Let us clarify the notation used in the rest of this chapter: As before, we denote the K -sparse signal of interest by $\mathbf{x} \in \mathbb{R}^N$. M represents the number of observations. The observation matrix is defined as $\Phi = [\phi_1 \ \phi_2 \ \dots \ \phi_N]$, where $\phi_i \in \mathbb{R}^M$ denotes the i th column of Φ . The observation vector is referred to as $\mathbf{y} \in \mathbb{R}^M$, where $\mathbf{y} = \Phi\mathbf{x}$. \mathcal{T}^i and \mathbf{r}^i denote the support estimate and the residue after the i th FBP iteration, respectively. $\tilde{\mathcal{T}}^i$ is the expanded support estimate after the forward step of the i th iteration. Finally, $\Phi_{\mathcal{J}}$ denotes the matrix of the columns of Φ indexed by \mathcal{J} , and $\mathbf{x}_{\mathcal{J}}$ is the vector of the elements of \mathbf{x} indexed by \mathcal{J} .

6.2.2 The Proposed Method

The FBP algorithm can now be outlined as follows: We initialize the support estimate as $\mathcal{T}^0 = \emptyset$, and the residue as $\mathbf{r}^0 = \mathbf{y}$. At iteration k , first the forward step expands \mathcal{T}^{k-1} by indices of the α largest magnitude elements in $\Phi^*\mathbf{r}^{k-1}$. This builds up the expanded support set $\tilde{\mathcal{T}}^k$. Then the projection coefficients are computed by the orthogonal projection of \mathbf{y} onto $\Phi_{\tilde{\mathcal{T}}^k}$. The backward step prunes $\tilde{\mathcal{T}}^k$ by removing the β indices with the smallest magnitude projection coefficients. This produces the final support estimate \mathcal{T}^k of the k th iteration. Finally, the projection coefficients \mathbf{w} for the vectors in $\Phi_{\mathcal{T}^k}$ are computed via the orthogonal projection of \mathbf{y} onto $\Phi_{\mathcal{T}^k}$, and the residue is updated as $\mathbf{r}^k = \mathbf{y} - \Phi_{\mathcal{T}^k}\mathbf{w}$. The iterations are carried on until $\|\mathbf{r}^k\|_2 < \varepsilon\|\mathbf{y}\|_2$. After termination of the algorithm at the l th iteration, \mathcal{T}^l gives the support estimate for \mathbf{x} , and the corresponding nonzero elements are set equal to the projection coefficients of \mathbf{y} onto \mathcal{T}^l . The pseudo-code of FBP is given in Algorithm 6.1.

As for the termination parameter ε , we choose it very small in practice (on the order of 10^{-6} for the experiments in this chapter) when the observations are noise-free. For noisy

Algorithm 6.1 FORWARD-BACKWARD PURSUIT

input: Φ, \mathbf{y}
define: $\alpha, \beta, K_{\max}, \varepsilon$
initialize: $\mathcal{T}^0 = \emptyset, \mathbf{r}^0 = \mathbf{y}, k = 0$

while true **do**
 $k = k + 1$
forward step:

$$\mathcal{T}_f = \arg \max_{\mathcal{J}:|\mathcal{J}|=\alpha} \|\Phi_{\mathcal{J}}^* \mathbf{r}^{k-1}\|_1$$

$$\tilde{\mathcal{T}}^k = \mathcal{T}^{k-1} \cup \mathcal{T}_f$$

$$\mathbf{w} = \arg \min_{\mathbf{w}} \|\mathbf{y} - \Phi_{\tilde{\mathcal{T}}^k} \mathbf{w}\|_2$$
backward step:

$$\mathcal{T}_b = \arg \min_{\mathcal{J}:|\mathcal{J}|=\beta} \|\mathbf{w}_{\mathcal{J}}\|_1$$

$$\mathcal{T}^k = \tilde{\mathcal{T}}^k - \mathcal{T}_b$$
projection:

$$\mathbf{w} = \arg \min_{\mathbf{w}} \|\mathbf{y} - \Phi_{\mathcal{T}^k} \mathbf{w}\|_2$$

$$\mathbf{r}^k = \mathbf{y} - \Phi_{\mathcal{T}^k} \mathbf{w}$$
termination rule:
if $\|\mathbf{r}^k\|_2 \leq \varepsilon \|\mathbf{y}\|_2$ **or** $|\mathcal{T}^k| \geq K_{\max}$ **then**
break
end if

end while

$\tilde{\mathbf{x}} = 0$
 $\tilde{\mathbf{x}}_{\mathcal{T}^k} = \mathbf{w}$
return $\tilde{\mathbf{x}}$

observations, ε should be selected depending on the noise level. To avoid the algorithm running for too many iterations in case of a failure, the maximum size of the support estimate is also limited by K_{\max} .

6.3 Relations to Other Greedy Algorithms

6.3.1 Two Stage Thresholding Algorithms

Two stage thresholding is defined as a general class of algorithms where the forward step consists of a modification of the sparse estimate followed by thresholding and projection of the residue onto the selected support, and the backward step consists of a second thresholding operation, which is also mostly followed by projection of the residue onto the pruned support estimate.

The most common examples of the TST-type algorithms are SP and CoSaMP. As for FBP, these algorithms are also based on iterative expansion and shrinkage of the support estimate. They allow for both the addition and removal of cK nonzero indices per iteration, where $c = 1$ for SP and $c = 2$ for CoSaMP. The main drawback of these algorithms is the equal forward and backward step sizes, as a result of which the support size should be kept fixed between the iterations. That is, these algorithms iteratively refine a support estimate with fixed size. Hence, they require an *a priori* estimate of the underlying sparsity level K . This is an important handicap for the practical application of these algorithms, since K is mostly unknown.

In [16], Maleki and Donoho propose an optimum TST scheme, which turns out to be a tuned version of the SP algorithm. They suggest utilizing an optimally tuned support size for a given $\{M, N\}$ pair. The tuning is performed using sparse signals with constant amplitude random sign (CARS) nonzero elements, which constitute the most difficult recovery problem for greedy methods. The optimum support size is selected proportional to the sparsity ratio $\frac{K}{M}$ corresponding to the 50% exact recovery rate for the actual $\frac{M}{N}$ ratio. The motivation behind this choice is selecting the support size as large as the maximum sparsity level which SP can exactly recover for the actual values of M and N . As the optimum $\frac{K}{M}$ rate is pre-computed for each $\frac{M}{N}$ value, the support size can be decided on-the-fly using the actual M and N values. The resultant tuned SP algorithm turns out to be best-performing TST scheme for the CARS sparse signals according to the empirical results in [16]. On the other hand, these results³ also indicate that overestimating the support size degrades the recovery accuracy. Hence, though this

³See Figure 5 in [16], where choosing the support size larger than the actual sparsity level degrades the recovery performance. According to this figure, the support size of SP should be exactly equal to the actual sparsity level for optimal performance.

tuned algorithm is acknowledged as the optimum TST scheme in [16], its performance is usually worse than the SP algorithm which requires an oracle to predict the actual sparsity level. Therefore, we employ the SP algorithm with an oracle, and not the tuned TST, in our experiments below. In addition, [16] only covers the empirical performance in the worst case scenario for the greedy methods involving the CARS sparse signals. Actually, the performances of greedy methods vary greatly for different distributions as demonstrated below by our empirical results. Though the CARS experiment is the limiting worst case, other greedy algorithms, such as OMP and FBP, yield better recovery results when the underlying sparse signals do not have constant amplitude nonzero elements.

In contrast to SP and CoSaMP, the FBP algorithm does not require an *a priori* estimate of the sparsity level K . Unlike the tuned TST, it does not necessitate a tuning of the support size either. As explained above, FBP enlarges the support estimate by $\alpha - \beta$ indices at each iteration until termination of the algorithm, which is based on the residual power instead of the sparsity level. Hence, neither the forward and backward steps nor the termination criterion require an estimate of the sparsity level. Among the simulations presented in the next section, we demonstrate a simple empirical strategy for choosing optimal step sizes, according to which, these can be chosen as a fixed ratio of the observation length. Moreover, the simulation results also indicate that the algorithm is quite robust to the choice of the forward and backward step sizes. This makes the FBP algorithm easily applicable in practice in contrast to SP and CoSaMP. However, this advantage comes at a cost, at least for now: The theoretical guarantees of FBP cannot be provided in a way similar to the SP or CoSaMP algorithms, which make use of the support size being fixed as K after the backward step. Consequently, we cannot provide the theoretical analysis of FBP in this work, and leave this as a possible research direction for the future. Note that, however, most of the theoretical analysis steps of SP or CoSaMP also hold for FBP. In addition, success of the forward step may be guaranteed with a condition similar to that for success of an A*OMP iteration presented in the previous chapter. The success condition for the backward step, however, still remains open.

6.3.2 Forward Greedy Algorithms

Sparse signal recovery schemes that enlarge the support estimate iteratively via forward selection steps can be classified as forward greedy algorithms. These involve algorithms that employ a forward step size of one, such as the matching pursuit (MP) [68] and OMP algorithms, in addition to more complicated methods which utilize selection of multiple indices at each forward step. Examples for the latter include MP variants such as the regularized OMP (ROMP) algorithm [27, 101], the generalized OMP algorithm (GOMP) [28, 103], and the Stagewise OMP (StOMP) algorithm [102].

The forward greedy algorithms have a fundamental drawback by definition: Since they possess no backward removal mechanism, any index that is inserted into the support estimate cannot be removed. That one or more incorrect elements remain in the support until termination may cause the recovery to fail. FBP, on the contrary, employs a backward step, which provides means for removal of atoms from the support estimate. This gives FBP the ability to cover up for the errors made by the forward step.

6.4 Empirical Analyses

This section is reserved for the demonstration of the FBP recovery performance in comparison to the basis pursuit (BP), SP, and OMP algorithms. For this purpose, we run recovery simulations involving different nonzero coefficient distributions, noiseless and noisy observations, and images. First, we compare the exact recovery rates, average recovery error, and run times of FBP with those of OMP, SP, and BP for signals with nonzero elements drawn from the Gaussian distribution. In order to generalize the results to a wide range of M and K along with different nonzero element distributions, we provide the empirical phase transition curves, which are obtained using the procedure defined in [16]. Meanwhile, these phase transition curves also serve for the purpose of investigating the optimal α and β choices. We also compare these phase transitions with those of the A*OMP algorithm from Chapter 5. Next, we demonstrate recovery from noisy observations. Finally, we test our proposal on images to illustrate the recovery performance for realistic coefficient distributions. Note that we run OMP with the residue-based termination criterion in the simulations below. Consequently, the abbreviation OMP, where we intentionally skip the subscript in the rest of this chapter, refers

actually to the version OMP_e . In addition, OMP shares the same set of termination parameters, i.e., ε and K_{\max} , with FBP.

6.4.1 Exact Recovery Rates and Reconstruction Error

First, we compare the exact recovery rates, recovery error, and run times of FBP using various α and β values with those of OMP, SP, and BP. In these simulations, the signal and observation sizes are fixed as $N = 256$ and $M = 100$ while K varies in $[10, 50]$. For each K , the recovery simulations are repeated over 500 test samples with randomly located nonzero elements. The nonzero elements of these samples are drawn randomly from the standard Gaussian distribution. As before, we call this type of sparse signals the Gaussian sparse signals. For each test sample, a different observation matrix is drawn randomly from the Gaussian distribution with mean zero and standard deviation $1/N$. ε is set to 10^{-6} and K_{\max} is 55 for both FBP and OMP. The recovery error is expressed in terms of the average normalized mean-squared-error (ANMSE), which is computed for each K over all involved K -sparse test samples as

$$\text{ANMSE} = \frac{1}{500} \sum_{i=1}^{500} \frac{\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2}{\|\mathbf{x}_i\|_2^2} \quad (6.1)$$

where $\hat{\mathbf{x}}_i$ is the recovery of the i th test vector \mathbf{x}_i . In addition, we present the exact recovery rates, which represent the ratio of perfectly recovered test samples to the whole test data. The exact recovery condition is specified as $\|\mathbf{x} - \hat{\mathbf{x}}\|_2 \leq 10^{-2} \|\mathbf{x}\|_2$, where $\hat{\mathbf{x}}$ denotes the recovery of \mathbf{x} .

Figure 6.1 and 6.2 depict the reconstruction performance of FBP with various choices of α and β for the Gaussian sparse signals in comparison to the OMP, BP, and SP algorithms. Figure 6.1 is obtained by varying α in $[2, 30]$, while the backward step size is selected as $\beta = \alpha - 1$ for each different forward step size. That is, the support estimate is expanded by one element per iteration, whereas the forward step size varies. For Figure 6.2, the forward step size is fixed as $\alpha = 20$, and the backward step size is altered in $[13, 19]$. This corresponds to changing the increment in the support size per iteration with a fixed forward step size. The run times of the FBP, SP, and OMP algorithms are also compared, while BP is excluded as it is incomparably slower than the other three algorithms.

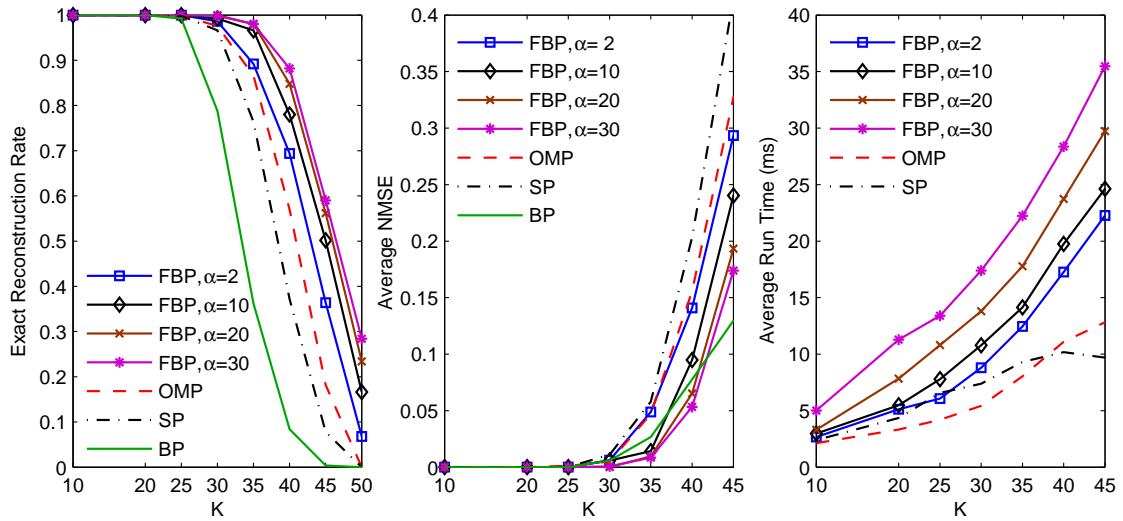


FIGURE 6.1: Reconstruction results over sparsity for the Gaussian sparse vectors. For FBP, $\beta = \alpha - 1$.

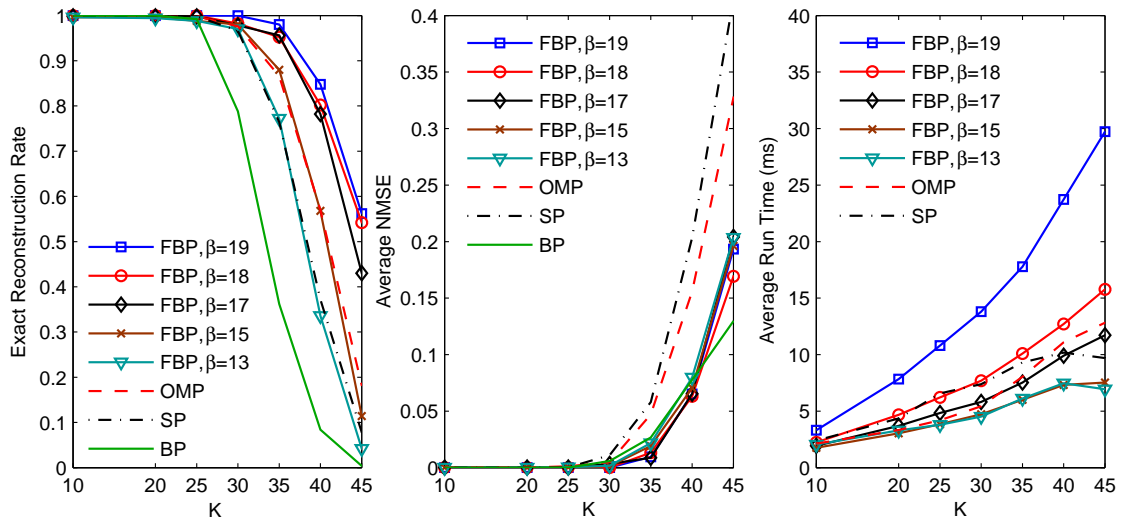


FIGURE 6.2: Reconstruction results over sparsity for the Gaussian sparse vectors. For FBP, $\alpha = 20$.

According to Figure 6.1, increasing α while keeping the support increment $\alpha - \beta$ fixed improves the recovery performance of FBP. We observe that the exact recovery rates of FBP are significantly better than the other candidates for all choices of α , even including the modest choice $\alpha = 2$. BP, SP, and OMP start to fail at around $K = 25$, where FBP is still perfect for all choices of α . Moreover, for $\alpha \geq 20$, the FBP failures begin only when $K > 30$. As for the ANMSE, FBP is the best performer when $\alpha \geq 20$. With this setting, BP can beat FBP in ANMSE only when $K > 40$. In addition, FBP yields

better recovery than OMP and SP for all choices of α .

In Figure 6.2, we observe that increasing β for a fixed α also improves the recovery performance. In this case, the exact recovery rates of FBP increase significantly with the backward step size, while the corresponding ANMSE values remain mostly unaltered. This indicates that when β is increased, nonzero elements with smaller magnitudes, which do not significantly improve the recovery error, can be more precisely recovered. In comparison to the other algorithms involved in the simulation, FBP is clearly the best performer for $\beta > 15$. Even with $\beta = 15$, FBP is still the best algorithm when both the exact recovery rates and ANMSE are considered together. Similar to the previous test case, BP can produce lower ANMSE than FBP only for $K > 40$.

As for the run times, we expectedly observe that increasing α or β slows down FBP. This is due to the decrease in the increment of the support size per iteration, which increases the number of iterations and the number of required orthogonal projection operations. Moreover, the dimensions of the orthogonal projection operations also increase with the forward step size. On the other hand, increasing $\alpha - \beta$ decreases the number of necessary iterations. As a result, FBP terminates faster. More important for this example, we observe that the run times of FBP, SP, and OMP are very close when $\alpha = 20$ and $\beta \leq \alpha - 2$. In case $\alpha = 20$ and $\beta = 17$, the speed of FBP and OMP are almost the same, whereas the reconstruction performance of FBP is significantly better than the other algorithms involved. With $\alpha = 20$ and $\beta = 15$, FBP is even faster than OMP and SP, while its performance is still better than these in general⁴.

To summarize the findings of this section, we observe that FBP performs better recovery than all other candidates in general for the Gaussian sparse signals. That its performance is better than the SP algorithm indicates that it also yields better recovery than the optimally tuned TST of [16] for this type of sparse signals. Moreover, these improvements can be obtained in quite short run times, which are equal to or better than those of the OMP and SP algorithms.

⁴Note that the speed of FBP can be further improved by removing the orthogonal projection operation after the backward step, at the expense of a slight degradation in the recovery performance.

6.4.2 Phase Transitions

The phase transitions are important for the empirical evaluation of sparse signal recovery algorithms over a wide range of the sparsity level and the observation length. Below, we present the empirical phase transition curves of the FBP algorithm obtained from recovery simulations involving three different nonzero element distributions in comparison to those of the OMP, SP, and BP algorithms. The first one of these distributions is the uniform sparse signals where the nonzero elements are distributed uniformly in $[-1, 1]$. The second type is the Gaussian sparse signals which have been investigated for the exact recovery rates and ANMSE above. The last ensemble involved is the CARS sparse signals where nonzero elements have unit magnitude with random sign. Below, we first compare the phase transitions of FBP with different α and β choices for the uniform sparse signals in order to investigate the optimality of these over the observation length. These simulations provide us an empirical strategy about how to choose the FBP step sizes in relation to M . Next, we compare FBP with a fixed setting to BP, OMP, and SP for all three test sets.

To explain how we obtain the empirical phase transitions, let us first define normalized measures for the observation length and the sparsity level as $\lambda = \frac{M}{N}$ and $\rho = \frac{K}{M}$. To obtain the empirical phase transition curves, we keep the signal length fixed at $N = 250$, and alter M and K to sample the $\{\lambda, \rho\}$ space for $\lambda \in [0.1, 0.9]$ and $\rho \in (0, 1]$ ⁵. For each $\{\lambda, \rho\}$ tuple, we randomly generate 200 sparse test signals and run FBP, OMP, BP, and SP algorithms for the recovery of each sparse signal. We employ an individual Gaussian observation matrix for each sparse signal. The termination parameters are selected as $\varepsilon = 10^{-6}$ and $K_{\max} = M$ for FBP and OMP. Specifying the exact recovery condition as $\|\mathbf{x} - \hat{\mathbf{x}}\|_2 \leq 10^{-2}\|\mathbf{x}\|_2$,⁶ where $\hat{\mathbf{x}}$ denotes the recovery of \mathbf{x} , the exact recovery rates are obtained for each $\{\lambda, \rho\}$ tuple and each algorithm. The phase transitions are then obtained using the methodology described in [16]. That is, for each λ , we employ a generalized linear model with logistic link to describe the exact recovery curve over ρ ,

⁵The λ axis is sampled with a resolution of 0.1, while the corresponding ρ values are chosen densely around the phase transition region for a specific λ in order to obtain a fine modelling of the transition region.

⁶This exact recovery condition is the same as the one in [16]. This choice has been made for the compatibility of the computed phase transitions with [16].

and then find the ρ value which yields 50% exact recovery probability⁷. Combining the results over the whole λ range, we end up with the empirical phase transition curve⁸.

The phase transitions provide us important means for finding an empirical way of choosing α and β optimally. As discussed in [16], the phase transition curve is mostly a function of λ . That is, it remains unaltered when N changes. Moreover, the transition region turns out to be narrower with N increasing. These claims are also supported by some other publications in the literature, such as [102, 110, 127, 128]. Hence, in order to find an optimal set of step sizes for FBP, we need to have a look at the empirical phase transitions with different α and β choices. For a better understanding of their optimality, α and β should not be fixed but be proportional to M . Trying to find fixed α and β values is subject to fail mainly for very low or very high λ values. In other words, it would not be possible to find a fixed optimal set $\{\alpha, \beta\}$ for the whole λ range even when we fix N . This is, however, possible when α is proportional to M , and β is related to the chosen α value. In order to find an optimal choice, we run two distinct sets of simulations: First, we vary α in $[0.1M, 0.4M]$, whereas $\beta = \alpha - 1$. Then we fix $\alpha = 0.2M$, and select β either in $[0.7\alpha, 0.9\alpha]$ or as $\alpha - 1$.

The phase transitions obtained by the procedure described above are depicted in Figure 6.3 for the Gaussian, uniform, and binary sparse signals. The graphs on the left side of Figure 6.3 illustrate phase transitions with different α values, while those on the right side show the changes with respect to different β values. These graphs indicate that the performance of FBP fundamentally improves with α and β , except for very high α choices. Another exception is the recovery of sparse signals with constant magnitude nonzero elements, which constitute the hardest problem for this type of algorithms. For this case, which is represented by the CARS ensemble, we observe that the gain with α is not significant, and the phase transitions remain unaltered when β changes. Another important observation that can be deduced from these results is that the performance of FBP is quite robust to the step size choices.

Concentrating on the forward step, the graphs on the left side of Figure 6.3 reveal that the phase transitions are stably improved with α until $\alpha = 0.3M$ for the uniform and Gaussian sparse signals. Choosing $\alpha = 0.4M$, in contrast, improves the phase transitions

⁷Note that, due to narrow phase transition regions [16], the region below the phase transition curve promises exact recovery with high probability for the corresponding recovery method.

⁸This procedure is the same as the computation of phase transitions in Chapter 3 and Chapter 5.

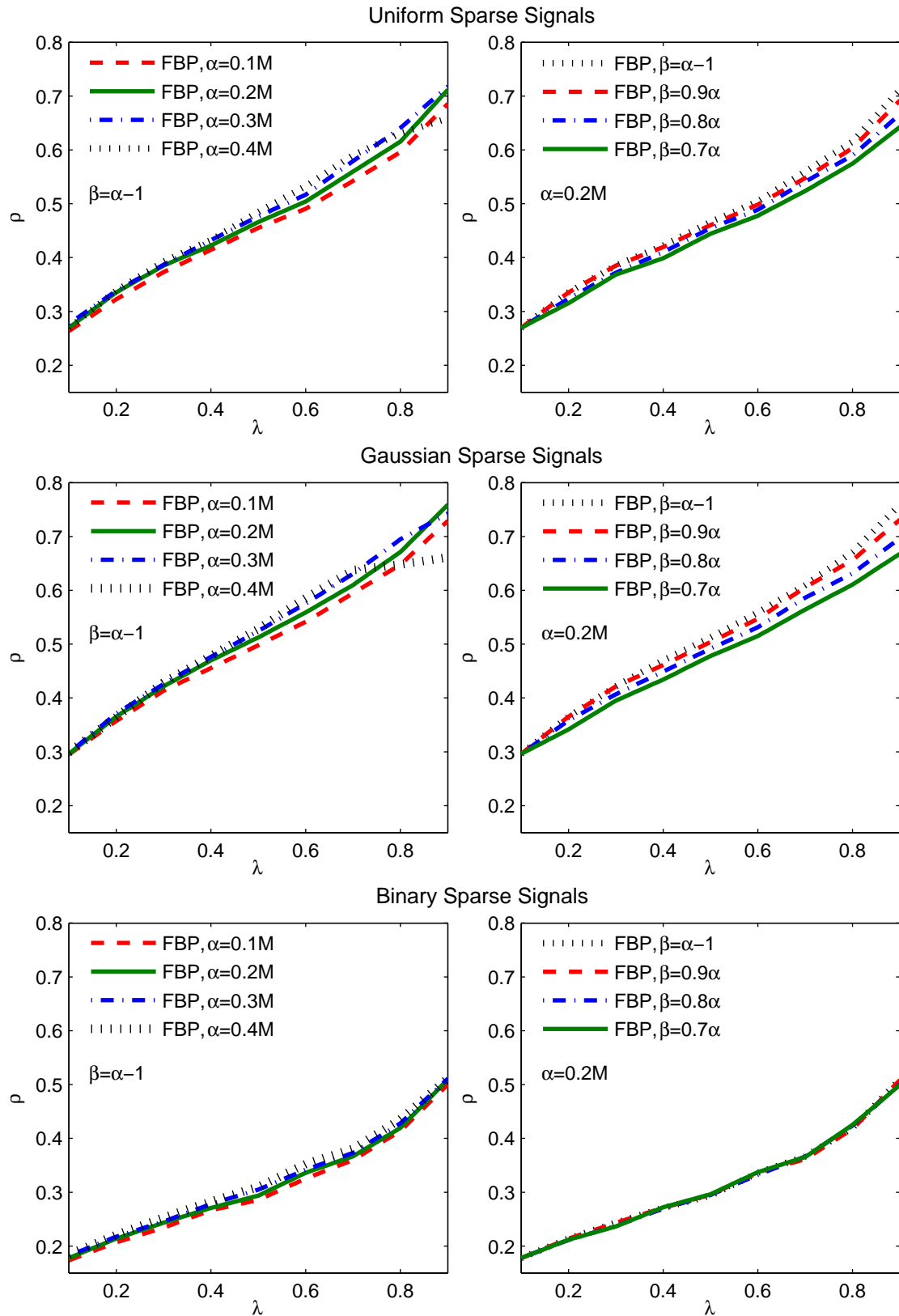


FIGURE 6.3: Phase transitions of FBP with different forward and backward step sizes for the uniform, Gaussian, and CARS sparse signals.

only for the mid- λ region, while the results get worse especially for the high λ values⁹. The reason for this degradation is the high value that $\alpha = 0.4M$ takes when M is large. As a result of this large forward step size, the size of the expanded support estimate exceeds M after the forward step when K is also large, i.e., the expanded support estimate already becomes larger than the spark¹⁰ of the dictionary, before the solution can be found. This leads to an ill-posed orthogonal projection problem, and causes the recovery to fail. Hence, we suggest using $\alpha = 0.3M$ for a globally optimum FBP scheme, while this value might be increased if the recovery problem lies in the mid- λ region. On the other hand, taking into account the computational complexity, we observe that there is not a significant decrement in the recovery performance when α is chosen smaller. As a consequence of this observation, we select $\alpha = 0.2M$ below for faster termination, and show that even this choice already leads to better phase transitions than OMP, BP, and SP for the uniform sparse signals. In fact, the graphs on the left side of Figure 6.3 state that the recovery performance of FBP is quite robust to the choice of the forward step size.

As for the backward step, it is obvious that the recovery accuracy decreases with β for the Gaussian and uniform sparse signals. However, we are also interested in observing how fast this occurs. The results state that the loss is slight for the low λ range. Though this degradation increases slightly with λ , we observe that the recovery performance of FBP is quite robust to the choice of the backward step size in addition to its stability over the forward step size, which has been discussed above. In comparison to the phase transitions of the other algorithms, which are not plotted on this figure but are available below in Figure 6.4, FBP provides better phase transition curves even with $\beta = 0.7\alpha$ for the uniform and Gaussian sparse signals, while BP can do slightly better only for the λ region around $0.8 - 0.9$. Remember that the β/α ratio commands the increment in the support size per FBP iteration. Reducing this ratio decreases the number of FBP iterations, and hence accelerates the recovery process. Therefore, these results reveal that it is possible to reduce the complexity of FBP until β is about 0.7α , while the phase

⁹We do not increment α over $0.4M$, however note that doing so would even further narrow the mid- λ range where the recovery is slightly improved, and widen the high λ region where the performance is degraded.

¹⁰Spark of a dictionary is defined as the smallest number for which RIP cannot be satisfied with any $\delta > 0$. Obviously, spark of an $M \times N$ dictionary cannot exceed M , since any set of $M + 1$ columns is linearly dependent.

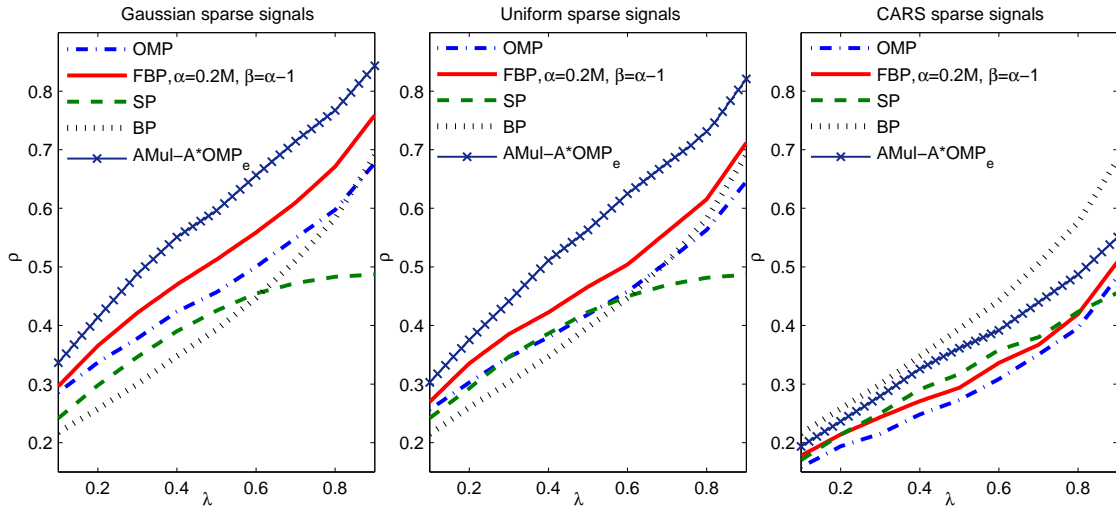


FIGURE 6.4: Phase transitions of FBP, BP, SP, OMP, and $\text{AMul-A}^*\text{OMP}_e$ for the Gaussian, uniform, and CARS sparse signals.

transition curves are still better than those of the BP, SP, and OMP algorithms for the recovery of uniform and Gaussian sparse signals.

Figure 6.4 compares the phase transition curves of FBP to those of A^*OMP , OMP, BP, and SP for the Gaussian, uniform, and CARS sparse signals, where the step sizes of FBP are fixed as $\alpha = 0.2M$ and $\beta = \alpha - 1$. First, we observe that A^*OMP outperforms FBP for all cases. This is an expected behavior since A^*OMP is a much sophisticated semi-greedy approach which employs a complicated, and hence time consuming, tree search. FBP, on the other hand, is a fast greedy algorithm that can solve the recovery problem faster, but possibly less accurately. The difference in the run times of the two approaches is obvious when one considers that FBP is approximately as fast as OMP, while A^*OMP is naturally much slower than it. As for the comparison with the other algorithms, FBP is the best performer for the Gaussian and uniform sparse signals. On the contrary, BP outperforms the others in the CARS case. For this case, SP also turns out to be partially and slightly better than FBP.

As before, we observe that the phase transition of BP is robust to the nonzero coefficient distribution, whereas the performances of the greedy and semi-greedy methods degrade for the CARS case. We observe that FBP, A^*OMP , and OMP curves show the highest variation among different distributions. This finding is similar to our observations for OMP and A^*OMP in the previous chapters. The performances of these algorithms are boosted when the nonzero values cover a wide range, such as for the Gaussian

distribution. On the other hand, nonzero values with equal magnitudes are the most challenging case due to the correlation maximization step. This behaviour is related to the decreased spread of the elements at the desired locations of the correlation vector for the CARS type signals, which has been discussed in Section 3.5.1 regarding the OMP algorithm.

6.4.3 Recovery from Noisy Observations

Next, we simulate recovery of sparse signals from noise-contaminated observations. For this purpose, we alter the observation model as

$$\mathbf{y} = \Phi \mathbf{x} + \mathbf{n} \quad (6.2)$$

where \mathbf{n} represents some additive observation noise. We model \mathbf{n} as white Gaussian noise, and alter the signal-to-noise ratio (SNR), which is defined as $20 \log \frac{\|\mathbf{y}\|_2}{\|\mathbf{n}\|_2}$, from 5 to 40 dB for the purpose of obtaining a general performance measure. In this simulation, FBP is run with $\alpha = 20$ and $\beta = 17$, i.e., $\alpha = 0.2M$ and $\beta = 0.85\alpha$, as we have seen above that OMP and FBP require similar run times for this choice. ε is selected with respect to the noise level, such that the remaining residual energy becomes equal to the noise energy after termination. The regularization parameter of BP is also adjusted in a similar fashion. The simulation is repeated for 500 Gaussian and 500 uniform sparse signals, where $N = 256$ and $M = 100$. For each test example, we employ an individual Gaussian observation matrix. The sparsity levels are selected as $K = 30$ and $K = 25$ for the Gaussian and uniform sparse signals, respectively. K_{max} is 55 as in the first set of simulations. Figure 6.5 depicts the recovery error for the noisy Gaussian and uniform sparse signals, while the run times are compared in Figure 6.6. Note that we express the recovery error in the decibel (dB) scale, calling it the distortion ratio, in order to make it better comparable with the SNR. Clearly, FBP yields the most accurate recovery among the candidate algorithms for both distributions, while BP can do slightly better than FBP only when SNR is about 5 dB. In addition, the run times reveal that FBP is not only the most accurate algorithm in this example, but is also as fast as OMP when $\alpha = 20$ and $\beta = 17$. Note that, increasing β beyond 17 would improve the recovery accuracy of the FBP algorithm, while the algorithm would require a slightly longer run time.

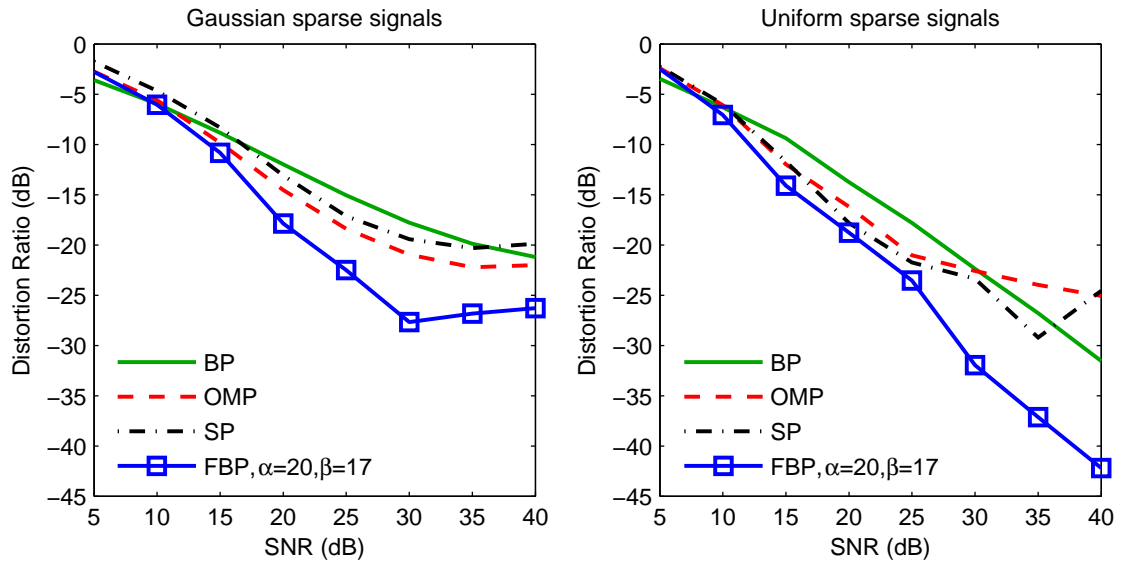


FIGURE 6.5: Average recovery distortion over SNR in case of noise contaminated observations. $K = 30$ and $K = 25$ for the Gaussian and uniform sparse signals, respectively.

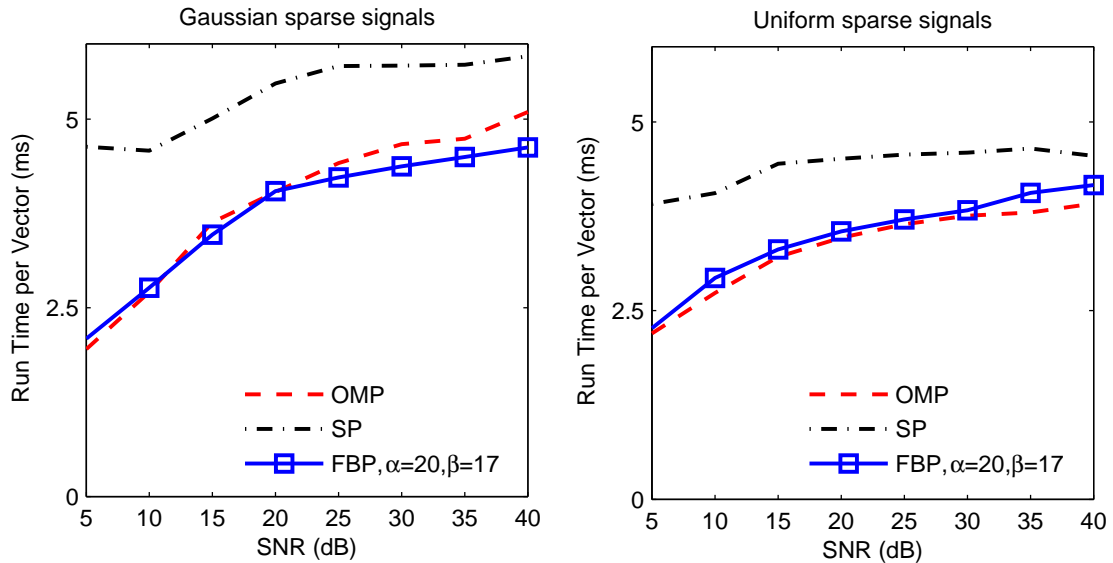


FIGURE 6.6: Average run time per test sample in case of noise contaminated observations. $K = 30$ and $K = 25$ for the Gaussian and uniform sparse signals, respectively.

6.4.4 Image Recovery Examples

To test FBP in a more realistic case, we demonstrate recovery of two 512×512 images below. As in the previous chapters, the recovery of these images is performed in blocks of size 8×8 , with the motivation of breaking the recovery problem into a number of smaller

and simpler subproblems. The reconstruction is performed in the 2D Haar Wavelet basis Ψ since the images are compressible in this domain. Note that, in this case, the reconstruction dictionary is not the observation matrix Φ itself, but its multiplication with the 2D Haar Wavelet basis, i.e., $\Phi\Psi$. That is, \mathbf{x} denoting the sparse wavelet coefficient vector of interest, the image itself is obtained as $\Psi\mathbf{x}$ after the recovery of \mathbf{x} from the observation $\mathbf{y} = \Phi\Psi\mathbf{x}$. For the first test, the image “bridge” is preprocessed such that each of its 8×8 blocks is K -sparse in the 2D Haar Wavelet basis, i.e., only the K largest magnitude wavelet coefficients are kept for each block. In the second case, the image “butterfly” is recovered without any such preprocessing. $M = 32$ observations are taken from each block of the images, where the entries of Φ are randomly drawn from the Gaussian distribution with mean zero and standard deviation $1/N$. The termination parameter of FBP is selected as $\varepsilon = 10^{-6}$. The forward step is selected as $\alpha = 10$, while the backward step size is either $\beta = 9$ or $\beta = 7$. Due to the different settings of the two problems, FBP is run with different K_{\max} values, as discussed below.

6.4.4.1 Demonstration on a Sparse Image

The first example is the recovery of the sparse image “bridge”. As discussed above, the sparseness of the image “bridge” is ensured by preprocessing prior to the recovery where each 8×8 block is forced to be K -sparse in the 2D Haar Wavelet basis by keeping the largest magnitude coefficients only. The sparsity level is selected as $K = 12$. For this sparse case, K_{\max} is selected as 20.

Figure 6.7 shows the preprocessed test image “bridge” on the upper left panel, while the BP recovery is on the upper right panel. FBP recovery with $\alpha = 10$, $\beta = 7$ can be found on the lower left panel, and FBP recovery with $\alpha = 10$, $\beta = 9$ is on the lower right panel. In this example, BP provides a peak signal-to-noise ratio (PSNR) of 29.9 dB, whereas the much simpler and faster FBP algorithm improves the recovery PSNR up to 32.5 dB when $\alpha = 10$ and $\beta = 9$. A careful investigation of the recovered images shows that FBP is able to improve the recovery at the detailed regions and edges. This example demonstrates that the simpler FBP algorithm is able to perform more accurate and faster recovery of a sparse signal with a realistic nonzero coefficient distribution than the much more sophisticated ℓ_1 norm minimization approach.

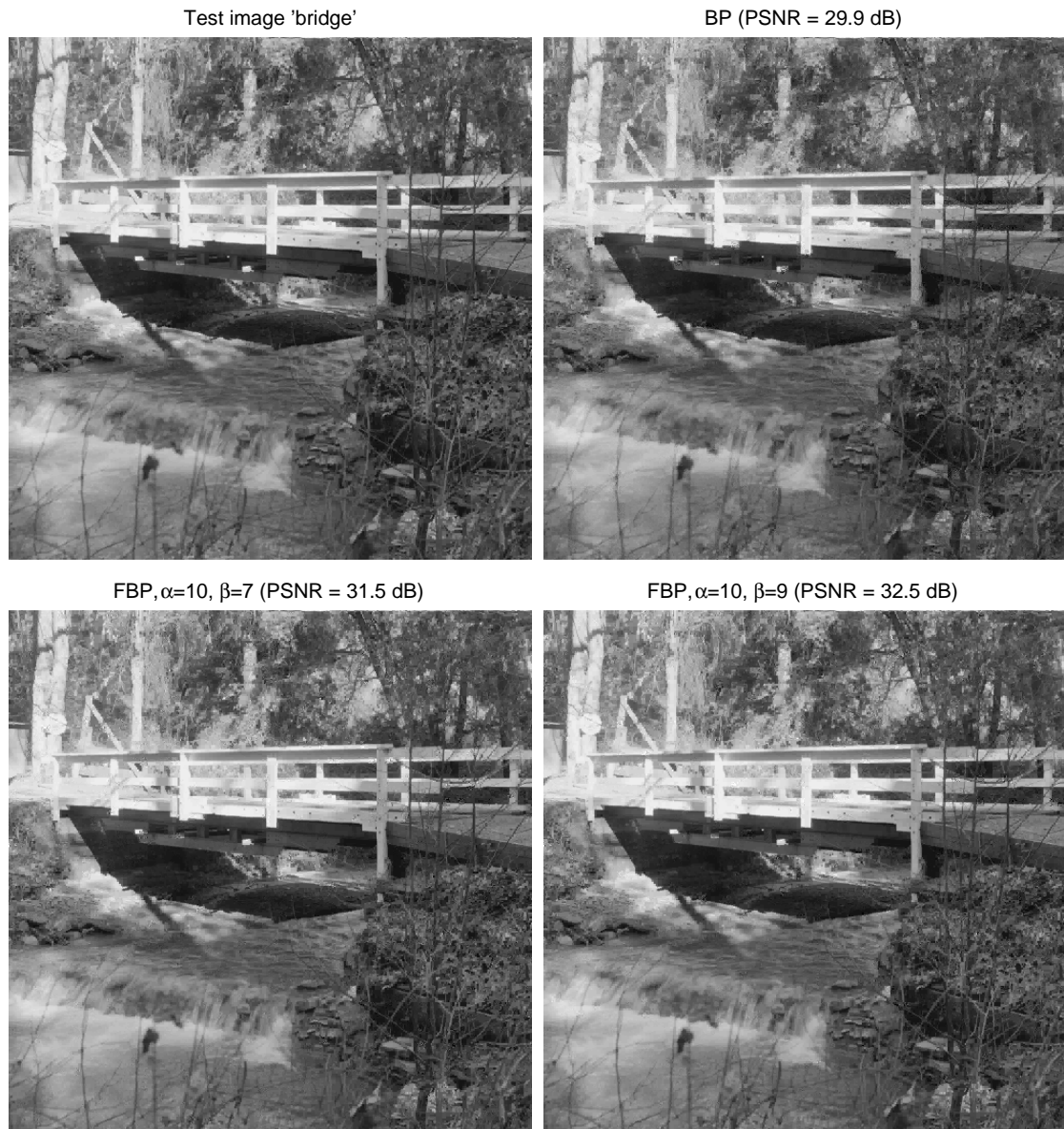


FIGURE 6.7: Recovery of the image “bridge” using BP and FBP. BP recovery yields 29.9 dB PSNR, while FBP provides 31.5 dB PSNR for $\alpha = 10$, $\beta = 7$ and 32.5 dB PSNR for $\alpha = 10$, $\beta = 9$.

6.4.4.2 Demonstration on a Compressible Image

As the underlying image “butterfly” is not sparse, the second image recovery example in this section has to deal with a harder problem. In this case, the goal of the recovery is obtaining the best K -sparse approximation by exploiting the compressibility of the image in the Wavelet transform domain. Since the recovery problem has a different structure of than the one above, FBP has better be run with $K_{\max} = 12$ in order to

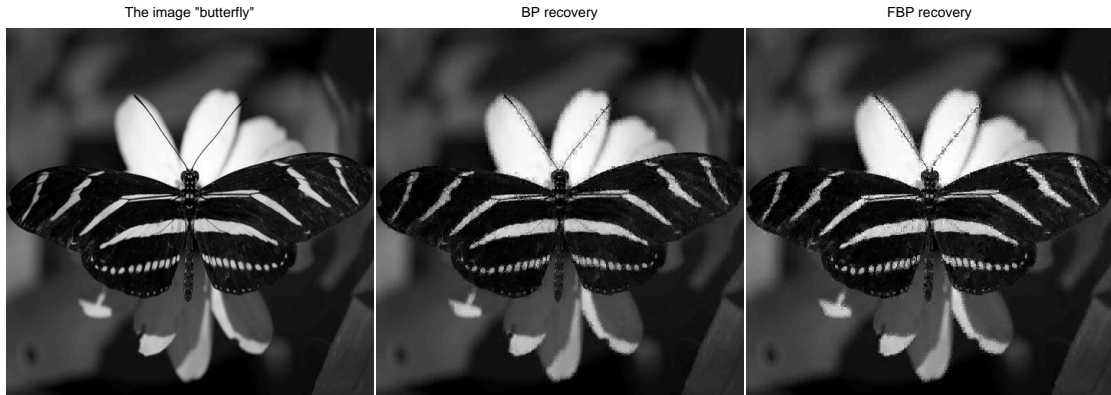


FIGURE 6.8: Recovery of the image “butterfly” using BP and FBP. BP and FBP yield 27.8 and 27.4 dB PSNR, respectively.

impose the actual goal of recovering the largest 12 wavelet coefficients for each block. For this test, we demonstrate FBP recovery only with $\alpha = 10$ and $\beta = 9$.

The recovery results for the image “butterfly” are depicted in Figure 6.8. The image “butterfly” can be seen on the left panel of the figure, while the middle and right panels of Figure 6.8 show the BP and FBP recoveries of the image, respectively. As for the PSNR, both of the algorithms end up with similar values¹¹. In particular, BP and FBP yield 27.8 and 27.4 dB PSNR, respectively¹². A careful visual comparison reveals that each of the algorithms perform somewhat better than the other one on some particular regions of the image, while the two approximations are very similar in terms of the total perceptual quality.

6.5 Summary

In this chapter, we have introduced the forward-backward pursuit algorithm for recovery of sparse signals from compressed measurements. FBP, which incorporates iterative forward and backward steps, falls into the category of TST algorithms [16]. The forward step enlarges the support estimate by α atoms, while the backward step removes $\beta < \alpha$ atoms from it. Hence, this two stage scheme iteratively expands the support estimate for the sparse signal, without requiring the sparsity level K *a priori*, as SP or CoSaMP

¹¹In fact, we have observed that such a behavior is common for a set of images as well.

¹²Note that the best 12-sparse representation of the image “butterfly” has 32.3 dB PSNR. That is, the maximum PSNR value that can be obtained with any algorithm searching for the best 12-sparse representation is 32.3 dB.

do. In comparison to forward greedy algorithms, FBP provides a backward step for removing possibly misidentified atoms from the solution at each iteration.

The recovery performance of FBP has been demonstrated in Section 6.5 in comparison to the OMP, SP, and BP algorithms. The simulations contain recovery of sparse signals with different distributions of the nonzero elements and recovery from noisy observations in addition to the demonstration of the algorithm on images. The results indicate that except for sparse signals with constant amplitude (the CARS ensemble), FBP can provide better exact recovery rates than the OMP, BP, and SP algorithms. We observe that the choice $\alpha = 0.3M$ and $\beta = \alpha - 1$ leads to the optimum empirical FBP performance, while α and β can be decreased in order to speed up the algorithm with a slight sacrifice of the recovery performance due to the robustness of the algorithm demonstrated by the phase transition curves. In case of the CARS sparse signals, where the ℓ_0 and ℓ_1 norms are equal, BP turns out to be the most accurate algorithm, outperforming the greedy candidates. The noisy recovery examples state that FBP is also robust to the observation noise, and provides more accurate recovery under noise than OMP, BP, and SP for the Gaussian and uniform sparse signals. Finally, the demonstration of FBP on images indicates the recovery abilities of the algorithm for signals with realistic nonzero element distributions. We have observed that the FBP recovery of a sparse image is better than the corresponding BP recovery. On the other hand, FBP and BP yield very close recovery accuracies for compressible images, while FBP is obviously faster.

Finally, in order to avoid any misinterpretation, we would like to note that our findings do not contradict with the results of Maleki and Donoho in [16]. The results in [16] indicate that the tuned SP algorithm is the optimal TST scheme for the CARS ensemble. Our results for the CARS case are parallel to this: SP performs better than FBP and OMP in the CARS scenario, where BP is the best performer. However, [16] does not contain adequate analysis for other types of sparse signals. Our results show that FBP provides better recovery than SP and BP when the magnitudes of nonzero elements are not comparable. We observe that the FBP performance gets even better when the magnitudes of nonzero elements start spanning a wider range, as for the Gaussian distribution. These indicate that SP is not necessarily the optimum TST scheme for all nonzero element distributions. From a global perspective, the CARS sparse signals are the most difficult case for greedy algorithms, and hence the corresponding recovery results can be taken as the worst case performance for greedy algorithms. Therefore, as

pronounced in [16], SP has the best worst-case performance among the TST schemes, while it is outperformed by FBP when the nonzero elements do not have comparable amplitudes anymore.

Chapter 7

A Mixed Integer Linear Programming Approach for Sparse Signal Recovery

7.1 Introduction

In this chapter, we propose using mixed integer linear programming (MILP) techniques to solve for an equivalent of the sparse signal recovery problem, which has been defined in Chapter 2 as the following ℓ_0 norm minimization

$$\begin{aligned} & \text{minimize} && \|\mathbf{x}\|_0 \\ & \text{subject to} && \mathbf{\Phi}\mathbf{x} = \mathbf{y}, \end{aligned} \tag{7.1}$$

where $\mathbf{x} \in \mathbb{R}^{M \times 1}$ is the K -sparse signal of interest, $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ is the observation matrix, and $\mathbf{y} \in \mathbb{R}^{M \times 1}$ denotes the vector of “compressed” observations. Previous works in the CS literature have examined the use of linear programming (LP) techniques intensively for the sparse signal recovery problem, however, MILP has not been yet explored for this purpose.

Exploiting MILP via the formulation presented in this chapter has a fundamental advantage over the mainstream sparse signal recovery methods: The proposed formulation is not an approximation of the ℓ_0 minimization problem in (7.1) as for the other methods,

but it turns out to be equivalent to (7.1). Consequently, the solution of the proposed formulation is exactly equal to the sparsest solution of the original recovery problem, once it is feasible.

In order to obtain the MILP equivalent of (7.1), we introduce in Section 7.2 an auxiliary binary vector \mathbf{z} of length N , on which the nonzero indices of \mathbf{x} are located by ones. Then, (7.1) can be cast into an equivalent MILP problem which is based on the joint optimization of \mathbf{z} and \mathbf{x} . Even though MILP problems are mostly NP-hard, addition of a few additional constraints based on a number of reasonable assumptions allow for the solution of this modified problem in reasonable time. In Section 7.3, the tractability of the proposed method is demonstrated by a number of simulations for recovery of sparse signals from noise-free measurements. These simulations not only reveal the performance of the proposed approach for recovery of sparse signals with different characteristics, but also compare it to a number of well-known algorithms in the field such as the subspace pursuit (SP), basis pursuit (BP), orthogonal matching pursuit (OMP), iterative hard thresholding (IHT), iterative support detection (ISD), smoothed ℓ_0 (SL0), and A*OMP.

7.2 The Equivalent MILP Formulation of the Sparse Signal Recovery Problem

The MILP equivalent of (7.1) may be obtained by exploiting a fundamental observation about the sparse signal recovery problem: (7.1) may be considered as an optimization problem involving two subproblems which should be solved simultaneously. The first one of these problems is identifying the locations of the nonzero elements in \mathbf{x} , i.e., the support of \mathbf{x} , and the other one searches for the values of these nonzero elements. Below, we introduce an auxiliary vector to define the former, while the latter appears as bound constraints on \mathbf{x} depending on the introduced auxiliary vector.

7.2.1 Problem Formulation

Let \mathcal{T} be the support of \mathbf{x} , and $\mathbf{x}_{\mathcal{T}}$ be the vector consisting of the elements of \mathbf{x} indexed by \mathcal{T} . Next, we define the binary auxiliary vector $\mathbf{z} = [z_1 \ z_2 \ \dots \ z_N]^T$ to mark the

nonzero locations of \mathbf{x} such that

$$z_i = \begin{cases} 1, & \text{if } i \in \mathcal{T}; \\ 0, & \text{otherwise.} \end{cases} \quad (7.2)$$

Now, the original problem (7.1) can be equivalently written as

$$\begin{aligned} & \text{minimize} && \mathbf{e}^T \mathbf{z} \\ & \text{subject to} && \mathbf{\Phi} \mathbf{x} = \mathbf{y}, \\ & && c_l z_i \leq x_i \leq c_u z_i, \quad i = 1, \dots, N, \\ & && z_i \in \{0, 1\}, \quad i = 1, \dots, N, \end{aligned} \quad (7.3)$$

where \mathbf{e} is a vector of ones, and $c_l, c_u \in \mathbb{R}$ are chosen large enough so that the range $[c_l, c_u]$ covers all of the nonzero values in \mathbf{x} . The bound constraints given in the third line of (7.3) force the nonzero elements of \mathbf{x} to appear only at the locations marked by \mathbf{z} . These enable us to solve for \mathbf{x} and \mathbf{z} simultaneously.

Though (7.3) is already enough for finding the correct support of \mathbf{x} , we also define the sparsity constraint as

$$\mathbf{e}^T \mathbf{z} \leq rM, \quad (7.4)$$

where $0 < r \leq 1$. This constraint sets an upper limit on the sparsity of the recovered vector, hence reduces the size of the feasible solution space. We discuss the choice of r below.

7.2.2 Implementation of the MILP Formulation

Next, we define a combined representation to implement the MILP formulation of the sparse signal recovery problem. For this purpose, let us first introduce an auxiliary vector

$$\mathbf{f} = [\mathbf{z}^T \quad \mathbf{x}^T]^T. \quad (7.5)$$

We also define the weight vector

$$\mathbf{w} = [\mathbf{e}^T \quad \mathbf{0}_{1 \times N}]^T, \quad (7.6)$$

where $\mathbf{0}_{b \times c} \in \mathbb{R}^{b \times c}$ denotes a matrix consisting of zeros only. Using these two definitions, the MILP equivalent of the sparse signal recovery problem (7.1) can be implemented as

$$\begin{aligned}
 & \text{minimize} && \mathbf{w}^T \mathbf{f} \\
 & \text{subject to} && \mathbf{A}_{\text{eq}} \mathbf{f} = \mathbf{b}_{\text{eq}}, \\
 & && \mathbf{A}_{\text{ineq}} \mathbf{f} \leq \mathbf{b}_{\text{ineq}}, \\
 & && z_i \in \{0, 1\}, \quad i = 1, \dots, N,
 \end{aligned} \tag{7.7}$$

where

$$\mathbf{A}_{\text{eq}} = [\mathbf{0}_{M \times N} \quad \Phi], \tag{7.8}$$

$$\mathbf{b}_{\text{eq}} = \mathbf{y}, \tag{7.9}$$

$$\mathbf{A}_{\text{ineq}} = \underbrace{\begin{bmatrix} -c_u & \mathbf{0} & 1 & \mathbf{0} \\ & \ddots & & \ddots \\ \mathbf{0} & -c_u & \mathbf{0} & 1 \\ c_l & \mathbf{0} & -1 & \mathbf{0} \\ & \ddots & & \ddots \\ \mathbf{0} & c_l & \mathbf{0} & -1 \\ 1 & \dots & 1 & 0 & \dots & 0 \end{bmatrix}}_{2N}, \tag{7.10}$$

$$\mathbf{b}_{\text{ineq}} = [\mathbf{0}_{1 \times 2N} \quad rM]^T. \tag{7.11}$$

Note that (7.8) and (7.9) represent the observation constraint $\Phi \mathbf{x} = \mathbf{y}$. The first $2N$ rows of (7.10) and (7.11) represent the bound constraints on the nonzero elements of \mathbf{x} , i.e., $c_l z_i \leq x_i \leq c_u z_i$, while the last rows of these correspond to the sparsity constraint (7.4).

7.2.3 Practical Issues

There exists a number of tools available to solve the MILP problems. In this work, we employ the IBM ILOG CPLEX optimization studio [129] to solve the problem (7.7). In practice, (7.7) might take too long to solve due to the large size of the feasible solution space, even when powerful solvers like CPLEX are employed. The parameters c_l , c_u ,

and r are very important for tractability of the optimization process, since they provide means for reducing the size of the feasible solution space. Below, we discuss proper choices of these parameters.

The parameters c_l and c_u define the range which the nonzero values of \mathbf{x} are allowed to span. If the chosen range is narrower than the actual range for \mathbf{x} , failure of the recovery is obvious. On the other hand, if the range $[c_l, c_u]$ is chosen too wide, then the bound constraints are clearly not tight enough, and they are not useful in reducing the size of the search tree employed in solving the MILP by the solver. Consequently, the computational effort increases along with the solution time. Hence, c_l and c_u should be chosen properly. Having said that, our main concern in this work is not finding the optimal $[c_l, c_u]$ range, but demonstrating the application of MILP in the sparse signal recovery problem. Hence, we do not attempt at finding the optimal c_l and c_u range, but employ appropriate assumptions during the simulations.

The sparsity constraint (7.4) also plays an important role in practice. Note that $r = 1$ is a natural upper bound due to the problem definition. Choosing r smaller, on the other hand, reduces the size of the feasible solution space, and therefore allows for faster termination of the algorithm. However, as for c_l and c_u , r should also not be chosen smaller than the actual sparsity level, since this makes the actual solution infeasible. For many practical applications, K is not known *a priori*, however $K \ll M$ holds in general. In accordance, we choose $r = 0.5$, i.e., $\|\mathbf{x}\|_0 \leq 0.5M$ in the simulations below, while this choice might be modified according to the *a priori* information about a particular recovery problem. In addition, choosing $r = 0.5$ also provides another important advantage. Following the assumption that Φ is full row rank, this choice guarantees that the optimization problem has only one possible solution when $K \leq \frac{M}{2}$.¹ This allows us to configure the optimization parameters such that CPLEX returns the first solution it encounters, without running until the actual termination point where all MILP subproblems are covered. This results in faster termination of the algorithm.

¹This follows from the uniqueness of any K -sparse solution when $2K$ -RIP is satisfied. See Chapter 2 for a discussion of the uniqueness property.

7.3 Simulations

Below, we demonstrate the performance of the proposed MILP solution for the sparse signal recovery problem in comparison to A*OMP, BP, SP, OMP, IHT, ISD, and SL0. As discussed above, the MILP problem is solved by running the “cplexmip” optimizer of the CPLEX optimization studio [129] from the MATLAB environment. We set a time limit of 100 seconds on CPLEX for each recovery, and terminate the optimization just after the first feasible solution is found. That is, if no solution is found in 100 seconds, the algorithm is assumed to fail². As discussed above, we set $r = 0.5$.

As for the other algorithms, A*OMP is run using the AStarOMP software, as before. The others are run using freely available software such as ℓ_1 -magic [108], Sparsify [119], Threshold-ISD [121], and the MATLAB implementation of SL0 [122]. A*OMP and OMP are run using the residue-based termination criterion with $\varepsilon = 10^{-6}$. That is, they run until $\|\mathbf{r}\|_2 \leq \varepsilon\|\mathbf{y}\|_2$, where \mathbf{r} denotes the residue of the observation \mathbf{y} . A*OMP parameters are set as $I = 3$, $B = 2$, and $P = 200$, and the adaptive-multiplicative cost model is employed with $\alpha = 0.97$. For SL0, we decrement the smoothing parameter σ slowly by 0.95 in order to reduce the risk of falling into local minima.

In the simulations, the candidate algorithms are run to recover sparse signals with different characteristics from noise-free measurements. Each test is repeated over 100 randomly generated sparse samples. The signal length is chosen as $N = 256$, while $M = 100$. The sparsity level K varies in $[10, 50]$. For each test sample, the elements of Φ are modelled as independent and identically distributed Gaussian random variables with mean zero and standard deviation $1/N$. The recovery results are expressed in terms of the average normalized mean-squared-error (ANMSE) and the exact recovery rates. The ANMSE is defined as

$$\text{ANMSE} = \frac{1}{100} \sum_{i=1}^L \frac{\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2}{\|\mathbf{x}_i\|_2^2} \quad (7.12)$$

where $\hat{\mathbf{x}}_i$ is the reconstruction of the i th test vector \mathbf{x}_i .

The tests involve sparse samples with different characteristics. The nonzero entries of these samples are selected from four different random ensembles. The nonzero entries of

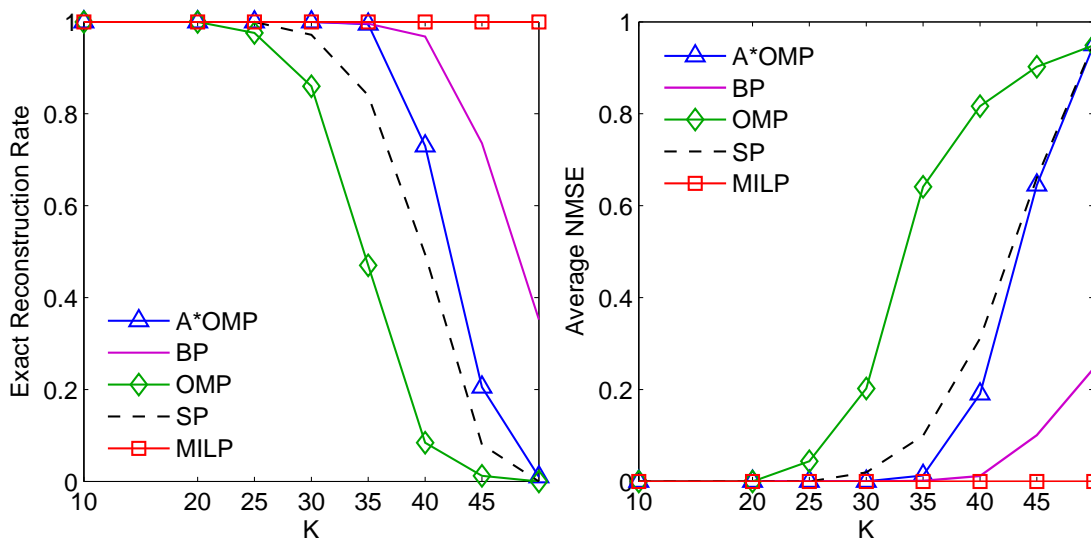
²Obviously, the MILP optimization does not actually fail. It may find the solution if the time constraint is removed, however, we would like to demonstrate a tractable application of MILP in the sparse recovery problem.

the so-called Gaussian sparse signals are drawn from the standard Gaussian distribution. Nonzero elements of uniform sparse signals are distributed uniformly in $[-1, 1]$. In addition to these, we consider two types of sparse signals with constant amplitude nonzero elements: The nonzero elements of the binary sparse signals are set to 1. Finally, the constant amplitude random sign (CARS) sparse signals involve nonzero elements with unit amplitude and random sign.

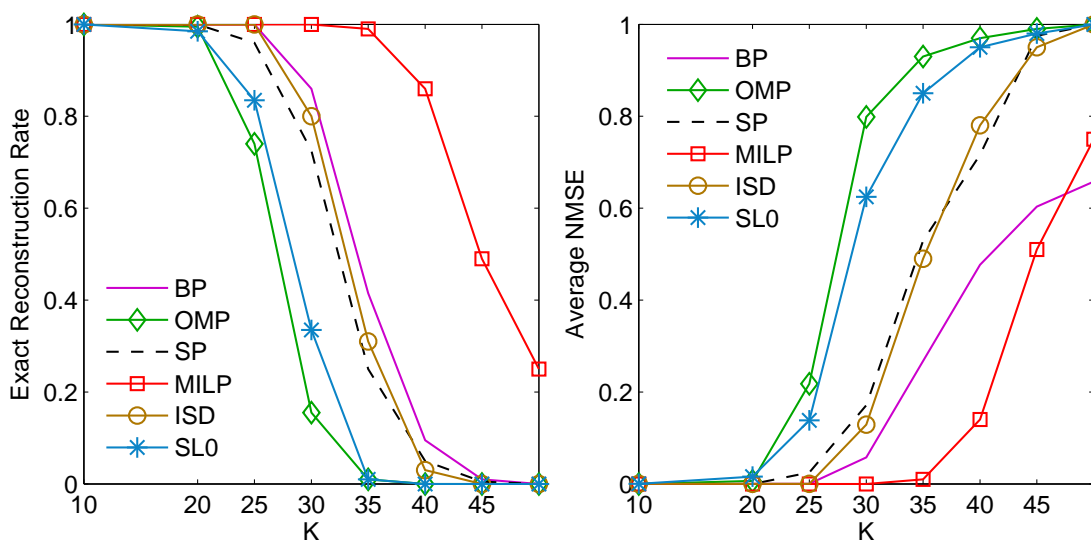
Figure 7.1(a) depicts the recovery results for binary sparse signals. For this case, we assume that the nonzero coefficients of \mathbf{x} lie in $[0, 1]$, i.e., $c_l = 0$ and $c_u = 1$. The other algorithms are also provided with some similar *a priori* information. Interestingly, we observe that once such *a priori* information is available, the proposed MILP formulation leads to the exact recovery of all binary sparse signals with sparsity level $K \in [10, 50]$. In practice, this provides a clear advantage for problems where the sparse signal is known to have nonzero elements with equal or similar values.

As for the CARS case, which is similar to the binary problem except the random sign, we set $c_l = 0$ and $c_u = 1$. Figure 7.1(b) depicts the superior recovery accuracy of MILP formulation for this case. We observe that the highest exact recovery rate is obtained by employing MILP. In addition, the ANMSE for the MILP formulation is exactly related to the exact recovery rate. That is, if MILP is able to find a solution in at most 100 seconds, this solution is correct. Otherwise, an empty solution is returned, and the normalized mean-squared-error of this solution is equal to unity. Hence, the ANMSE becomes equal to one minus the exact recovery rate of the MILP formulation. This indicates that the solution found by the MILP is exactly equal to the exact solution of the original ℓ_0 norm minimization problem, as discussed above.

The recovery results for the Gaussian and uniform sparse signals are illustrated in Figures 7.2(a) and 7.2(b). For the uniform sparse signals, we assume that the signal is known to lie in $[-1, 1]$, that is $c_l = -1$ and $c_u = 1$. For the Gaussian ensemble, we set $-c_l = c_u = \|x\|_\infty$. We observe that MILP formulation still yields the highest accuracy for the uniform sparse signals, whereas A*OMP performs very close to it. When the nonzero entries are normally distributed, A*OMP has the highest recovery accuracy, while SL0 and ISD also perform better than MILP. Clearly, the recovery accuracy of MILP degrades when the range which is spanned by the nonzero elements of the underlying sparse signals gets wider. Among the examples we considered, Gaussian sparse



(a) Binary sparse signals



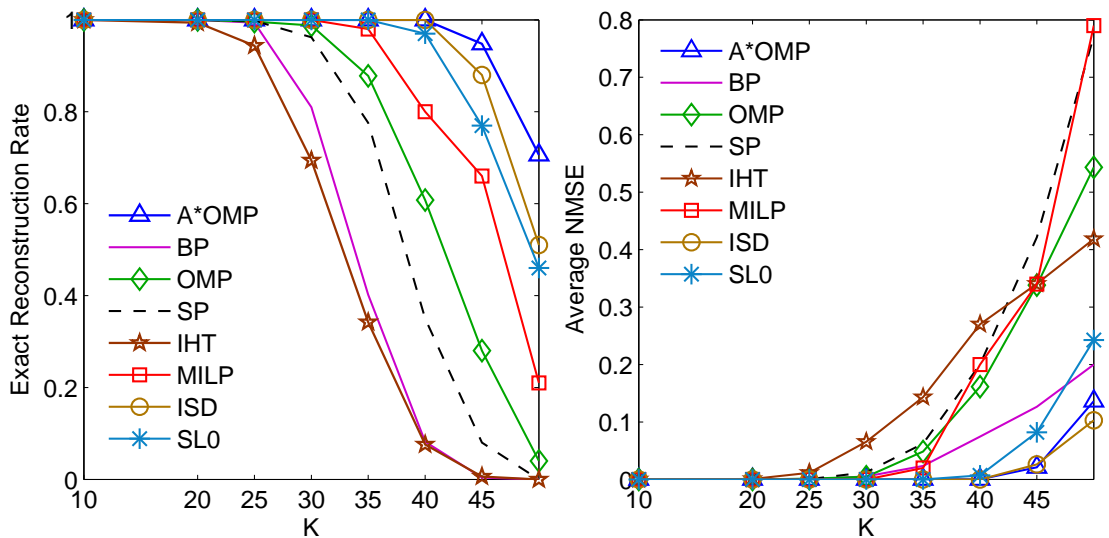
(b) CARS sparse signals

FIGURE 7.1: Average recovery results for the binary and CARS sparse signals. Each test is repeated over 100 random test samples. The signal length is 256, and the observation length is 100. The observation matrices are drawn from the Gaussian distribution.

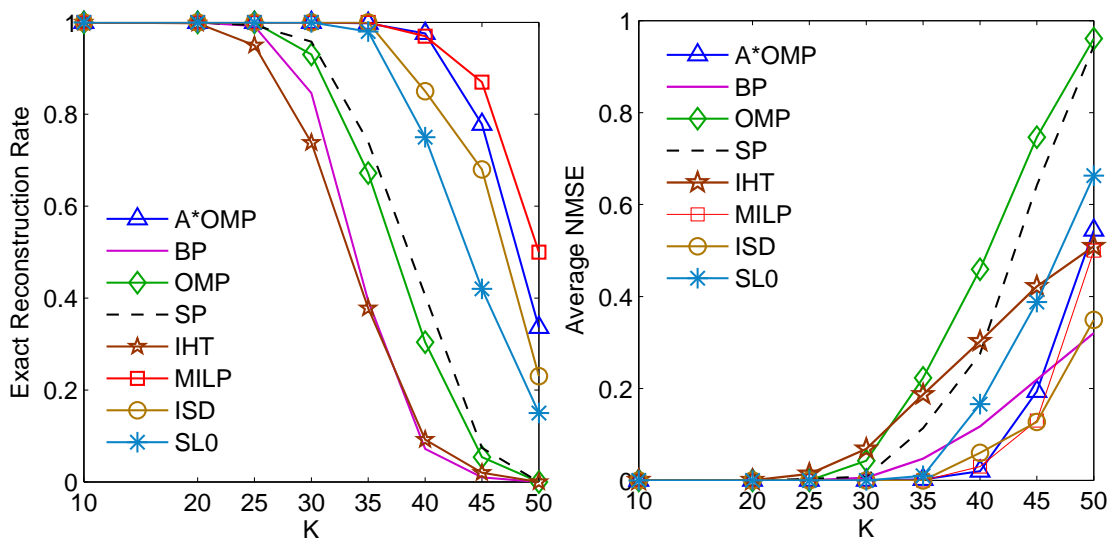
signals are ones with the widest span of nonzero elements, hence they constitute the case where MILP shows the worst performance³.

In addition to the recovery accuracy, the run times of the MILP optimization are also

³ This is again related to the constraint on the run time of the search. If the algorithm were allowed to run freely until it finds the first feasible solution, this solution would be the correct one for any type of sparse signals. However, as mentioned above, we keep the time constraint for the tractability of the proposed approach.



(a) Gaussian sparse signals



(b) Uniform sparse signals

FIGURE 7.2: Average recovery results for the uniform and Gaussian sparse signals. Each test is repeated over 100 random test samples. The signal length is 256, and the observation length is 100. The observation matrices are drawn from the Gaussian distribution.

extremely important for the evaluation of the proposed approach. In fact, most integer programming problems are naturally NP-hard. However, the average run times depicted in Table 7.1 state that the proposed formulation can be solved in reasonable time for the recovery of sparse signals having constant amplitude nonzero elements with appropriate assumptions which effectively reduce the size of the feasible solution space. We observe that the run time increases when K exceeds 40 for CARS case. This is due to the failed

recoveries, for each of which the algorithm runs for 100 seconds. For cases where MILP formulation provides exact recovery of all signals, the run times are reasonable for many applications.

TABLE 7.1: Average run time in seconds per sparse vector

	K				
	10	20	30	40	50
Binary	0.18	0.19	0.19	0.2	0.34
CARS	0.23	0.27	0.37	22.1	81.9

7.4 Conclusions and Future Work

In this chapter, we have concentrated on a new formulation for the sparse signal recovery problem. This formulation casts the problem into an equivalent MILP problem. Though MILP problems are mostly NP-hard, introduction of appropriate constraints help making it tractable for our case.

We have demonstrated the sparse signal recovery performance of the proposed approach via a number of simulation experiments involving sparse signals with different characteristics. These simulations indicate that the proposed approach yields high recovery rates when the underlying sparse signals have equal amplitude nonzero elements. Especially for the binary sparse signals we have observed that the MILP formulation yields exact recovery until $K = \frac{M}{2}$ under some appropriate assumptions. Moreover, the algorithm is reasonably fast for such signals. The recovery accuracy of the proposed approach, however, begins to degrade when the nonzero elements vary in amplitude, in which case some other candidates yield similar or better recovery accuracy. That is, the MILP optimization actually requires longer run time than the allowed time interval to find the solution in these problems, hence it returns an empty solution for a higher number of test samples in the limited time. Taking the complexity of the proposed algorithm also into account, we may conclude that the proposed approach is favorable for the recovery of sparse signals with constant or similar amplitude nonzero elements, especially the binary ones, where it provides both high recovery accuracy and reasonably high termination speed.

Before concluding, we would like to note that future work on the constraints is necessary to take the full advantage of the MILP in sparse signal recovery. Methods for finding tight bounds on the nonzero elements of the underlying sparse signals might especially be of interest. In addition, it is also worth investigating other possible constraints to further reduce the size of the feasible solution space. One example of the latter might be exploiting structured sparsity, where the size of the feasible region may be further reduced by problem-specific signal structures. In addition, the presented MILP reformulation is quite suitable to be solved with the Benders decomposition technique [130] of integer programming. Implementing this formulation with Benders decomposition may alleviate the computational burden, and hence, may decrease the run times for the search. Finally, we believe that rapid advancements in computer hardware will be a vital key for the practical use of such methods in the near future.

Chapter 8

Summary and Future Work

8.1 Summary

In this dissertation, we have concentrated on search-based methods for recovery of sparse signals from reduced sets of measurements. For this purpose, we have not only introduced novel recovery techniques, but also presented a RIP-based theoretical analysis of the well-known OMP algorithm.

In Chapter 3, we have developed an online RIP-based recovery condition for OMP with more iterations than the sparsity level K of the underlying sparse signal. Though we cannot convert this online condition into exact recovery guarantees for all K -sparse signals, we show that it might still be satisfied online despite failures among the first K iterations if the number of correct and incorrect indices in the support estimate satisfy some bounds. In comparison to the state-of-the-art exact recovery guarantees which require $6K$ to $30K$ iterations [8–10], our online recovery condition may guarantee exact recovery within $\frac{3}{2}K$ iterations when these bounds hold. Furthermore, the bound on the number of incorrect indices is also supported by histograms, showing that this bound becomes even loose in the recovery simulations performed. In addition, Chapter 3 also contains a number of empirical results which compare the recovery performance of OMP with SP and BP via phase transitions.

Chapters 4 and 5 have been devoted to the application of best-first search for the sparse signal recovery problem. For this purpose, we have introduced the A*OMP algorithm which combines the A* search technique with OMP-like extension of the tree branches in

Chapter 4. A^* search provides powerful means for solving the sparse recovery problem due to its ability to deal with different path lengths simultaneously via its auxiliary function mechanism. For this purpose, we have discussed a number of cost models which allow for effective compensation of the path length differences in our specific recovery problem. In addition, we have defined a number of pruning techniques in order to keep the size of the search tree limited. The proposed cost models and pruning techniques effectively reduce the tree size, and hence, make the search tractable in practice. This issue has been discussed in Chapter 4 in terms of the complexity-accuracy trade-off it allows for. In Chapter 5, we have presented RIP-based theoretical analysis of A^* OMP. Our fundamental theoretical finding reveals the strong recovery abilities of A^* OMP which requires a less restrictive RIP condition than OMP. Moreover, we have also compared different termination criteria, which has led us to the observation that employing the residue-based termination is more optimal than the sparsity-based one. In addition to these theoretical findings, the simulation results in Chapters 4 and 5 present an extensive empirical evaluation of A^* OMP in comparison to other mainstream recovery approaches. These results unveil the strong recovery abilities of A^* OMP, especially with the adaptive-multiplicative cost model and residue-based termination which improve both the speed and accuracy of the search.

Next, we have introduced FBP, which is a novel iterative TST-type algorithm, in Chapter 6. Similar to other TST-type algorithms, such as SP and CoSaMP, FBP also incorporates consequent forward and backward stages at each iteration. However, in contrast to SP and CoSaMP, the backward step size of FBP is not equal to the forward step size, but is smaller than it. As a result of this, FBP allows for iterative expansion of the support from scratch. This removes the need of an oracle to provide the sparsity level, which other TST algorithms require *a priori*. The simulation results, which have been presented in Chapter 6, not only illustrate the recovery performance of the proposed approach, but also discover the optimal choice for the forward and backward step sizes. Supported by the phase transitions, the forward and backward step sizes can be simply selected as a fixed ratio of the number of observations. This makes the FBP algorithm a tractable TST-scheme that can be employed trivially in practice.

Finally, we have presented a new formulation for the sparse signal recovery problem in Chapter 7. The presented formulation can be solved by MILP techniques, where the feasibility of the solution follows from addition of some reasonable constraints. In

contrast to other sparse signal recovery algorithms, the obtained MILP problem is exactly equivalent to the original sparse recovery problem. Hence, its solution is exactly equal to the desired sparsest one if a feasible solution can be found. This observation is supported by the presented simulation results, where the proposed approach either finds the correct solution in some limited time or the search is terminated without any solution. Despite the MILP problems are mostly NP-hard, these results demonstrate the tractability of the solution with proper definition of the constraints which reduce the size of the feasible solution space. To conclude, we have observed that MILP technique is especially effective for sparse signals with constant amplitude nonzero elements, where the recovery accuracy is superior to the other mainstream algorithms under some reasonable assumptions. Specifically, if the underlying sparse signals are binary, MILP provides perfect recovery when $K \leq \frac{M}{2}$.

8.2 Suggestions for Future Work

The findings of this dissertation may provide a basis for possible future work in the field regarding a number of perspectives.

First, the online recovery guarantees of OMP, developed in Chapter 3, may be both improved and generalized by future research. For the former, tighter bounds should be established on the number of correct and false indices which guarantee that the developed online condition becomes less restrictive than the K -step exact recovery condition in a particular iteration. Such tighter bounds would confirm the validity of the presented online condition for a larger portion of sparse signals. As for the latter, theoretical guarantees for the existence of support estimates satisfying the necessary conditions should be developed. That is, in case the existence of support estimates with a sufficiently high number of correct indices in addition to a sufficiently small number of false indices could be guaranteed, it would be trivial to generalize the developed online condition as an exact recovery guarantee for all K -sparse signals.

The A*OMP method, presented in Chapter 4 and Chapter 5, may also benefit from the future work suggested above for the theoretical guarantees of OMP. Since the theoretical analysis of A*OMP is very close to that of OMP, improvements on the online recovery

guarantees of OMP may also be trivially translated for A*OMP. In addition, we believe that the algorithm may also be significantly improved as part of future research regarding a number of other perspectives. First, it would be possible to improve not only the accuracy but also the speed of the search by some modifications of the path extension mechanism such as exploring not only a single node but a group of nodes, i.e., a subpath, during each extension of the best path. It might also be beneficial to decide the lengths of these subpaths online using adaptive strategies which depend on measures such as the correlation between the residue and the dictionary atoms, the residual power, or the path length. Similar adaptive strategies may also be employed for deciding the number of explored children online as well. As another possible modification, structured sparsity, which covers sparsity models such as tree-structured dictionaries, block sparsity, clustered sparse signals, etc., might be incorporated with the proposed tree search strategy to exploit specific properties of the underlying sparse problem whenever possible. For such extensions, it would be sufficient to replace the path extension mechanism of A*OMP with a model-based one. As a result of using a problem-specific path extension mechanism, both the recovery accuracy and the speed of the algorithm may be improved. Furthermore, it is also possible to incorporate any matching pursuit strategy for exploring the children of the best path instead of the OMP-like extension. Such strategies may improve the recovery speed and the recovery accuracy as well. On the other hand, the speed of the algorithm may benefit from any strategy that speeds up the correlation or orthogonal projection steps. Regarding the correlation step, one possible strategy is clustering the dictionary atoms in a tree structure as for the tree-based pursuit algorithm [37]. This strategy represents the dictionary by a tree where each inner node is a common representation of its child nodes, while the leaf nodes are themselves the dictionary atoms. Then, selection of the B best dictionary atoms can be performed iteratively from the root to the leaves by following the best B candidate nodes at each level. Finally, the proposed multiplicative and adaptive multiplicative cost models may be also employed with the A* search in other search problems as well. We believe these cost models would increase the performance and the efficiency of the A* search in many applications.

As for the FBP method of Chapter 6, theoretical exact recovery guarantees still remain open as an important future research direction. As part of the future work, the algorithm may also significantly benefit from varying the step sizes online, which could

improve not only the speed but also the accuracy of the FBP recovery. For example, the backward step size might be selected very small for the first iterations, and then iteratively increased during the recovery process until it is comparable to the forward step size. Another, perhaps more sophisticated, strategy might be adaptive selection of the step sizes throughout the recovery. One possible means for this purpose is exploiting the correlations of the dictionary atoms to the residue. In addition to these, FBP may also easily benefit from structured sparsity as for A*OMP whenever possible. Moreover, any strategy that speeds up the correlation or the orthogonal projection step would speed up FBP as well.

We also foresee a number of interesting future research directions regarding the MILP formulation developed in Chapter 7. First of all, future work on the constraints, which are very important for the termination speed of the algorithm, is necessary to take the full advantage of the MILP optimization in the sparse signal recovery problem. Among the constraints, methods for estimating tight bounds on the values of the nonzero elements of the underlying sparse signals would be of great interest. In addition, it is also worth investigating other possible constraints to further reduce the size of the feasible solution space. One example of the latter is exploiting structured sparsity whenever possible. In such cases, the size of the feasible region may be significantly reduced by appropriate constraints exploiting problem-specific signal structures. Finally, the presented MILP reformulation is quite suitable to be solved with the Benders decomposition technique [130] of integer programming. Implementing this formulation with the Benders decomposition may alleviate the computational burden, and hence, may decrease the run times for the search.

To conclude, there exists a number of possible future research directions based on the findings of this dissertation. Hence, we believe that our findings will play an important role for the future work in the field.

Bibliography

- [1] E. Candès and T. Tao, “Decoding by linear programming,” *IEEE Trans. Inf. Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.
- [2] E. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.
- [3] D. L. Donoho, “Compressed sensing,” *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.
- [4] E. Candès and T. Tao, “Near-optimal signal recovery from random projections: universal encoding strategies?,” *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5406–5425, 2006.
- [5] J. A. Tropp and S. J. Wright, “Computational methods for sparse solution of linear inverse problems,” *Proc. IEEE*, vol. 98, no. 6, pp. 948–958, June 2010.
- [6] Y. C. Pati, R. Rezaifar, and P. S. Krishnaprasad, “Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition,” in *Proc. 27th Asilomar Conference on Signals, Systems and Computers*, Los Alamitos, CA, 1993, vol. 1, pp. 40–44.
- [7] J. Wang and B. Shim, “On the recovery limit of sparse signals using orthogonal matching pursuit,” *IEEE Trans. Signal Process.*, vol. 60, no. 9, pp. 4973–4976, Sept. 2012.
- [8] T. Zhang, “Sparse recovery with orthogonal matching pursuit under RIP,” *IEEE Trans. Inf. Theory*, vol. 57, no. 9, pp. 6215–6221, Sept. 2011.

- [9] S. Foucart, “Stability and robustness of weak orthogonal matching pursuits,” *Recent Advances in Harmonic Analysis and Applications, Springer Proceedings in Mathematics & Statistics*, vol. 25, pp. 395–405.
- [10] J. Wang and B. Shim, “Improved recovery bounds of orthogonal matching pursuit using restricted isometry property,” 2012, submitted to *IEEE Trans. Inform. Theory*, available as <http://arxiv.org/abs/1211.4293>.
- [11] S. Koenig, M. Likhachev, Y. Liu, and D. Furcy, “Incremental heuristic search in AI,” *AI Mag.*, vol. 25, no. 2, pp. 99–112, 2004.
- [12] R. Dechter and J. Pearl, “Generalized best-first search strategies and the optimality of A^* ,” *J. ACM*, vol. 32, no. 3, pp. 505–536, 1985.
- [13] F. Jelinek, *Statistical Methods For Speech Recognition*, chapter 6, MIT Press, Cambridge, MA, USA, 1997.
- [14] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths,” *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 12, pp. 100–107, July 1968.
- [15] P. E. Hart, N. J. Nilsson, and B. Raphael, “Correction to a formal basis for the heuristic determination of minimum cost paths,” *SIGART Newsletter*, , no. 37, pp. 28–29, 1972.
- [16] A. Maleki and D. L. Donoho, “Optimally tuned iterative reconstruction algorithms for compressed sensing,” *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, no. 2, pp. 330–341, Apr. 2010.
- [17] W. Dai and O. Milenkovic, “Subspace pursuit for compressive sensing signal reconstruction,” *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2230–2249, May 2009.
- [18] D. Needell and J. A. Tropp, “CoSaMP: Iterative signal recovery from incomplete and inaccurate samples,” *Appl. Comp. Harmonic Anal.*, vol. 26, pp. 301–321, 2008.
- [19] E. Candès, M. Rudelson, T. Tao, and R. Vershynin, “Error correction via linear programming,” in *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, Pittsburgh, PA, Oct. 2005, pp. 295–308.

- [20] J. A. Tropp and A. C. Gilbert, "Signal recovery from partial information via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [21] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde, "Model-based compressive sensing," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1982–2001, Apr. 2010.
- [22] G. Peyre, "Best basis compressed sensing," *IEEE Trans. Signal Process.*, vol. 58, no. 5, pp. 2613–2622, May 2010.
- [23] A. Kyrillidis, G. Puy, and V. Cevher, "Hard thresholding with norm constraints," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, Mar. 2012, pp. 3645–3648.
- [24] D. Malioutov and M. Malyutov, "Boolean compressed sensing: Lp relaxation for group testing," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, Mar. 2012, pp. 3305–3308.
- [25] A. Y. Carmi, L. Mihaylova, and D. Kanevsky, "Unscented compressed sensing," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, Mar. 2012, pp. 5249–5252.
- [26] H. Mansour and Ö. Yilmaz, "Support driven reweighted ℓ_1 minimization," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, Mar. 2012, pp. 3309–3312.
- [27] D. Needell and R. Vershynin, "Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit," *Found Comput Math*, vol. 9, no. 3, pp. 317–334, June 2009.
- [28] B. Shim, J. Wang, and S. Kwon, "Generalized orthogonal matching pursuit," *IEEE Trans. Signal Process.*, vol. PP, no. 99, 2012.
- [29] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations," *J. Fourier Anal. Appl.*, vol. 14, pp. 629–654, 2008.
- [30] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Appl. Comp. Harmonic Anal.*, vol. 27, no. 3, pp. 265–274, 2009.

- [31] V. Cevher and S. Jafarpour, “Fast hard thresholding with Nesterov’s gradient method,” in *Neuronal Information Processing Systems, Workshop on Practical Applications of Sparse Modeling*, Whistler, Canada, 2010.
- [32] R. Garg and R. Khandekar, “Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property,” in *Proceedings of the 26th Annual International Conference on Machine Learning*, New York, NY, USA, 2009, ICML ’09, pp. 337–344, ACM.
- [33] I. Daubechies, M. Defrise, and C. De Mol, “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint,” *Communications on Pure and Applied Mathematics*, vol. 57, no. 11, pp. 1413–1457, Nov. 2004.
- [34] T. Blumensath, “Accelerated iterative hard thresholding,” *Signal Processing*, vol. 92, no. 3, pp. 752–756, Mar. 2012.
- [35] T. Blumensath and M. E. Davies, “Gradient pursuits,” *IEEE Trans. Signal Process.*, vol. 56, no. 6, pp. 2370–2382, June 2008.
- [36] T. Blumensath and M. E. Davies, “Stagewise weak gradient pursuits,” *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4333–4346, Nov. 2009.
- [37] P. Jost, P. Vandergheynst, and P. Frossard, “Tree-based pursuit: Algorithm and properties,” *IEEE Trans. Signal Process.*, vol. 54, no. 12, pp. 4685–4697, Dec. 2006.
- [38] S. F. Cotter and B. D. Rao, “Application of tree-based searches to matching pursuit,” in *Acoustics, Speech and Signal Processing (ICASSP), 2001 IEEE International Conference on*, 2001, pp. 3933–3936.
- [39] G. Z. Karabulut, L. Moura, D. Panario, and A. Yongacoğlu, “Integrating flexible tree searches to orthogonal matching pursuit algorithm,” *Vision, Image and Signal Processing, IEE Proceedings -*, vol. 153, no. 5, pp. 538–548, Oct. 2006.
- [40] P. Schniter, L. C. Potter, and J. Ziniel, “Fast Bayesian matching pursuit,” in *Information Theory and Applications Workshop, 2008*, Feb. 2008, pp. 326–333.
- [41] R. Chartrand, “Nonconvex compressed sensing and error correction,” in *Acoustics, Speech and Signal Processing (ICASSP), 2007 IEEE International Conference on*, 2007, vol. 3.

- [42] R. Chartrand, “Exact reconstruction of sparse signals via nonconvex minimization,” *IEEE Signal Process. Lett.*, vol. 14, no. 10, pp. 707–710, 2007.
- [43] R. Chartrand, “Fast algorithms for nonconvex compressive sensing: MRI reconstruction from very few data,” in *Proc. IEEE Int. Symp. Biomedical Imaging: From Nano to Macro ISBI '09*, 2009, pp. 262–265.
- [44] R. Chartrand and W. Yin, “Iteratively reweighted algorithms for compressive sensing,” in *Acoustics, Speech and Signal Processing (ICASSP), 2008 IEEE International Conference on*, 2008, pp. 3869–3872.
- [45] E. Candès, M. Wakin, and S. Boyd, “Enhancing sparsity by reweighted ℓ_1 minimization,” *Journal of Fourier Analysis and Applications*, vol. 14, pp. 877–905, 2008.
- [46] Y. Wang and W. Yin, “Sparse signal reconstruction via iterative support detection,” *SIAM Journal on Imaging Sciences*, vol. 3, no. 3, pp. 462–491, 2010.
- [47] I. Daubechies, R. DeVore, M. Fornasier, and C. S. Güntürk, “Iteratively reweighted least squares minimization for sparse recovery,” *Comm. Pure Appl. Math.*, vol. 63, no. 1, pp. 1–38, Jan. 2010.
- [48] Shihao Ji, Ya Xue, and L. Carin, “Bayesian compressive sensing,” *IEEE Trans. Signal Process.*, vol. 56, no. 6, pp. 2346–2356, June 2008.
- [49] S. D. Babacan, R. Molina, and A. K. Katsaggelos, “Bayesian compressive sensing using Laplace priors,” *IEEE Trans. Image Process.*, vol. 19, no. 1, pp. 53–63, 2010.
- [50] M. F. Duarte, M. B. Wakin, and R. G. Baraniuk, “Fast reconstruction of piecewise smooth signals from random projections,” in *Online Proceedings of the Workshop on Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, Rennes, France, 2005.
- [51] C. La and M. N. Do, “Tree-based orthogonal matching pursuit algorithm for signal reconstruction,” in *Image Processing, 2006 IEEE International Conference on*, Oct. 2006, pp. 1277–1280.
- [52] M. F. Duarte, M. B. Wakin, and R. G. Baraniuk, “Wavelet-domain compressive signal reconstruction using a hidden Markov tree model,” in *Acoustics, Speech and*

- Signal Processing (ICASSP), 2008 2012 IEEE International Conference on*, 2008, pp. 5137–5140.
- [53] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, “Algorithms for simultaneous sparse approximation. Part I: Greedy pursuits,” *Signal Processing*, vol. 86, no. 3, pp. 572–588, Mar. 2006.
- [54] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, “Simultaneous sparse approximation via greedy pursuit,” in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP '05)*, Mar. 2005, vol. 5.
- [55] Y. C. Eldar and M. Mishali, “Robust recovery of signals from a structured union of subspaces,” *IEEE Trans. Inf. Theory*, vol. 55, no. 11, pp. 5302–5316, 2009.
- [56] Y. C. Eldar and M. Mishali, “Block sparsity and sampling over a union of subspaces,” in *Proc. 16th Int Digital Signal Processing Conf*, 2009, pp. 1–8.
- [57] M. Stojnic, F. Parvaresh, and B. Hassibi, “On the reconstruction of block-sparse signals with an optimal number of measurements,” *IEEE Trans. Signal Process.*, vol. 57, no. 8, pp. 3075–3085, 2009.
- [58] D. Baron, M. F. Duarte, S. Sarvotham, M. B. Wakin, and R. G. Baraniuk, “An information theoretic approach to distributed compressed sensing,” in *Allerton Conference on Communication, Control, and Computing*, Allerton, IL, Sept. 2005.
- [59] D. Baron, M. B. Wakin, M. F. Duarte, S. Sarvotham, and R. G. Baraniuk, “Distributed compressed sensing,” Tech. Rep., Rice University, Department of Electrical and Computer Engineering, 2005.
- [60] M. B. Wakin, S. Sarvotham, M. F. Duarte, D. Baron, and R. G. Baraniuk, “Recovery of jointly sparse signals from few random projections,” in *Proceedings of the Workshop on Neural Information Processing Systems (NIPS)*, Vancouver, Canada, Dec. 2005, pp. 1435–1442.
- [61] M. F. Duarte, S. Sarvotham, D. Baron, M. B. Wakin, and R. G. Baraniuk, “Distributed compressed sensing of jointly sparse signals,” in *Proceedings of the 39th Asilomar Conference on Signals, Systems and Computation*, Pacific Grove, CA, Nov. 2005, pp. 1537–1541.

- [62] V. Cevher, M. F. Duarte, C. Hegde, and R. G. Baraniuk, “Sparse signal recovery using Markov random fields,” in *Proceedings of the Workshop on Neural Information Processing Systems (NIPS)*, Vancouver, Canada, Dec. 2008.
- [63] C. Hegde, M. F. Duarte, and V. Cevher, “Compressive sensing recovery of spike trains using a structured sparsity model,” in *Workshop on Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, 2009.
- [64] V. Cevher, P. Indyk, C. Hegde, and R. G. Baraniuk, “Recovery of clustered sparse signals from compressive measurements,” in *International conference on Sampling Theory and Applications (SAMPTA)*, 2009.
- [65] M. F. Duarte, V. Cevher, and R. G. Baraniuk, “Model-based compressive sensing for signal ensembles,” in *Proceedings of the 47th annual Allerton conference on Communication, control, and computing*, 2009, Allerton’09, pp. 244–250.
- [66] S. Chen, D. L. Donoho, and M. Saunders, “Atomic decomposition by basis pursuit,” *SIAM J. on Sci. Comp.*, vol. 20, no. 1, pp. 33–61, 1998.
- [67] D. L. Donoho and M. Elad, “Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization,” in *Proc. Natl. Acad. Sci.*, Los Alamitos, CA, Mar. 2003, vol. 100, pp. 2197–2202.
- [68] S. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
- [69] D. L. Donoho and X. Huo, “Uncertainty principles and ideal atomic decomposition,” *Information Theory, IEEE Transactions on*, vol. 47, no. 7, pp. 2845–2862, Nov. 2001.
- [70] J.-L. Starck, M. Elad, and D. L. Donoho, “Image decomposition via the combination of sparse representations and a variational approach,” *Image Processing, IEEE Transactions on*, vol. 14, no. 10, pp. 1570–1582, Oct. 2005.
- [71] J.-L. Starck, M. Elad, and D. L. Donoho, “Redundant multiscale transforms and their application for morphological component separation,” in *Advances in Imaging and Electron Physics*, vol. 132 of *Advances in Imaging and Electron Physics*, pp. 287–348. Elsevier, 2004.

- [72] C. Jutten H. Mohimani, M. Babaie-Zadeh, “A fast approach for overcomplete sparse decomposition based on smoothed ℓ_0 -norm,” *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 289–301, Jan. 2009.
- [73] M. Rudelson and R. Vershynin, “Geometric approach to error-correcting codes and reconstruction of signals,” Tech. Rep., Dept. of Mathematics, Univ. of California at Davis, 2005.
- [74] N. Ouarti and G. Peyre, “Best basis denoising with non-stationary wavelet packets,” in *Proc. 16th IEEE Int Image Processing (ICIP) Conf*, 2009, pp. 3825–3828.
- [75] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *Image Processing, IEEE Transactions on*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.
- [76] A. K. Fletcher, S. Rangan, V. K. Goyal, and K. Ramchandran, “Analysis of denoising by sparse approximation with random frame asymptotics,” in *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, Sept. 2005, pp. 1706–1710.
- [77] A. K. Fletcher, S. Rangan, V. K. Goyal, and K. Ramchandran, “Denoising by sparse approximation: Error bounds based on rate-distortion theory,” *EURASIP J. Appl. Signal Process.*, pp. 1–19, 2006.
- [78] J. A. Tropp, “Just relax: Convex programming methods for identifying sparse signals in noise,” *Information Theory, IEEE Transactions on*, vol. 52, no. 3, pp. 1030–1051, march 2006.
- [79] M. Elad and I. Yavneh, “A plurality of sparse representations is better than the sparsest one alone,” *IEEE Trans. Inf. Theory*, vol. 55, no. 10, pp. 4701–4714, Oct. 2009.
- [80] M. E. Tipping, “Sparse Bayesian learning and the relevance vector machine,” *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, September 2001.
- [81] D. P. Wipf and S. Nagarajan, “A new view of automatic relevance determination,” in *Advances in Neural Information Processing Systems 20*, J.C. Platt, D. Koller, Y. Singer, and S. Roweis, Eds., pp. 1625–1632. MIT Press, Cambridge, MA, 2008.

- [82] D. Wipf and S. Nagarajan, “Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions,” *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, no. 2, pp. 317–329, 2010.
- [83] D. P. Wipf, B. D. Rao, and S. Nagarajan, “Latent variable Bayesian models for promoting sparsity,” *IEEE Trans. Inf. Theory*, vol. 57, no. 9, pp. 6236–6255, Sept. 2011.
- [84] T. Zhang, “Adaptive forward-backward greedy algorithm for learning sparse representations,” *IEEE Trans. Inf. Theory*, vol. 57, no. 7, pp. 4689–4708, July 2011.
- [85] M. E. Tipping and A. Faul, “Fast marginal likelihood maximisation for sparse Bayesian models,” in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.
- [86] D. P. Wipf and B. D. Rao, “Sparse Bayesian learning for basis selection,” *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2153–2164, 2004.
- [87] D. P. Wipf and B. D. Rao, “An empirical Bayesian strategy for solving the simultaneous sparse approximation problem,” *IEEE Trans. Signal Process.*, vol. 55, no. 7, pp. 3704–3716, 2007.
- [88] M. A. T. Figueiredo, “Adaptive sparseness for supervised learning,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 9, pp. 1150–1159, 2003.
- [89] A. Cohen, W. Dahmen, and R. Devore, “Compressed sensing and best k-term approximation,” *Journal of the American Mathematical Society*, vol. 22, no. 01, pp. 211–231, Jan. 2009.
- [90] T. T. Cai and T. Jiang, “Limiting laws of coherence of random matrices with applications to testing covariance structure and construction of compressed sensing matrices,” *Ann. Statist.*, vol. 39, no. 3, pp. 1496–1525, 2011.
- [91] J. A. Tropp, “Greed is good: Algorithmic results for sparse approximation,” *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.
- [92] J. A. Tropp, “Norms of random submatrices and sparse approximation,” *Comptes Rendus Mathematique*, vol. 346, pp. 1271–1274, Dec. 2008.

- [93] D. L. Donoho, “For most large underdetermined systems of linear equations, the minimal ℓ_1 -norm solution is also the sparsest solution,” *Comm. Pure Appl. Math.*, vol. 59, no. 6, pp. 797829, 2006.
- [94] A. Fletcher and S. Rangan, “Orthogonal matching pursuit from noisy random measurements: A new analysis,” in *Advances in Neural Information Processing Systems 23*, Vancouver, 2009.
- [95] M. A. Davenport and M. B. Wakin, “Analysis of orthogonal matching pursuit using the restricted isometry property,” *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4395–4401, sept. 2010.
- [96] M. Rudelson and R. Vershynin, “Sparse reconstruction by convex relaxation: Fourier and Gaussian measurements,” in *40th Annual Conference on Information Sciences and Systems*, 2006, pp. 207–212.
- [97] J. D. Blanchard, C. Cartis, and J. Tanner, “Compressed sensing: How sharp is the restricted isometry property?,” *SIAM Rev.*, vol. 53, no. 1, pp. 105–125.
- [98] B. Bah and J. Tanner, “Improved bounds on restricted isometry constants for Gaussian matrices,” *SIAM. J. Matrix Anal. & Appl.*, vol. 31, no. 5, pp. 2882–2898.
- [99] S. Park and H. N. Lee, “On the derivation of RIP for random Gaussian matrices and binary sparse signals,” in *ICT Convergence (ICTC), 2011 International Conference on*, Sept. 2011, pp. 120–124.
- [100] E. Candès, “The restricted isometry property and its implications for compressed sensing,” *Comptes Rendus Mathématique*, vol. 346, no. 9-10, pp. 589–592, May 2008.
- [101] D. Needell and R. Vershynin, “Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit,” *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, pp. 310–316, Apr. 2010.
- [102] D. L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck, “Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit (stomp),” Tech. Rep., Statistics Dept., Stanford Univ., Mar. 2006.

- [103] Jian Wang and Byonghyo Shim, “Exact reconstruction of sparse signals via generalized orthogonal matching pursuit,” in *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on*, Nov. 2011, pp. 1139–1142.
- [104] Y. Nesterov, “Smooth minimization of non-smooth functions,” *Mathematical Programming*, vol. 103, pp. 127–152, 2005.
- [105] R. Tibshirani, “Regression shrinkage and selection via the LASSO,” *J. Royal Statist. Soc. B*, vol. 58, pp. 267–288, 1996.
- [106] M. A. Saunders, “PDCO: Primal-dual interior-point method for convex objectives,” Tech. Rep., Stanford University, Systems Optimization Laboratory, Nov. 2002.
- [107] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, “An interior-point method for large-scale ℓ_1 -regularized least squares,” *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, pp. 606–617, 2007.
- [108] E. Candès and J. Romberg, “ ℓ_1 -magic: Recovery of sparse signals via convex programming,” Tech. Rep., California Institute of Technology, Oct. 2005.
- [109] M. R. Osborne, B. Presnell, and B. Turlach, “A new approach to variable selection in least squares problems,” *IMA J. Numer. Anal.*, vol. 20, pp. 389–403, 2000.
- [110] D. L. Donoho and Y. Tsaig, “Fast solution of ℓ_1 -norm minimization problems when the solution may be sparse,” *IEEE Trans. Inf. Theory*, vol. 54, no. 11, pp. 4789–4812, Nov. 2008.
- [111] S. Foucart and M.-J. Lai, “Sparsest solutions of underdetermined linear systems via ℓ_q -minimization for $0 < q \leq 1$,” *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 395–407, 2009.
- [112] A. Blake and A. Zisserman, *Visual Reconstruction*, MIT Press, 1987.
- [113] S. Ashkiani, M. Babaie-Zadeh, and C. Jutten, “Error correction via smoothed ℓ_0 -norm recovery,” in *Statistical Signal Processing Workshop (SSP), 2011 IEEE*, June 2011, pp. 289–292.

- [114] M. F. Duarte and Y. C. Eldar, “Structured compressed sensing: From theory to applications,” *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4053–4085, Sept. 2011.
- [115] H. V. Pham, Dai W., and O. Milenkovic, “Sublinear compressive sensing reconstruction via belief propagation decoding,” in *Information Theory (ISIT) 2009, IEEE International Symposium on*, July 2009, pp. 674–678.
- [116] N. B. Karahanoglu and H. Erdoğan, “A* orthogonal matching pursuit: Best-first search for compressed sensing signal recovery,” *Digital Signal Processing*, vol. 22, no. 4, pp. 555–568, July 2012.
- [117] N. B. Karahanoglu and H. Erdoğan, “Compressed sensing signal recovery via A* orthogonal matching pursuit,” in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, May 2011.
- [118] N. B. Karahanoglu, “AStarOMP software package,” <http://myweb.sabanciuniv.edu/karahanoglu/research/>.
- [119] T. Blumensath, “Sparsify software package,” <http://users.fmrib.ox.ac.uk/~tblumens/sparsify/sparsify.html>.
- [120] N. B. Karahanoglu and H. Erdoğan, “A comparison of termination criteria for A*OMP,” in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, Aug. 2012.
- [121] “Threshold-ISD software package,” <http://www.caam.rice.edu/~optimization/L1/ISD/>.
- [122] “SL0 matlab code,” <http://ee.sharif.edu/~SLzero/>.
- [123] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, “A simple proof of the restricted isometry property for random matrices,” *Constr. Approx.*, vol. 28, pp. 253263, 2008.
- [124] S. Mendelson, A. Pajor, and N. Tomczak-Jaegermann, “Uniform uncertainty principle for bernoulli and subgaussian ensembles,” *Constr. Approx.*, vol. 28, pp. 277289, 2008.

-
- [125] M. Rudelson and R. Vershynin, “On sparse reconstruction from Fourier and Gaussian measurements,” *Comm. Pure Appl. Math.*, vol. 61, pp. 10251045, 2008.
- [126] N. B. Karahanoglu and H. Erdođan, “Forward-backward search for compressed sensing signal recovery,” in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, Aug. 2012.
- [127] R. Berinde, A. C. Gilbert, P. Indyk, H. Karloff, and M. J. Strauss, “Combining geometry with combinatorics: A unified approach to sparse signal recovery,” in *Proc Allerton Conf. Commun. Control, Comput.*, 2008.
- [128] D. L. Donoho and J. Tanner, “Counting the faces of randomly-projected hypercubes and orthants, with applications,” *Discrete Comput. Geom.*, vol. 43, no. 3, pp. 522–541, Apr. 2010.
- [129] “IBM ILOG CPLEX Optimization Studio V12.4,” <http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/>.
- [130] R. K. Martin, *Large Scale Linear and Integer Optimization: A Unified Approach*, Kluwer Academic Publishers, Boston, 1999.