

REPORT DOCUMENTATION PAGE				<i>Form Approved</i> OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) OCTOBER 2018		2. REPORT TYPE PhD DISSERTATION (Post Print)		3. DATES COVERED (From - To) SEP 2016 – SEP 2018	
4. TITLE AND SUBTITLE COMPRESSIVE SENSING MATRIX DESIGN AND EVALUATION WITH COGNITIVE RADIO APPLICATIONS				5a. CONTRACT NUMBER IN-HOUSE	
				5b. GRANT NUMBER N/A	
				5c. PROGRAM ELEMENT NUMBER 62788F	
6. AUTHOR(S) John J. Kelly				5d. PROJECT NUMBER G2TP	
				5e. TASK NUMBER MI	
				5f. WORK UNIT NUMBER PR	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/Information Directorate Rome Research Site/RIGC 525 Brooks Road Rome NY 13441-4505				8. PERFORMING ORGANIZATION REPORT NUMBER N/A	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/Information Directorate Rome Research Site/RIGC 525 Brooks Road Rome NY 13441-4505				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/RI	
				11. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-RI-RS-TP-2019-002	
12. DISTRIBUTION AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA Case Number: 88ABW-2018-5393. DATE CLEARED: 25 OCT 2018					
13. SUPPLEMENTARY NOTES Author final manuscript. Pub in ProQuest. 25 Oct 2018. 190 pages. Binghamton University. This work is copyrighted. This is a work of the United States government and is not subject to copyright protection in the United States.					
14. ABSTRACT This dissertation is aimed at evaluating a sensing matrix in an overcomplete compressive sensing (CS) system. The main theoretical problems addressed are: Given a redundant (overcomplete)dictionary, a sensing matrix, and a greedy approximation method such as orthogonal matching pursuit, what tools can ascertain the suitability of a given sensing matrix, in the course of performing CS with the dictionary? In general, how does the relationship between the sensing matrix and dictionary (regarding structure/composition) affect the greedy support estimation of a sparse signal?					
15. SUBJECT TERMS compressed sensing, sensing matrix, overcomplete dictionary, greedy approximation, dynamic spectrum access, spectrum sensing, sensor data falsification, spectrum falsification detection					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 189	19a. NAME OF RESPONSIBLE PERSON JOHN J. KELLY
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (Include area code) N/A

COMPRESSIVE SENSING MATRIX DESIGN AND EVALUATION
WITH COGNITIVE RADIO APPLICATIONS

BY

JOHN J. KELLY

BS, SUNY Institute of Technology at Utica/Rome, 1998
MS, SUNY at Binghamton, 2011

DISSERTATION

Submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate School of
Binghamton University
State University of New York
2019

DISTRIBUTION A.

Approved for Public Release: Distribution Unlimited. Case
Number 88ABW-2018-5393 2018 10 25

Accepted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate School of
Binghamton University
State University of New York
2019

February 8, 2019

Mark Fowler, Chair and Faculty Advisor
Department of Electrical and Computer Engineering, Binghamton University

Scott Craver, Member
Department of Electrical and Computer Engineering, Binghamton University

Jessica Fridrich, Member
Department of Electrical and Computer Engineering, Binghamton University

Xiaohua (Edward) Li, Member
Department of Electrical and Computer Engineering, Binghamton University

Harold Lewis III, Outside Examiner
Department of Systems Science and Industrial Engineering, Binghamton University

Abstract

This dissertation is aimed at evaluating a sensing matrix in an overcomplete compressive sensing (CS) system. The main theoretical problems addressed are: Given a redundant (overcomplete) dictionary, a sensing matrix, and a greedy approximation method such as orthogonal matching pursuit, what tools can ascertain the suitability of a given sensing matrix, in the course of performing CS with the dictionary? In general, how does the relationship between the sensing matrix and dictionary (regarding structure/composition) affect the greedy support estimation of a sparse signal?

An abundance of published work describes CS matrices constructed randomly from certain classes of distributions. With high probability, these CS matrices preserve sparse signal information, for signals formed from a linear combination of a few columns from an *orthonormal* dictionary matrix. For a given non-orthonormal overcomplete dictionary, this research applies a known non-iterative parametric equation to construct a class of optimal sensing matrices. The resulting (non-unique) optimal compressive sensing matrix linearly transforms the dictionary columns to be maximally decorrelated, in an aggregate sense. This research is directed at *evaluating* a given sensing matrix with respect to an optimal sensing matrix.

The approach here complements existing *sensing matrix evaluation* literature, and is focused on comparing Gram matrices (a key tool of linear algebra). While a few Gram matrix properties have been used for evaluating CS matrices, e.g. the max/mean Gram entry, much of the geometric information encoded within the Gram has previously gone unused. Here, additional aspects of the Gram matrix (e.g. the distribution of its entries) are developed and utilized. Distances between Gram matrix pairs formed from perturbed versus unperturbed dictionaries are computed, and then related to the resulting effect on the CS support estimation problem.

One application for sensing matrix evaluation is facilitating spectrum sharing in cognitive radio (CR) networks. The feasibility of CS matrix optimization for a spectrum dictionary to efficiently approximate spectrum from a potential CR user is explored, as a means to detect spectrum falsification (a significant problem in certain CR networks). Gram matrix comparison methods are applied to ascertain the effect of changes in the spectrum dictionary over time.

Dedication

To my father who ignited in me a deep love of wisdom and learning,
to my mother who taught me how to dream big and continues to inspire me in countless ways,
and to my wife whose love and support have never faltered.

Acknowledgments

I foremost wish to express my sincere gratitude to my advisor, Dr. Mark L. Fowler. His guidance and encouragement throughout my graduate studies have been invaluable.

Additionally, I am indebted to committee members Dr. Scott Craver, Dr. Jessica Fridrich, and Dr. Xiaohua (Edward) Li for the time and effort they put into teaching their classes, as well as for their careful reading and evaluation of this dissertation. Thank you to the helpful librarians, supportive colleagues, and to all my teachers and coaches I've ever had; without all of you, this dissertation would never been possible.

Contents

List of Tables	xiii
List of Figures	xiv
1 Introduction	1
1.1 Application: Secure Spectrum Sensing in CRNs	1
1.2 Solution Approach: Sparse and Compressive Approximation	2
1.3 Novel Contributions	3
1.4 Dissertation Organization	4
2 Linear Signal Approximation Preliminaries	7
2.1 Signal Spaces	7
2.2 Linear Transformations	7
2.3 Orthonormal Basis Dictionaries	7
2.4 Overcomplete Dictionary Representations	8
2.5 Linear Transformations used in this Work	9
3 Signal Approximation Models	11
3.1 Introduction to General Linear Models	11
3.2 Sparse Approximation	12
3.3 Compressive Sensing	13
3.3.1 CS Problem Formulation	13
3.3.2 Sensing Matrices	15
3.4 A Geometric Analysis of Sensing Matrices and Dictionaries	16
3.4.1 Dictionary Atoms as Vectors	16
3.4.2 Coherence	16
3.4.3 Significance of Dictionary Atom Separation	19

3.4.4	Distance or Affinity Matrices	21
3.4.5	Measuring Atom Separation via the Gram Matrix	22
3.5	Greedy Recovery Methods	23
3.5.1	Introduction	23
3.5.2	Orthogonal Matching Pursuit	26
3.5.3	Inner Product Correlation	28
4	Sensing Matrix Optimization for Compressive Sensing	30
4.1	Introduction	30
4.2	Class of Optimal Effective Dictionaries	31
4.3	Class of Optimal Sensing Matrices	33
4.4	Updating the Optimal Sensing Matrix in Response to Dictionary Changes	36
4.5	Related Methods and Topics	37
5	Sensing Matrix Evaluation	39
5.1	Introduction	39
5.1.1	Purpose	39
5.1.2	Chapter Organization	40
5.2	Optimized and Stale Effective Dictionary Nomenclature	40
5.3	Constructing Stale Effective Dictionaries	41
5.4	Problem Formulation and the Difference Operator	44
5.5	Defining Three Approaches to Sensing Matrix Evaluation	45
5.5.1	Approach 1: Sensing Matrix Optimization Objective Function	46
5.5.2	Approach 2: Distances between Sets of Gram Matrix Entries	48
5.5.3	Approach 3: Distances between Gram Matrices	51
5.6	Focus on Approach 2: Distance Measures between Sets of Gram Matrix Entries	52
5.6.1	Probability Mass and Distribution Functions	52
5.6.2	PMF of Gram Matrix Entries: A Geometric Interpretation	53
5.6.3	EDF of Gram Matrix Entries: A Geometric Interpretation	53
5.6.4	Overview of Statistical Distances	57
5.6.5	Kolmogorov-Smirnov Test	57
5.6.6	Applying the KS Test to Sensing Matrix Evaluation	58

5.6.7	Kullback-Leibler Divergence	59
5.6.8	Applying KLD to Sensing Matrix Evaluation	59
6	Cognitive Radio Network Spectrum Analysis	60
6.1	Spectrum Approximation for Cognitive Radio Networks	60
6.1.1	Brief Introduction to Cognitive Radio Networks	60
6.1.2	Overview of Greedy Spectrum Approximation	61
6.1.3	Spectrum Report Models	64
6.1.4	GDB as an Overcomplete Dictionary	65
6.1.5	Approximation Methods for GSA	67
6.1.6	Updating the Stale GDB and Evaluating Sensing Matrices	68
6.1.7	Related Spectrum Approximation Methods	69
6.1.8	Practical GSA Considerations	70
6.1.9	The Relative Comparison of the GSA Process	70
6.2	Spectrum Verification for Cognitive Radio Networks	72
6.2.1	Overview	72
6.2.2	Spectrum Falsification Heuristic 1 (δ_{MATCH}): Number of Matched Sensors	72
6.2.3	Spectrum Falsification Heuristic 2 (δ_{RECON}): Reconstruction Error	73
6.2.4	Spectrum Falsification Heuristic 3 (δ_{SIGN}): Correlation Sign	74
6.2.5	Related Methods	75
7	Simulation and Test	78
7.1	Overview	78
7.1.1	Test Approach: Test Cases and Functional Areas Tested	78
7.1.2	Simulation Parameters	79
7.1.2.1	Approximation Method Configurations Simulated	80
7.1.2.2	Dictionaries Simulated	81
7.1.2.3	Sensing Matrices Simulated	82
7.1.2.4	Stale Effective Dictionary Construction: Dictionary and Sensing Matrix Perturbations	82
7.1.3	Detailed Review of Test Cases	83
7.1.4	Simulation Notes	83

7.2	Gaussian Dictionary (TC-01 through TC-03)	84
7.2.1	TC-01: Gaussian Dictionary, Baseline Support Estimation Performance . .	84
7.2.1.1	Support Estimation vs Number of Measurements	84
7.2.1.2	Support Estimation vs SNR	86
7.2.1.3	Support Estimation vs Sparsity	87
7.2.2	TC-02: Gaussian Dictionary with Dictionary Evolution	88
7.2.2.1	Gram Analysis	88
7.2.2.2	Support Estimation per Sensing Matrix	89
7.2.2.3	Support Estimation vs Difference Operator	90
7.2.3	TC-03: Gaussian Dictionary, Sensing Matrix Evolution	92
7.2.3.1	Gram Analysis	92
7.2.3.2	Support Estimation per Sensing Matrix	92
7.2.3.3	Support Estimation vs Difference Operator	94
7.3	CRN Spectrum Database Dictionary (TC-04 through TC-06)	96
7.3.1	Spectrum Approximation and Verification Simulation Overview	96
7.3.2	TC-04: Spectrum Dictionary with Sensing Matrix Evolution (Inversion) . .	99
7.3.2.1	Support Estimation per Sensing Matrix	99
7.3.2.2	SSDF Inversion Detection Heuristics	99
7.3.3	TC-05: Spectrum Dictionary with Sensing Matrix Evolution (Spoofing) . .	99
7.3.3.1	Support Estimation per Sensing Matrix	99
7.3.3.2	SSDF Spoofing Detection Heuristics	99
7.3.4	TC-06: Spectrum Dictionary with Sensing Matrix Evolution (Shifting) . .	101
7.3.4.1	Support Estimation per Sensing Matrix	101
7.3.4.2	SSDF Shifting Detection Heuristics	103
7.3.4.3	Support Estimation vs Difference Operator	103
7.3.5	Discussion on Spectrum Dictionary Results	107
7.4	Compression and Computational Complexity (TC-07)	108
8	Conclusion and Further Research	110
8.1	Sensing Matrix Evaluation Considerations	110
8.2	Additional Greedy Spectrum Approximation and Verification Considerations . . .	111

9	Appendices	113
9.1	Symbols and Operators	113
9.2	Compressive Sensing Matrix Measures	115
9.2.1	Sparse Rank (SPARK)	115
9.2.2	Null Space Property	115
9.2.3	Cumulative Coherence	116
9.2.4	t-Averaged Coherence	116
9.2.5	Restricted Isometry Property	117
9.3	Sensing Matrix Optimization Derivations	118
9.3.1	Derivation of Optimal Effective Dictionary	118
9.3.2	Derivation of Optimal Gram Matrix	121
9.3.3	Derivation of Minimum Objective Function Error	122
9.3.4	Derivation of Optimal Sensing Matrix	123
9.4	Additional Information on Linear Algebra Topics	129
9.4.1	Symmetric Matrices	129
9.4.2	Positive Semi-Definite Matrices	129
9.4.3	Affine Sets, Convex Sets and Cones	130
9.4.4	The Set of Positive Semi-Definite Matrices form a Cone	131
9.4.5	Geometric Interpretation of Gram Matrix Entries	133
9.4.6	Geometric Perspectives on the Covariance and Gram Spaces	135
9.4.7	Metrics, Norms and Matrix Norms	135
9.5	Cognitive Radio Networks	136
9.5.1	Section Overview	136
9.5.2	Dynamic Spectrum Access	136
9.5.3	Cognitive Radio Networks	137
9.5.4	Cooperative Spectrum Sensing	138
9.5.4.1	Types of CSS	138
9.5.4.2	Centralized Spectrum Access System	139
9.5.4.3	The Geographic Database	139
9.5.5	Additional Aspects of the CRN Signal Model	141
9.5.6	CRN Security	143

9.5.6.1	Overview	143
9.5.6.2	Spectrum Sensing Data Falsification	144
9.5.6.3	SSDF Cases Addressed by the Proposed Method	147
9.6	Additional Applications of Gram Matrix Distances	148
9.7	SSDF Prototyping Algorithms	148
9.7.1	Build CR Network GDB	149
9.7.2	Call CRN Spectrum Analysis Simulation	149
9.7.3	Construct Candidate SU Spectrum Report	150
9.7.4	Monte Carlo SSDF Test	150
9.7.5	Compute SSDF Detection Simulation Statistics	150
9.8	Core Prototyping Code	151
9.8.1	Overview	151
9.8.2	Orthogonal Matching Pursuit	151
9.8.3	Inner Product Correlation	153
9.8.4	Sensing Matrix Optimization	153
9.8.5	Simple Demo of Sensing Matrix Optimization	155
9.8.6	Gaussian Dictionary Simulation Code	157
9.8.7	Sensing Matrix Evaluation	164
10	References	165
11	List of Acronyms	174

List of Tables

1	Variable Definitions for Stale and Updated Sensing Matrices and Dictionaries . . .	41
2	Two Methods for Stale Effective Dictionary Construction	44
3	Simulation Test Case Names and Locations	78
4	List of Approximation Configurations Simulated.	81
5	Details of Main Simulation Test Cases	84
6	Simulation Parameters	97
7	Table of Symbols	114

List of Figures

1	Coherence vs Number of Compressive Measurements	20
2	Methods to Compute Atom Separation	21
3	Intuitive demonstration of the effect on the geometry of dictionary atoms in low dimensions.	35
4	Dictionary Evolution for Stale Effective Dictionary Construction	42
5	Sensing Matrix Evolution for Stale Effective Dictionary Construction	43
6	Failure of Single Parameter Gram Matrix Measures	50
7	PMF of Gram Matrix Entries	54
8	EDF of Gram Matrix Entries	54
9	Interpretation of PMF of Gram Matrix Entries	56
10	Cognitive Radio Sensor Node Grid	62
11	Cognitive Radio Network Setup	63
12	Processing Chain of a Cognitive Radio Sensor	63
13	Candidate Secondary User Spectrum Request	64
14	Hidden Emitters	65
15	Creation of Geographic Database	66
16	Spectrum Sensing Data Falsification Data Functional Diagram.	72
17	Support Estimation Results for Gaussian Dictionary versus Number of Measurements	85
18	Support Estimation Results for Gaussian Dictionary versus SNR	86
19	Support Estimation Results for Gaussian Dictionary versus Sparsity	87
20	Gram Analysis, Dictionary Evolution, Gaussian Dictionary	89
21	Support Estimation Results for Gaussian Dictionary with Dictionary Evolution. . .	90
22	Support estimation versus difference operators for Gaussian dictionary with dictionary evolution.	91
23	Gram Analysis, Sensing Matrix Evolution, Gaussian Dictionary	92
24	Support Estimation Results for Gaussian Dictionary with Sensing Matrix Evolution.	93

25	Support estimation versus difference operator for Gaussian dictionary with sensing matrix evolution.	95
26	CRN Sensors and Transmitter Layout	97
27	Example Image of Spectrum Database	98
28	Support Estimation, Spectrum Dictionary w/Spectral Inversion	100
29	SSDF Detection Heuristics for Spectral Inversion	101
30	Support Estimation, Spectrum Dictionary w/Spectral Spoofing	102
31	SSDF Detection Heuristics for Spectral Spoofing	103
32	Support Estimation, Spectrum Dictionary w/Spectral Shifting	104
33	SSDF Detection Heuristics for Spectral Shifting	105
34	Support estimation versus difference operator for spectrum dictionary with sensing matrix evolution	106
35	CDF KS Statistics for Gram Matrices of Evolving Dictionaries	107
36	Computational Complexity and Compression	109
37	Euclidean Distance Between Atom Pairs	134

1 Introduction

1.1 Application: Secure Spectrum Sensing in CRNs

Dynamic spectrum access (DSA) and cognitive radio (CR) technologies have been the focus of increasingly intense recent research. As envisioned, CR transceivers will utilize unoccupied frequencies in their vicinity so as to maximize radio availability and throughput. A key part of CR success is the task of spectrum sensing and classification, i.e. the CRs near-real time awareness of the available spectrum in its vicinity.

DSA technologies such as software defined radio enable efficient use of spectral resources in cognitive radio network (CRN) systems. To achieve the anticipated 5G wireless services and associated quality of service required, additional bandwidth above what is available today will be needed [1]. As wireless *densification* becomes a reality, large metropolitan areas will see heterogeneous networks with many sensors per geographic area emerge [1]. The accumulation of more users, more sensor systems, and finite available bandwidth will need to be optimally addressed. Various CRN network designs have been studied and discussed in the literature, e.g. proposed CRN architectures include underlay, overlay, and interweave systems [1, 2]. All of these architectures involve spectrum sharing of unused or intermittently used spectrum between primary users (PU) and secondary users (SU), and can be utilized to expand the available number of channels (at some nominal bandwidth). However, with sensor densification [1], and increasingly prevalent channel overlap and re-use (in particular at the micro- and pico-system levels), CRN computational and logistical requirements can increase exponentially, to the point where the number of sensors exceeds the number of fixed channels. In addition, the potential for co-channel interference as seen by cognitive radios (CR) in overlapped coverage areas increases dramatically in this scenario.

Spectrum sensing is a key task that a CRN will need to perform. Relying on spectrum sensing reports generated from a single sensor in a dense network may be unreliable due to obstructions in the path of the signal, signal fading, receiver physical constraints and traffic loading considerations [3]. Collaborative sensing can help to overcome such single-sensor impairments. Spectrum measurement sensor reports from multiple sensors can be fused to reconstruct the local spectrum,

thereby ascertaining whether the spectrum reported by the SU (as well as the SU itself) is to be trusted, sanctioned or even denied access to the system [3].

While collaborative spectrum sensing aids in a variety of CR tasks, it also renders the CR system vulnerable to spectrum sensing data falsification (SSDF) attacks [4]. Although there are a variety of active research areas within CR security, the tools developed in this paper are intended to address the detection of a subset of SSDF attacks, utilizing sparse approximation and compressive sensing techniques. One example of a spectrum falsification, as indicated in [4], is for a malicious SU to invert its sensing results prior to reporting them, which causes otherwise available spectrum to appear used, and active spectrum to appear available, thereby reducing availability and creating the potential for interference. This present work presumes the existence of a geographic database (GDB) which, for the purposes of this paper, is a matrix that is constructed and updated via collaborative energy-detection based spectrum reports from trusted sensors. The GDB is maintained by a controlling spectrum authority, centrally located at a fusion center (FC). In sparse approximation terms, the GDB is the dictionary matrix and the spectrum reports are columns or atoms of the dictionary.

1.2 Solution Approach: Sparse and Compressive Approximation

Sparse signal models and transform compression are two classes of signal processing tools increasingly used across a wide variety of domains. While they are each separate classes of tools, they are often utilized as a complementary pair. Applications range from exploratory data analysis, to dimensionality reduction on manifolds, to signal reconstruction and analysis [5]. Tools based on sparsity and random projections are coursing through various veins of current research, e.g. the related research areas of sparse approximation and compressed sensing (CS), specifically *discrete CS*.

Many recent research publications have focused on applying compressive sensing (CS) to CR tasks [6]. While much CS research appears focused on reducing the burden of the ADC in the front end of CR receivers, some research has been aimed at utilizing CS to reduce the computational burden of analyzing and characterizing the spectrum of a CR system, particularly as applied to the generation of radio environment maps (REM) or spectrum hole identification [7–10]. One way to generate a REM is through spectrum cartography, i.e., the plotting of which radio frequency (RF) channels are in use over a geographic area. Reference [7] uses a variant of orthogonal matching pursuit (OMP) for spectrum cartography intending to construct a REM based on received signal

strength (RSS). Their method capitalizes on the sparsity of the active PUs over the coverage area. Using the same principles and similar approach, CS has also been applied to the problem of sparse target localization [11], a form of RF fingerprinting.

The application for CS sensing matrix optimization presented in this dissertation uses a slight variant of OMP to detect SSDF attacks in the reported spectrum of a candidate SU. OMP is used to de-mix the composite reported spectrum, and use spectrum reports from sensor nodes (SNs) with overlapped coverage sectors to reconstruct the reported spectrum as a linear combination of a few SN reports. The method provides a potential means to approximate spectrum and to detect SSDF attacks. Sparsity in the context of this application translates to the relatively few SNs needed to approximate the reported spectrum of an honest CR. Traditional CS utilizes a randomly generated sensing matrix to achieve the desired compression while minimally distorting the orthonormal basis (ONB) dictionary. Here, recent results in sensing matrix optimization (e.g. [12]) are applied which strives to orthogonalize the non-orthogonal GDB dictionary matrix. In addition, the proposed sensing matrix is chosen with pseudorandom components, which provide some level of anonymity for the candidate CR. The resulting proposed sensing matrix simultaneously acts to

- Compressively reduce the communications overhead of having to report the spectrum to the FC,
- Reduce the aggregate computational burden of OMP at the FC,
- Optimally separate the non-orthogonal components of the GDB dictionary, thereby increasing probability of detecting SSDF attacks, and
- Mildly obfuscate the spectrum of the candidate CR from other users through multiplication with a pseudo-random matrix (although neither spread spectrum techniques nor cryptographic codes are applied here).

In addition, the proposed system model incorporates discrepancies in the frequency support in the GDB, due to fading, terrain, or obstructions in the signal path.

1.3 Novel Contributions

The main novel contributions of this dissertation include:

- Novel methods of Gram matrix comparison which move the state of the art from single parameter measures to distribution-based measures, as well as comparisons based on quantitative

measures of histograms as opposed to qualitative.

- A general sparse approximation framework applied to CRN security, named *Greedy Spectrum Approximation* (GSA), is developed and specifically aimed at performing SSDF detection (i.e. to perform spectrum verification) within database-driven cooperative spectrum sensing (CSS) CR systems
- A system model which incorporates high user density, frequency reuse, sensor node overlap and additive white Gaussian noise (AWGN) in the context of SSDF detection
- A new SSDF attack, called the *shifted spectrum attack*, is presented, simulated, and analyzed
- A method of spectrum verification, named *Greedy Spectrum Verification* (GSV) that uses compression and obfuscation of the spectrum sent to the FC via a randomized sensing matrix
- Application of a closed form compressive sensing matrix optimization method (as presented in [12]) to improve spectrum matching, and establishment of novel connections between the method in [12] and ‘pre-processing’ procedures cited in other publications, e.g. [11]
- A comparison of four different types of sensing matrices (identity, random, optimized, stale)
- Identification and analysis of a vulnerability within the native OMP algorithm making it susceptible to spectrum inversion attacks, as well as a slight modification of the OMP algorithm to overcome it
- Addressing the issue of when to update the optimal sensing matrix by monitoring the geometry of the resulting effective dictionary

These contributions resulted in a conference paper [13], as well as a patent application [14].

1.4 Dissertation Organization

The remainder of this dissertation is outlined as follows:

- **Section 2** provides a tutorial-level overview of linear signal spaces, transformations, and signal expansion.
- **Section 3** contains background on sparse approximation using overcomplete dictionaries, in relation to CS and CSRD. This section also presents a very brief review of well-known sensing matrix design measures for CS applications. Next, the section discusses the geometry of the columns of a dictionary (known as atoms), and defines the Gram matrix as the primary tool that will be used to study dictionary geometry. The section concludes with an overview

of correlation and orthogonal matching pursuit in the context of greedy methods used to reconstruct vectors that are sparse in a given dictionary.

- Next, **Section 4** reviews research seeking to compute an optimal sensing matrix for CSRD, with respect to the effect the sensing matrix has on the separation of the dictionary atoms. The product of the sensing matrix and dictionary is known as the effective dictionary. In addition, the Gram matrix provides a measure of spread of a matrices columns. Therefore the effect of the sensing matrix on the columns of the dictionary is computed from the Gram matrix of the resulting effective dictionary. One sensing matrix optimization method is reviewed in detail which is the focus of the dissertation, and provides a closed-form method of sensing matrix optimization found in the literature.
- A dynamically changing dictionary presents challenges for practical sensing matrix optimization, and so **Section 5** identifies three classes of tools under the heading of *Difference Operators*. These operators are intended to compare Gram matrices of effective dictionaries within the context of the sensing matrix optimization in CS in redundant dictionaries (CSRD) presented in Section 4, with the aim of determining when to update the sensing matrix in response to changes in the columns of the dictionary matrix. The aim is specifically to compare Gram matrices of effective dictionaries formed from optimal versus sub-optimal effective dictionaries. Some nomenclature regarding stale sensing matrix/dictionaries is first defined, followed by two methods to construct stale effective dictionaries. Next the sensing matrix evaluation (SME) problem is formulated, and the three approaches for SME are described. Use of the sensing matrix objective (SMO) function is described, as are some novel applications of well-known probability distribution distance metrics applied to a pair of Gram matrix entries. Also mentioned, but not studied in depth, are methods in the literature which take into account the geometry of the space of Gram matrices. The section finishes with an emphasis on the theoretical development of computing distances between distributions of Gram matrix entries.
- **Section 6** contains an overview of the tools developed in the previous sections as applied to spectrum approximation and verification in cognitive radio networks (CRNs). First, Section 6.1 provides a brief overview of CRN design, with an emphasis on collaborative spectrum sensing (CSS) and detection of spectrum sensing data falsification (SSDF) attacks in database-driven CRNs (a.k.a. Byzantine attacks). It also provides details on a novel approach to CSS, named *Greedy Spectrum Approximation (GSA)* in a CSS database-driven CRN which views

the spectrum geographic database (GDB) as an overcomplete dictionary. The method involves utilizing sparse approximation or CSRD (using either random or optimal sensing matrices), by viewing the spectrum database as a type of overcomplete dictionary. Next, Section 6.2 outlines a novel approach to SSDF detection, named *Greedy Spectrum Verification (GSV)* in a CSS database-driven CRN. The method involves operating on the output of the GSA method of Section 6.1, and presents three metrics that can be used to detect certain types of SSDF.

- **Section 7** presents the results of a MATLAB simulation of the theory presented in Sections 4, 5, 6.1, and 6.2. The intent is to understand the tools and to demonstrate their limitations and feasibility. The sensing matrix and Gram comparison tools presented in Section 5 are prototyped and assessed for their ability to measure or monitor the effective dictionary geometry and its effect on CSRD regarding support estimation performance and specifically with GSV (regarding SSDF detection performance). Two classes of dictionaries are evaluated: pseudo-randomly generated dictionaries (drawn from a Gaussian distribution) and a synthetically-generated spectrum GDB.
- **Section 8** abstracts some specific generalizations of the results of this research and suggests potential future areas of fertile research.

2 Linear Signal Approximation Preliminaries

2.1 Signal Spaces

Frequently, real-world systems and signals can be adequately modeled as arising from a linear process, and as a result a popular approach in signal processing is to apply linear signal and system models [15]. It is typical to linearly model signals as a sum of a few vectors that ‘live’ in a particular vector space. Using a vector space model for signals allows for simplicity when combining signals, whereby adding a scaled version of one signal vector to a scaled version of another signal vector results in a third relevant signal vector. The signals in this work are concerned with *normed* vector spaces (see Section 9.4.7), where the Euclidean inner product is defined for the vector space, and induces the ℓ_2 – *norm* [16]. The normed vector-space point of view permits the use of familiar notions such as length, which allows distances and angles to be computed, facilitating the comparison and contrast of signal vectors [15–18].

2.2 Linear Transformations

Linearly transforming a signal into a different basis or redundant representation (such as a frame or overcomplete dictionary) can sometimes provide a more succinct or compressed representation of the signal. Linear transformations are used to decompose a signal (a.k.a. *analysis*) or to expand a signal (a.k.a. *synthesis*, *reconstruction*, or *approximation*) as a weighted linear combination of other signals [17]. In signal processing, an example of a classic linear transformation is the Fourier transform [17]. In practical digital signal processing (DSP), linear transformations typically take the form of a matrix, such that the linear transformation is realized as a matrix multiplication with a single signal vector (or a signal matrix of multiple signal vectors) [17].

2.3 Orthonormal Basis Dictionaries

If the columns of a linear transformation matrix are linearly independent, the set of columns forms a *basis*. I.e., no column in the set (other than possibly a column whose entries consist of all zeros)

can be constructed by linear a combination of the remaining columns [19, 20]. Another way this is expressed is to say that no column lies within the vector space that is *spanned* by the other columns in the set. Note that a set of vectors need not be mutually orthogonal in order to be considered a basis. Orthogonal linear transformation matrices whose columns form an orthonormal basis (ONB) are heavily used by engineers and scientists for a myriad of applications; such transforms are referred to as orthonormal transforms and consist of an orthonormal matrix (in the case of real entries, or a *unitary matrix* for complex entries) [19, 20]. An orthonormal matrix is a matrix whose columns are each of unit length, and are pair-wise orthogonal [17] such that the inner product of each column with itself is equal to one, and the inner product of any column with any other column of the matrix is zero [19, 20].

2.4 Overcomplete Dictionary Representations

In addition to ONBs, researchers have studied *overcomplete* systems of equations, used to approximate a signal vector (overcomplete dictionaries can also be used to represent a set of vectors, but here the focus will remain on approximation of a single column vector). In approximation theory, there is a substantial amount of research devoted to increasing the number of atoms in the dictionary beyond that of an ONB [21] to represent a vector. In overcomplete representations of a vector, the columns of the dictionary exhibit a certain level of linear dependency, providing a richer, more robust, and more subtle representation [20, 21]. There exists atoms within the dictionary that lie within the span of other atoms, i.e. within a given overcomplete dictionary, some atoms are linear combinations of other atoms. Overcomplete systems can be formed in a variety of ways, e.g.,

- Combining several different bases into one (e.g. identity and discrete Fourier transform matrix) [22], or
- Partitioning a single column vector containing discrete measurements into shorter vectors of equal size and storing the lower-dimensional vectors as columns in an overcomplete matrix [23].

In general, the data comprising dictionaries can be based on stored digital sensor measurements (realizations), or a mathematical model of the data [24] . Overcomplete dictionaries provide a number of advantages beyond orthonormal bases (ONB), including *sparsity* (expanding any signal in a large class using only a small number of coefficients), and *separation* (clearly separating signals

that are a linear combination of simpler signals, using the transform coefficients). Simultaneously, it is advantageous if overcomplete dictionaries retain these characteristics of ONBs [17, 20]:

- Stability: Small perturbations to the coefficients should yield small changes in the reconstructed signal
- Speed: Numerical computation of the coefficients should be $O(N)$ or $O(N\log(N))$

A well-written account of the shift in published literature from orthogonal transforms to overcomplete representations can be found in [24], where it is noted that “... the dictionary can only be as successful as its underlying model...” [24]. Models are, by definition, simplistic when compared with the complexity of the actual processes which they are used to represent. In place of a model of the data to form a dictionary, actual measurements may be used (if available). Therefore, a dictionary can be viewed as a type of database and continued greater advances in communications technology are enabling ever-increasing databases to be updated more frequently than ever before. Dictionaries that incorporate actual measurements in near real-time are able to change in response to user needs, environmental changes, or feedback from the performance of a given system. As a result, the database or dictionary can, through use of a feedback signal or dictionary updates, adaptively become tailored to the particular application and current environment. However, when the feedback signal contains noise, interference or inaccuracies (as it almost always does in practice), then some type of discrimination, pre-processing (e.g. dictionary learning), filtering or smoothing is usually applied, so that the dictionary performs well for the intended application [24–26]. Such an adaptive dictionary feedback process can also require additional overhead in the form of increased bandwidth, and/or higher computational complexity.

2.5 Linear Transformations used in this Work

Linear matrix transformations are critical to the topics of compressive sensing and spectrum analysis as used in this dissertation. The main application of the tools developed in this dissertation, specifically sensing matrix optimization and evaluation, is the task of approximating and verifying a *candidate* spectrum vector in a cognitive radio network. The main signal model is a linear expansion model for the vector x , representing a candidate spectrum report that is to be approximated by the spectrum reports generated by trusted receivers local to the candidate receiver.

Within this context, Fourier transforms are used to transform the digital signal from the discrete

time domain into the digital frequency domain. However, it will be assumed that this transformation has already been performed (within the calculation of the periodogram) prior to performing power spectral density (PSD) estimation and verification. Also, a linear transformation called a *sensing matrix* Φ is used to compress the PSD estimates.

3 Signal Approximation Models

3.1 Introduction to General Linear Models

Signal expansion (approximation) systems aim to select a subset of the columns (atoms) from a given dictionary matrix and estimate a set of coefficients or column weights used to form a linear combination that is adequate to describe a signal vector. Signal expansion is closely related to *subset selection or support estimation*, a.k.a., *least-absolute shrinkage and selection operator or LASSO*, which is a selection method for sparse solutions to linear regression problems [27].

In particular, the spectrum approximation techniques studied in this dissertation aim to represent a vector x (referred to as a *spectrum report*) as a linear combination of a small number of columns from a matrix whose columns consist of a set of trusted spectrum reports. This matrix is referred to as a spectrum *dictionary* Ψ , and a spectrum report vector can be approximated as a linear combination of columns from Ψ :

$$x \approx \Psi s, \quad (3.1)$$

where $x \in \mathbb{R}^{Nx1}$, $\Psi \in \mathbb{R}^{NxM}$, and $s \in \mathbb{R}^{Mx1}$. The column vector $s \in \Sigma_k \equiv \{s : \|s\|_o \leq k \ll M\}$, such that s is a k -sparse vector and Σ_k is the (non-linear) vector space of k -sparse signals [16]. An N -dimensional k -sparse signal vector has k non-zero elements, while the rest of its $N - k$ entries are zero. The set of k indices at which s is non-zero is called the *support* of the vector s and is represented as $\text{supp}(s)$, so that $|\text{supp}(s)| = k$. The quantity $\|s\|_o$ is called the *ell-zero* (ℓ_o) semi-norm of s (see Section 9.4.7). The cardinality of the support of s , represented by $|s|$, is equal to the ℓ_o semi-norm such that $|\text{supp}(s)| = k = \|s\|_o$. Yet another way to express the ‘cardinality of the support of s ’ and the ℓ_o semi-norm of x is to say that the *sparsity* of s is k .

For the $N \times M$ dictionary matrix Ψ , when $N < M$, Ψ is referred to as an *overcomplete dictionary* or *redundant dictionary*. Fig. 27 depicts an example of an overcomplete matrix. The columns of the dictionary are called *atoms* and the i^{th} atom is denoted by ψ_i , i.e., a trusted SN spectrum report. In short, the vector x is to be approximated by the columns of the dictionary matrix Ψ and the

relationship takes the form of a linear system of equations. In particular, this dissertation applies to problems where $N < M$ making Ψ an overcomplete dictionary, and consequently the linear system of equations in Eq. (3.1) is an *underdetermined system of equations*, with either an infinite number of solutions or no solutions depending on whether or not x is in the column space of Ψ [28, 29].

3.2 Sparse Approximation

Approximation problems arise in a variety of contexts with many subtlety different goals. When the goal of an approximation problem is to select a *sparse* subset of overcomplete dictionary columns and associated expansion coefficients to be used in a linear combination to approximate (i.e. *expand*) a signal, the problem is known in the literature as the *sparse approximation* (SA) problem [30, 31]. The SA problem can be stated as: [30]

“For a given class of dictionaries, how does one design a fast algorithm that provably calculates a nearly optimal sparse representation of an arbitrary input signal?”

The goal in SA is to expand the signal x defined in Eq. (3.1) in a few of the atoms (columns) of the dictionary Ψ , i.e. to form a *k-sparse* signal expansion $\tilde{x}$ of x in terms of Ψ . SA historically precedes the advent of CS. In SA (and CS) sparsity acts as a regularizer or constraint that is placed upon the under-determined system of equations in order to make the problem solvable. The SA problem can be formulated as an optimization problem [30]:

$$\begin{aligned} & \underset{s}{\text{minimize}} && \|s\|_0 \\ & \text{subject to} && \|x - \Psi s\|_2 \leq \epsilon, \end{aligned} \tag{3.2}$$

for some $\epsilon > 0$. An alternative form of the SA problem when the sparsity k is fixed to k_{max} can be formulated as [32, 33]:

$$\begin{aligned} & \underset{s}{\text{minimize}} && \|x - \Psi s\|_2 \\ & \text{subject to} && \|s\|_0 = k_{max}. \end{aligned} \tag{3.3}$$

The reconstructed approximation is formulated as:

$$\hat{x} = \Psi s \approx x, \quad (3.4)$$

where s is a k -sparse vector of coefficients, whose magnitude and index location of non-zero entries are to be estimated.

The SA problem is essentially to find the *best k -term approximation* of x with regards to the sparsity of the coefficient s and the Euclidean distance to x [34]. As presented in the literature, the ℓ_2 - norm minimization in Eq. (3.2) and Eq. (3.3) is frequently accomplished via least-squares (LS) techniques [22, 30, 35, 36]. Solving the SA problem exactly has been shown to be NP-hard [22,32,33], and so either iterative greedy approximation methods are frequently used to approximate a solution, or a simpler version of the SA problem is solved by the minimizing the ℓ_1 -norm and linear programming optimization methods are used.

Greedy approximation methods and linear programming are frequently used to solve CS problems as well as SA problems. Although SA inherently does not include any compression, for a given set of measurements x , the optimal approximating coefficient indices are generally not unique [12].

3.3 Compressive Sensing

3.3.1 CS Problem Formulation

Building on the model in Eq. (3.1), a more general signal model utilized in this work is the compressed system of linear system of equations, and can be formulated as

$$\begin{aligned} y &= \Phi x \\ &\approx \Phi \Psi s = As, \end{aligned} \quad (3.5)$$

where $y \in \mathbb{R}^m$ is termed the (compressed) *measurement vector* and $\Phi \in \mathbb{R}^{m \times N}$ is the *sensing matrix*, such that $m \ll N$. The vector s is again k - sparse. The matrix $A \in \mathbb{R}^{m \times M}$ is referred to as the *effective dictionary*, and is composed of the matrix product of the sensing matrix and dictionary, $A = \Phi \Psi$. Note also that since $N < M$, the relationship among the sparsity of s , and the dimensions of the sensing and dictionary matrices is $k \leq m \ll N < M$. Note that in SA, since there is no compression, $m = N$ and the sensing matrix simply reduces to the $N \times N$ identity matrix: $\Phi = I_{N \times N}$.

To reduce either the sampling rate/storage requirements, or to improve data rate or computational complexity of atom selection or signal recovery of x , one can pre-multiply the signal x by a compressive sensing matrix Φ , in order to reduce the dimensionality of the system of equations [16]. The resulting compressed linear problem is known as the *compressive sensing* (CS) problem. The problem is to approximate the compressed measurements y by $\hat{y}$

$$\hat{y} = \Phi \hat{x} \approx y = \Phi x. \quad (3.6)$$

From Eq. (3.4), $\hat{x} = \Psi s \approx x$, so the problem becomes

$$\begin{aligned} & \underset{s}{\text{minimize}} \quad \|s\|_0 \\ & \text{subject to} \quad \|y - As\|_2 \leq \eta, \end{aligned} \quad (3.7)$$

where $A = \Phi\Psi$ and $y = \Phi x$. As with SA, when the CS problem includes a fixed constraint on the sparsity, the CS problem can be alternatively formulated as [32,33]

$$\begin{aligned} & \underset{s}{\text{minimize}} \quad \|y - As\|_2 \\ & \text{subject to} \quad \|s\|_0 \leq k_{max}. \end{aligned} \quad (3.8)$$

Since y , s , and x are vectors belonging to a finite-dimensional vector space, the CS problem formulated above is known as *discrete* CS (as opposed to *infinite-dimensional* CS) [16]. This dissertation concerns the discrete form of CS, and will be referred to simply as 'CS' from here on. The main conceptual contribution that CS appears to provide is the realization that a sparse signal can be represented by a relatively small number of randomly generated linear measurements [37].

A sparse signal representation can be used to reconstruct (i.e. recover) signals, or to perform sparse signal processing in a compressed domain to ease computational burden [5]. The number of measurements m required for successful recovery has been shown to be on the order of $m \approx k \log(N/k)$ [5, 16, 22, 38]. Historically, the CS problem was initially performed mainly on $N \times N$ ONB dictionaries (i.e., columns of Ψ are orthonormal). Later publications began to address CS in redundant dictionaries [24, 39, 40].

3.3.2 Sensing Matrices

There is a large body of existing research which has analyzed sensing matrices for CS and attempted to estimate a lower bound on the number of measurements required for a high probability of reconstruction. In orthonormal basis dictionaries, random sensing matrices are frequently employed due to the simplicity and optimality of generating a sensing matrix ($\Phi \in \mathbb{R}^{m \times N}$, $m \ll N$) by randomly sampling the entries from a particular set of distributions (referred to as *sub-Gaussian* distributions). Such sensing matrices have the restricted isometry property (RIP, Sec. 9.2.5) with high probability for orthonormal dictionaries. The RIP is but one of many matrix properties used to characterize sensing matrices for CS; Section 9.2 contains a brief review of some of them. One popular sub-Gaussian based sensing matrix [22, 38] is to take the entries $\phi_{i,j}$ of the $m \times N$ sensing matrix Φ to be independent realizations of a Gaussian random variable with mean 0 and standard deviation $1/\sqrt{m}$,

$$\phi_{i,j} \sim \eta(0, 1/m). \quad (3.9)$$

However, there are certain applications where the engineer is not permitted free-reign regarding the selection of a random sensing matrix. Sometimes the constraints of the application hinder random sensing matrix construction or prevent it altogether. There exist applications in which the sensing matrix must have some structure imposed on, e.g., magnetic resonance imaging [22]. Additionally, there may be some sensing matrices that are more suited to certain applications than others. For example, one sub-Gaussian distribution presented in the literature [38, 41] contains only three distinct entries: 0, +1, and -1, where up to 2/3 of the entries are allowed to be 0 while maintaining its stability as CS matrix (i.e., it obeys the RIP; see Sec. 9.2.5). Performing CS with such a sparse sensing matrix is less computationally complex, as it requires no multiplications.

In some instances, an overcomplete dictionary can provide a more subtle representation of a signal, however the linear dependency between the atoms also presents a greater risk of selecting the incorrect atom when noise is present. When using CS with a redundant dictionary, the considerations for proper sensing matrix selection changes as compared with the orthonormal case. The goal moves from preserving the orthonormality of the orthonormal dictionary to finding a sensing matrix that makes a non-orthonormal dictionary more orthonormal, i.e. making the atoms more mutually orthogonal. If there is knowledge of the dictionary and sufficient computing resources to perform a singular value decomposition, the sensing matrix can be tailored to the dictionary, as is shown in

3.4 A Geometric Analysis of Sensing Matrices and Dictionaries

3.4.1 Dictionary Atoms as Vectors

A dictionary may be interpreted as a single geometric object, or as being composed of a collection of geometric objects. The dictionary used in this work is formulated as an $N \times M$ matrix (where $N < M$) with real entries, i.e. as a collection of dictionary atoms or columns comprising the matrix. Therefore, the dictionary can be viewed as a collection of column vectors in N -dimensional Euclidean vector space. The geometry of a dictionary matrix can be conceived by imagining each of the M columns or atoms as a vector that ‘lives’ in an N -dimensional vector space. Assuming the dictionary columns are normalized (i.e. each column is divided by its respective length, making the length of each column to be unity), the separation (or spread) between each pair of columns can be found by computing the Euclidean distance between dictionary column vectors. Similarly, to measure the separation between atoms, the angular separation between each pair of atoms can be computed.

The literature contains many matrix properties that are used to design sensing matrices for compressive sensing (see Section 9.2). This section addresses the *coherence* property of CS matrices, which is fundamental to the understanding of the main tools developed in this dissertation.

3.4.2 Coherence

The *coherence* of a matrix $A \in \mathbb{R}^{m \times M}$ with ℓ_2 -normalized columns is defined as [22]

$$\mu(A) = \max_{1 \leq i, j \leq M \text{ and } i \neq j} |\langle a_i, a_j \rangle|, \quad (3.10)$$

or equivalently, the inner product $|\langle a_i, a_j \rangle|$ can be written using properties of vector multiplication $|a_i^T a_j|$

$$\mu(A) = \max_{1 \leq i, j \leq M \text{ and } i \neq j} |a_i^T a_j|, \quad (3.11)$$

where a_i and a_j for $i \in [1, M]$ are the i^{th} and j^{th} *normalized* columns of the matrix A , respectively. Specifically, ℓ_2 -normalized columns means that each element of the matrix A is divided by the ℓ_2 -norm of the column in which the element resides, i.e. $\|a_i\|_2 = 1$ for $i \in [1, M]$. If A does not have

normalized columns, then the coherence can be computed as

$$\mu(A) = \max_{1 \leq i \neq j \leq M} \frac{|\langle a_i, a_j \rangle|}{\|a_i\|_2 \|a_j\|_2}. \quad (3.12)$$

An alternative formula for the coherence in terms of the matrix inf norm is [42]

$$\begin{aligned} \mu(A) &= \max_{i \neq j} |\langle A_i, A_j \rangle| \\ &= \|A^T A - I\|_\infty \\ &= \|G(A) - I\|_\infty, \end{aligned} \quad (3.13)$$

where again the matrix A has normalized columns, and the matrix $G = A^T A$ is called the *Gram matrix* of the matrix A . Thus, the coherence is the maximum absolute value of the off-diagonal entry of the matrix formed from $A^T A$. It can be shown that the range of possible values for the coherence is bounded as

$$\mu(A) \in \left[\sqrt{\frac{M-m}{m(M-1)}}, 1 \right]. \quad (3.14)$$

The lower bound on the coherence of an $m \times M$ matrix is referred to as the *Welch bound* [16,30,43,44] and is represented as $\underline{\mu}$

$$\mu(A) \geq \underline{\mu} \equiv \sqrt{\frac{M-m}{m(M-1)}} \quad (3.15)$$

If a dictionary matrix meets the Welch bound with equality, the dictionary is called an *equiangular tight frame* (ETF) [30]. When the number of measurements m is much smaller than the dimension N of the desired vector x , i.e. when $m \ll N$, then the Welch bound can be approximated as [16,30]

$$\underline{\mu}(A) \approx \frac{1}{\sqrt{m}}, \text{ when } m \ll M. \quad (3.16)$$

Coherence is a fundamental property of a matrix that measures the extent to which two atoms from a dictionary are alike. It is a “blunt instrument” in the sense that it only measures the most extreme correlations in the dictionary [30]. A matrix with a small coherence is called an *incoherent matrix*. In words, the coherence is “the largest absolute normalized inner product among the atoms of the dictionary” [25]. As [45] notes, the coherence tries to measure the most likely pair(s) of dictionary atoms to be confused when executing a recovery or pursuit algorithm.

The concept of coherence has been extended in several direction. Some publications have de-

defined coherence measures for certain types of sparse signal models, e.g. block-sparsity. Other publications aim to relate the coherence more directly to the number of required measurements required for recovery.

The sparse rank (spark), null space property (NSP), and restricted isometry property (RIP) (see Section 9.2) provide guarantees for the recovery of k -sparse signals. However, verifying that a matrix satisfies any of these properties would typically require a combinatorial search over all $M C_k$ submatrices of the dictionary. While the coherence also provides guarantees, the coherence is more computationally efficient than the NSP or RIP. As a general rule of thumb, a small coherence implies better performance from a CS recovery algorithm, so the coherence provides a means to evaluate the expected performance of a system.

Reference [30] points out the following relationship between the sparsity of a vector and the dictionary to attain some results in solving sparse systems of equations proven by [30, 46, 47]:

If the inequality

$$\|\alpha\|_o < \frac{1}{2} \left(1 + \frac{1}{\mu(A)} \right) \quad (3.17)$$

is satisfied, then the following are true statements:

- The vector α is the sparsest vector to explain x in terms of the dictionary A , i.e. it is the solution of

$$\min_{\alpha} \|\alpha\|_o \text{ s.t. } x = D\alpha. \quad (3.18)$$

- The basis pursuit (BP) algorithm for approximating α is tantamount to exactly solving the following linear program for α ,

$$\min_{\alpha} \|\alpha\|_1 \text{ s.t. } x = D\alpha. \quad (3.19)$$

- The OMP algorithm is guaranteed to succeed in solving (3.18).

Note that the optimization problems in Eqs. (3.18) and (3.19) are noiseless versions of the problem.

3.4.3 Significance of Dictionary Atom Separation

Learning the correct dictionary is an important step in overcomplete sparse approximation applications. While a learned dictionary may contain many atoms applicable to the relevant input signals of interest (i.e. the input signals can be linearly described by the dictionary columns), the learned dictionary may not have the incoherence required for good performance, where the term *good* is defined relative to the application. Sometimes the goal is to simply attain an adequate approximation of an input signal, in which case, it matters little which particular atoms are selected and therefore separation among atoms becomes less relevant as long as an accurate approximation is found. Other applications may depend on identifying the correct atoms that are correlated with an input, in which case, the separation between atoms becomes more important. In this dissertation, the application is focused more on the latter, namely identifying which atoms formed the input signal. In this case, good sensing matrices can act to reduce the coherence in learned dictionaries [25, 48, 49]. In [50], although not explicitly defining the coherence as the maximum inner product between columns of a dictionary, the concept of *highly coherent structures with respect to the dictionary* within the input signal is discussed. There is also relatively early mention of the incoherence of bases in regards to the ℓ_1 uncertainty principle [51].

Separation among dictionary atoms is relevant when performing tasks in SA, i.e. it is useful to measure or accurately estimate the *spread* of the columns of a dictionary. A *good* dictionary in SA is not only applicable to the task (adequately matching components of the desired signal) and therefore *approximating* the signal set of interest, but in some applications (such as is the focus of this work) a dictionary should also provide *resolvability* amongst its atoms. That is, atoms of a ‘good’ dictionary are relatively spread out from each other, as measured either by their angular separation, or their Euclidean distance. Adequate approximation and atomic separation are typically at odds with each other. E.g., in greedy sparse approximation, more atoms in the dictionary translates to a higher likelihood that the dictionary can match the input signal and sparsely represent the input signal. However, packing more atoms closer together in geometric space, along with additive measurement noise can also result in a higher likelihood that a greedy recovery algorithm selects the wrong atom. Therefore, this tradeoff must be recognized within the design. Note that there may be other applications (beyond the scope of this work) in which it may not be as important to distinguish one atom from another, as long as a sufficiently accurate approximation can be made. The concept of measuring the closeness of packed atoms in regards to SA and compressive sensing tasks has

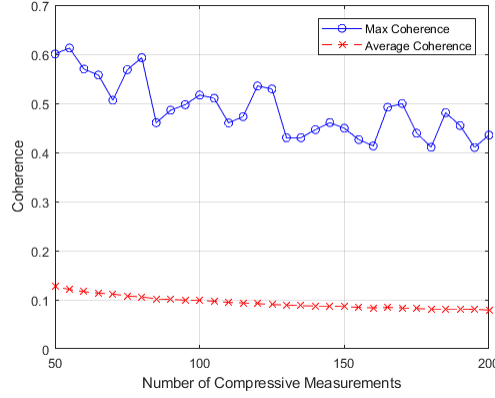


Figure 1: A dictionary is generated by randomly selecting its elements according to a Gaussian distribution, and a sensing matrix is randomly generated, for a series of measurements. The coherence of each resulting effective dictionary is computed, as is the averaged coherence (i.e. the mean off-diagonal entry of the effective dictionary formed from the sensing matrix and dictionary matrix). The trend is seen to be that more compression (a lower number of measurements) results in atoms being forced closer together.

been treated to some extent in [31].

When performing CSRD, that is SA with compression, the issue of atom separation becomes even more significant. The effect of performing sparse approximation under compression is to push the dictionary atoms closer together, i.e. to make them even more coherent than the atoms were prior to compression, as shown in Fig. 1. When trying to correlate a compressed vector with a compressed (effective) dictionary, the atoms become more difficult to distinguish than in the non-compressed case. Thus, the notion of *optimized* sensing matrix in CSRD to separate compressed dictionary atoms becomes a particularly helpful tool.

The paper [26] provides multiple ways to measure the spread or distance between a pair of column-normalized dictionary atoms, e.g.:

- Cosine (Gram entry): $g_{ij} = \cos(\theta_{ij}) = |\mathbf{a}_i^T \mathbf{a}_j|$
- Angle (or Arclength): $s_{ij} = \theta_{ij} = \arccos(g_{ij})$
- Sine (a.k.a. representation error): $r_{ij} = \sin(\theta_{ij})$
- Euclidean (Direct) Distance: $d_{ij} = \|\mathbf{a}_i - \mathbf{a}_j\|_2$

Fig. 2 shows a pair of atoms from a normalized dictionary, as well as some ways to compute distances between the pair of atoms. For a column-normalized dictionary, the relationship between a Gram matrix entry ($\cos(\theta_{ij})$) and the Euclidean distance d_{ij} between the two associated atoms ($\mathbf{a}_i$ and $\mathbf{a}_j$) can be found [52]. Applying the *law of cosines* to the triangle with sides of lengths

$\|\mathbf{a}_i\|_2 = 1$, $\|\mathbf{a}_j\|_2 = 1$, and d_{ij} gives

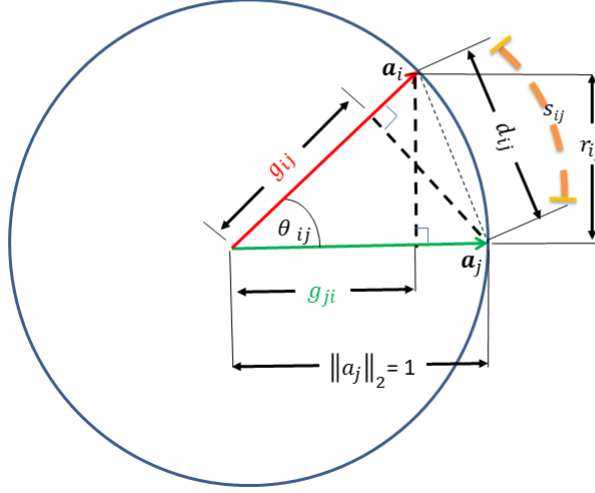


Figure 2: A pair of atoms $\mathbf{a}_i, \mathbf{a}_j$ from a normalized dictionary and the various distances that can be defined to compute separation. Note that a larger angle (not shown) can be formed between the two atoms measuring $2\pi - \theta$ radians, but this angle is not considered in this work.

$$\begin{aligned}
 d_{ij} &= \sqrt{\|\mathbf{a}_i\|_2^2 + \|\mathbf{a}_j\|_2^2 - 2|\cos(\theta_{ij})|} \\
 &= \sqrt{1 + 1 - 2|\cos(\theta_{ij})|} \\
 &= \sqrt{2 - 2|\cos(\theta_{ij})|}.
 \end{aligned} \tag{3.20}$$

3.4.4 Distance or Affinity Matrices

In many signal processing tools and applications, there arises a need to capture the pair-wise relationship among a set of measurements. In graph theory, the matrix that captures the relationships between pairs of nodes connected by edges is known as an *affinity matrix* or *adjacency matrix* [53]. Entries of *Euclidean distance matrices* contain the pair-wise Euclidean distance between pairs of vectors in a measurement set [52, 54, 55]. A matrix that is closely related to the Euclidean distance matrix is the Gram matrix.

3.4.5 Measuring Atom Separation via the Gram Matrix

Various sensing matrix properties have been defined in terms of the Gram matrix of the effective dictionary in order to measure the effectiveness of a sensing matrix for a given dictionary. E.g., the following coherence-related properties have been developed in the research literature

- Coherence [22]
- Cumulative Coherence [22, 30]
- t-Averaged Coherence [45]
- Average Coherence [56]

In addition, the Gram matrix utility is underscored by its:

- Computability (whereas other CS sensing matrix properties are not practically computable, e.g. such as the RIP)
- Strong geometric interpretation
- Frequent appearance in pivotal tools of linear algebra (e.g., singular value decomposition (SVD))
- Many applications in diverse areas

Gram matrices arise naturally in many applications, e.g. parameterization of non-negative polynomials, the *kernel trick method* in machine learning [57], semi-definite programming in convex optimization problems [52], greedy CS recovery algorithms [35], and covariance matrix analysis/principle component analysis [57]. Linear algebra provides an intuitive framework for many of these applications. The Gram matrices also arise in many of the fundamental tools commonly used in linear algebra, e.g. eigenvalue decompositions, the singular value decomposition (SVD), and the Moore-Penrose pseudo-inverse [29].

Looking at the definition and structure of the absolute value of the Gram matrix ($G(A) = A^T A$) will provide some intuitive insight into its use and interpretation. Each entry of the Gram matrix is formed from the scalar-product between a pair of columns of the matrix A . The strong geometric properties of the scalar-products provides an intuitive indication of the geometric significance of Gram matrix entries. The scalar-product is a type of inner product, and each entry in the Gram matrix provides a measure of the angle that each column vector of A makes with every other column vector of A . As a result, the entries of $G(A)$ provide a rough measure of the Euclidean distance or

spread of columns (or atoms) of A , i.e., the Gram matrix provides a table whose entries measure a dictionary's pair-wise atomic separation [31].

In many previous publications concerning analysis of CS systems using overcomplete dictionaries, the goal has been to derive a single measure such as coherence (maximum Gram entry [22]) or average coherence (mean Gram entry [56]) that predicts the effectiveness a given sensing matrix or a sensing matrix drawn from some distribution, mainly as a function of the number of samples (or rows of the sensing matrix). However, in this work the goal is directed to a relative comparison of sensing matrices which provides insight into the overall intrinsic geometry of the dictionary, as well as the effect of the sensing matrix on the geometry of the dictionary. The motivating premise for this approach stems from the following reasons:

- The overall geometry of a dictionary is generally not accurately reducible to single-parameter simplification (e.g., the coherence)
- The literature concerning comparison of dictionary geometries appears to be lacking
- The methods used to analyze and compare dictionaries may be used to form the basis for more advanced CS tasks, e.g., dictionary learning

In summary, the maximum absolute off-diagonal inner product between any two pairs of dictionary atoms is the coherence. It provides a coarse idea of the most extreme correlation present in the dictionary. Although it is easy to compute, and provides some theoretical benefit, it provides very little insight into the geometry of the dictionary. Section 5 contains additional tools that can provide more geometric insight into the dictionary, albeit at the expense of additional computational resources. The tools in that section are aimed at evaluating a sensing matrix relative to an optimized sensing matrix.

3.5 Greedy Recovery Methods

3.5.1 Introduction

Adaptively constructing a signal via a small subset of the dictionary usually comes with the cost of increased computation and complexity of a nonlinear problem [16, 20, 24]. Linear approximation problems are considered *adaptive* since the set of atoms to include in order to approximate a signal (the support of the signal in the dictionary) is likely to change when the signal changes [16, 20].

Reconstructing a signal that is sparse in a dictionary represents a nonlinear problem; it is nonlinear because a linear combination of two or more k – *sparse* signals results in a signal that is no longer k – *sparse*, therefore any linear combination of k – *sparse* signals lies outside the space of k – *sparse* signals [16]. Each k – *tuple* of dictionary atoms forms a k -dimensional subspace within the ambient M -dimensional Euclidean vector space (since the total number of atoms in the dictionary is M , the collection of all dictionary atoms forms an M – *dimensional* vector space). Naively testing each set of k atoms (i.e. each subspace) one set at a time, out of M total atoms to see which set of atoms accurately represents the signal would require testing $\binom{M}{k}$ subspaces. Doing so is computationally prohibitive in many problems of interest [16] and as a result sparse adaptive signal approximation tools are sought. Compressive sensing (CS) can help reduce the computational burden by compressing each N -dimensional atom down to $m < N$ dimensions, as well to apply the principle of sparsity to reduce the feasible set of possible solutions.

Some classes of adaptive signal expansion methods include the method of frames, best orthogonal basis, greedy methods and ℓ_1 -norm minimization, a.k.a., basis pursuit (BP). One well-known greedy method is matching pursuit (MP), which uses a sequential selection scheme that iteratively builds a basis, one vector at a time. Typically this method does not give sparse representations [20,58]. The greedy method focused on in this dissertation is orthogonal matching pursuit (OMP), which is studied in Section 3.5.2.

The process of reconstructing an accurate representation (i.e. approximation or expansion) of a signal in terms of dictionary atoms is referred to as *signal recovery*. In addition to signal recovery, SA and CS can be applied to numerous signal processing applications (detection, estimation, clustering, etc.) [5]. Reconstructing a signal via a linear approximation requires two estimation sub-tasks:

- *Support estimation* of the set of dictionary column (atom) indices (a.k.a. the *support* of the signal in the dictionary), and
- Estimating the coefficients that are applied to the atoms in the linear combination .

Broadly, the published research regarding solving or estimating a sparse solution to a linear system of equations is divided into [22,35]:

- ℓ_1 -*Based Optimization* methods,
- Greedy Methods, and

- Thresholding Algorithms.

In ℓ_1 minimization, the sparse ℓ_0 model is simplified, i.e., relaxed (this is generally known as *convex* relaxation), and the ℓ_0 model is replaced by the simpler ℓ_1 objective. Various methods to solve the resulting ℓ_0 optimization problem are studied in the literature.

ℓ_1 optimization recovery methods rely on the objective function being convex. However, the native objective function in SA and CS is non-convex, namely SA and CS innately stipulate minimization of the ℓ_0 norm of a function. When using ℓ_1 optimization, the hopeful operating assumption is that the simpler ℓ_1 objective function adequately captures the sparse behavior of the more complex ℓ_0 objective. While this is easy to see in two dimensions, higher dimensions pose a challenge to visualize. A convex ℓ_1 optimization algorithm (i.e. a linear program) is then selected to solve the simplified ℓ_1 -based sparse model.

However, in greedy methods, a more complex ℓ_0 objective function is used, and instead of seeking an overall global optimal solution, an approximate solution is iteratively (*greedily*) built. Greedy (approximation or recovery) methods operate using the more complex ℓ_0 -based model. However, greedy methods in general are only locally optimal, in the sense that the choices made at a given iteration are optimal for that iteration only; the greedy choices already made may in the end result in a globally suboptimal solution. The significance of the term *greedy* is that, at each iteration of the algorithm, a hard threshold-based decision is made in accordance with some locally optimal criterion, i.e. the algorithm takes the best that it can get at each iteration. Much of the content of this section is adapted from the chapter [35], and is a good first source for understanding greedy algorithms. Greedy methods are generally conceptually easier than optimization methods, but typically offer less rigorous guarantees than convex optimization based methods. Hard thresholding strikes a greater balance between performance guarantees and ease of implementation [35].

Reference [35] defines greedy algorithms as a set of iterative methods that, at each step make a locally optimal decision, i.e. a step is taken that will give the maximum improvement of the objective function, at the current iteration, but there is no guarantee it will find the global optimum arguments that optimize the objective function.

Initialized with a zero vector, greedy methods iteratively estimate a set of indices at which the vector takes on non-zero values. This is referred to as the *greedy selection* step and is alternated with an estimation step in which the values for the non-zero components are computed. As a general rule, theoretical performance guarantees are typically weaker than those of convex-based methods [35].

3.5.2 Orthogonal Matching Pursuit

Orthogonal matching pursuit (OMP) is a subset of the larger framework of greedy recovery algorithms in CS. The term *recovery* is intended to convey the process by which compressed vectors (or more generally in some applications, matrices) are un-compressed, recovering the original un-compressed vector (matrix). The OMP algorithm has been well studied [35, 59, 60], and is listed in Algorithm 1, which is based on [60].

Algorithm 1 OMP Algorithm

Input: $A = [a_1, a_2, \dots, a_M] : mxM, y : mx1, k_{max}$
Initialize:
 $r^0 = y, \Lambda^0 = \emptyset, \ell = 0, \tilde{A} = [a_1 / \|a_1\|_2, a_2 / \|a_2\|_2, \dots, a_M / \|a_M\|_2]$
while $\ell \leq k_{max}$ **do**
 match: $h^\ell = \tilde{A}^T r^\ell$
 identify: $\Lambda^{\ell+1} = \Lambda^\ell \cup \left\{ \max_j |h^\ell(j)| \right\}$
 update LS estimate: $\hat{x}^{\ell+1} = \underset{z: \text{supp}(z) = \Lambda^{\ell+1}}{\text{argmin}} \|y - \tilde{A}z\|_2$
 update residual: $r^{\ell+1} = y - \tilde{A}\hat{x}^{\ell+1}$
 update index: $\ell = \ell + 1$
end while
Output: $\Lambda^{\ell+1}, \hat{x}^{\ell+1}, r^{\ell+1}$

Note that the OMP input matrix A can be taken to be either a dictionary or an effective dictionary, i.e. OMP can be applied to either SA or CS problems. The approximation for the vector x is updated iteratively by orthogonally projecting y onto the selected columns of the matrix A , written as A_T , where T is defined the currently selected index support set.

Within the LS estimation step, OMP minimizes the objective function

$$\hat{x}^{\ell+1} = \min_{z: \text{supp}(z) = \Lambda^{\ell+1}} \|y - \tilde{A}z\|_2. \quad (3.21)$$

Note that the constraint $z : \text{supp}(z) = \Lambda^{\ell+1}$ is a restriction on the vector z to be a k -sparse vector with $k = \#\Lambda^{\ell+1}$, where $\#\Lambda^{\ell+1} \equiv$ cardinality of the set $\Lambda^{\ell+1}$. Alternatively, the constraint can be written in terms of a subset of the columns of $\tilde{A}$

$$\hat{x}^{\ell+1} = \min_z \|y - \tilde{A}_{\Lambda^{\ell+1}} z\|_2, \quad (3.22)$$

where $\tilde{A}_{\Lambda^{\ell+1}}$ is the restriction of the matrix A to the columns specified by the index set $\Lambda^{\ell+1}$. The solution to this LS estimation can be accomplished as

$$\hat{x}^{\ell+1} = \tilde{A}_{\Lambda^{\ell+1}}^+ y, \quad (3.23)$$

where $\tilde{A}_{\Lambda^{\ell+1}}^+$ is the Moore-Penrose inverse of $\tilde{A}_{\Lambda^{\ell+1}}$ and is defined as

$$\tilde{A}_{\Lambda^{\ell+1}}^+ \equiv \left(\tilde{A}_{\Lambda^{\ell+1}}^H \tilde{A}_{\Lambda^{\ell+1}} \right)^{-1} \tilde{A}_{\Lambda^{\ell+1}}^H, \quad (3.24)$$

assuming that $\tilde{A}_{\Lambda^{\ell+1}}^H \tilde{A}_{\Lambda^{\ell+1}}$ is invertible [29].

In contrast with another well-known recovery method known as *matching pursuit* (MP), OMP performs a LS minimization over all of the previously selected coefficients. In addition, OMP will not select a given index more than once. Due to the LS orthogonal nature of OMP, the residual at each iteration is orthogonal to the subspace spanned by the atoms that have already been selected.

OMP is more complex than MP, mostly due to the orthogonalization (LS) step, but it is this step that also results in increased performance, specifically with respect to index selection accuracy. From a computational complexity perspective, the most costly operations involved in OMP are the matrix-vector products [35]. In addition to the naive approach, [36] presents three different implementations of the OMP algorithm using the QR decomposition, Cholesky decomposition, and the matrix inversion lemma. There also exist iterative algorithms for solving the LS portion of OMP, e.g. conjugate gradient methods [35].

One drawback of OMP is that, once an incorrect atom has been chosen during the matching portion, it is not identified and removed in subsequent iterations of the OMP algorithm. As a result, if an atom index has been selected that is outside the support of the input signal to be approximated, then more than k iterations of OMP are required to accurately approximate a signal that is k -sparse in the dictionary [22]. One potential solution to address this is to run the OMP algorithm for more than k iterations.

OMP is also known as the *orthogonal greedy algorithm* in approximation theory [35]. An alternative subset selection method based on the Viterbi algorithm has been explored, and is related to OMP and ordinary least squares (OLS) [61]. Although computationally more expensive than OLS, the Viterbi-based method is reported to have more accurate index selection accuracy than both OMP and OLS [61]. The OMP algorithm has been studied specifically with respect to the RIP in [40] providing some useful notation and observations of OMP in the context of orthogonal projections onto the dictionary and its orthogonal complement.

In [42], the authors look at coherence-based guarantees of OMP, specifically using the average coherence and the minimum-to-average ratio of the input signal. They use these quantities to define what they call the *strong coherence* property. If the strong coherence property is satisfied by the $m \times N$ sensing matrix, then when $m \approx O(k \log N)$ measurements are used, OMP is predicted to recover a k -sparse signal with high probability. However, the results they develop appear to be applicable only to k -sparse signals, i.e. only to orthonormal dictionaries.

In [62], the authors present some results (similar to [42]) that bound the performance of OMP for sparse signal recovery using the minimum magnitude of the sparse input signal. In [40], the authors provide a nice analysis of some seemingly innocuous statements regarding OMP in relation to the RIP. They show that if the sensing matrix obeys the RIP, then the residual vector formed within OMP can be shown to be bounded in a predictable way.

In [36], the authors provide three different implementations of the OMP algorithm, including methods that utilize a QR decomposition, the matrix inversion lemma, and the Cholesky matrix decomposition. The native or naive implementation of OMP is discussed and utilized in this dissertation.

3.5.3 Inner Product Correlation

Inner product correlation (IPC) is included in this dissertation to compare to the performance of OMP. IPC is essentially the first step of OMP, which is to match the atoms of the dictionary to the input signal via an inner product between the transposed dictionary and the input signal vector. It does not perform a LS orthogonalization, and so is less computationally complex, but it also does not have the ability to identify atoms as accurately as OMP.

To perform IPC, a *normalized correlation profile* is computed as:

$$z_j = |c_j| / \|\Psi_j\|_2, \quad (3.25)$$

where Ψ_j is the j^{th} column of Ψ , z_j is the j^{th} entry of the $M \times 1$ vector z , and c_j is the j^{th} entry of the vector c , defined as

$$c = \Psi^T x. \quad (3.26)$$

The vector z is a normalized correlation profile representing the relative similarity of each atom in

the dictionary to the input signal to be approximated x . The indices of the highest k elements of z correspond to the k atoms of the dictionary which are most similar to that of the input signal x . A sub-optimal k -sparse solution to the previously formulated optimization problem can be taken as $s = I$ where I is an index set corresponding to the indices of the top k entries of z . However, the potential linear dependency and redundancy present in the overcomplete dictionary Ψ tends to limit the effectiveness of this approach.

4 Sensing Matrix Optimization for Compressive Sensing

4.1 Introduction

Many publications have addressed the idea of performing CS in redundant (overcomplete) dictionaries (CSRD), where the sampling matrix is generated randomly e.g., [37, 39]. Conditions for successful recovery in CSRD were developed in [39]. Later research involved optimization of the sensing matrix by tailoring the sensing matrix to the dictionary being used [12]. Reference [45] was one of the first widely disseminated publications to suggest the optimization of the sensing matrix in the context of CS and developed what was termed an *t-averaged mutual coherence* property of the effective dictionary A as a predictive indicator of the performance of CS recovery. It is interesting to note that although [45] developed the *t-averaged mutual coherence*, the sensing matrix optimization algorithm presented in that publication used an objective function which did not directly utilize the average mutual coherence. Instead, it minimized the Frobenius norm of the difference between a target Gram matrix and the computed Gram matrix in order to compute a sensing matrix that reduced the Frobenius norm of the entries of the effective dictionary Gram matrix. Other methods of sensing matrix optimization that followed [45] were presented in [25] and [49].

In non-orthogonal dictionaries, randomly selected sensing matrices have been shown to be less effective in terms of recovery performance when compared with optimized sensing matrices, which are constructed with knowledge of the sparsifying dictionary [12]. This was initially demonstrated in using randomly generated dictionaries and later extended to images [25]. Instead of striving to maintain orthogonality (as CS with orthogonal dictionaries does), a main goal of sensing matrix design in CS with over complete dictionaries is to scale and rotate the dictionary atoms so as to separate the atoms far apart. Various CS sensing matrix optimization schemes have been developed in the literature e.g., [45], [48], [25].

The *effective* (or *projected*) dictionary is the compressed dictionary given by $A = \Phi\Psi$. The Gram matrix of the effective dictionary was defined in Section 3.4.2 as:

$$G(A) \equiv A^T A, \quad (4.1)$$

for $A \in \mathbb{R}^{m \times M}$ [17, 22, 57]. This dissertation is mainly concerned with the absolute value of the entries of $G(A)$. The Gram matrix of a complex matrix is defined similarly, except that the transpose operation is replaced by the complex conjugate operation. The Gram matrix is also sometimes referred to as the *Gramian* matrix in the literature [17, 18]. Note that this is different from the *controllability Gramian* that is used in control theory [63]. In addition, while not the focus of this dissertation, it is noted that the theory underlying CSRD is tightly coupled with that of *frames* [12, 64, 65]. Frame theory is a deep and well-studied area. While there are many good references available a helpful tutorial can be found in [21, 66].

4.2 Class of Optimal Effective Dictionaries

Before deriving an optimal *sensing matrix*, it is of benefit to study the geometry of the desired *effective dictionary* that is formed from using an optimal sensing matrix. The effective dictionary that is formed from the product of the optimal sensing matrix and the dictionary for which it was optimized is known as the *optimal effective dictionary*. The geometry of the atoms of a dictionary are described by the Gram matrix of the dictionary and an aggregate measure of the Gram matrix would be helpful. Specifically, an overall measure of the spread or angular separation of the all atoms in the dictionary stored in the Gram matrix would prove useful. As a succinct measure of atomic separation, the Frobenius norm is often used to summarize the Gram matrix of the effective dictionary $G(A)$ [25, 45, 48], and therefore to summarize the average geometric separation of the columns of A . The *Frobenius* norm of a real matrix A is defined as:

$$\|A\|_F = \sqrt{\text{tr}(A^T A)} = \sqrt{\sum_{i=1}^m \sum_{j=1}^M |a_{ij}|^2} = \sqrt{\sum |\sigma_i(A)|^2}. \quad (4.2)$$

In addition, a search of the current literature indicates that sensing matrix comparisons and sensing matrix optimization methods commonly utilize the Frobenius norm of the effective dictionary's Gram matrix [12, 25, 45, 49], particularly within the objective function. An objective function which strives to minimize the off-diagonal entries of the Gram matrix of the effective dictionary A can be formulated as:

$$\underset{\Phi \in \mathbb{R}^{m \times N}}{\text{minimize}} \quad \|I_L - A^T A\|_F^2 = \|I_L - G(A)\|_F^2, \quad (4.3)$$

where $A \equiv \Phi\Psi$ [12]. While various iterative schemes have been proposed to find an optimal Φ for a given Ψ (e.g., [25, 67]), the method in [12] has derived a general closed-form expression for the *class* of optimal sensing matrices in terms of the singular values and left-singular vectors of the dictionary. The method presented in [12] first derives a necessary and sufficient form that the optimal effective dictionary must take, i.e., the form of the effective dictionary when the optimal sensing matrix is used. It is given as

$$A_{opt} = U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T, \quad (4.4)$$

where $U \in \mathbb{R}^{m \times m}$, $V \in \mathbb{R}^{\tilde{N} \times \tilde{N}}$ are arbitrary orthogonal matrices. The derivation of Eq. (4.4) can be found in 9.3.1. This condition stipulates that the m singular values of the optimal effective dictionary are all ones. When the effective dictionary has this form, then the Gram matrix of the effective norm is as close to an appropriately sized identity matrix as possible, using the Frobenius norm in the objective function. The objective function in Eq. (4.3) is defined as

$$\rho \equiv \|I_L - A^T A\|_F^2. \quad (4.5)$$

The Gram matrix formed from the optimal effective dictionary makes the objective function in Eq. (4.3) as low as possible. As [12] outlines, the objective function value when the optimal effective dictionary is used is given by

$$\rho_{min} = M - m, \quad (4.6)$$

i.e., the difference between the number of atoms M and the number of compressive measurements m . A detailed proof of this result is given in Section 9.3.3. Reference [12] also notes that there exist some degrees of freedom present in the optimal effective dictionary. For example, V and U are arbitrary orthogonal matrices. I.e., as long as these two matrices are chosen to each be orthogonal, a minimum of the objective function will be achieved.

4.3 Class of Optimal Sensing Matrices

Having developed a general form for an optimal effective dictionary that minimizes the residual error of the off-diagonal elements of the effective dictionary's Gram matrix, the stage is now set to derive a class of sensing matrices that result in an optimal effective dictionary. Reference [12] provides a general closed-form expression for the optimal sensing matrix that will produce an optimal effective dictionary (an optimal spreading of its atoms) and therefore have minimum objective function residual error, i.e., $\rho = \rho_{min} = M - m$. The form of the optimal sensing matrix is:

$$\Phi_{opt} = U \begin{bmatrix} I_m & 0 \end{bmatrix} \begin{bmatrix} V_{11}^T \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T, \quad (4.7)$$

where $U \in \mathbb{R}^{m \times m}$, $V_{11}^T \in \mathbb{R}^{\bar{N} \times \bar{N}}$ are arbitrary orthogonal matrices and the singular value decomposition of the dictionary is given by $\Psi = U_d \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} V_d^T$. The sensing matrix Φ_{opt} is optimal in the sense that, for the given dictionary Ψ , the sensing matrix Φ_{opt} minimizes the effective Gram's Frobenius norm. Alternatively stated, when the sensing matrix Φ is selected in accordance with the form specified in Eq. (4.7), the effective dictionary A has the form given in Eq. (4.4). The derivation of Eq. (4.7) can be found in Section 9.3.4.

It should be stressed that the above equation provides a closed-form expression for a not a single matrix, but a *class* of matrices that satisfy the minimum effective Gram's Frobenius norm. Similar to the optimal effective matrix equation, reference [12] notes that there exist some degrees of freedom present in the optimal sensing matrix. For example, V_{11} and U are arbitrary orthogonal matrices. I.e., as long as these two matrices are chosen to each be orthogonal, a minimum of the objective function will be achieved. As long as the matrices V_{11} and U are each orthogonal matrices, they may be selected in whatever way is deemed best for the given problem.

In order to build some intuition for the CS matrix optimization a simple analysis experiment was conducted to begin to visually grasp understanding of the optimization in low dimensions. An $N \times M$ dictionary Ψ was constructed by randomly populating a 3×4 matrix with entries randomly drawn from a uniform distribution. An $m \times N$ random sensing matrix Φ_{rand} was constructed by randomly populating a 2×3 matrix per Eq. (3.9) using a normal distribution with zero mean and variance

$1/m = 1/3$. An optimal sensing matrix was constructed via Eq. (4.7) using random orthogonal matrices generated for U and V_{11} . Fig. 3 shows the results of the experiment and the effect on the geometric spread of the atoms. The original dictionary atoms shown in subfigure (a) in three dimensions are compressed into two dimensions using both a random sensing matrix (subfigure (b)) as well as an optimized sensing matrix (subfigure (c)). The sensing matrix optimization objective function value is shown for each case. It is seen that the optimized sensing matrix results in a lower objective function value than the random case, and that the effect of the optimized sensing matrix is to spread the vectors out more so than in the random sensing matrix case. To generate these graphs, the following procedure was used:

1. At the MATLAB command prompt, enter
`optSampMatrixIntroDemo` and hit *< Enter >*.
2. Wait for the function to complete.
3. Upon completion, the graphs are displayed and are saved to the current MATLAB directory path.

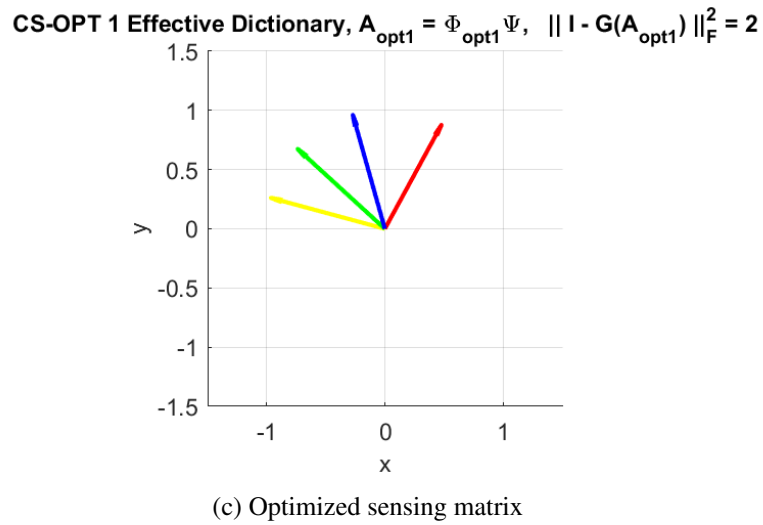
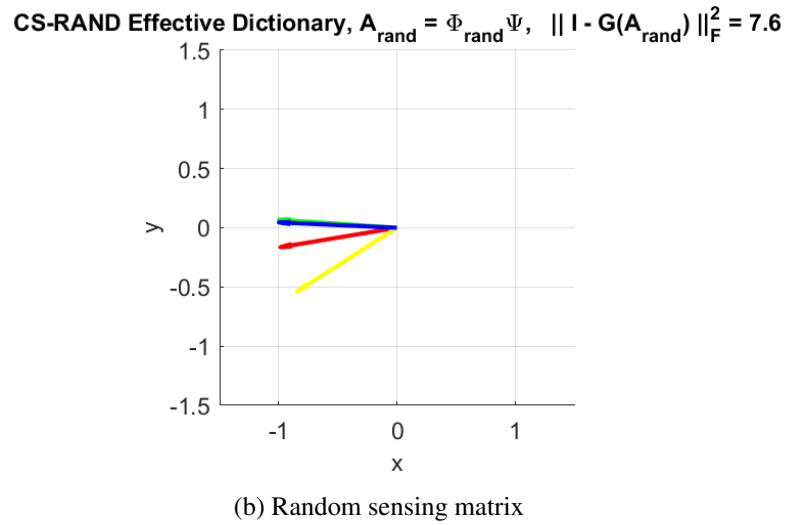
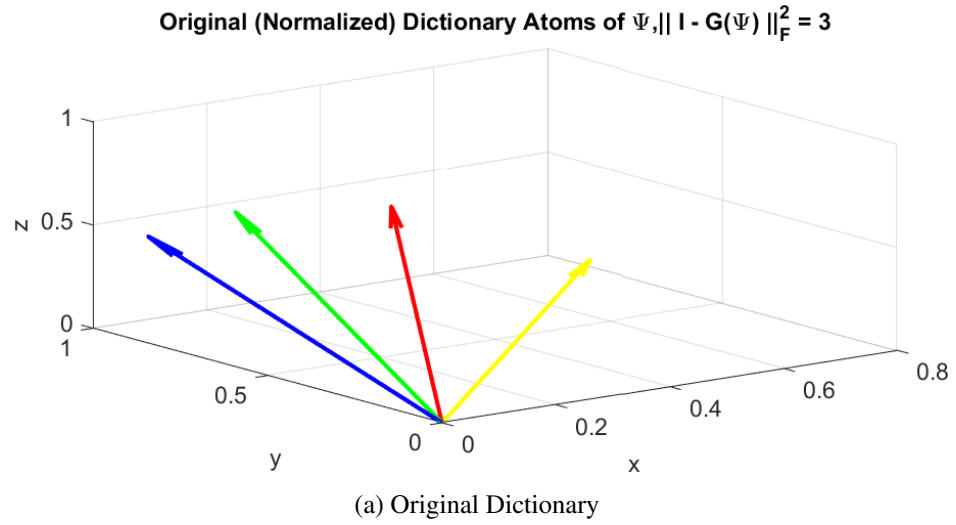


Figure 3: Intuitive demonstration of the effect on the geometry of dictionary atoms in low dimensions.

4.4 Updating the Optimal Sensing Matrix in Response to Dictionary Changes

As the dictionary changes the associated optimized sensing matrix may need to be updated as well. However, sensing matrix optimization requires the computation of either an eigenvalue or singular value decomposition. Instead of updating the sensing matrix each time the dictionary changes, it is possible that the sensing matrix may need to be updated only after the dictionary has been changed enough, i.e., the that perhaps the sensing matrix updating when the dictionary change significantly affects the geometry of the overall effective dictionary $A = \Phi\Psi$.

In light of these points, three items of study are noted:

- Other than using the objective function in Eq. (4.3), how can the optimality of a sensing matrix be evaluated with respect to a given dictionary?
- What is the effect on support estimation if a stale sensing matrix is used on an updated dictionary, or if an optimized sensing matrix is used on a perturbed version of the dictionary for which it was optimized?
- Since the optimized sensing matrix is based on the SVD of the dictionary, how can the SVD be updated efficiently?

The first bullet is addressed in this dissertation by comparing the Gram of the stale effective dictionary to a Gram of the optimal (updated) effective dictionary, by developing tools to compare Gram matrices and distances between Gram matrix elements as developed in Section 5. The second bullet will be addressed empirically in Section 7. As there appear to be multiple methods to efficiently compute the SVD of a rank-1 updated matrix (e.g. see Section 12.5 of [68]), the third bullet is a good candidate for future research beyond the scope of this current dissertation.

With regards to real-time implementation of sensing matrix optimization in a setting where the dictionary might periodically change, two key tasks that need to be performed are to:

- Evaluate the sensing matrix, i.e. measure the extent of non-optimality (deviation from the Gram of an optimized effective dictionary) induced by the stale sensing matrix/updated dictionary combination
- Decide whether or not to update the sensing matrix

Deciding when to update the sensing matrix is highly application-dependent. However, two approaches are now described for deciding when to update the sensing matrix. One approach to

refreshing the sensing matrix is to blindly update the sensing matrix, on a *periodic basis*. Another approach may be to update the sensing matrix on a *data-driven basis*. That is, the sensing matrix is updated whenever the dictionary changes, whether it changes by a miniscule amount or a great deal. Depending on the effect of the sensing matrix on the geometry of the updated dictionary, the stale sensing matrix may still be very close to optimal. That is, the mere fact that the dictionary undergoes an update may not necessitate updating the sensing matrix. The dictionary update may have caused the projected atoms to become closer (equating to worse CSRD atom selection), farther away (better CSRD atom selection) from each other, or to not have changed their relative position much at all.

4.5 Related Methods and Topics

The method used in this dissertation and described in [12] for optimal sensing matrix design is related to some other methods published in the literature which utilize CS for spectrum cartography [7], [8] and sparse target localization [11]. These publications use a common so-called pre-processing step that acts to orthogonalize the effective dictionary $A = \Phi\Psi$. The pre-processing calls for first finding the Moore-Penrose pseudoinverse of A , denoted as A^+ . Next an orthogonal basis is found for the range of A^T . Then, the effective matrix is pre-multiplied by $T = QA^+$, where $Q = \text{orth}(A^T)^T$. The processed effective dictionary is then

$$A_{proc} = TA = QA^+. \quad (4.8)$$

When orthogonalized, the compressed measurements $y = Ax$ become

$$\begin{aligned} y &= A_{proc}Ax \\ &= QA^+Ax \\ &= Qx. \end{aligned} \quad (4.9)$$

Now the rows of Q form an orthogonal basis and therefore Q satisfies the RIP. Alternatively, the method of [12] has a similar approach and the resulting optimized sensing matrix is in general more incoherent with the dictionary than a randomly generated sensing matrix. The resulting optimized effective dictionary is more orthogonalized than the input dictionary. The work in this dissertation based on [12] presents a more generalized form for an optimal sensing matrix (Eq. (4.7)) than that

shown in (4.8), allowing for additional degrees of freedom in the optimal sensing matrix design.

A technique known as (orthogonal) *Procrustes analysis* is related to, but different from sensing matrix optimization. Procrustes analysis has been used to reconstruct a set of input data points so that the reconstruction aligns with a set of anchor points [54]. After centering the data, the method involves finding a rotation and reflection that optimally maps (in a LS sense) the set of centered points X to the set of anchor points Y , i.e. finding the orthogonal matrix Q that minimizes the Euclidean distance in the following problem:

$$\underset{Q: QQ^T=I}{\text{minimize}} \quad \|QX - Y\|_F^2. \quad (4.10)$$

The orthogonal Procrustes problem seeks the orthogonal matrix Q that is applied to the centered data X , whereas in the sensing matrix optimization problem in this dissertation the sensing matrix sought (Φ) is rectangular $m < N$. Also in sensing matrix optimization, the objective function involves constraints on the Gram of the effective dictionary matrix, whereas in the orthogonal Procrustes problem, the objective function explicitly involves the data matrix.

5 Sensing Matrix Evaluation

5.1 Introduction

5.1.1 Purpose

This chapter introduces and formulates the problem of sensing matrix evaluation (SME) in the context of compressive sensing in redundant dictionaries (CSRD). Suppose that an ‘optimal’ sensing matrix is computed for a particular dictionary, as per the methods described in Sec. 4.3. Optimality is defined with respect to the sensing matrix objective function Eq. (4.5), i.e. the Frobenius norm of the off-diagonal entries of the Gram matrix of the effective dictionary attains the minimum value ρ_{min} as defined in Eq. (4.6). Now suppose that the effective dictionary is changed by either altering the dictionary (while keeping the sensing matrix the same), or altering the optimized sensing matrix (while keeping the dictionary the same). Altering the effective dictionary matrix will likely change the Euclidean distance between pairs of the effective dictionary atoms, thereby changing the geometry of the space that is spanned by the effective dictionary atoms.

Section 3.4 and Section 4 discussed using the Gram matrix $G(A) = A^T A$ of the effective dictionary $A = \Phi\Psi$ to describe the geometry of the space spanned by the columns of A . The off-diagonal Gram matrix entries are related to the Euclidean distance between pairs of effective dictionary atoms (columns). Therefore, comparing two Gram matrices ($G(A_1)$ and $G(A_2)$) reveals information about the geometric differences between the effective dictionaries A_1 and A_2 . Specifically, finding the distance between the Gram matrix in the stale case and the Gram matrix in the optimized case provides a basis to discern between a stale sensing matrix and an optimized one.

This chapter explores efficient computational methods for comparing two Gram matrices. There are multiple mathematical tools already developed to measure the distance between a pair of distributions, as well as distances between matrices (in particular, distances between *symmetric* matrices). This chapter reviews some of these methods and applies them to find the distance between Gram matrices of effective dictionaries for CSRD sensing matrix evaluation. The primary intended application of such Gram matrix comparisons is aimed at evaluating the *staleness* of a sensing matrix due to a time-evolving dictionary.

5.1.2 Chapter Organization

Section 5.2 defines some symbols and nomenclature that proves useful in the context of sensing matrix evaluation. Section 5.3 describes two ways to construct stale effective dictionaries: either by altering the optimized sensing matrix or altering the dictionary that was used during sensing matrix optimization. Section 5.4 outlines the problem, and introduces the notion of a generalized difference operator for measuring differences between a pair of sensing matrices to ascertain their suitability for a particular dictionary. Section 5.5 describes three approaches to compare Gram matrices, including utilizing the sensing matrix optimization (SMO) objective function, applying distribution function distances to Gram matrix entries, and using more advanced techniques to compute distances between Gram matrices within the space of positive semidefinite matrices (PSD) along a convex cone. While the sensing matrix objective function and Gram matrix distances are both valid approaches to Gram matrix comparison (and therefore to sensing matrix comparison), the focal method that is studied is to compare Gram matrices on the basis of the distribution of their entries. Therefore, in Section 5.6, an in-depth analysis of an approach to Gram matrix comparison by comparing the distribution of entries of each Gram matrix.

5.2 Optimized and Stale Effective Dictionary Nomenclature

In order to provide a clarifying structure, some nomenclature and symbols are now defined which are intended to facilitate the presentation. A dictionary updated at time t is denoted Ψ_t . An *update* to a dictionary is defined as a change to one or more of its elements, specifically a change to one or more of the dictionary columns. The sensing matrix that is optimized for Ψ_t (per the method presented in Sec. 4.3) is represented by Φ_t . The resulting effective dictionary is referred to as an *updated effective dictionary*, and is represented as $A_t \equiv \Phi_t \Psi_t$.

As time proceeds beyond t , the dictionary Ψ_t evolves to $\Psi_{t+1}, \Psi_{t+2}, \dots$ and it becomes possible that the sensing matrix Φ_t (optimized for Ψ_t) may no longer minimize Eq.(4.5), the objective function used in sensing matrix optimization. If an effective dictionary is composed of a sensing matrix Φ_{t_1} and a recently updated dictionary Ψ_{t_2} , where $t_1 \neq t_2$ then Φ_{t_1} is called a *stale sensing matrix*. The resulting matrix product $A_{t_1, t_2} = \Phi_{t_1} \Psi_{t_2}$ is termed a *stale effective dictionary*. The tools in this chapter are intended to measure the differences in geometry between A_{t_1} and A_{t_1, t_2} .

Table 2 summarizes the various variables that are of use in this context. If the sensing matrix is not accordingly updated, then the geometry of the resulting effective dictionary as measured by the Gram matrix will likely change.

Table 1: Variable Definitions for Stale and Updated Sensing Matrices and Dictionaries

Variable Name	Symbol	Summary
Dictionary	Ψ_t	Dictionary updated at time $t \in 0, 1, 2, \dots$
Optimized Sensing Matrix	Φ_t	Sensing matrix that has been optimized for Ψ_t .
Updated Effective Dictionary	$A_t = \Phi_t \Psi_t$	Effective dictionary formed from updated sensing matrix and updated dictionary.
Stale Effective Dictionary	$A_{t_1, t_2} = \Phi_{t_1} \Psi_{t_2}$	Effective dictionary formed from <i>stale</i> sensing matrix (optimized for dictionary that was last updated at time t_1) and updated dictionary (updated at time t_2), where $t_1 \neq t_2$.
Updated Gram Matrix	$G(A_t) = A_t^T A_t$	Gram matrix formed from updated effective dictionary.
Stale Gram Matrix	$G(A_{t_1, t_2}) = A_{t_1, t_2}^T A_{t_1, t_2}$	Gram matrix formed from stale effective dictionary.

5.3 Constructing Stale Effective Dictionaries

To model the time evolution of the dictionary and achieve a *stale* or *perturbed effective dictionary*, columns of the original dictionary are altered. There are two methods explored to achieve a stale or perturbed effective dictionary. Either 1) use a single sensing matrix and evolve the dictionaries from Ψ_1 through Ψ_4 , or 2) construct optimized sensing matrices Φ_1 through Φ_4 based on perturbed versions of the dictionary Ψ_1 through Ψ_4 , and then multiply each sensing matrix by the original dictionary Ψ_0 , i.e.:

- **Method 1: Dictionary Evolution:** Construct an optimal sensing matrix based on the unperturbed dictionary and multiply each perturbed dictionary Ψ_i by the optimal sensing matrix Φ_0
- **Method 2: Sensing Matrix Evolution:** For each perturbed dictionary Ψ_i , construct four sensing matrices Φ_t and multiply the original unperturbed dictionary Ψ_0 by each sensing

matrix Φ_t

Method 1 (dictionary evolution) involves the following steps:

1. Construct an optimized sensing matrix for the un-perturbed dictionary Ψ_0 to form Φ_0
2. Construct four versions of the dictionary $\Psi_1, \Psi_2, \Psi_3, \Psi_4$, each of which differ increasingly from Ψ_0
3. Multiply Φ_0 by each of the four dictionaries to form four stale effective dictionaries
 $A_{0,1} = \Phi_0 \Psi_1, A_{0,2} = \Phi_0 \Psi_2, \dots$, etc.

The process for Method 1 (dictionary evolution) is illustrated in Fig. 4.

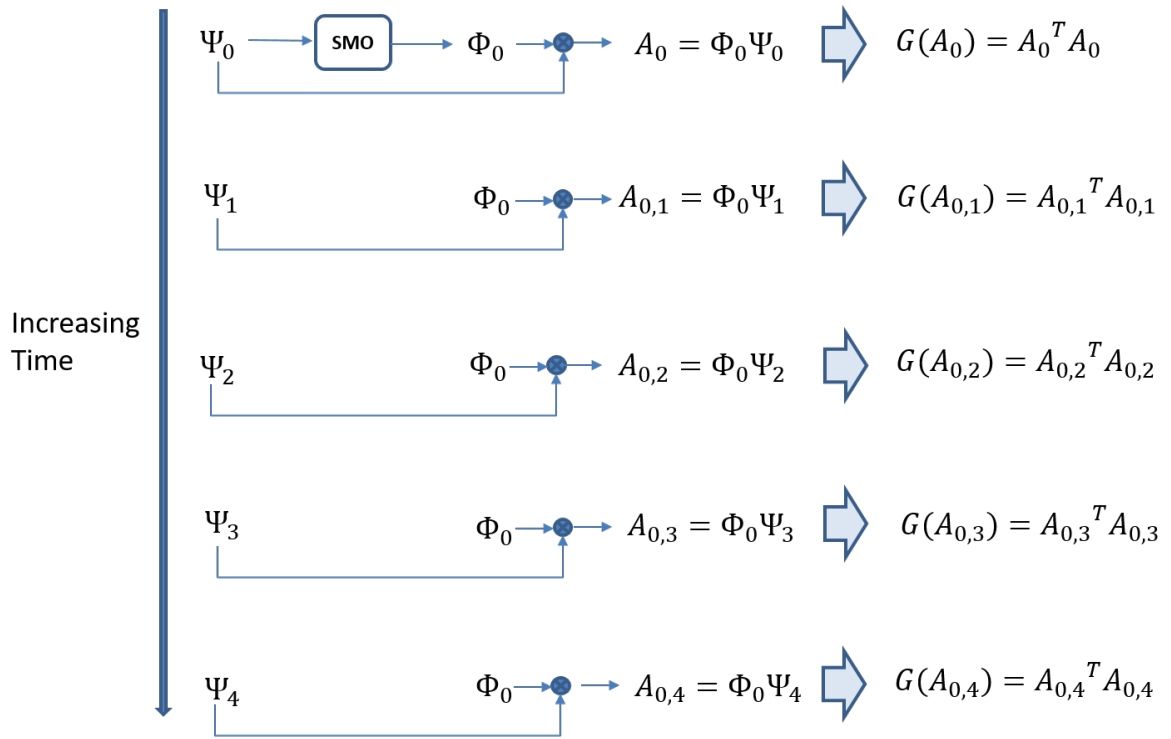


Figure 4: Method 1: Dictionary Evolution for Stale Effective Dictionary Construction. After sensing matrix optimization (SMO), evolution of the the dictionary over time results in an increasingly stale effective dictionary matrix. An optimized sensing matrix may not remain so if it is not re-optimized for the current dictionary. The potential effect is a change in the geometry of the effective dictionary, as measured by its Gram matrix. Comparing stale Gram matrices to the updated Gram matrix computed at the time of SMO can aid in measuring the effectiveness of the sensing matrix in orthogonalizing the most recently updated dictionary.

Method 2 (sensing matrix evolution) involves the following steps:

1. Construct an optimized sensing matrix for the un-perturbed dictionary Ψ_0 to form Φ_0
2. Construct four versions of the dictionary $\Psi_1, \Psi_2, \Psi_3, \Psi_4$, each of which differ increasingly from Ψ_0

3. Construct four optimized sensing matrices using each of the four different dictionaries to form $\Phi_1, \Phi_2, \Phi_3, \Phi_4$
4. Multiply Φ_i for $i = 1, 2, 3, 4$ by the un-perturbed dictionary Ψ_0 to form four stale effective dictionaries $A_{1,0} = \Phi_1 \Psi_0, A_{2,0} = \Phi_2 \Psi_0, \dots$, etc.

Both methods require the construction of an optimized sensing matrix based on the un-perturbed dictionary, and the construction of four perturbed dictionaries, therefore *Step 1)* and *Step 2)* for Method 2 are the same as in Method 1. The process for sensing matrix evolution is illustrated in Fig. 5.

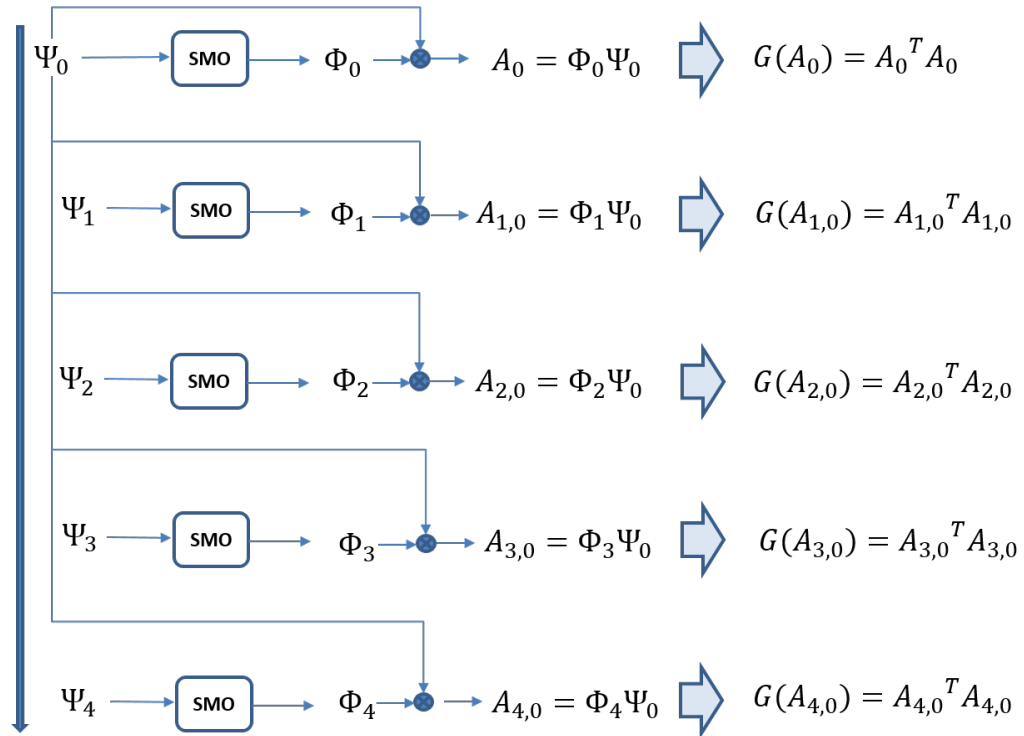


Figure 5: Method 2: Sensing Matrix Evolution for Stale Effective Dictionary Construction. After SMO, evolution of the the sensing matrix over time results in an increasingly perturbed effective dictionary. An optimized sensing matrix may not remain so if it is not re-optimized for the current dictionary. The potential effect is a change in the geometry of the effective dictionary, as measured by its Gram matrix. Comparing stale Gram matrices to the updated Gram matrix computed at the time of SMO can aid in measuring the effectiveness of the sensing matrix in orthogonalizing the most recently updated dictionary.

In Method 1 (Dictionary Evolution) the (updated) optimized sensing matrix Φ is multiplied by each of the four increasingly perturbed versions of the original dictionary Ψ_0 to form the four stale effective dictionaries. In Method 2 (Sensing Matrix Evolution) each of the four sensing matrices are optimized using an increasingly perturbed version of Ψ_0 , and multiplied then by the original dictionary Ψ_0 to form the four stale effective dictionaries.

Table 2: Two methods for stale effective dictionary construction and the resulting effective dictionary in each case.

Time (t)	Method 1: Dictionary Evolution	Method 2: Sensing Matrix Evolution	Percent of Columns Different between Ψ_t and Ψ_0
1	$A_{0,1} = \Phi_0 \Psi_1$	$A_{1,0} = \Phi_1 \Psi_0$	25%
2	$A_{0,2} = \Phi_0 \Psi_2$	$A_{2,0} = \Phi_2 \Psi_0$	50%
3	$A_{0,3} = \Phi_0 \Psi_3$	$A_{3,0} = \Phi_3 \Psi_0$	75%
4	$A_{0,4} = \Phi_0 \Psi_4$	$A_{4,0} = \Phi_4 \Psi_0$	100%

For both Method 1 and Method 2, the relationship between the original unperturbed dictionary Ψ_0 and the perturbed dictionaries are:

- Ψ_1 : 25% of the columns are different from Ψ_0
- Ψ_2 : 50% of the columns are different from Ψ_0
- Ψ_3 : 75% of the columns are different from Ψ_0
- Ψ_4 : 100% of the columns are different from Ψ_0

Table 3 lists the effective dictionaries constructed for each stale construction method.

5.4 Problem Formulation and the Difference Operator

In order to formulate the sensing matrix evaluation problem, it is first assumed that sensing matrix optimization has been performed, as outlined in Eq. (4.7) of Sec. 4.3 (repeated here for convenience)

$$\Phi_{opt} = U \begin{bmatrix} I_m & 0 \end{bmatrix} \begin{bmatrix} V_{11}^T \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T,$$

such that a sensing matrix resulting in the minimum objective function value (defined in Eq. (4.3) and repeated below) is found as

$$\underset{\Phi \in \mathbb{R}^{m \times N}}{\text{minimize}} \quad \|I_L - A^T A\|_F^2 = \|I_L - G(A)\|_F^2 \equiv \rho_{min} = M - m,$$

with respect to a particular dictionary Ψ , where $SVD(\Psi) = U_d \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} V_d^T$. The matrix U is an arbitrary orthonormal matrix, so it represents a “substantial degree of freedom” in finding an optimal sensing matrix [12]. In this work, the entries of the matrix U are computed by first randomly selecting them according to a standard normal distribution (i.e., a discrete normally-distributed random

variable with mean 0 and standard deviation 1 represented as $U \sim \mathcal{N}(0, 1)$ and then orthonormalizing columns of U (possibly through the Gram-Schmidt orthogonalization procedure), and that $V_{11} = I_{\bar{N}}$, i.e. the identity matrix of appropriate dimension $\bar{N}$.

Using these assumptions, the sensing matrix optimization finds one of many sensing matrices that attains the minimum objective function value ($\rho_{min} = M - m$) with respect to the dictionary Ψ . After sensing matrix optimization occurs, if the dictionary changes, e.g., entries or a row or column of Ψ is updated, then it is possible that the previously optimized sensing matrix ceases to be optimal.

To measure the difference between the Gram matrix of an updated effective dictionary $G(A_{t_2})$ relative to a stale effective dictionary $G(A_{t_1, t_2})$ define a general *difference operator* $\mathcal{D}$ as

$$\mathcal{D}_{meas} \{G(A_{t_1, t_2}), G(A_{t_2})\}, \quad (5.1)$$

where $G(A_{\bullet})$ is an $M \times M$ Gram matrix for the $m \times M$ matrix A . The general difference operator $\mathcal{D}_{meas}$ is made specific by indicating a particular measure via the subscript. There are three approaches presented to measuring difference between Gram matrices, which are addressed in the next section.

5.5 Defining Three Approaches to Sensing Matrix Evaluation

In this sub-section, the three Gram matrix comparison approaches are reviewed at a high level, which are aimed at measuring the effectiveness of an optimized sensing matrix in spreading out the atoms of a given dictionary (specifically to compare the pairwise spread of atoms between optimized and non-optimized effective dictionaries). More generally, the methods presented allow one to compare the geometry of two data sets in terms of their respective Gram matrices.

Clearly, if one desired to detect a change between two dictionary matrices Ψ_{t_1} and Ψ_{t_2} , then the Frobenius norm of the difference between dictionary matrices can be computed, i.e., $\|\Psi_{t_1} - \Psi_{t_2}\|_F$. However, this difference on its own does not provide any insight into the effect of the sensing matrix Φ_{t_1} on the dictionaries. Therefore, the effective dictionaries $A_{t_2} = \Phi_{t_2} \Psi_{t_2}$ and $A_{t_1, t_2} = \Phi_{t_1} \Psi_{t_2}$ are analyzed instead of the mere dictionaries.

CSRD performance is enhanced when using sensing matrices that spread out the atoms of the dictionaries. Therefore, in addition to operating on the effective dictionaries, the difference in geo-

metric spread between optimized and stale effective dictionary atoms is important to capture. Consequently, the Gram matrix of each effective dictionary are chosen the objects of study, namely $G(A_{t_2})$ and $G(A_{t_1, t_2})$, because the entries of the Gram matrix encode information about the Euclidean distance between dictionary atoms. Within this context it is noted that [26] defines a distance measure between dictionaries in the non-compressive setting. The measure in [26] is interesting because it is independent of the order of the dictionary columns and is based on the sum of angles formed between atoms from one dictionary and the the pairwise closest corresponding atoms in the other dictionary.

In this subsection, three approaches are explored for *sensing matrix evaluation* (SME), i.e. comparing a pair of Gram matrices, where each Gram matrix is formed from two versions of an effective dictionary. Three approaches are presented for Gram matrix comparison as applied to SME:

- **Approach 1:** Evaluate the sensing matrix *optimization objective function*, $\mathcal{D}_\rho$
- **Approach 2:** Measure the distance between the sets of each Gram matrices entries
 - Kolmogorov-Smirnov test: $\mathcal{D}_{KS}$
 - Kullback-Leibler divergence: $\mathcal{D}_{KL}$
- **Approach 3:** Measure the distance between the Gram *matrices* within the set of symmetric, positive-semi-definite matrices

SME approaches 1 and 2 are emphasized in this work. Under approach 3, additional difference operators can be defined with regards to distance metrics between Gram matrices, however such operators will not be extensively discussed in this dissertation. Some measures associated with Approach 3 are only generally outlined in Sec. 5.5.3 and are listed for completeness.

5.5.1 Approach 1: Sensing Matrix Optimization Objective Function

One possible (and logical) technique to measure differences between the geometries of two effective dictionaries is to recompute the value of the sensing matrix objective function ρ (Eq. (4.5)) used in the sensing matrix optimization equation, but using the stale effective dictionary to compute the

difference in optimality. The sensing matrix optimization objective function is repeated below

$$\begin{aligned}\rho &\equiv \|I_L - A^T A\|_F^2 \\ &= \|I_L - (\Phi\Psi)^T(\Phi\Psi)\|_F^2,\end{aligned}$$

where Φ is a sensing matrix that was constructed via the optimization method of section 4.3. Re-computing the optimization objective function with the stale version of the sensing matrix, such that the Gram of the stale effective dictionary $G(A_{t_1,t_2})$ is used, results in an objective function value of greater than the minimum value of $M - m$

$$\rho_{t_1,t_2} = \|I_L - G(A_{t_1,t_2})\|_F^2 \geq M - m. \quad (5.2)$$

Equality is attained when an optimal sensing matrix is used

$$\rho_{t_2} = \|I_L - G(A_{t_2})\|_F^2 = M - m \equiv \rho_{min}. \quad (5.3)$$

Subtracting these two objective function values gives the difference measure $\mathcal{D}_\rho$

$$\begin{aligned}\mathcal{D}_\rho \{G(A_{t_1,t_2}), G(A_{t_2})\} &= |\rho_{t_1,t_2} - \rho_{min}|, \\ &= \left| \|I_L - G(A_{t_1,t_2})\|_F^2 - \rho_{min} \right|, \\ &= \left| \|I_L - (\Phi_{t_1}\Psi_{t_2})^T(\Phi_{t_1}\Psi_{t_2})\|_F^2 - \rho_{min} \right|,\end{aligned} \quad (5.4)$$

where Φ_{t_1} is a sensing matrix that was constructed via the optimization method of section 4.3, but using a previous (stale) version of the dictionary, Ψ_{t_1} . Therefore the construction can no longer be guaranteed to be optimal in the sense of minimizing the objective function, with respect to the current dictionary Ψ_{t_2} .

Blindly recalculating the objective function periodically after dictionary updates can be useful. However, doing so gives little insight into the overall geometric difference between the updated and stale effective dictionaries. E.g., after a dictionary updated, the stale sensing matrix may cause some of the Gram entries of the effective dictionary to increase in absolute value (atoms may move farther apart), while simultaneously causing an equal number of other Gram entries to decrease in absolute value (atoms may move closer together). In this case, the sum of Gram entries computed in the objective function would not differ greatly between the stale and updated effective dictionaries, and

so would not be an effective method to ascertain the associated geometric difference in the effective matrix due to a stale sensing matrix. The sensing matrix optimization using the method of section 4.3 is guaranteed to minimize only the *sum* of the off-diagonal entries of the Gram matrix of the effective dictionary. Note that the mean of the off-diagonal entries of the Gram matrix is just a scaled version of the sum of the off-diagonal entries.

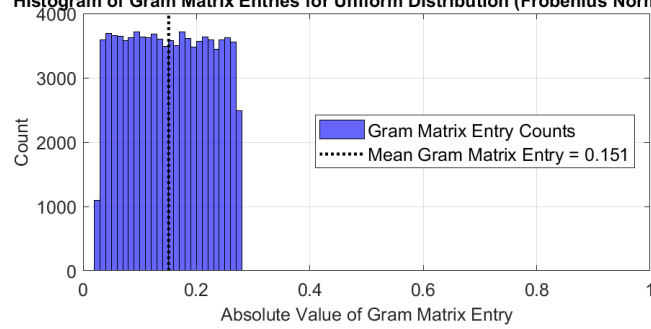
Viewing the off-diagonal entries of the Gram matrix as a collection of data elements with a probability mass function, the mean of this distribution provides a measure of the center of the distribution only, and the ability to measure a change in the effect of the sensing matrix on the *geometry* of the dictionary is somewhat limited. Instead of simply analyzing a single statistic like the maximum value or the mean, a more geometrically comprehensive statistic can be computed by comparing the entire distributions of off-diagonal Gram matrix entries obtained from the stale and updated Gram matrices, as is studied next in Sec. 5.5.2.

5.5.2 Approach 2: Distances between Sets of Gram Matrix Entries

As an alternative to comparing the mean of Gram matrix entries, the distance between Gram matrix entry distributions provides more detailed geometric information about the changes in the effective dictionary due to dictionary updates. It is suggested that traditional statistical distances can be applied to a distribution of Gram matrix entries to detect a shift in effective dictionary geometry. Statistical distances can be used to monitor the difference between the distributions of entries of the Gram matrix of the optimized CS system and Gram matrix entries of the stale CS system, seeking a change in the geometry of the effective dictionary. These statistical distances (presuming that their computational complexity is low enough relative to re-generating the sensing matrix) could be used to alert the system that an updated sensing matrix should be generated. To compare Gram matrices distributions, traditional methods of probability distribution distances can be used, of which there are many [69]. Two well-known measures to compare distributions are the *Kolmogorov-Smirnov* test and the *Kullback-Leibler* divergence. The method of computing distances between distributions of Gram matrix entries to discern differences in geometry is treated in detail in this dissertation and warrants its own section (see Sec. 5.6). In order to provide some motivation for seeking a measure beyond the sensing matrix objective function, an example using a contrived data set to simulate the entries of the Gram matrix is presented in Fig. 6. Data sets (representing Gram matrix entries) are constructed to demonstrate that Gram matrices with very similar Frobenius norms, or

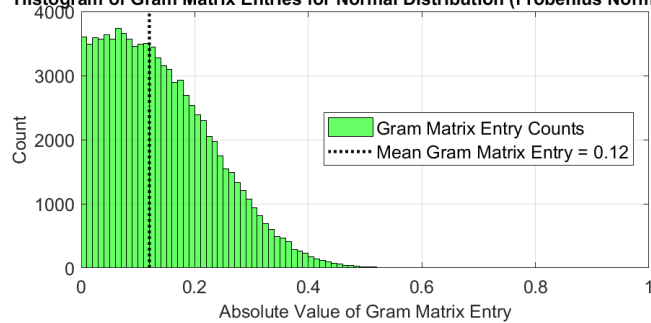
with very similar means of their entries can have very different histogram distributions. The different histogram distributions represent significantly different underlying effective dictionary geometries. Therefore, single-parameter measures of the Gram matrix (e.g. only the mean Gram matrix entry, or only the Frobenius norm of the Gram matrix), may in some dictionaries not provide a good measure of the relative geometry of the atoms of an effective dictionary. Consequently, other Gram comparison methods are sought which more comprehensively encode the geometry of the effective dictionary.

Histogram of Gram Matrix Entries for Uniform Distribution (Frobenius Norm = 50.3)



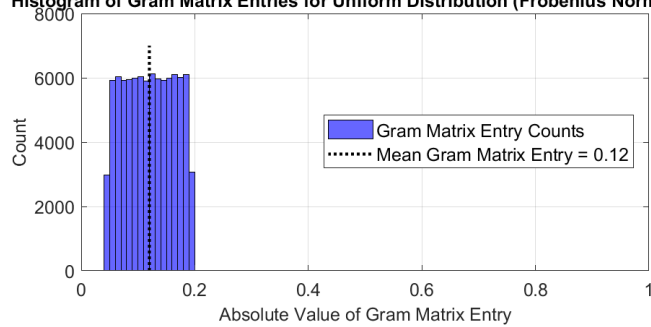
(a) Histogram 1

Histogram of Gram Matrix Entries for Normal Distribution (Frobenius Norm = 50.9)



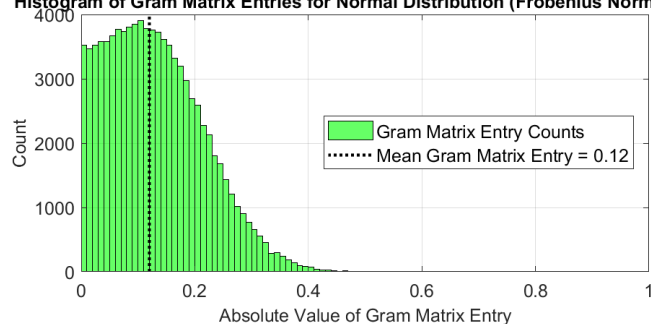
(b) Histogram 2

Histogram of Gram Matrix Entries for Uniform Distribution (Frobenius Norm = 38.4)



(c) Histogram 3

Histogram of Gram Matrix Entries for Normal Distribution (Frobenius Norm = 46.8)



(d) Histogram 4

Figure 6: Failure of Single Parameter Gram Matrix Measures. Histograms of synthetically generated Gram matrices are shown. Neither the Frobenius norm nor the mean Gram matrix entry reflect the geometry of the histograms, and therefore the geometry of the underlying effective dictionaries.

5.5.3 Approach 3: Distances between Gram Matrices

Analysis of the Gram matrix entries distribution is relatively straightforward, and provides geometrical information concerning dictionary geometry changes, but it disregards structural and geometrical information about the space in which a (PSD) Gram matrix resides in relation to all other Gram matrices. Other technical methods such as the log-Euclidean distance can be used to leverage such known structure [57]. Equivalently stated, studying only the values of the Gram matrix entries ignores the fact that the space of finite-dimensional positive semi-definite Gram matrices live on a convex cone [57] (see also Section 9.4.4). This section lists some of the results in the literature that begin to study more appropriate distance functions to measure the difference between a pair of positive semi-definite matrices, as found in [57, 70].

Reference [57] identifies the following measures between covariance matrices, and can also be applied to measure the distance between Gram matrices:

- Euclidean Metric
- Log-Euclidean Metric
- Bregman Divergence
- Alpha-Beta Divergences

Each of these measures have different combinations of properties that may be suited to a particular application, e.g.

- Geodesic distance
- Affine invariance
- Scale invariance
- Unitary invariance
- Inversion invariance
- Inner product distance
- Complete metric

The measures and properties listed above are included for completeness. Further analysis and simulation as applied to Gram matrix comparison are not explored in this dissertation, but are left to a prospective research effort.

5.6 Focus on Approach 2: Distance Measures between Sets of Gram Matrix Entries

This section focuses on Approach 2 to sensing matrix evaluation. One can compare two Gram matrices by comparing the two data sets formed from the entries of each Gram matrix, where the entries are viewed as a collection of data points. In the current context, one Gram matrix corresponds to the updated effective dictionary and the other Gram matrix corresponds to the stale effective dictionary. While there exist many different types of statistical tools that have been developed to compare discrete distributions of data, here we explore two. The two tools used in this dissertation to compare sets of Gram matrix entries are the Kolmogorov-Smirnov (KS) statistic and the Kullback-Leibler divergence (KLD).

Using the entire set of Gram matrix entries (as opposed to a single parameter) provides more detailed geometric information concerning the projected dictionary than simply using the mutual coherence. E.g, the mutual coherence uses only the entry (entries) that attain the maximum value of all the entries in the Gram matrix and represents the worst-case (closest) Euclidean distance among all pairs of projected dictionary atoms (see Sec. 3.4.2).

5.6.1 Probability Mass and Distribution Functions

Two basic statistical functions are now defined. Let $X \equiv X_1, X_2, \dots, X_N$ be a set of samples from a discrete distribution. The first statistical function to be defined is the *probability mass function* (PMF) or empirical probability density

$$f(x_i) = \frac{1}{n} \sum_{i=1}^n \mathbf{I}(x_i \leq X_i < x_{i+1}), \quad (5.5)$$

where $[x_i, x_{i+1}]$ is the i^{th} sub-interval or bin, and there are n bins altogether.

The second tool to define is the *cumulative distribution function* (CDF) (or empirical distribution function (EDF) in the discrete case) of the entries of the Gram matrix of the effective dictionary. The CDF is defined as follows and is adapted from [69]. Again, let $X \equiv X_1, X_2, \dots, X_n$ be a set of samples from a discrete distribution. Then, the empirical distribution function (EDF) of the sample set is

$$F(x) = \frac{1}{n} \sum_{i=1}^n \mathbf{I}(X_i \leq x). \quad (5.6)$$

where $\mathbf{I}(\rho)$ is the *indicator function*, which is equal to 1 if the condition ρ is true, and equal to 0 if

the condition ρ is false, X_i is the subset of n samples from X that are less than or equal to x .

5.6.2 PMF of Gram Matrix Entries: A Geometric Interpretation

For the analysis proposed, the PMF of the absolute value of Gram matrix entries can be viewed through the lens of a histogram, as in Fig. 8. Many publications make use of the histogram (or very similar graphs) of Gram entries as a way to visually and *qualitatively* compare the optimality of CS sensing matrices [12, 25, 45, 67]. One histogram-based interpretation of the PMF of Gram matrix entries is as follows. Each pair of column-normalized atoms in the dictionary (a_i, a_j) is associated with two angles subtended between them: one (smaller) angle within the range $[-\pi, \pi]$ radians and another (larger) angle outside this range. The i^{th}, j^{th} Gram matrix entry is the cosine of the smallest angle between atom a_i and atom a_j .

When constructing the PMF of Gram matrix entries, consider only the absolute value of the Gram matrix entries, i.e. the absolute value of cosine of dictionary atom angles within $[-\pi, \pi]$, making all the atom angles within the range $[0, \pi]$. For a given point on the PMF, the height of its corresponding bin in the PMF histogram indicates the number of atom pairs whose absolute value of the cosine of their angles reside within the range indicated by the respective histogram PMF bin.

The histogram of Gram matrix entries can be roughly interpreted as a type of greedy algorithm performance *risk profile* in the following manner. Each entry of the Gram matrix of the effective dictionary provides information concerning the distance between a pair of effective dictionary atoms (e.g., Section 3.4.3, Eq. (3.20) relates the angle between two atoms vectors to the Euclidean distance between the two vectors). The Gram matrix entry PMF conveys the relative risk that a randomly selected pair of dictionary atoms (dictionary columns) resides close to each other in Euclidean distance. The closer two atoms are, the higher risk there is that a greedy recovery algorithm such as OMP confuses one atom for the other atom in the presence of measurement noise.

5.6.3 EDF of Gram Matrix Entries: A Geometric Interpretation

This section addresses a more abstract view of the geometric relationships among dictionary atoms using a graph-theoretical interpretation of the EDF. Similar notions are separately used in the context of *t-average coherence* (see [45] and Section 9.2.4) as well as structure detection in brain neuron spike analysis [71, 72]. However, [45] does not geometrically interpret the EDF as a distribution function of Gram matrix entries as is done in this dissertation, and [71, 72] is not applied to com-

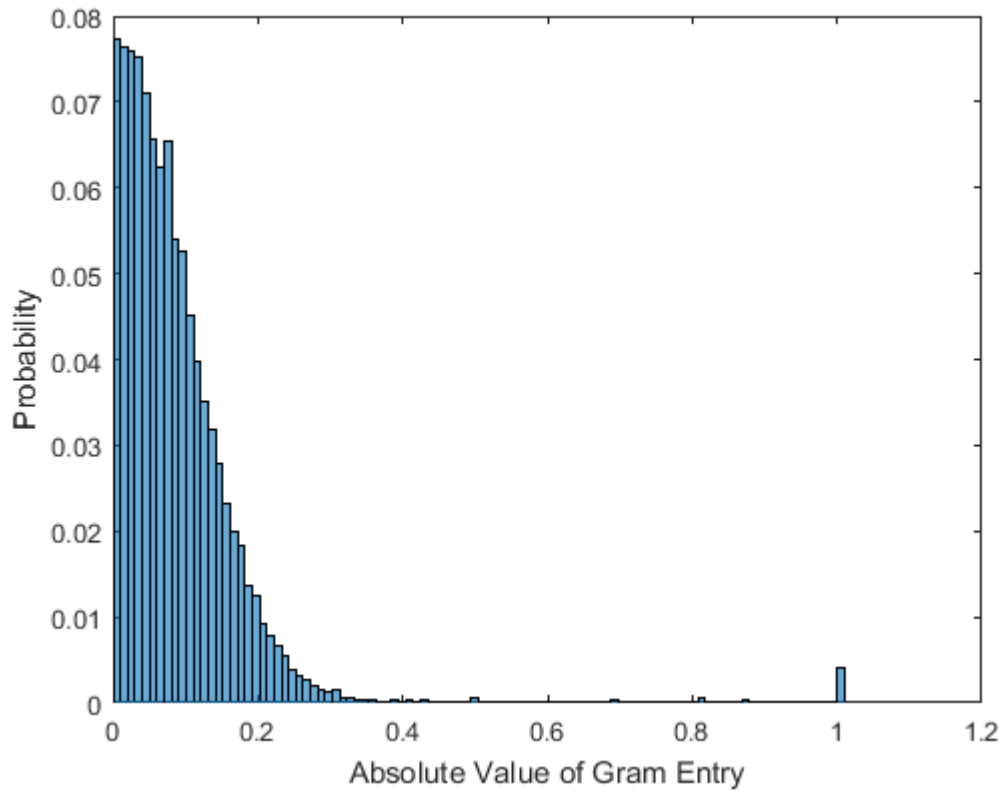


Figure 7: PMF of the absolute value of Gram matrix entries of a column-normalized dictionary. The histogram is computed by dividing the interval $[0,1]$ into 100 sub-intervals or bins.

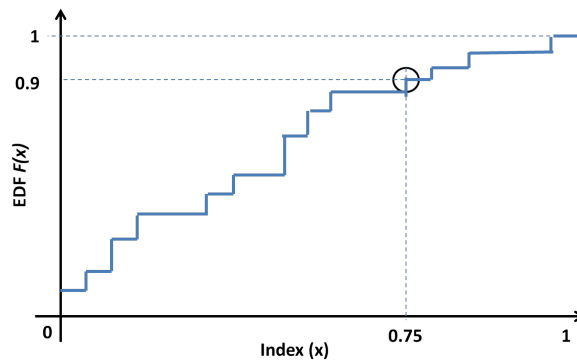


Figure 8: Representative graph depicting a general EDF of the absolute value of off-diagonal Gram matrix entries of a column-normalized dictionary. The circled point on the graph can be interpreted as stating that 90% of the Gram entries are less than or equal to 0.75.

pressive sensing or sensing matrix evaluation.

The EDF of Gram matrix entries provides information concerning the *growth rate* of graph edges in an undirected, unweighted graph. One can represent each atom as a node in an undirected graph, then each atom pair is represented by a pair of graph nodes. Select a distance threshold δ , and draw an edge between each pair of graph nodes if the associated atoms are within the distance threshold δ . Then an edge between a pair of nodes indicates that the Euclidean distance between the two atoms is below the fixed threshold δ . The total number of edges in the undirected graph indicate the total number of atom pairs that are within the distance threshold δ .

More subtle geometric information about the space spanned by the atoms of a dictionary can be obtained if the total number of graph edges is studied over a range of distance thresholds [71, 72]. Envision balls centered at the tip of each atom vector. Starting with a ball radius of zero, incrementally increase the radius of each ball. A given ball radius can be interpreted as a threshold. When two balls intersect, an edge is drawn between the corresponding nodes of the graph. The total number of edges (intersecting ball pairs) at each threshold (ball radius) is recorded. The number of graph edges is tracked as a function of threshold and plotted on an EDF. This process is related to a concept in computational topology known as a *filtration* [71, 72].

Since the effective dictionary atoms are normalized, the absolute value of the Gram matrix entries fall in the range $[0,1]$. As stated, Gram matrix entries are the absolute value of the cosine of the smallest angle formed between atom pairs, which is directly related to the distance between the two vector tips. A Gram matrix entry of 0 corresponds to the maximum possible angular displacement (π radians between atom pairs and a Gram matrix entry of 1 corresponds to an angular distance of 0 between atom pairs (i.e. overlapping atom vectors). As one traverses the resulting EDF from 0 to 1 on the abscissa (i.e. the x-axis), the profile along the ordinate (i.e. the y-axis) indicates the number of atoms pairs (undirected graph edges) that are included in the cumulative count as the balls are inflated (i.e. as atom pairs are counted in the reverse order of their angular separation-from largest to smallest angle). Each bin on the abscissa of the EDF corresponds to a particular distance threshold, and the associated ordinate on the EDF represents the number of atoms that are closer than that distance threshold (ball radius), as shown in Fig. 8.

Equivalently, this process can be perceived as iteratively thresholding the entries of the Gram matrix, and counting only those entries whose value is above a specified threshold [71, 72]. Fig. 9 depicts a simplified dictionary in two dimensions, and how the geometry of the space spanned by

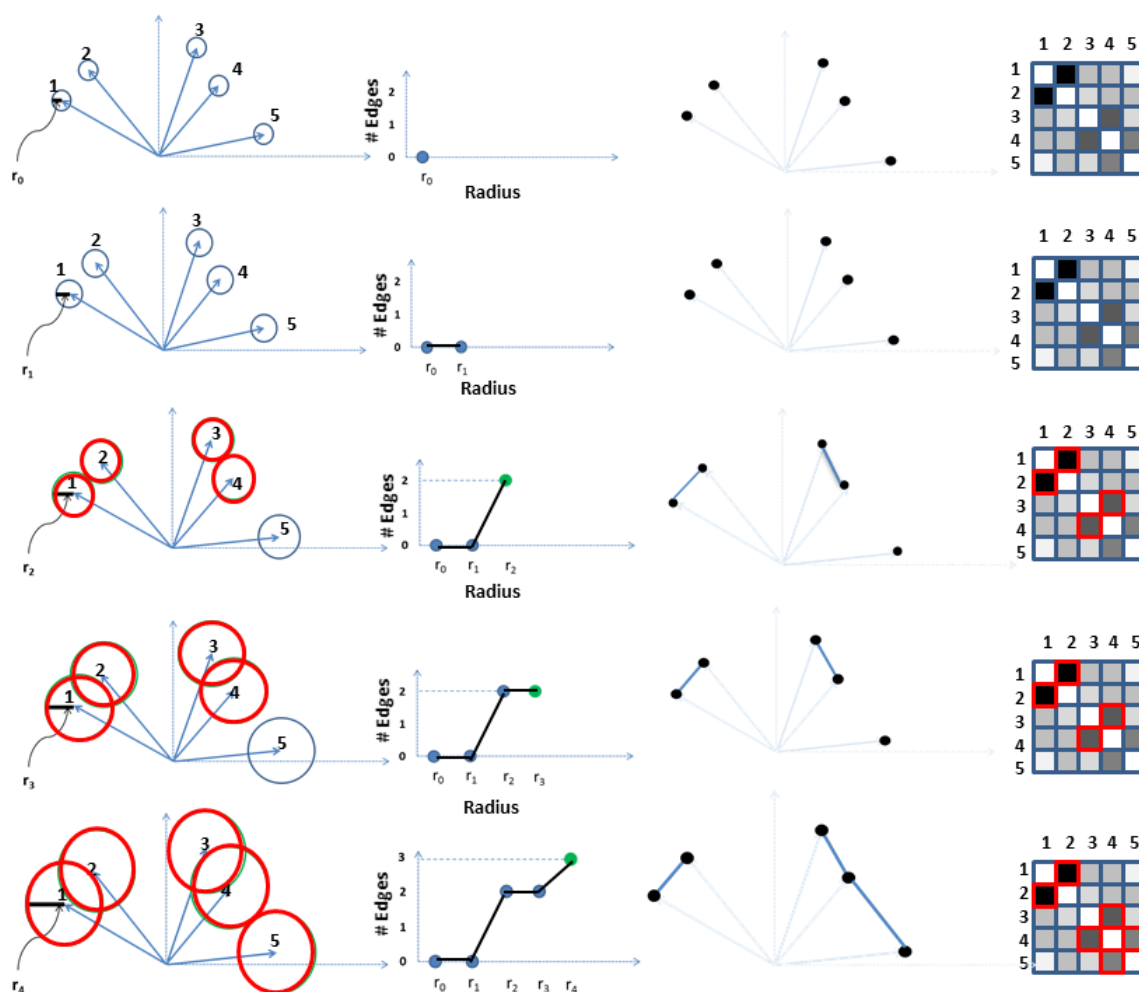


Figure 9: Iteratively ‘growing’ a Gram matrix by inclusion. Five iterations of thresholding dictionary atom pairs based upon their pairwise distances. At each iteration, only those atom pairs that are within the current threshold distance are retained (i.e. only those atoms whose respective spheres intersect). This is tantamount to iteratively thresholding the entries of the associated Gram matrix, and retaining only the associated atom pairs that are closer than the current distance threshold. In the graphs on the right, only the Gram matrix entries outlined in red are retained.

the atoms of a dictionary can be analyzed in terms of the pairwise angles and distances between its atoms. The red circles on the left side graphs show the atoms in the dictionary that are close enough to include (count) at the current iteration. The graphs in the second column represent an EDF of the number of atoms pairs counted at each threshold, as a function of inter-atomic distance. The graphs in the third column represent the corresponding undirected graphs. The Gram matrices in the last column on the right are color-coded to show the atom pairs in red are close enough to each other such that they are included in the count at each specified threshold (ball radius).

5.6.4 Overview of Statistical Distances

Reference [73] identifies two significant applications of statistical distances in data analysis:

- Parametric statistical inference (estimating parameters of a presumed distribution)
- Goodness of fit (measuring distances between a data set and a known distribution)

Many statistical distances are used for both purposes, and there are generally two types of distances:

- Distances between distribution functions, e.g. Kolmogorov-Smirnov, Cramer-Von Mises, Anderson-Darling
- Distances between density functions, e.g. Kullback-Leibler Divergence, Bregman Divergences, Pearson's Chi-Square, Hellinger Distance

This dissertation examines two of these measures in particular: the Kolmogorov-Smirnov (KS) statistic and the Kullback-Leibler (KLD) divergence. Note that while the KS distance can be used to measure discrepancies between either two data sets or between a data set and a parametric model, this dissertation uses KS to distinguish between two discrete finite data sets.

5.6.5 Kolmogorov-Smirnov Test

The KS test is a non-parametric statistical *goodness of fit* test that was originally developed to determine if an empirical data sample set belongs to a given continuous underlying CDF [69]. Smirnov later extended the KS-test to compare two empirical data distributions, i.e. EDFs [69]. To begin, define $X_1, X_2, \dots, X_m$ and $Y_1, Y_2, \dots, Y_n$ as two independent samples from populations with unknown EDFs, defined as F_X and F_Y . In addition, define F_m and F_n as the corresponding empirical EDFs. Then, the hypothesis test for the equivalence of the two distributions is formulated as [69]:

$$\begin{aligned} H_0 : F_X(x) &= F_Y(x) \forall x \\ H_1 : F_X(x) &\neq F_Y(x) \exists x \end{aligned} \tag{5.7}$$

The KS-statistic for comparing two distributions is [69]:

$$D_{m,n} = \sup_x |F_m(x) - F_n(x)| \tag{5.8}$$

The KS-statistic translates to the maximum difference between the distribution function of $F_m(X)$ and the distribution function of $F_n(x)$.

There is a significant body of literature that applies the KS-statistic to perform a binary hypothesis test as to whether two data sets are drawn from the same distribution. In addition to this, however, the *KS-statistic* $D_{m,n}$ can be used as a rough measure of distance between two data sets without performing a statistical test, *per se*. The KS-statistic can provide a relative measure, in some sense, of how *far* one data set is from another data set.

5.6.6 Applying the KS Test to Sensing Matrix Evaluation

In this section, the KS statistic is applied as a distance measure between the entries of a stale Gram matrix $G(A_{t1}, A_{t2})$ and an updated Gram matrix $G(A_{t2})$. Define $G_{t2}(\cdot)$ to be the discrete finite data set consisting of the absolute value of the entries that lie above the main diagonal of the *updated* Gram matrix. Similarly, define $G_{t1,t2}(\cdot)$ to be the discrete finite data set consisting of the absolute value of the entries that lie above the main diagonal of the *stale* Gram matrix. The total number of elements in either the stale or updated Gram matrix is $M \times M$, so the total number of elements above the main diagonal of either Gram matrix is

$$N_{triu} = \frac{M(M-1)}{2}. \quad (5.9)$$

Therefore, each of the data sets $G_{t1}(\cdot)$ and $G_{t1,t2}(\cdot)$ contains N_{triu} elements of real scalars taking values on $[0, 1]$. Define $F_{t2}(x)$ to be the EDF of the the data set $G_{t2}(\cdot)$. The N_{t2} points used to evaluate the EDF are within the interval $[\min(G_{t2}(\cdot)), \max(G_{t2}(\cdot))]$. This interval is sub-divided into N_{t2} sub-intervals, whose bin edges are collected in the N_{t2} -element vector x_{t2} . Then the EDF of the absolute value of the above-diagonal entries of the updated Gram matrix $F_{t2}(x)$ is given by

$$F_{t2}(x) = \frac{1}{N_{t2}} \sum_{i=1}^{N_{t2}} \mathbf{I}(G_{t2}(\cdot) \leq x_{t2}(i)). \quad (5.10)$$

Similarly, define $F_{t1,t2}(x)$ to be the EDF of the stale Gram matrix $G_{t1,t2}(\cdot)$. The $N_{t1,t2}$ points used to evaluate the EDF are within the interval $[\min(G_{t1,t2}(\cdot)), \max(G_{t1,t2}(\cdot))]$. This interval is sub-divided into $N_{t1,t2}$ sub-intervals, whose bin edges are collected in the $N_{t1,t2}$ -element vector $x_{t1,t2}$. Then the EDF $F_{t1,t2}(x)$ is given by

$$F_{t1,t2}(x) = \frac{1}{N_{t1,t2}} \sum_{i=1}^{N_{t1,t2}} \mathbf{I}(G_{t1,t2}(\cdot) \leq x_{t1,t2}(i)). \quad (5.11)$$

The KS statistic can then act as a difference operator between the two Gram matrices

$$\mathcal{D}_{KS} \{G(A_{t_1,t_2}), G(A_{t_2})\} = \sup_x |F_{t_1,t_2}(x) - F_{t_1}(x)| \quad (5.12)$$

where the argument x indexes the EDFs over the interval $(-\infty, \infty)$. The N_{bin} -element vector x represents the N_{bin} bin edges into which the interval is divided. The vector x can be expressed in the MATLAB language as $[-\infty; \text{sort}([G(A_{t_1,t_2}(:)); G(A_{t_2}(:))]); \infty]$. The associated KS-statistic offers some interesting geometric interpretation. One interpretation of the KS-statistic in light of the geometry of the EDF of Gram matrix entries is that the KS-statistic is the largest difference between two dictionaries, regarding the number of atom pairs that are at least a given distance apart from each other.

5.6.7 Kullback-Leibler Divergence

The Kullback-Leibler divergence (KLD) measures the extent to which one probability distribution differs from another probability distribution. The KLD is also referred to as *relative entropy*, and is a measure of the information gain when one distribution is used over a second distribution [74]. The KLD from one discrete probability distribution $p(x)$ to a second discrete distribution $q(x)$ is

$$\mathcal{D}_{KL}(p||q) = \sum_{i=1}^n p(i) \log \frac{p(i)}{q(i)}. \quad (5.13)$$

The KLD is not a metric since it does not possess the property of symmetry, i.e., $\mathcal{D}_{KL}(p||q) \neq \mathcal{D}_{KL}(q||p)$. In spite of this, the KLD can be used to measure the difference between using one Gram matrix over the other.

5.6.8 Applying KLD to Sensing Matrix Evaluation

The KLD resulting from using the PMF $f_{t_1,t_2}(x)$ of the absolute value of Gram matrix entries $G(A_{t_1,t_2})$, instead of the PMF $f_{t_1}(x)$ of absolute value of Gram entries $G(A_{t_2})$ is

$$\mathcal{D}_{KL}(f_{t_1}||f_{t_1,t_2}) = \sum_{i=1}^n f_{t_1}(i) \log \frac{f_{t_1}(i)}{f_{t_1,t_2}(i)}, \quad (5.14)$$

where n is the number of bins over which the KLD is evaluated.

6 Cognitive Radio Network Spectrum Analysis

6.1 Spectrum Approximation for Cognitive Radio Networks

6.1.1 Brief Introduction to Cognitive Radio Networks

Rampant reductions in size, weight, and power requirements of several enabling technologies including GPS, spectrally agile transmitters (e.g. software-defined radios or SDRs), batteries, and Wi-Fi are contributing to a deluge of networked mobile radio frequency (RF) devices in use today [75]. Another enabler of mobile networked devices is availability of RF spectrum. RF devices require unoccupied spectrum (channels) on which to transmit and receive information.

In commercial applications, RF devices are generally used according to a static set of spectrum standards or policies to prevent users from interfering with each other. Many (though not all) traditional RF spectrum licenses are fixed such that they designate which RF channels are available to be used by certain types of users, for a particular purpose, over a particular geographic area, and unless the license is modified (a relatively lengthy process), the license does not change over time. These fixed licenses or channels are allocated to specific a licensee, whether the channels are heavily used, or not used at all. In addition, large-scale RF studies indicate that many RF bands are underutilized most of the time [76]. This fixed radio spectrum licensing and policy is creating an ‘artificial spectrum scarcity’ in the sense that users who have a need for spectrum are not allowed to access those underutilized bands [77, 78]. Unlicensed frequency bands such as those used for Wi-Fi networks provide users more ubiquitous connectivity.

Along with some anticipated form of network infrastructure, a set of CRs that are interdependent and collaboratively exchange information with each other form a *cognitive radio network* (CRN). The CRN architecture is a promising prospect for more effective management of RF spectrum by allowing opportunistic dynamic spectrum access to those who need it. In general, CRN network components can be roughly divided into subscriber (user) equipment and network equipment. In CRNs, subscribers are further categorized generally as either incumbent primary users (PUs) or secondary users (SUs) who are allowed to use the available spectrum, provided that they do not

significantly interfere with PUs operations [78]. A cognitive radio user who wishes to gain access to the CRN is referred to as a candidate secondary user (CSU).

Spectrum sensing is the acquisition of knowledge concerning what RF channels are used and which ones are available. While there are various forms of spectrum sensing, this research assumes that the spectrum will be sensed cooperatively, i.e. each SU in the system will submit a spectrum report to a centralized managing authority known as the spectrum access system (SAS). Furthermore, it is presumed that the SAS will house a database of spectrum reports that are submitted by a set of trusted sensor nodes (SNs) that are scattered throughout the CRN coverage area (CA). The spectrum database located at the SAS is known as a geographic database (GDB). The SAS will use the spectrum reports submitted from the SNs to help verify spectrum reports that arrive at the SAS from CSUs desiring CRN access. If an incoming spectrum report from a CSU deviates too far from local SN spectrum reports, then the SAS can decide to ignore the suspect spectrum report, deny the CSU access, etc. When a user intentionally falsifies a spectrum report, this is known as spectrum sensing data falsification (SSDF).

The following sub-sections on spectrum approximation provides a method to construct or approximate spectrum reports submitted by a CSU as a function of the spectrum reports residing in the GDB. Section 9.5 contains additional details on CRN architectures. GSA is subsequently applied to greedy spectrum verification (GSV). GSV is outlined in Section 6.2 and provides a set of heuristics to measure the deviation between CSU and SN spectrum reports for the purpose of detecting SSDF.

6.1.2 Overview of Greedy Spectrum Approximation

The name *greedy spectrum approximation* (GSA) is used to describe the process of approximating (or expanding) the spectrum reported by a CSU, as a linear function of a small number of trusted sensor node (SN) sector spectrum estimates, using a greedy algorithm, such as orthogonal matching pursuit (OMP). To begin the formulation of the CSS scenario, assume that there exists a well-defined two-dimensional (flat) geographical CA over which a CRN is operating (see Fig. 10). The SU CR users (transmitters) are assumed to be stationary both spatially and spectrally, i.e. the SUs do not change their channel and do not change geographic location over the course of the spectrum approximation.

The GSA approach presented here involves the use of a fixed network of trusted SNs sites scattered about the CA and a centralized fusion center known as the spectrum access system (SAS),

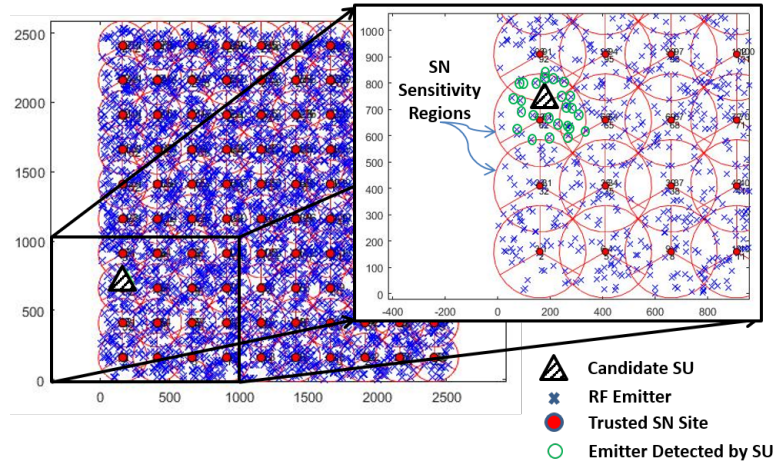


Figure 10: Grid of sensors and transmitters scattered over geographic coverage area. Red circles around a trusted sensor node indicate its sensitivity region.

as seen in Fig. 11. Conceptually, the SNs could be either other trusted SUs or a dedicated infrastructure of sensors. In this dissertation, a single SN site is composed of three co-located directional (sectored) antennas, pointing at 0, 120, and 240 degrees azimuth, respectively.

Each sectored SN antenna senses the spectrum within its sector, constructs a spectrum report for its respective sector, and forwards the report (possibly via multiple SNs) to the centralized SAS for aggregation, analysis, and incorporation into the GDB. The GDB then represents the totality of trusted knowledge regarding spectrum usage throughout the system SA. The set of all spectrum reports collected over the SA and stored in the GDB resides at the SAS as shown in Fig. 15. As subsequently formulated, each spectrum report is assumed to be a vector of common length N , and each element of a given spectrum report contains an estimate of the power sensed at particular channel.

In addition to utilizing spectrum reports directly from trusted SNs, the SAS can conceivably update the GDB with spectrum reports from SUs that are trusted. One way to ascertain the trust of an SU is to verify the spectrum reported by the SU. When a SAS (along with a GDB and access to a set of trusted SNs) receives spectrum reports from a candidate SU (CSU), a decision must be made by the SAS as to whether to trust the report or not. Furthermore, a decision to trust the CSU must also be made as well. If the candidate SU provides a spectrum report that does not correlate well enough with the local trusted SNs, then the spectrum report will not be incorporated into the GDB and the candidate SU may be denied access to spectrum. Therefore, the main goal in the application of GSA is to use the SNs to evaluate and verify the CSUs spectrum report, as compared to the

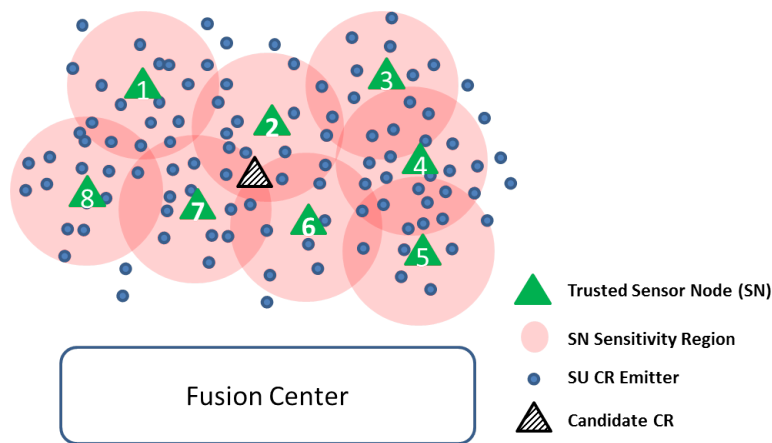


Figure 11: Collaborative CRN with SU CR emitters, fusion center, trusted sensor nodes, and associated sensitivity regions. Although not explicitly shown here, each sensitivity region is divided into three equal sectors.

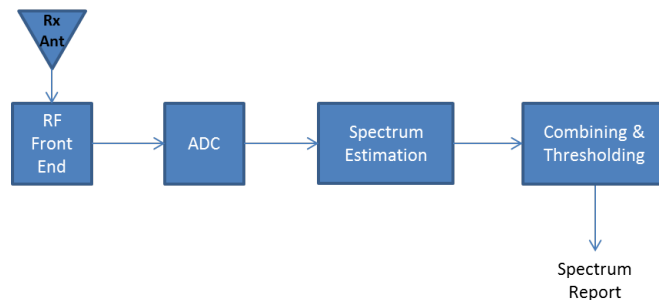


Figure 12: Processing Chain of a Cognitive Radio Sensor

local SNs surrounding the CSU. A rough sequence of the RF processing chain used to construct a spectrum report is shown in Fig. 12.

The intended use-case model is summarized in the following process:

1. CSU issues spectrum request (query, see Fig. 13)
2. Local SNs that are trusted by the CRN receive and forward request to SAS
3. SAS acknowledges and verifies the spectrum request and issues a sensing matrix
4. SNs forward sensing matrix to the CSU
5. Candidate SU senses its local spectrum with the sensing matrix provided and sends an obfuscated spectrum report to the local SNs
6. Local SNs receive and forward obfuscated spectrum report to SAS
7. Using the GDB, the SAS performs spectrum approximation

The approach taken in the GSA method is to attempt to reconstruct the CSUs spectrum report as a linear combination of a small number of spectrum reports that are stored in the GDB (see Section

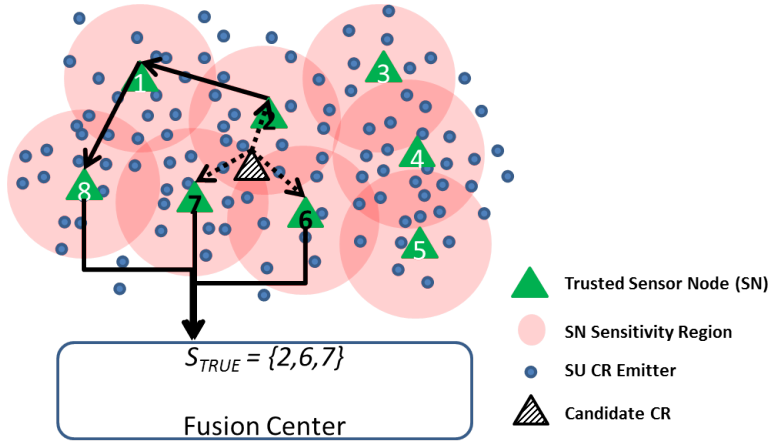


Figure 13: Candidate CR broadcasts a spectrum request to local SNs (dashed lines). The spectrum request is then collaboratively forwarded (solid lines) to the SAS where the SN IDs that first sensed the request are identified and collected in a set (S_{TRUE}) containing SN IDs local to the candidate CR.

6.1.4). The signal model describing the spectrum report generation from the CSU and trusted SNs is detailed in Section 6.1.3.

6.1.3 Spectrum Report Models

Define the number of RF bandlimited transmitters in the CA as N_e . The power received at the CSU due to the i^{th} emitter is denoted as $P_{T,i}L_i$ where $P_{T,i}$ represents the power transmitted from the i^{th} emitter. L_i represents the signal loss incurred as the signal travels from the i^{th} emitter to the candidate SU.

Let $N_{E,CR}$ represent the number of emitters whose signal arrives at the candidate SU above a certain power threshold p_{thresh} , where $N_{E,CR} < N_e$. Figure 14 depicts the $N_{E,CR}$ emitters whose power sensed at the CSU is above p_{thresh} as lying within the *sensitivity region* of the CSU, indicated by the red circle. If a signal is received by the CSU (or a SN) at a power at or above p_{thresh} , then the transmitter that sent the signal lies within the sensitivity region of the CSU (or a SN). The sensitivity region is the geographic area local to the CSU (or a SN) within which are located transmitters able to be ‘seen’ by the CSU (or SN).

Let $p_{CR} \in \mathbb{R}^{N_S}$ be a column vector containing N_S digital received signal strength measurements made by the candidate SU, collected at a sampling rate of F_S . Define $P_{CR} \in \mathbb{R}^{N_S}$ as the digital power spectrum estimate associated with the vector p_{CR} , estimated at the CSU. The signal strength vector p_{CR} received at the CSU is then modeled as the sum of $N_{E,CR} < N_e$ bandlimited

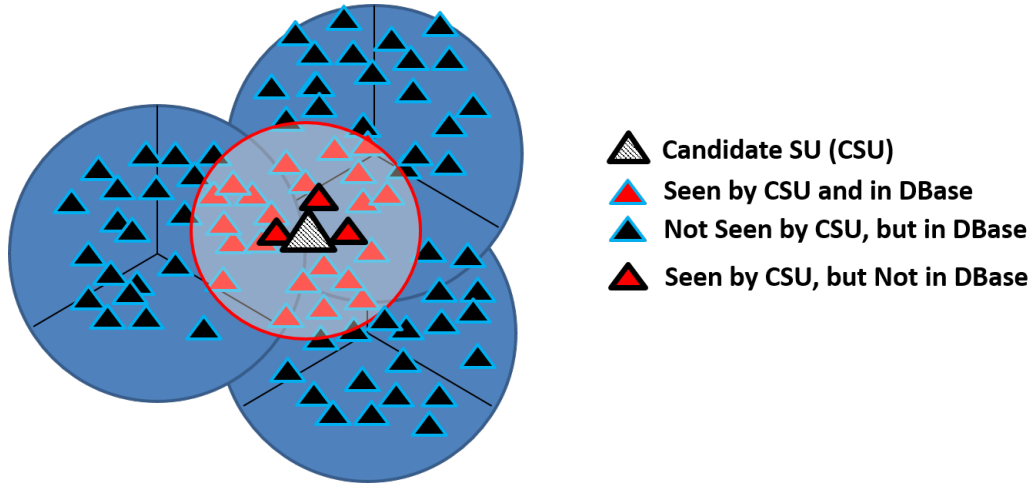


Figure 14: Emitters may be hidden from either the CSU or the SNs.

signals from $N_{E,CR}$ transmitters after having undergone free-space propagation, along with additive noise. The vector P_{CR} results from computing the periodogram of the measurement vector p_{CR} [79, 80].

Next, channelization is applied to the power spectrum estimate P_{CR} at the candidate SU such that the spectrum spanning F_S Hz is subdivided into $N = F_S/B$ frequency bins, where each bin is B Hz wide and contains the sum of all signal power within its bandwidth. The resulting candidate SU *spectrum report* is represented as

$$x \in \mathbb{R}^{Nx1} \quad (6.1)$$

6.1.4 GDB as an Overcomplete Dictionary

The channelized spectrum profile estimated at each SN sector, referred to as a *SN spectrum report*, is constructed in a manner analogous to that of the candidate SU, and is represented as

$$\psi_j \in \mathbb{R}^N, \quad (6.2)$$

where $j \in [1, M]$ for M SN sectors scattered throughout the CA. Additional details regarding signal model are included in Section 9.5.5.

The GDB is viewed as an overcomplete (i.e., redundant) $N \times M$ real non-negative matrix (for $N < M$). Each spectrum sensing report generated by a given SN sector comprises a slice of the GDB, i.e., each column of the GDB represents the channels sensed by a particular SN sector and each row

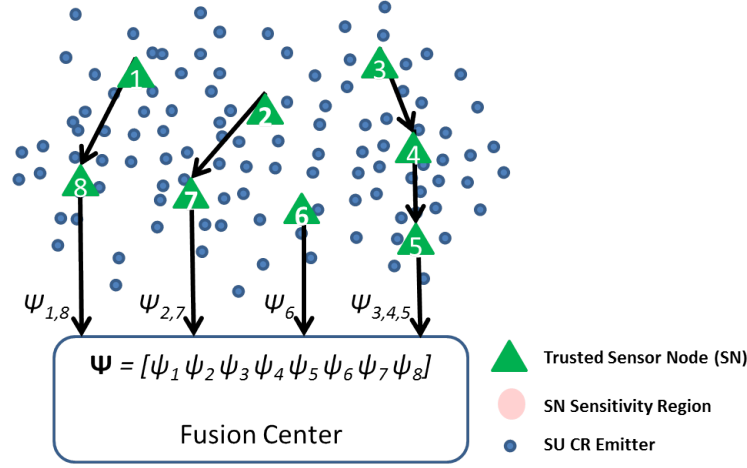


Figure 15: Spectrum reports (ψ_i) are collaboratively forwarded to the centralized fusion center and collected column-wise to form the geographic database.

represents a single frequency or channel. Each entry in the matrix is the power estimated at the i^{th} channel by the j^{th} SN sector. The system collaboratively forwards each SN spectrum report to the FC where all M channelized spectrum profiles ψ_j are collected into a matrix $\Psi \in \mathbb{R}^{N \times M}$ comprising the GDB. The j^{th} column of Ψ is ψ_j and represents the channelized power spectrum measured at the j^{th} SN sector

$$\Psi = [\psi_1 \dots \psi_M]. \quad (6.3)$$

The set of frequencies to be searched for available and occupied spectrum is called the *frequency search space*, Ω . The frequency search space is formulated as the contiguous and finite set of frequencies $\{F_{min}, \dots, F_{max}\}$. The frequency search space is binned into N mutually exclusive channels (i.e. the spectrum is *channelized*, without any overlap between adjacent channels). The resulting set of channels is called the *channel search space*, and is indexed by $j \in \{1, \dots, N\}$. The channel search space is assumed to be universally known throughout the system.

At the SAS, the complete set of spectrum reports from trusted SNs are stored within the GDB, as shown in Fig.15. The assumption is that there exists more SNs than possible channels, the GDB is modeled as a matrix, specifically an overcomplete (see Section 2.3) dictionary Ψ . To be precise, let Ψ be an $N \times M$ real non-negative matrix, for $N < M$.

Note that due to the versatile and user-driven nature of mobile devices, the GDB is actually a dynamic function of time and geographic location. In this dissertation, it is assumed that the CRNs channel usage is stationary over the duration of the CRs channel acquisition. At a single time instant, channel re-use is possible. E.g., two SUs may be using the same channel (at a low enough

power) at two different locations in the geographic CA of the CRN at the same time. In short, the spectrum usage is assumed to be stationary with respect to time, but the spectrum usage is permitted to vary over the CA. Given these assumptions, the GDB is assumed to accurately reflect the current spectrum usage in the associated CA.

6.1.5 Approximation Methods for GSA

The GSA approach involves sparsely approximating the channelized spectrum report x from the CSU, i.e., linearly combining a small fraction of the total number of SN spectrum profiles ψ_j to approximate x . The approximation of the spectrum report x is denoted $\hat{x}$ and can be represented as

$$\hat{x} = \Psi s \approx x, \quad (6.4)$$

where s is a k -sparse vector of coefficients, whose magnitude and locations are to be estimated. The approach is to apply the approximation methods in Section 3 and Section 4 to find the *best k -term approximation* of x in terms of the sparsity of the coefficient s and the Euclidean distance to x .

There are five approaches explored to match the CSU spectrum to SN sector spectrum in the GDB Ψ . The first is sparse approximation (Section 3.2) using OMP (Section 3.5.2), referred to as SA-OMP. The second is sparse approximation using inner product correlation (IPC, Section 3.5.3). The third is compressive sensing (CS, Section 3.3) with a random sensing matrix (CS-RAND), and the fourth approximation method is CS with an optimized sensing matrix (CS-OPT, Section 4). Lastly, Section 7.3 studies a noisy variant of the CS-OPT case involving a stale or perturbed effective dictionary for four increasingly perturbed effective dictionaries, and is termed *CS-OPTNOI*.

The SA-OMP and IPC approaches are formulated according to Eq. (3.3)

$$\begin{aligned} & \underset{s}{\text{minimize}} \quad \|x - \Psi s\|_2 \\ & \text{subject to} \quad \|s\|_0 = k. \end{aligned} \quad (6.5)$$

The compressed versions of the problem CS-RAND, CS-OPT, and CS-OPTNOI construct a compressive sensing matrix Φ and apply it to the spectrum report x to form $y = \Phi x$, and to the GDB dictionary Ψ to form the effective dictionary $A = \Phi \Psi$. In this case, the problem can be cast according to Eq. (3.8)

$$\begin{aligned} & \underset{s}{\text{minimize}} && \|y - As\|_2 \\ & \text{subject to} && \|s\|_0 \leq k_{max}. \end{aligned}$$

The next sub-section addresses the CS-OPTNOI case with regard to the issue of evaluating the resulting effective dictionary when a stale sensing matrix is applied to the GDB.

6.1.6 Updating the Stale GDB and Evaluating Sensing Matrices

Within the context of Section 4.4, updating the optimal sensing matrix in response to an update in the GDB spectrum dictionary may be necessary. As time progresses, any stationarity assumptions on the GDB becomes increasingly invalid. Therefore, a way to update the GDB with new measurements becomes necessary.

The same terms in relation to dictionary updates that were defined in Section 4.4 apply here as well. For practical implementation of GSA, instead of updating the sensing matrix each time a row or column of the GDB changes, it is conjectured that the sensing matrix need only be updated after some percentage of GDB entries have been updated.

In light of these points, three items of study are relevant:

- Without knowing if a given sensing matrix is stale or updated, how can the sensing matrix be evaluated in regards to a GDB? I.e. how may the condition of sub-optimal sensing matrix and dictionary pair be detected?
- What is the effect on support estimation (specifically, on the spectrum approximation and verification methods presented in this section) if a stale sensing matrix is used on an updated GDB?
- Since the optimized sensing matrix is based on the SVD of the GDB, how can the SVD be updated efficiently?

The first two bullets are addressed in this dissertation by comparing the Gram of the stale effective dictionary to a Gram of the optimal (updated) dictionary. Tools to measure distances between Gram matrices and distances between Gram matrix elements that were developed in Section 5 are applied and related to support estimation. This is addressed empirically in Section 7.3. As there

appear to be multiple methods to efficiently compute the SVD of a rank-1 updated matrix (e.g. see Section 12.5 of [68]), the third bullet is a good candidate for future research.

In non-orthogonal dictionaries, such as the GDB, randomly selected sensing matrices have been shown to be less effective in terms of recovery performance when compared with optimized sensing matrices [12], which are constructed with knowledge of the GDB, i.e. using Eq. (4.7). The CS-OPT approximation method can be applied to the GDB, whereby the SAS generates a pseudo-random orthonormal matrix U , and then performs an SVD on the GDB. The singular values and left-singular vectors returned are then used to compute an optimal sensing matrix for the GDB which is then passed to a candidate CR. The candidate CR can then sense its local spectrum, compute the spectrum profile, and transform it by the optimal sensing matrix. It then forwards the compressed and obfuscated measurements to the SAS. With this approach, the SUs channel usage information is masked in two ways:

- Pseudo-random Matrix Multiplication
- Non-Invertible Dimensionality Reduction

In applications where the dictionary is time-variant, the sensing matrix optimization method needs to be augmented with a technique to update it. Such is the case with spectrum approximation in dynamic spectrum assignment (DSA) applications. The sensing matrix is guaranteed to be optimized only as long as the dictionary does not change. As soon as the dictionary changes, the resulting effective dictionary (the dictionary multiplied by the sensing matrix) may no longer be optimal in terms of the Frobenius norm of its off-diagonal Gram matrix entries. As defined in Section 5.2, the resulting effective dictionary matrix in this case is called *stale*. This case where the dictionary changes after sensing matrix optimization, is referred to as the approximation method CS-OPTNOI, i.e., a noisy version of a previously optimized sensing matrix.

6.1.7 Related Spectrum Approximation Methods

The main difference between existing publications concerning sparse transmitter location that were reviewed and the greedy spectrum approximation method presented in this dissertation, is that in all of the sparse transmitter location references reviewed, the number of transmitting sources (CRs) are relatively sparse over the coverage area. These types of methods are ideally suited to situations where a small number of primary users is to be detected or located. The research for this dissertation

concerns RF spectrum that is sparse in a known dictionary. I.e. the spectrum can be reconstructed adequately by a linear combination of a few columns from the dictionary matrix.

In addition to knowledge concerning the PU, information about other SUs will also need to be utilized. Hence, the CR will need to know which channels are being used by other SUs in its area, beyond just what channels the PU is using. This is particularly true in dense networks, as the number of users becomes large. Descriptions of methods that utilize CS for some aspect of spectrum sensing in a related, but different manner than GSA way can be found in [6–11].

6.1.8 Practical GSA Considerations

It should be pointed out that there exists an essential difference between the PSD estimated at the CSU and the PSD estimate of a given sector's local trusted SN. For a given i^{th} transmitter, the distance between the transmitter and the SN will in general not be equal to the distance between the transmitter and the CSU. Consequently, the path-loss experienced by the signal as it travels to the SN will be different from that of the path loss incurred as the signal travels to the CSU. Therefore, the received power at the antenna of the SN due to the i^{th} transmitter will not be equal to the power received by CSU's antenna due to the i^{th} transmitter. Fig. 14 depicts three classes of SUs: those detected by the CSU and the SNs (and are in the GDB), those SUs detected by the CSU only, and those SUs detected by the SNs only. As a result, perfect linear approximation of the spectrum report at the CSU in terms of the SN spectrum reports (GDB atoms) is in general not possible. However, for the purposes of SSDF detection the linear approximation may suffice. As long as the closest (most local to the CSU) SNs are able to represent the CSU spectrum report better than all of the other remaining (non-local) SNs, then the SSDF detection should work. This represents a relative comparison of the spectrum reconstruction in that the reconstructed spectrum is judged relative to the SNs that are known to be local to the querying CSU.

6.1.9 The Relative Comparison of the GSA Process

Correlation of the reported spectrum with the entire GDB dictionary is deemed necessary in order to correlate with other SN spectrum reports from all over the CA. This allows the algorithm to discriminate against SUs that are falsifying their location by spectrum spoofing, or by reporting spectrum from another part of the CA altered in some way. This matching technique of the algorithm to use a correlation measure relative to all other SNs in the GDB prevents the need for having to derive and calculate an absolute correlation threshold against which correlation is measured.

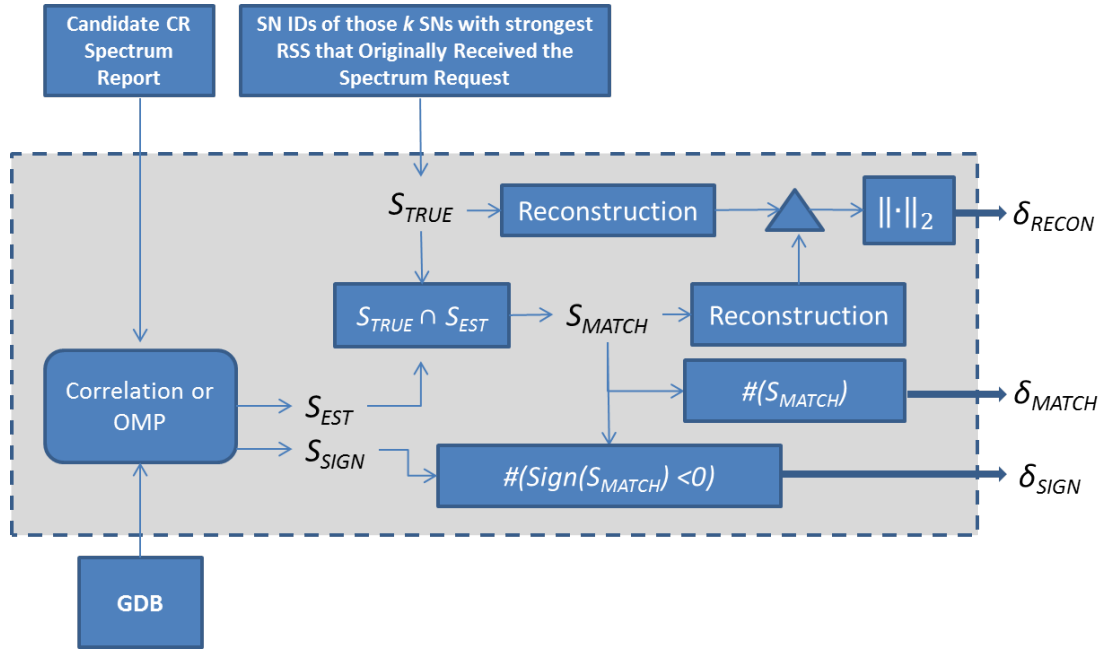


Figure 16: Spectrum Sensing Data Falsification Data Functional Diagram.

6.2 Spectrum Verification for Cognitive Radio Networks

6.2.1 Overview

After performing greedy spectrum approximation (GSA, see Section 6.1), the SAS is then able to perform greedy spectrum verification (GSV). The outputs from the orthogonal matching pursuit (OMP) process used here are the set of matched trusted sensor node (SN) IDs, the expansion coefficients, and correlation sign. These three outputs are used either alone or in combination to formulate three detection heuristics.

6.2.2 Spectrum Falsification Heuristic 1 (δ_{MATCH}): Number of Matched Sensors

The spectrum access system (SAS) compares the reported spectrum from the candidate secondary user (CSU) with the spectrum in the sensor node (SN) spectrum reports (atoms) in the geographic database (GDB) to identify the relevant SNs that are reported as being close to the requesting CSU. This task involves model selection, namely identifying a small number of SNs from the GDB whose spectrum best correlates with the candidate CR. Each SN is assigned a unique SN identification number (SN ID) that corresponds to a single column of the GDB. In implementing the proposed method, it is necessary to define two sets of SN IDs:

- S_{SN} : The set of SN IDs that first received the CSU spectrum request and are in the vicinity

of the CSU

- S_{EST} : The set of SN IDs that are returned from the collaborative spectrum approximation process, as performed by the SAS, by correlating the atoms in the GDB with CSUs spectrum report

The intersection of these two sets is used to derive the SSDF detection heuristics. The IDs of the SNs in the vicinity of the candidate CR who received the initial spectrum request are collected in a set denoted as S_{SN} . Let the set of IDs returned from either the inner product correlation (IPC) or orthogonal matching pursuit (OMP) process be represented by S_{EST} . The *matched sensor set* S_{MATCH} is the intersection of the set S_{SN} and S_{EST} and can be written as

$$S_{MATCH} = S_{SN} \cap S_{EST}. \quad (6.6)$$

The cardinality of the matched sensor set (i.e., the number of SN IDs within S_{MATCH}) is then

$$\delta_{MATCH} = \#S_{MATCH}. \quad (6.7)$$

The quantity δ_{MATCH} is used as an SSDF detection heuristic. The *matched sensors heuristic* δ_{MATCH} is simply the number of SN IDs that lie in the intersection of the sets S_{SN} and S_{EST} .

6.2.3 Spectrum Falsification Heuristic 2 (δ_{RECON}): Reconstruction Error

Using δ_{MATCH} , the spectrum reported from the candidate CR x_{CR} can be approximated by reconstructing it from the k most correlated atoms from the GDB Ψ , where the correlation is either the IPC method, or using OMP. The reconstruction can be accomplished by applying (3.4)

$$\hat{x}_{CR} = \Psi s_{MATCH}. \quad (6.8)$$

When using the IPC, s_{MATCH} is a k -sparse binary vector whose support is given by the indices listed in S_{MATCH} . Alternatively, for the matching methods that use OMP (namely SA-OMP, CS-RAND, CS-OPT, and CS-OPTNOI) the coefficient vector s_{MATCH} is computed within the OMP procedure, but only for those indices that are found in the set S_{MATCH} .

The computation of the ‘trusted’ spectrum is found as

$$\hat{x}_{SN} = \Psi s_{SN} \quad (6.9)$$

where s_{SN} is a k -sparse binary vector whose support is given by the sensor node indices listed in S_{SN} . The *reconstruction error heuristic* δ_{RECON} is found as the difference between the reconstruction formed from the atoms identified in the matching process (as listed in S_{MATCH}) with the reconstruction formed from the atoms (where each atom corresponds to a known SN) in the GDB that are known to be local to the CR (as identified in S_{SN})

$$\delta_{RECON} = \hat{x}_{CR} - \hat{x}_{SN}. \quad (6.10)$$

The reconstruction error heuristic (δ_{RECON}) is used as a detection heuristic for SSDF.

A modification of the GSV process for the reconstruction heuristic could also be made using a non-centralized approach in conjunction with a centralized SAS. This hybrid approach may reduce network traffic overhead by directing the SNs (who receive the CSU spectrum request) to share and collaborate prior to forwarding the CSU spectrum report to the SAS. Specifically, the SNs could share their respective spectrum reports amongst each other and collectively verify the CSU spectrum report. Each SN could approximate the CSU spectrum report using a greedy algorithm with a small dictionary formed from the columns of each of the local SNs, and then test the veracity of the reported spectrum by computing the difference between the reconstructed spectrum and the CSUs reported spectrum. Then, the outcome of this test could be forwarded to the SAS, thereby verifying the CSU spectrum and reducing overall network traffic. However, doing so would not be able to identify the presence of spectrum spoofing SSDF.

6.2.4 Spectrum Falsification Heuristic 3 (δ_{SIGN}): Correlation Sign

Using the primary heuristic δ_{MATCH} , another SSDF detection heuristic is formed by ascertaining whether the reported spectrum correlates positively or negatively with the atoms in the GDB. Even if S_{MATCH} contains relatively many SN IDs (corresponding to a relatively high correlation), an inverted SSDF attack may be occurring. If a high δ_{MATCH} is observed with the condition that many of the matches were made under a negative correlation, then this can indicate the presence of an inverted spectrum attack. A small addition to the native OMP algorithm needs to be made in order to capture the sign of the correlation and to reliably detect such spectral inversion attacks using

OMP. The OMP algorithm typically presented in the literature uses the absolute value or square of the correlation [35] which disregards the sign of the correlation, as is done in the *identify* step of the OMP algorithm, as shown in Algorithm 1 in Section 3.5.2. In order to capture this, the sign of the entry of $h^\ell(j)$ given in Algorithm 1 that corresponds to the maximum entry of $h^\ell(j)$ needs to be stored at each of the k iterations. The result is that for each index returned in S_{EST} , there is a corresponding binary number indicating the sign that the correlation took when the index was selected during the OMP process. However, instead of operating on all the indices within S_{EST} , we concern ourselves only with those indices that are known to match the set of trusted SN indices, and therefore work exclusively with the in the matched index set S_{MATCH} . A heuristic to measure the number of negatively correlated atoms selected during OMP can be formulated by counting the total number of negative correlations occurring in S_{MATCH} and comparing it to the cardinality of the index set.

$$\delta_{SIGN} = \frac{\#of\ negative\ correlations\ in\ S_{MATCH}}{\delta_{MATCH}}. \quad (6.11)$$

6.2.5 Related Methods

Many methods in the literature attempt to measure the consistency of spectrum sensing results reported by various SNs, regarding the occupancy of a *single* channel, i.e., on single binary reports aggregated from multiple sensors over a relatively wide area. In this dissertation, the method compares multiple channels simultaneously. While these simplified scenarios may possibly be of benefit for certain applications, a more realistic and complex case occurs when each individual sensing report contains the power (or simply a binary digit indicating presence or absence) locally sensed over several channels.

Furthermore, the method presented here strives to *reconstruct* the spectrum report from multiple other spectrum reports, as opposed to just correlate with it. The perspective taken here is that there are multiple spectrum reports that correlated with the candidate spectrum report, and therefore a greedy technique such as orthogonal matching pursuit is used.

In addition, CRN *densification* falls within the scope of this present work. The dense network as

addressed in this work senses many users and facilitates geolocation and therefore SSDF detection. However, the increased number of users and frequency reuse also affects the ability to identify a particular CR based on its reported spectrum. More on dense and ultra-dense networks can be found in [2].

The attack model presented in [81] describes some attacks in which malicious passive nodes can infer the location of a CR (in particular of a SU) based on the channels used by the CR, as well as some methods intended to prevent such information leakage. This present work is similar in some respects to [81], but instead of making inferences regarding a CR on its frequency (channel) *usage* as in [81], in this dissertation the inference is based on which active channels are *sensed* by the CR, i.e., on the channels the CR reports as being used by other devices, not which channels are used or queried by the CR itself.

GSA as proposed here is based on each trusted sensor node (SN) contributing a portion (albeit, an overlapping portion) of the candidate secondary users (CSU) spectrum. However, if it is known definitively that sensors are very close to each other over the entire coverage area, then the spectrum reported by each of the SNs tends to be highly correlated. In that case, multi-sensor compressive sensing (CS) algorithms can be applied, e.g. simultaneous orthogonal matching pursuit (SOMP), to account for the joint-sparsity of the SN reports. In the CS literature, this situation is known as the multiple measurement vector (MMV) CS scenario. Further research could study the different types of MMV recovery algorithms as applied to such an ultra-dense network.

A related method utilizing coverage and channel information that strives to estimate transmitter location for radio environment map (REM) construction is RF fingerprinting [82, 83]. Publications regarding RF fingerprinting typically utilize RSS and round trip time delay and are processed on a per-channel basis. Many RF fingerprinting methods use a propagation model, sometimes augmented with drive-tested measurements to estimate the location of a transmitter. A common CR application of RF fingerprinting is to locate or identify a small number of active transmitters (usually PUs) within the coverage area [84]. In contrast, the method presented here does not rely on RSS predicted by a propagation model, but rather matches a receiver's PSD to the joint frequency pattern sensed by a small number of local SNs close to the receiver.

Reference [6] presents a survey of CS applications in CRNs. CRN tasks such as spectrum sensing, parameter estimation (channel state, direction of arrival, signal-to-noise ratio (SNR), sparsity order, etc.), and REM database construction (a.k.a. spectrum cartography) are reviewed (i.e., active

transmitter localization). The current paper shares some methods in common with these techniques, but is distinct in nature. E.g., in sparse transmitter localization it is assumed that there are only a few active transmitters. In this paper, there are many active transmitters, but the number of spectrum reports from trusted nodes that are required to adequately approximate the candidate CRs estimated PSD is assumed sparse. Also, the vast majority of publications address the problem of PU signal identification without regard for other local active SUs. Here, the PSD reported over a finite set of channels (whether occupied by other SUs or the PU itself) is used a feature vector to assess the candidate CR. To this author's knowledge, no publications have attempted to address the problem of SSDF detection by utilizing sparse approximation or CS, where the PSD itself is dense, yet the PSD can be sparsely approximated in an overcomplete dictionary.

7 Simulation and Test

7.1 Overview

7.1.1 Test Approach: Test Cases and Functional Areas Tested

The test approach is to perform numerical simulation of the algorithms described in Sections 3, 4, 5, and 6, and utilize the simulation results to verify comprehension and feasibility of the algorithms. The focus is on testing the effectiveness of the approximation methods mainly for support estimation, as well as evaluating the ability of the difference operators to measure differences in distributions of Gram matrices formed from different effective dictionaries. The approximation performance translates to the ability to detect spectrum sensing data falsification (SSDF), which is also explored.

There are seven documented simulation test cases (TC), numbered TC-01 through TC-07, which are listed in Table 4.

Table 3: Simulation Test Case Names and Locations

Test Case Label	Test Case Name	Document Section Number
TC-01	Gaussian Dictionary, Baseline Support Estimation Performance	7.2.1
TC-02	Gaussian Dictionary with Dictionary Evolution	7.2.2
TC-03	Gaussian Dictionary with Sensing Matrix Evolution	7.2.3
TC-04	Spectrum Dictionary with Sensing Matrix Evolution (Inversion)	7.3.2
TC-05	Spectrum Dictionary with Sensing Matrix Evolution (Spoofing)	7.3.3
TC-06	Spectrum Dictionary with Sensing Matrix Evolution (Spectral Shifting)	7.3.4
TC-07	Compression and Computational Complexity	7.4

The simulation test cases are designed to evaluate the performance of three classes of tools (with regards to their respective functional performance areas):

- Effective Gram Matrix Histograms: With regards to their ability to characterize the optimality of effective dictionaries
- Approximation Methods: With regards to their ability to identify the support of a signal within a dictionary, and to reconstruct the signal

- Difference Operators: With regards to their ability to distinguish (compute distance) between effective dictionaries on the basis of their Gram matrix histograms, as well as their ability to relate the computed distance to approximation performance.

The *effective Gram matrix histogram* is a histogram that is computed from the entries of the Gram matrix formed from an effective dictionary, and measures the geometry of the effective dictionary. The approximation methods perform support estimation and signal reconstruction. The difference operators tie together the changes in geometry encoded in the effective Gram matrix histogram and the approximation methods performance. The SSDF detection methods follow directly from the approximation methods, and as a result are functionally lumped together.

In addition to the above, the reduction in computational complexity resulting from using compressive sensing matrices is briefly simulated. There exists some level of computational burden required to compute the optimized sensing matrix, as it requires the computation of an SVD of the dictionary with complexity $\mathcal{O}(\min(m^2n, mn^2))$ [68]. From a computational efficiency perspective, it is of value to minimize the number of SVDs that are computed over time. In order to minimize the number of SVDs taken, it is useful to quantify and interpret the impact of the sensing matrix on the geometry of the current dictionary (particularly in the case of a sensing matrix that was optimized for a previous version of the dictionary.) Section 7.4 contains a short simulation study of the trade-off between the SVD and the compression regarding computational complexity.

7.1.2 Simulation Parameters

The size of the dictionary in every test case was held constant at 200×300 . The simulation parameters that were varied within TC-01 (Baseline Support Estimation) were:

- Number of Measurements (m)
- Signal-to-Noise Ratio (SNR)
- Sparsity (k)

The number of measurements (m) and sparsity (k) for TC-02 through TC-06 was held constant at $m = 70$, and $k = 5$, respectively. The list of simulation test parameters that were used within TC-02 and TC-03 includes:

- SNR: 2 dB
- Dictionary Type: Gaussian

- Sensing Matrix: Identity, Pseudo-Random, Optimized, Stale
- Stale Effective Dictionary Construction (or Perturbation) Type: Sensing Matrix or Dictionary
- Percent of Atoms Perturbed in Dictionary: 25, 50, 75, or 100%

The list of simulation test parameters that were simulated for TC-04 through TC-06 includes:

- SNR: 8 dB
- Dictionary Type: Spectrum Database
- Sensing Matrix: Identity, Pseudo-Random, Optimized, Stale
- Stale Effective Dictionary Construction (or Perturbation) Type: Sensing Matrix
- Percent of Atoms Perturbed in Dictionary: 25, 50, 75, or 100%

The test parameters in TC-02 through TC-06 are elaborated on in the following sub-sections. For test case TC-07 (Compression and Complexity), the parameter that was varied was the number of OMP iterations.

7.1.2.1 Approximation Method Configurations Simulated

Five combinations of sensing matrix types/recovery methods were simulated for this dissertation. Each combination of sensing matrix and recovery method defines an *approximation configuration*. The five approximation configurations simulated are:

- Identity Sensing Matrix with inner product correlation (CORR)
- Identity Sensing Matrix with OMP (SA-OMP)
- Pseudo-random CS Matrix with OMP (CS-RAND)
- Optimized CS Matrix with OMP (CS-OPT)
- Stale/Noisy Effective Dictionary with OMP (CS-OPTNOI)

CS-OPTNOI is actually a noisy variant of CS-OPT, but is counted as a separate case to clarify the simulation test cases. These are also listed in Table 5, organized by sensing matrix and recovery method. There are two different recovery or matching methods simulated: inner-product correlation (IPC) and orthogonal matching pursuit (OMP), both of which are discussed in Section 3.5.

The first approximation configuration simulated is termed *CORR*, and uses the inner product correlation method defined in Section 3.5.3 along with an $N \times N$ identity sensing matrix. IPC is simulated as a comparison to the OMP matching method. It uses the normalized correlation profile z without orthogonalization, as formulated in Eq. (3.25), and returns the indices corresponding to

Table 4: List of Approximation Configurations Simulated. A dash '-' indicates that corresponding combination was not tested.

		Sensing Matrix			
		Identity ¹	Random ²	Optimized ²	Stale ²
Recovery Method	IPC	CORR	—	—	—
	OMP	SA-OMP	CS-RAND	CS-OPT	CS-OPTNOI

1. Matrix is not compressive 2. Matrix is compressive

the top k values of z as the estimation of dictionary support. Next, the *SA-OMP* approximation configuration is simulated and consists of utilizing the OMP algorithm reviewed in Section 3.5.2 along with an $N \times N$ identity sensing matrix.

While CORR and SA-OMP have no compression, the CS approach is also simulated by pre-multiplying the input signal to be approximated x and the dictionary by a pseudo-random *sensing matrix* Φ (see Section 3.3.2), resulting in a set of compressed measurements y . This is referred to as *CS-RAND*. The elements of the $m \times N$ sensing matrix are chosen according to Eq. (3.9). The solution to the optimization problem appearing in Eq. (3.8) is approximated using OMP. The OMP process returns the k -sparse coefficient vector s representing the support of the measurement vector in terms of the dictionary. The non-zero indices of the vector returned is the estimate of dictionary support.

The last approximation configuration tested is referred to as *CS-OPTNOI*. There are four evolutions of CS-OPTNOI, depending on the extent of the staleness (or evolution) of the effective dictionary.

7.1.2.2 Dictionaries Simulated

For TC-02 through TC-06, the test cases are sub-divided based on two classes of dictionaries: a generic pseudo-random dictionary (formed from a Gaussian distribution) and a cognitive radio network spectrum dictionary. The results for the Gaussian dictionary are contained in Section 7.2 and, the results for the spectrum database dictionary are in Section 7.3. In each case, the dictionaries incorporate pseudo-randomly generated elements, but the CRN spectrum database dictionaries are constructed in such a way as to roughly simulate the physical constraints of a sectorized, cellular-based RF network. For each of the two dictionary types simulated, there are also noisy or *perturbed* versions of the original dictionary Ψ_0 resulting in stale effective dictionaries which are incorporated

into the simulation.

It should be noted that there were actually a family of generic pseudo-random dictionaries that were generated by populating the dictionary matrix entries with pseudo-random values from three probability distributions (Gaussian, Weibull, and uniform distributions). However, due to limited space and resources, only the results from the Gaussian dictionary (*gauss*) and CRN spectrum dictionary (*spect*) are presented.

7.1.2.3 Sensing Matrices Simulated

There are four different types of sensing matrices were studied and are tested in this dissertation:

- Identity matrix
- A column-normalized random sensing matrix whose entries are generated by pseudo-randomly sampling a standard normal distribution, as discussed in Section 3.3.2
- An optimized sensing matrix that is computed via the methods presented in Section 4.3
- A sensing matrix that was formerly 'optimized' for a particular dictionary, but since being optimized, has incurred updates to its columns, resulting in a stale effective dictionary

7.1.2.4 Stale Effective Dictionary Construction: Dictionary and Sensing Matrix Perturbations

Recall from Section 3.3.1 that the effective dictionary is formed from the matrix multiplication of a sensing matrix with the dictionary. There are two methods for constructing a stale effective dictionary, as defined in Section 5.2. For the cases involving a stale/noisy effective dictionary, four different stale effective dictionaries of increasingly perturbed entries are constructed from the original, as discussed in Section 5.3. Two stale effective dictionary construction methods were applied, as outlined in Section 5.3.

For the pseduorandom Gaussian dictionary, both Method 1 (Dictionary Evolution), and Method 2 (Sensing Matrix Evolution) are simulated. To perturb or evolve the dictionary columns for both methods for the Gaussian dictionary, new columns were drawn from the same Gaussian distribution of mean 0 and standard deviation 1.

For the CRN spectrum database (GDB) dictionary, only Method 2 (Sensing Matrix Evolution) was utilized. The reason that effective dictionary evolution method 1 (evolving the dictionary) was not applied to the spectrum dictionary was that it proved impractical to generate the four different perturbed CRN spectrum maps to generate the four perturbed spectrum dictionaries.

7.1.3 Detailed Review of Test Cases

First, in Section 7.2.1, a baseline simulation for support estimation performance is performed for the Gaussian dictionary. Support estimation performance versus the number of compressive measurements, SNR, and sparsity are shown for CORR, SA-OMP, CS-RAND, and CS-OPT.

Next, in Section 7.2.2 for the Gaussian dictionary with *dictionary evolution*, the following graphs are shown for CORR, SA-OMP, CS-RAND, CS-OPT, and CS-OPTNOI:

- **Gram Analysis:** Histograms of the Gram matrix entries for effective dictionaries corresponding to each of the four sensing matrix types
- **Support Estimation Performance:** Histograms of the number of matched atoms for each of the four sensing matrix types
- **Gram Difference Operators:** The mean matching performance versus the three difference operators (Sensing Matrix Optimization Objective Function, KS statistic, and KL divergence)
- **SSDF Detection Results**

In Section 7.2.3, similar graphs are then generated for the Gaussian dictionary with *sensing matrix evolution*.

Next, the Gram Analysis and Support Estimation is performed for the spectrum dictionary (GDB) with sensing matrix evolution, as a means of measuring the ability to perform spectrum approximation, and SSDF detection for the three SSDF types (inversion, spoofing, and shifting). The Gram Difference Operators are computed for the SSDF Shifting Spectrum case. The support estimation performance is evaluated for all the approximation methods (CORR, SA-OMP, CS-RAND, CS-OPT, and CS-OPTNOI) using the GDB as the dictionary. Table 6 lists a summary of the simulation test cases, and where they can be found in this document. Lastly, Section 7.4 evaluates the impact of compression and sensing matrix optimization on the OMP process, in terms of computational complexity over multiple iterations.

7.1.4 Simulation Notes

The simulation graphs were computed using MATLAB (R2017b) simulation software, which included various toolboxes. The MATLAB software was run on a 64-bit desktop computer with a 3.80 GHz processor and 32.0 GB of memory. Short test procedures including MATLAB commands are included in each subsequent sub-section to replicate similar results to those found in the graphs. The MATLAB code for a subset of the core MATLAB functions is included in Section 9.8.

Table 5: Details of Main Simulation Test Cases (Not Including the Complexity/Compression Test Case: TC-07)

Dictionary	Evolution Method (for CS-OPTNOI)	SSDF Tested	Gram Analysis	Approximation Performance	Difference Operator	Test Case, Section
Gaussian	n/a	n/a	n/a	Figs. 17, 18, 19	n/a	TC-01, 7.2.1
	Dictionary	n/a	Fig. 20	Fig. 21	Fig. 22	TC-02, 7.2.2
	Sensing Matrix	n/a	Fig. 23	Fig. 24	Fig. 25	TC-03, 7.2.3
Spectrum	Sensing Matrix	Inverted	n/a	Fig. 28, 29	n/a	TC-04, 7.3.2
		Spoof	n/a	Fig. 30, 31	n/a	TC-05, 7.3.3
		Shifted	n/a	Fig. 32, 33	Fig. 34, 35	TC-06, 7.3.4

7.2 Gaussian Dictionary (TC-01 through TC-03)

7.2.1 TC-01: Gaussian Dictionary, Baseline Support Estimation Performance

In this subsection, some baseline comparisons of OMP, correlation, random compression and optimized compression with respect to support estimation are studied. The dictionary is a 200×300 matrix populated with entries drawn pseudorandomly from a standard normal distribution. The comparisons are varied across number of measurements (m), signal to noise ratio (SNR) and the number of atoms (k , sparsity) used to construct each input vector. The vectors to be approximated (input vectors) are constructed by randomly selecting atoms from the dictionary and forming a linear combination of the atoms using coefficients that are selected uniformly on the interval $[0,1]$. The performance metric used is the percentage of Monte Carlo trials in which the support was perfectly recovered. For each simulation, there were 1000 Monte Carlo trials performed for each m , SNR , and k .

7.2.1.1 Support Estimation vs Number of Measurements

The results from the first baseline comparison are shown in Fig. 17. As might be expected, the number of trials resulting in perfect support recovery increases with the number of compressive measurements (m). The number of compressive measurements is varied from 10 to 200 in increments of 10 for both CS-RAND (CS using a random sensing matrix) and CS-OPT (CS using an optimized sensing matrix). Also as expected, the optimized sensing matrix performs better across all measurements, and tend to converge as the number of measurements approaches the dimension of the atoms ($N = 200$). It is interesting to note that at $m = N = 200$, the performance decreases sharply. This result was seen repeatedly over many different runs of the simulation. The signal to be approximated was constructed of $k = 5$ atoms from the dictionary, and its SNR was brought down to 2 dB by adding Gaussian noise.

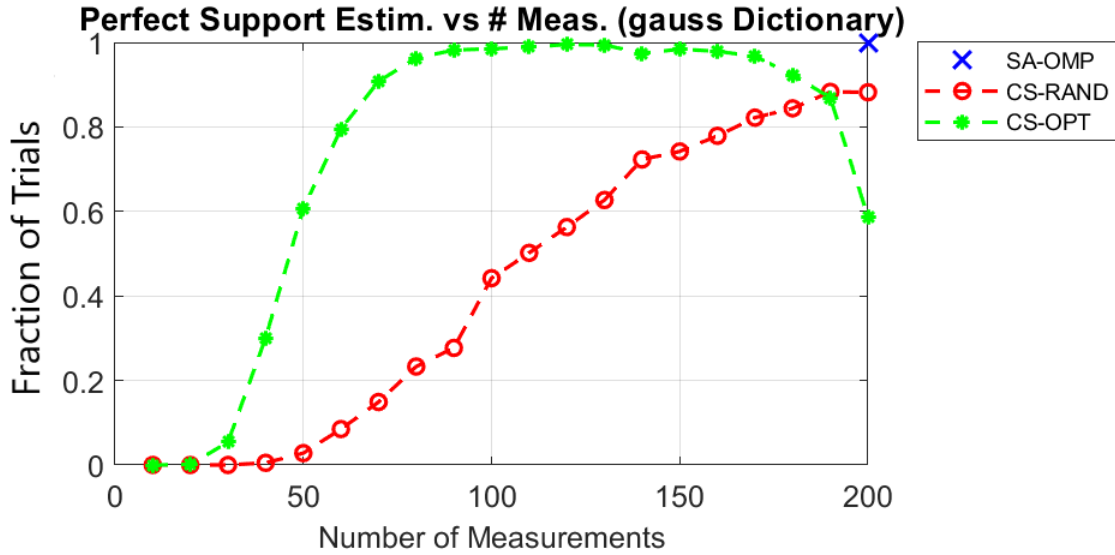


Figure 17: Support estimation versus number of compressive measurements for SNR = 2 dB, N=200, M=300, k=5, over 1000 Monte Carlo trials.

Although there is no compression in SA-OMP, Fig. 17 also shows the results for SA-OMP at the full number of measurements (200). It is interesting to note that neither CS-RAND nor CS-OPT attains the performance of SA-OMP at the full number of measurements. It is hypothesized that the reason for this is that SA-OMP involves a trivial sensing matrix (i.e., SA-OMP applies only the identity sensing matrix to the noisy input signal). CS-RAND and CS-OPT each apply a sensing matrix which operates on the noisy input signal, and has the effect of amplifying the noise:

$$\begin{aligned}
 y &= \Phi x \\
 &= \Phi (x_o + \eta) \\
 &= \Phi x_o + \underbrace{\Phi \eta}_{\text{Amplified Noise}}
 \end{aligned} \tag{7.1}$$

Therefore, even though at the full number of measurements there is no compression, performance degradation due to noise amplification is observed as a result of amplification of the existing noise, known as *noise-folding* [85–87]. The mere presence of the sensing matrix (even though it's not compressive) introduces measurement noise into the signal that is not present in SA-OMP without sensing matrix multiplication.

The following steps can be used to create a similar version of Fig. 17:

1. At the MATLAB command prompt, enter
`optmatdemo_vs_meas('gauss')` and hit *< Enter >*.
2. Wait for the function to complete.
3. Upon completion, the graphs are displayed and are saved to the current MATLAB directory path.

7.2.1.2 Support Estimation vs SNR

Next, the results from the second baseline comparison is shown in Fig. 18. The general trend is that the performance increases with SNR. In this simulation, the graph marked 'CORR' is inner-product correlation, and represents the first few steps in the OMP process, but without orthogonalization or compression. SA-OMP returns the best atom selection accuracy. It utilizes no compression and uses OMP to decompose the input vector in terms of atoms from the dictionary. Next, CS-OPT and CORR relatively close in performance. It is interesting to note a decrease in performance from SA-OMP to CORR. This is likely a direct result of the orthogonalization step used by OMP, but absent in CORR which acts to subtract out the contribution from each atom separately. This results in a more accurate atom matching process. Also, the optimized sensing matrix used in CS-OPT with compression, performs about as well as CORR without compression. Lastly, compression through use of a random sensing matrix results in the worst performance.

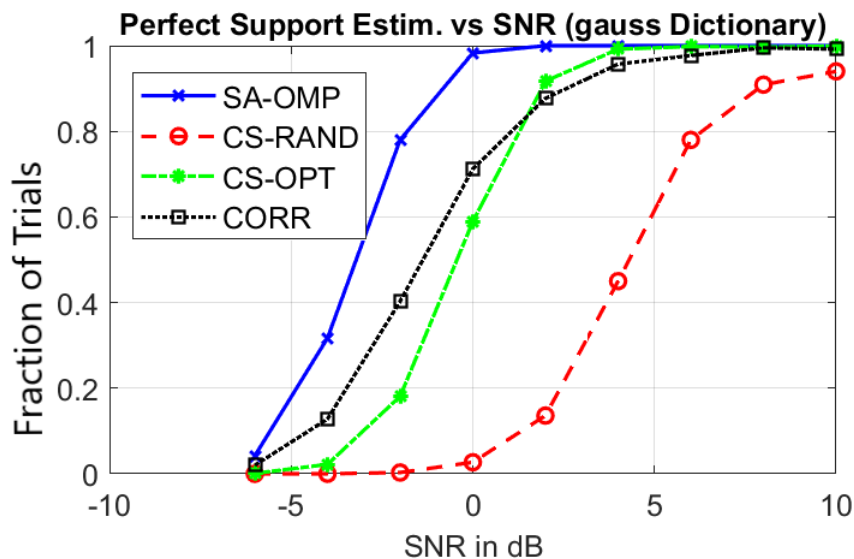


Figure 18: Support estimation versus SNR for $m=70$, $N=200$, $M=300$, $k=5$, over 1000 Monte Carlo trials.

The following steps can be used to create a similar version of Fig. 18:

1. At the MATLAB command prompt, enter
`optmatdemo vs snr('gauss')` and hit *< Enter >*.
2. Wait for the function to complete.
3. Upon completion, the graphs are displayed and are saved to the current MATLAB directory path.

7.2.1.3 Support Estimation vs Sparsity

Last in the baseline analysis are the results from the support estimation performance as a function of sparsity, shown in Fig. 19. This graph is very similar to the analysis of measurements in Fig. 18. It is hypothesized that this is a direct result of the strong relationship among sparsity and the number of measurements as seen in the lower bound on the number of compressed measurements (m) needed for recovery of a k -sparse, N – dimensional signal: $m \approx k \log(N/k)$, as discussed in Section 3.3.1.

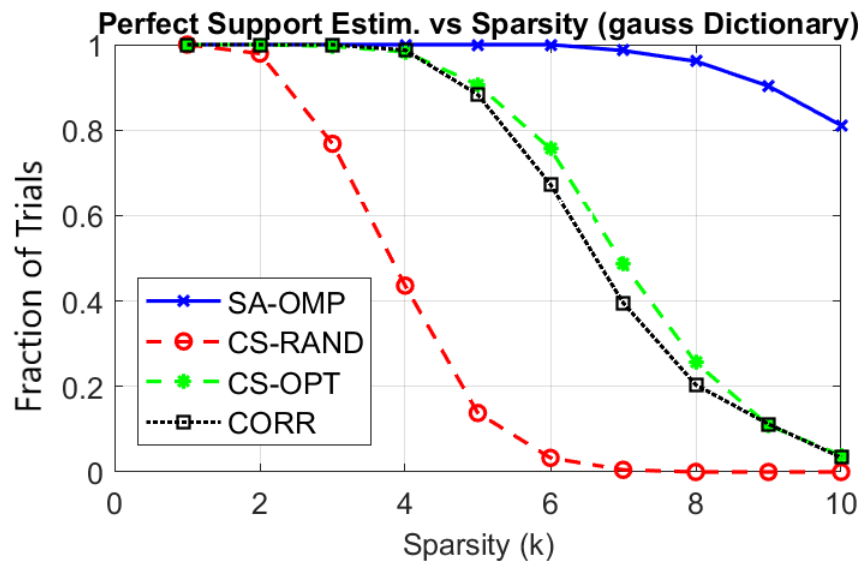


Figure 19: Support estimation versus sparsity for SNR=2 dB, $m=70$, $N=200$, $M=300$, over 1000 Monte Carlo trials.

The following steps can be used to create a similar version of Fig. 19:

1. At the MATLAB command prompt, enter
`optmatdemo vs sparsity('gauss')` and hit *< Enter >*.
2. Wait for the function to complete.
3. Upon completion, the graphs are displayed and are saved to the current MATLAB directory path.

7.2.2 TC-02: Gaussian Dictionary with Dictionary Evolution

Within this subsection dictionary entries are populated by drawing entries pseudo-randomly from a standard Gaussian (zero-mean, unity variance) distribution. The stale effective dictionaries for the CS-OPTNOI cases are constructed via Method 1 of Section 5.3 (dictionary evolution), i.e. by using a single sensing matrix with dictionaries that differ by 25%, 50%, 75%, 100% from the original dictionary used to optimize the sensing matrix.

7.2.2.1 Gram Analysis

Fig. 20 depicts four histograms of the absolute value of Gram matrix entries corresponding to the four sensing matrix types multiplied by dictionaries with Gaussian entries. The *Stale* case corresponds to the case in which the most stale or the most evolved dictionary was used to generate the 'optimized' sensing matrix. As is expected, the tail of the histogram corresponding to CS-RAND extends out further than that of the initial (original) dictionary, due to the random compressive sensing matrix applied. Also, the optimized case (CS-OPT) shows a reduction of the tail and an increase in the number of Gram entries closer to zero. This indicates that the atoms are being driven further apart (more incoherent) than in the CS-RAND case. Lastly, the CS-OPTNOI (Stale) case indicates that the spread of atoms is slightly better than that of the CS-RAND case, but worse than the CS-OPT case.

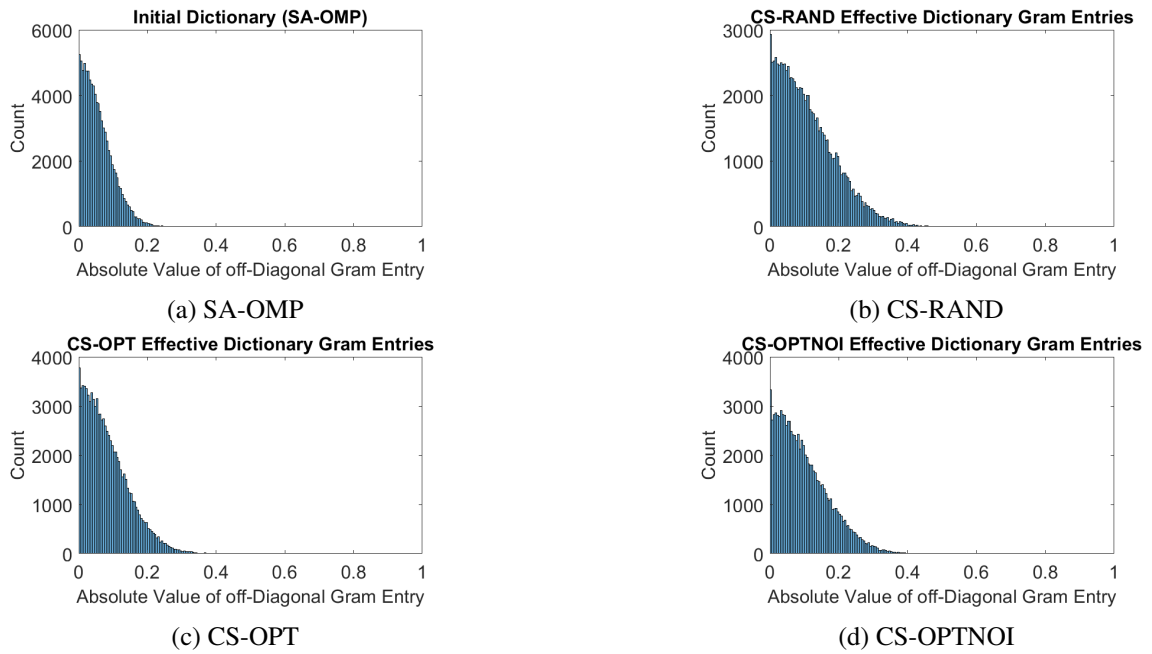


Figure 20: Histograms of Off-Diagonal Gram Matrix Entries of the Effective Dictionaries for the four types of sensing matrices.

7.2.2.2 Support Estimation per Sensing Matrix

Fig. 21 displays histograms of the results from a Monte Carlo analysis (1000 trials) of support estimation performed with the four sensing matrices: identity (SA-OMP), random (CS-RAND), optimized (CS-OPT) and optimized-but-stale (CS-OPTNOI). As expected, the SA-OMP approach with no compression returns the highest atom selection accuracy. CS-RAND results in a large decrease in atom selection accuracy, however, using an optimized sensing matrix in CS-OPT greatly restores the atom selection performance almost to that of SA-OMP, but with compression. CS-OPTNOI represents a decrease in performance compared to CS-OPT, as expected, since the dictionary used is entirely different, though identically distributed as the dictionary that used to optimize the sensing matrix. It is interesting to note that CS-OPTNOI performs better than CS-RAND. This accords with the Gram analysis results of these sensing matrices, as seen in Fig. 20. Specifically, there are more Gram entries for CS-OPTNOI near zero than there are in the CS-RAND case.

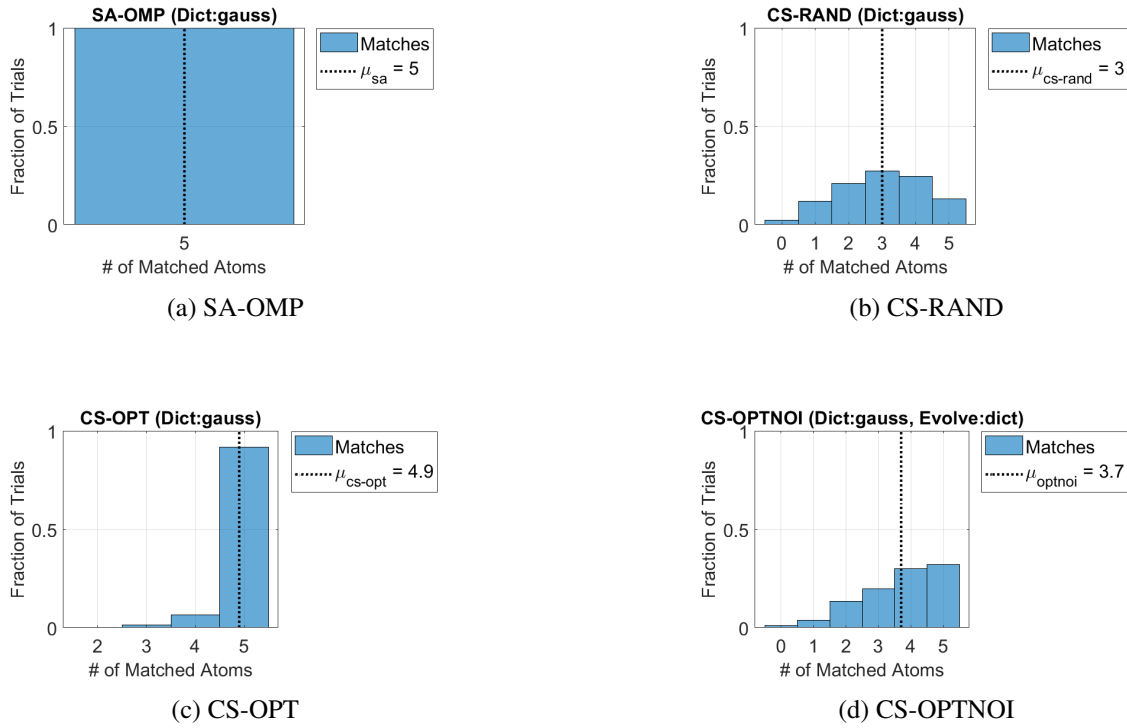
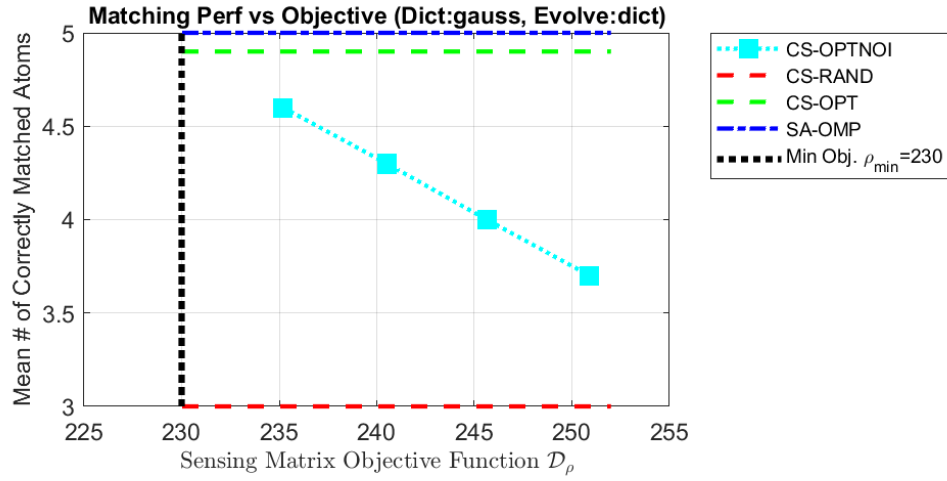


Figure 21: Support estimation simulation results using a pseudo-randomly generated 200×300 dictionary matrix whose entries were drawn from a Gaussian distribution, over 1000 Monte Carlo trials. The input SNR of the signal was 2 dB, and the number of compressive measurements where applicable was $m = 70$. The CS-OPTNOI case represents the worst case sensing matrix staleness.

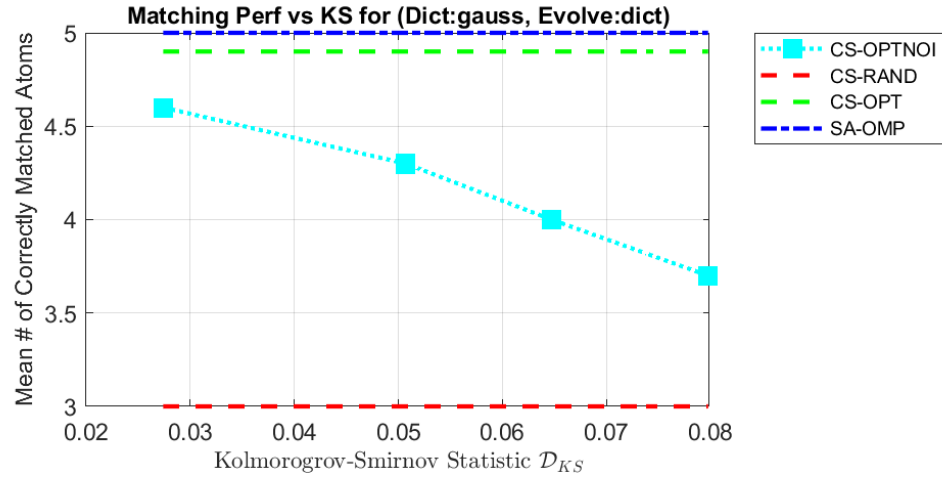
7.2.2.3 Support Estimation vs Difference Operator

Fig. 22 shows the average number of correct atoms selected at each iteration of a Monte Carlo simulation (with 1000 trials) as the sensing matrix is increasingly altered over four levels of evolution, making the effective dictionary increasingly evolved. The difference between the optimal effective dictionary and the evolved or stale effective dictionary is measured by applying the three difference operators that were developed in Section 5. As can be seen, the difference between optimal and evolved effective dictionaries can be gauged via each distance operator.

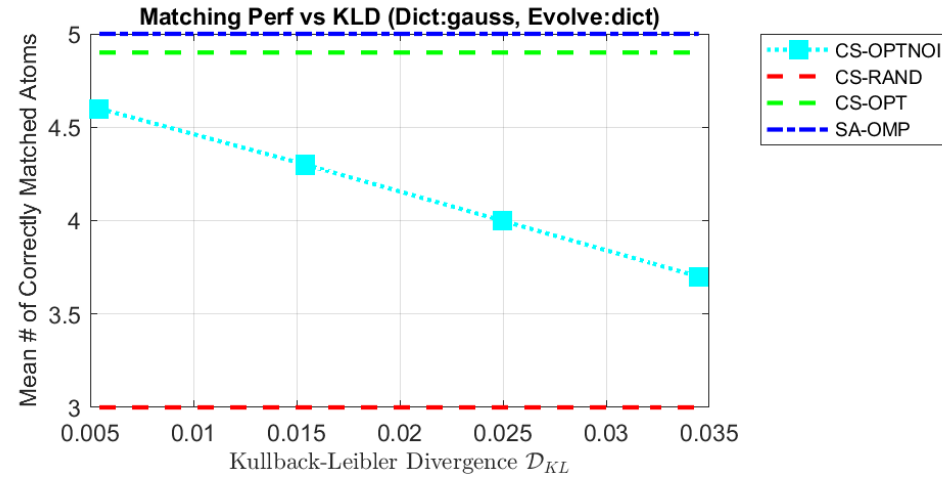
In each of the three graphs of Fig. 22, the support estimation performance for SA-OMP, CS-RAND, and CS-OPT does not vary over each of the four levels of evolution, but are included on the graph for reference. For the objective function distance operator, the minimum (optimal) objective function value is shown for reference. It is interesting to note that in each case, the estimation performance trend is similar as the sensing matrix evolves from that of an optimized sensing matrix to that of a randomly generated sensing matrix.



(a) Objective function difference operator, Gaussian-Distributed Entries



(b) Kolmogorov-Smirnov Statistic difference operator, Gaussian-Distributed Entries



(c) Kullback-Leibler Divergence difference operator, Gaussian-Distributed Entries

Figure 22: Support estimation versus difference operators for Gaussian dictionary with dictionary evolution.

7.2.3 TC-03: Gaussian Dictionary, Sensing Matrix Evolution

Within this subsection dictionary entries are populated by drawing entries pseudo-randomly from a standard Gaussian (zero-mean, unity variance) distribution. The stale effective dictionaries for the CS-OPTNOI cases are constructed via Method 2 of Section 5.3 (sensing matrix evolution), i.e. by using a series of sensing matrices with the original dictionary. The sensing matrices were optimized using dictionaries that differ by 25%, 50%, 75%, 100% from the original dictionary.

7.2.3.1 Gram Analysis

Fig. 23 depicts four histograms of the absolute value of Gram matrix entries corresponding to the four sensing matrix types multiplied by dictionaries with Gaussian entries. The trends in these histograms for sensing matrix evolution mirrors that of the dictionary evolution Gram analysis histograms shown in Fig. 20.

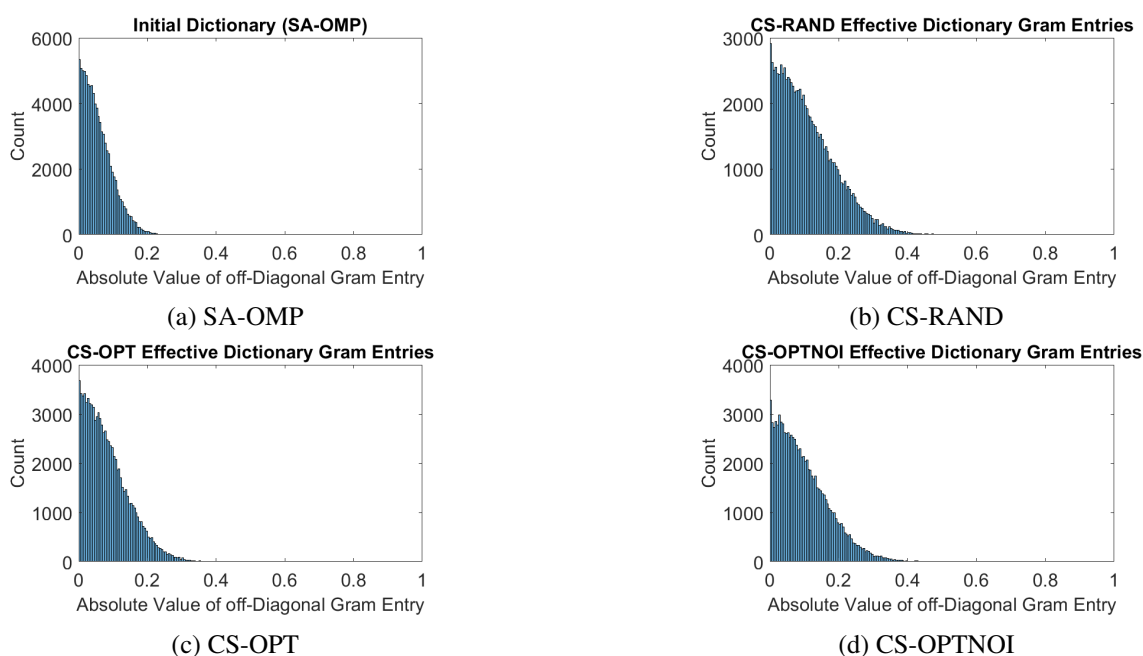


Figure 23: Histograms of off-diagonal Gram matrix entries of the effective dictionaries for the four types of sensing matrices.

7.2.3.2 Support Estimation per Sensing Matrix

Fig. 24 displays histograms of the results from a Monte Carlo analysis (1000 trials) of support estimation performed with the four sensing matrices: identity (SA-OMP), random (CS-RAND), optimized (CS-OPT) and optimized-but-stale (CS-OPTNOI). As expected, the SA-OMP approach

accuracy, however, using an optimized sensing matrix in CS-OPT greatly restores the atom selection performance almost to that of SA-OMP, but with compression. CS-OPTNOI represents a decrease in performance compared to CS-OPT, as expected, since the dictionary used is entirely different, though identically distributed as the dictionary that used to optimize the sensing matrix. It is interesting to note that CS-OPTNOI performs better than CS-RAND. This accords with the Gram analysis results of these sensing matrices, as seen in Fig. 20. Specifically, there are more Gram entries for CS-OPTNOI near zero than there are in the CS-RAND case.

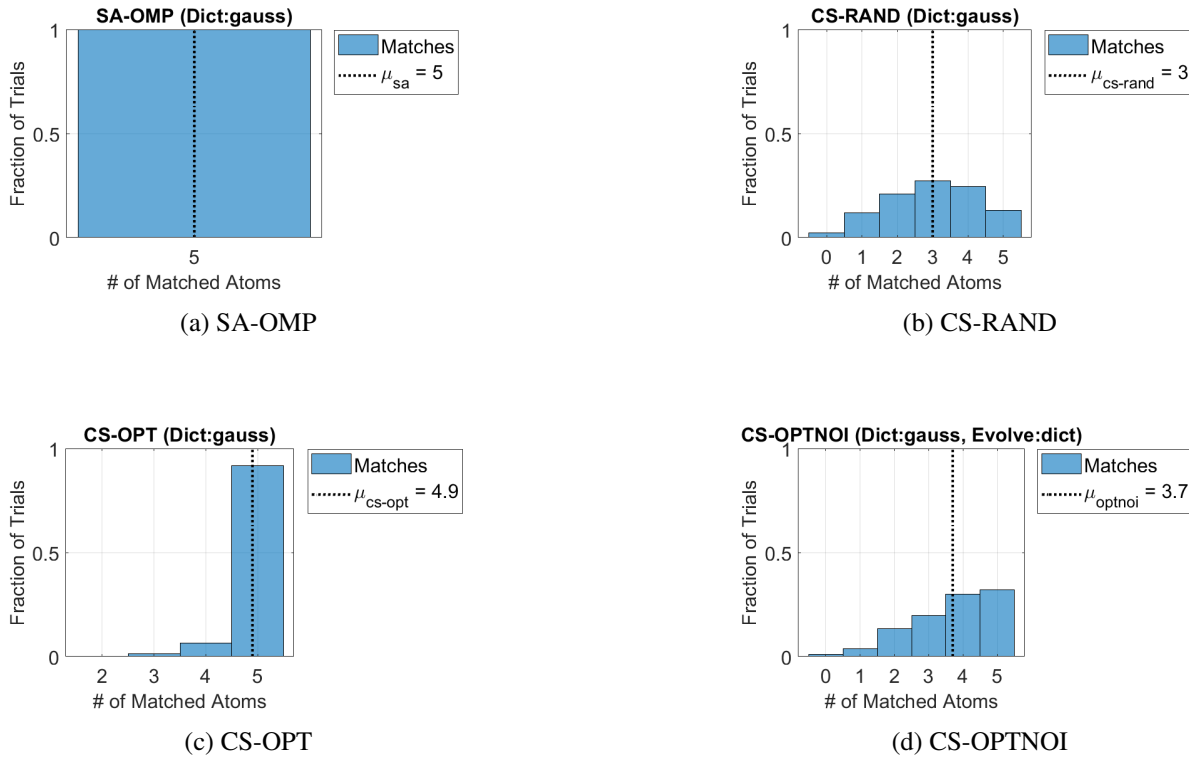
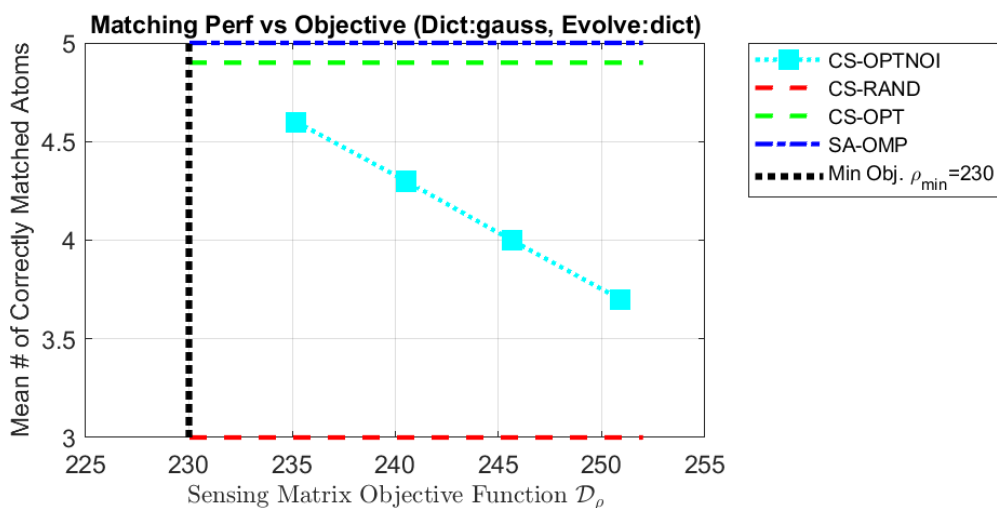


Figure 24: Support estimation simulation results using a pseudo-randomly generated 200×300 dictionary matrix whose entries were drawn from a Gaussian distribution, over 1000 Monte Carlo trials. The input SNR of the signal was 2 dB, and the number of compressive measurements where applicable was $m = 70$. The CS-OPTNOI case represents the worst case sensing matrix staleness.

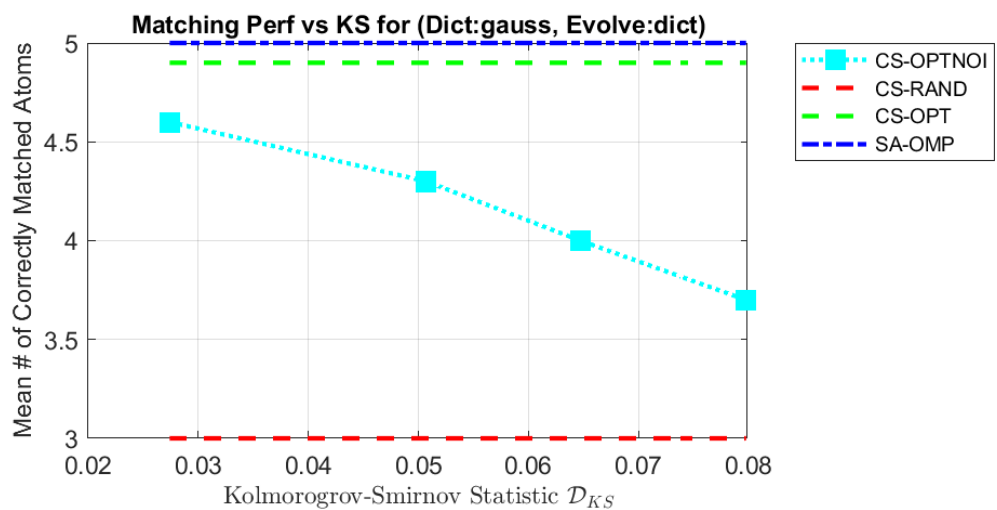
7.2.3.3 Support Estimation vs Difference Operator

Fig. 25 shows the average number of correct atoms selected at each iteration of a Monte Carlo simulation (with 1000 trials) as the sensing matrix is increasingly altered over four levels of evolution, making the effective dictionary increasingly evolved. The difference between the optimal effective dictionary and the evolved or stale effective dictionary is measured by applying the three difference operators that were developed in Section 5. As can be seen, the difference between optimal and evolved effective dictionaries can be gauged via each distance operator.

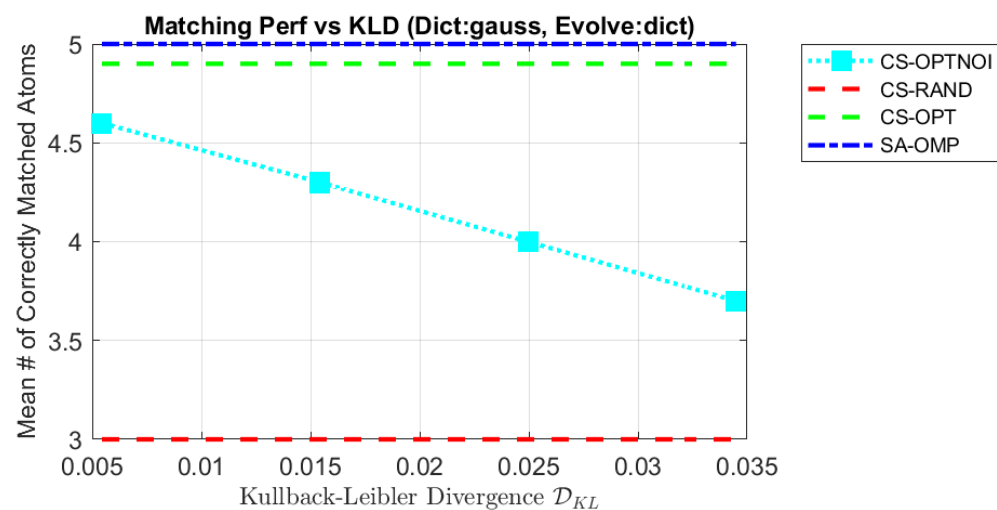
In each of the three graphs of Fig. 25, the support estimation performance for SA-OMP, CS-RAND, and CS-OPT does not vary over each of the four levels of evolution, but are included on the graph for reference. For the objective function distance operator, the minimum (optimal) objective function value is shown for reference. It is interesting to note that in each case, the estimation performance trend is similar as the sensing matrix evolves from that of an optimized sensing matrix to that of a randomly generated sensing matrix.



(a) Objective function difference operator, Gaussian-Distributed Entries



(b) Kolmogorov-Smirnov Statistic difference operator, Gaussian-Distributed Entries



(c) Kullback-Leibler Divergence difference operator, Gaussian-Distributed Entries

Figure 25: Support estimation versus difference operators for Gaussian dictionary with dictionary evolution.

7.3 CRN Spectrum Database Dictionary (TC-04 through TC-06)

7.3.1 Spectrum Approximation and Verification Simulation Overview

For the purposes of the spectrum database simulation, two SSDF states are defined: *SSDF Present* and *SSDF Absent*. For SSDF Present, three different types of SSDF are simulated. Both SSDF states are tested concurrently, for each of the three types of SSDF:

- SSDF Absent: No Attack
- SSDF Present: Inversion/Spoofing/Shifting

The prototype experiments described in this section are aimed at evaluating the greedy spectrum approximation and verification (GSA, GSV respectively) methods detailed in Sections 6.1 and 6.2 with regards to:

- The ability of the approximation methods to match the spectrum reports from the CSU with the spectrum reports from SNs that are close to the CSU
- The effectiveness of using the outputs from the approximation methods to detect SSDF present versus SSDF absent using three *detection heuristics* within the simulation:
 - δ_1 Number of Matched Sensors
 - δ_2 Reconstruction Error (model residual)
 - δ_3 Correlation Sign

Using the results, coarse empirical receiver operator characteristic (ROC) curves are computed for the reconstruction error metric, δ_2 . The simulation of the RF network includes the following:

- Geographic Location of Each Transmitter: The set of N_E transmitters are scattered randomly throughout the CA
- Time Shift of Each Transmitted Signal: Within the observation window, the start time of each transmitted signal varies randomly according to a uniform distribution
- Center Frequency of Each Transmitted Signal: The center frequency of each emitted signal varies according to a uniform random distribution throughout the frequency search space

The input simulation parameters are listed in Table 7. Each CR emitter is simulated as an approximately bandlimited signal (a sinc pulse) of bandwidth 5 MHz, at a random center frequency. In addition, a random time shift within the sensing interval is applied. AWGN is added to the signal

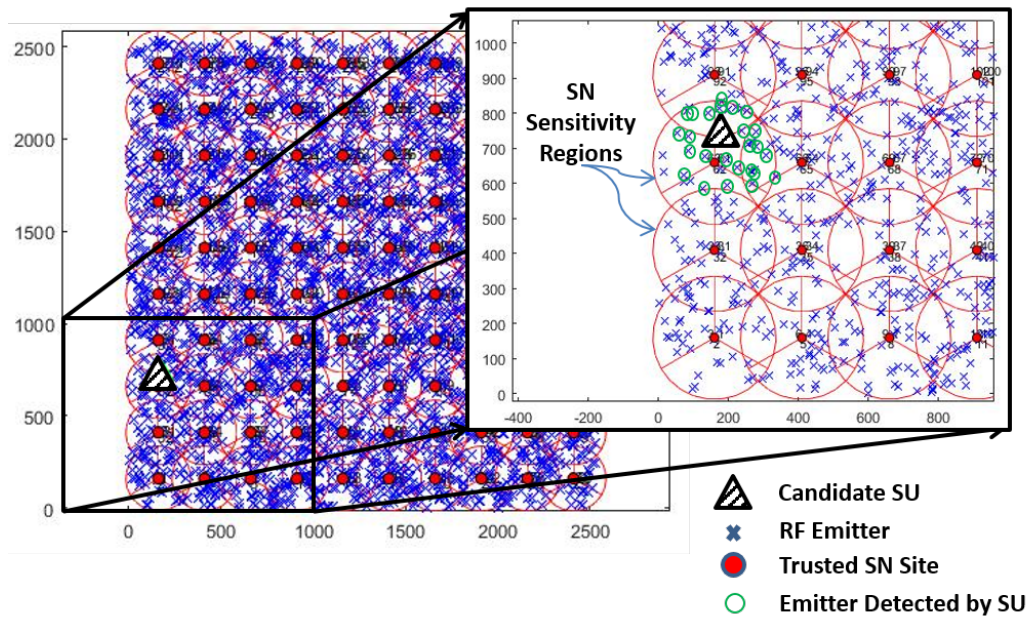


Figure 26: Grid of sensors and transmitters used in the simulation.

prior to spectrum estimation to bring the SNR down to 8dB. For the candidate SU and each SN, the spectrum is channelized into 200 spectral subbands each of which is 2.5 MHz wide. A thresholding decision is made as to whether there exists spectrum in each channel or not. If the spectral power is less than $p_{thresh} = -70dBm$ then the energy for that channel is set to zero below a suitable threshold. The power threshold is adjusted relative to the emitter transmitter power such that due to free-space path loss, emitters beyond approximately 150m from either the SNs or the candidate SU are removed from the sensing reports.

Table 6: Simulation Parameters

Parameter Name	Value	Symbol
Coverage Area	2.5kmX2.5km	CA
Emitters/CA	3500 emitters	N_E
Search Bandwidth	500 MHz	
Number of Spectral Subbands	200	N
Bandwidth/Subband	2.5 MHz	B_{sub}
Emitter Bandwidth	5 MHz	B_E
Received Power Threshold	-70 dBm	p_{thresh}
Number of Sensors/CA	300	M
Number of Compressive Measurements	70	m
Number of Monte Carlo Trials	1000	N_{iter}
Input SNR	8 dB	
Sparsity Estimate	5	k

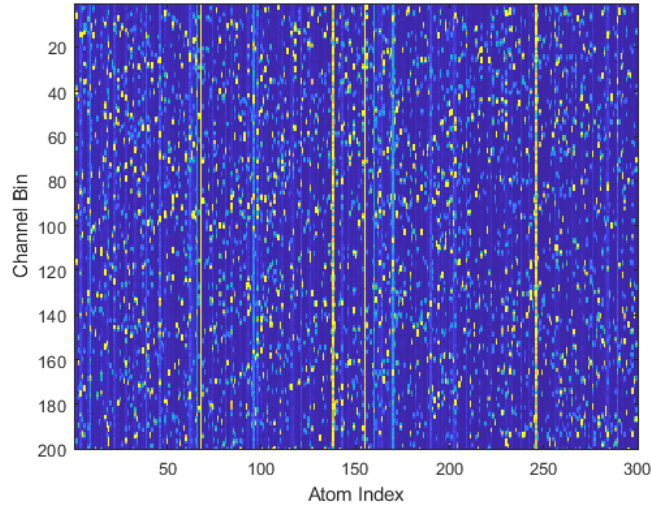


Figure 27: Image of the elements comprising a synthetic spectrum database (GDB) dictionary matrix, where there are more columns (300) than rows (200).

For simplicity during the simulations, the 300 SNs are uniformly distributed on a grid over the coverage area (Fig.26). Prior to running the simulation, the GDB is constructed containing spectrum reports from each of the SNs. Fig.27 depicts an image of the elements of a sample GDB computed in MATLAB.

For each iteration of the Monte Carlo simulation, a random location is selected within the coverage area representing the location of the candidate SU location. Next, it is determined which SU emitters lie within 150 m of the candidate SU location. To simulate the possibility that the GDB is stale and also to account for possible obstructions in the signal path between the candidate SU and the emitters, 4-5 emitters out of this set are randomly omitted from the candidate SU sensor report. Next, the *true* SN IDs are identified as the six closest SNs to the candidate SU. The set of estimated SN IDs is then found by the four matching methods discussed (correlation, SA-OMP, CS-RAND, and CS-OPT). The set of estimated SN IDs is compared to the true SN ID set and the number of matches is found. Once the number of matches is found, the other two detection heuristics are then computed.

The algorithm was run for each SSDF case (No SSDF, spectrum inversion, spectrum spoofing, and shifted spectrum SSDF). Whenever OMP is used, the desired sparsity level is set to $k = 5$. For the sensing matrix optimization, $U \in R^{m \times m}$ is a pseudo-randomly generated orthonormal matrix (generated using the command *orth(rand(m))* in MATLAB), and $V_{11}^T \in R^{N \times N}$ is set to be the identity matrix of appropriate size. A list of high-level algorithm descriptions for the CRN spectrum analysis simulation that generated the results in this dissertation is included in Section 9.7.

7.3.2 TC-04: Spectrum Dictionary with Sensing Matrix Evolution (Inversion)

7.3.2.1 Support Estimation per Sensing Matrix

In Fig. 28 the performance trend is generally that SA-OMP performs best with respect to matching, followed by CS-OPT, CORR, and lastly CS-RAND matches the least number of SN IDs on average. In regards to the cases when SSDF is present, a different trend is present: CORR performs best in the sense that it matches the fewest SN IDs under the inverted spectrum SSDF attack, followed by SA-OMP, CS-RAND, and lastly CS-OPT.

7.3.2.2 SSDF Inversion Detection Heuristics

Fig. 28 and Fig. 29 show the issue that the CS-based methods of CS-RAND and CS-OPT have regarding detecting spectral inversion. This can be explained by observing that SA-OMP, CS-RAND and CS-OPT all use OMP which as is seen in Section 3.5.2, relies solely on the magnitude or the square of the inner product to perform the correlation, i.e., the sign of the correlation is disregarded. However, the reconstruction metric (δ_2) as well as the correlation sign metric (δ_3) both appear to effectively detect the presence of the spectral inversion SSDF (99.3% and 96.4%, respectively) with relatively low false alarm rates of 2.9% and 0.4%, respectively.

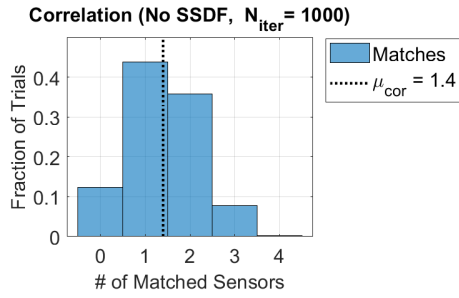
7.3.3 TC-05: Spectrum Dictionary with Sensing Matrix Evolution (Spoofing)

7.3.3.1 Support Estimation per Sensing Matrix

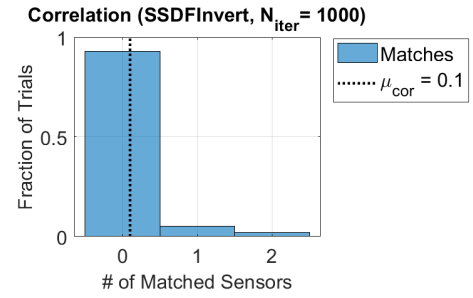
Fig. 30 shows the matching results for the SSDF spectral spoofing case. Correlation (a) is shown to perform the worst, and SA-OMP in (c) appears to do the best at matching. The presence of the random compression matrix in (e) degrades the matching performance as expected, and the optimized CS matrix in (g) greatly increases the matching ability. In all SSDF spoofing cases (b,d,f, and h), there are very few matches.

7.3.3.2 SSDF Spoofing Detection Heuristics

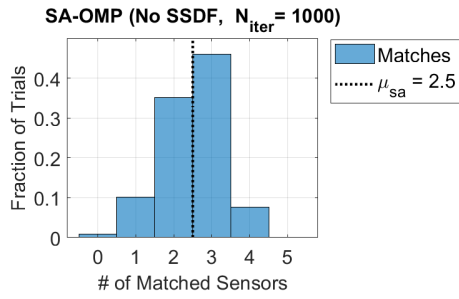
Fig. 31 (a) shows that the δ_1 heuristic (computed using CS-OPT) detected the spectral spoofing SSDF in approximately 91.8% of the trials with a false alarm rate of about 3.5%. The empirical ROC in Subfigure (b) reflects the matching performance results, and also shows the CS-OPTNOI case, whose performance lies between that of CS-OPT and CS-RAND. The CS-OPTNOI case reflects



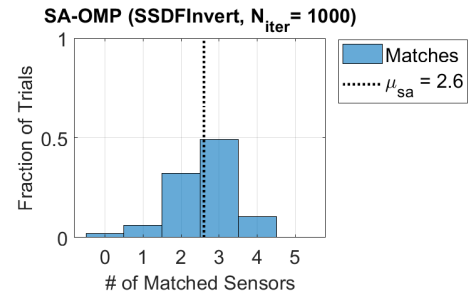
(a) No SSDF with correlation



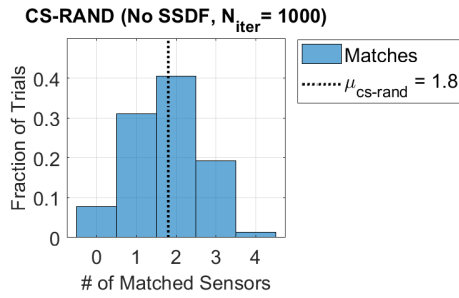
(b) Inversion with correlation



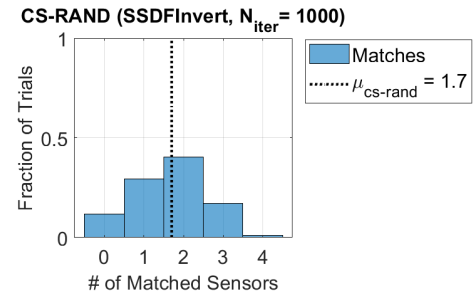
(c) No SSDF with SA



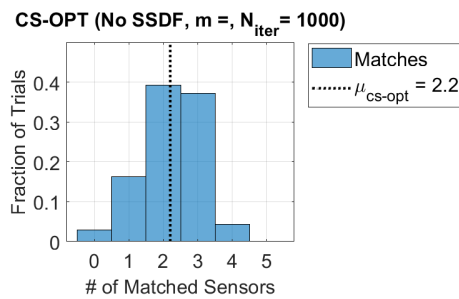
(d) Inversion with SA



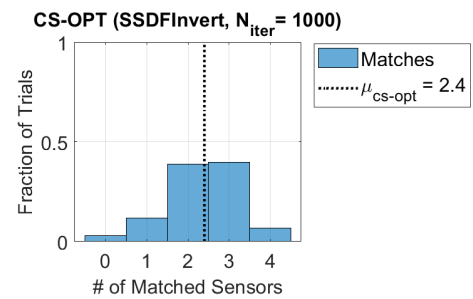
(e) No SSDF with CSRAND



(f) Inversion with CSRAND

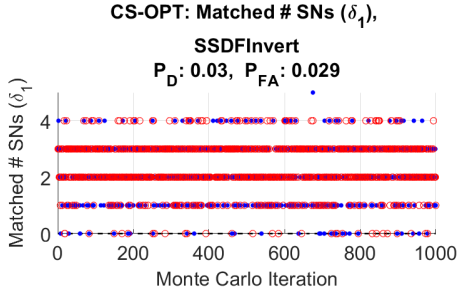


(g) No SSDF with CSOPT

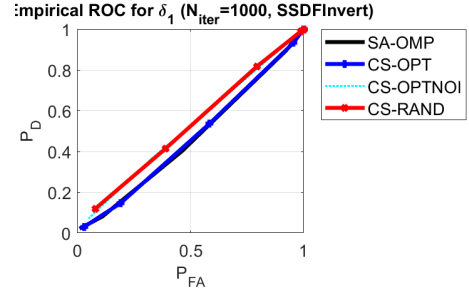


(h) Inversion with CSOPT

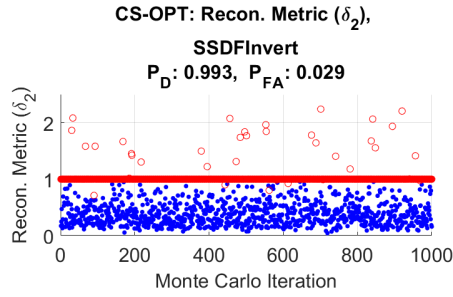
Figure 28: SSDF detection results. The case with no SSDF results is compared to the case with spectral inversion SSDF.



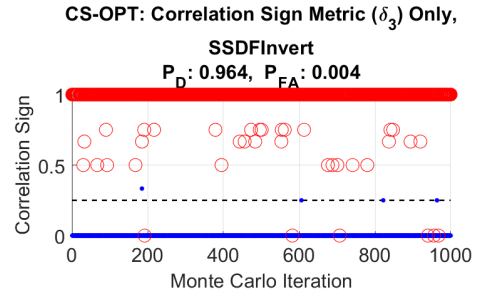
(a) δ_1 detection metric



(b) δ_1 ROC



(c) δ_2 detection metric



(d) δ_3 only detection metric

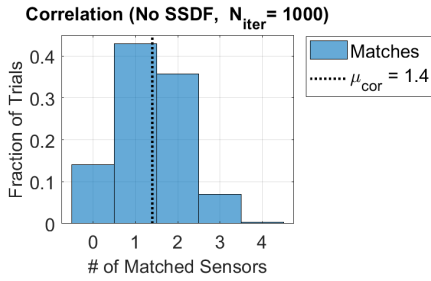
Figure 29: Simulation results. Detection heuristics for the spectral inversion SSDF. Note that while Subfigure (b) shows the four main approximation configurations, Subfigures (a,c, and d) reflect only the CS-OPT configuration. The CS-OPTNOI case in Subfigure (b) reflects the most perturbed version of the effective dictionary.

the Aost Aerturbed version Af Ahe Affective Aictionary. The δ_2 SSDF Aetection Aeuristic Ahowed a probability Af Aetection Af about A3.5% with a 3.5% Aalse alarm Aate. As Aight be Aexpected, looking at Ahe Aonly Ahe Aorrelation Aign in Subfigure Ad) Aroved Ao Aot be an Affective detection Aeuristic.

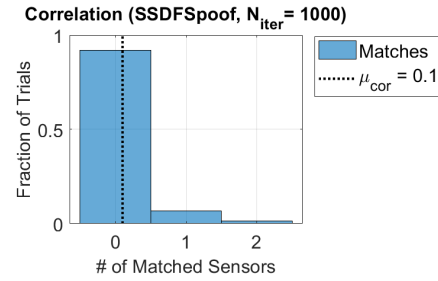
7.3.4 TC-06: Spectrum Dictionary with Sensing Matrix Evolution (Shifting)

7.3.4.1 Support Estimation per Sensing Matrix

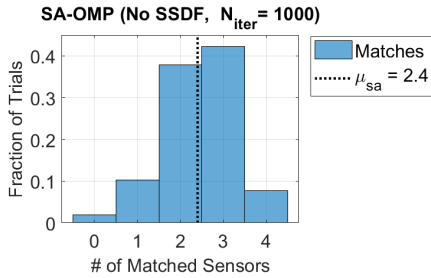
Fig. 32 shows the matching results for the spectral shifting SSDF. The performance is generally very similar, with the exception of CS-OPT, which tends to match more SN IDs than the other methods under this SSDF. The reason for CS-OPT matching more spectrally-shifted SN IDs than the other methods might be because the CS-OPT sensing matrix is based on the GDB, as opposed to the other matching methods. With many densely located SUs, it is likely that a shifted spectrum report would be correlated with spectrum sensed by other SNs. Lastly, Fig. 30 depicts the spectral spoofing SSDF scenario and shows that all four methods exhibit comparable performance with regards to discriminating spectrally spoofed spectrum reports from those spectrum reports in the GDB.



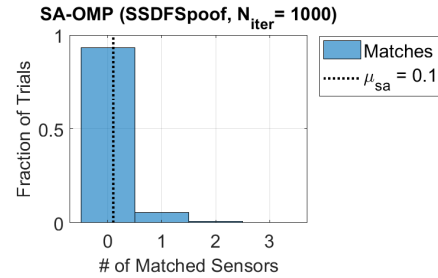
(a) No SSDF with correlation



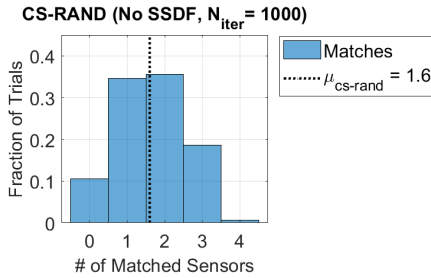
(b) Spoofing with correlation



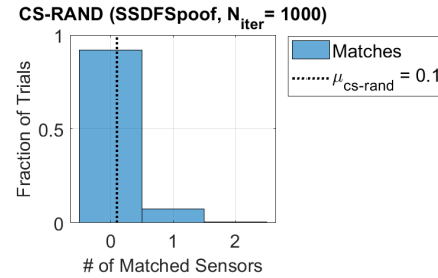
(c) No SSDF with SA



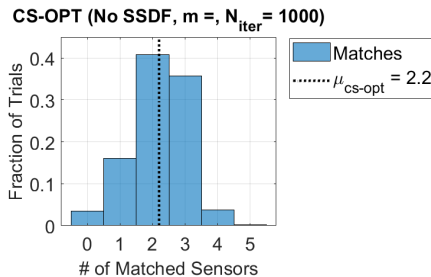
(d) Spoofing with SA



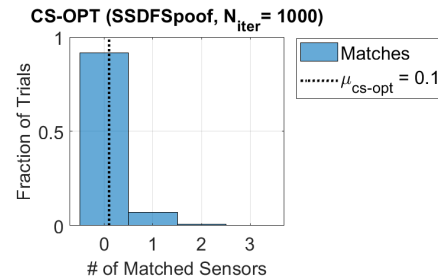
(e) No SSDF with CSRAND



(f) Spoofing with CSRAND



(g) No SSDF with CSOPT



(h) Spoofing with CSOPT

Figure 30: Simulation results. The case with no SSDF results is compared to the case with spectral spoofing SSDF.

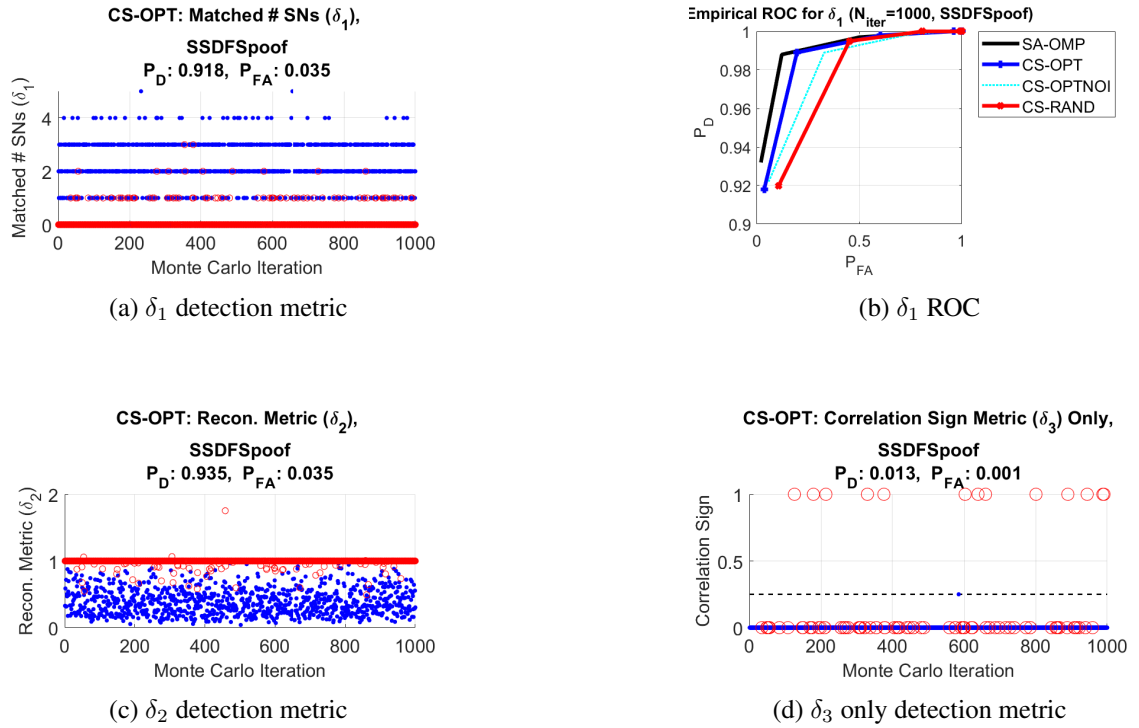


Figure 31: Simulation results. Detection heuristics for the spectral spoofing SSDF. Note that while Subfigure (b) shows the four main approximation configurations, Subfigures (a,c, and d) reflect only the CS-OPT configuration. The CS-OPTNOI case in Subfigure (b) reflects the most perturbed version of the effective dictionary.

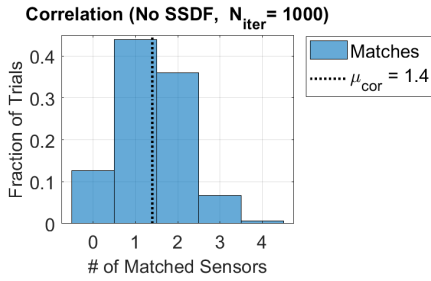
7.3.4.2 SSDF Shifting Detection Heuristics

Fig. 33 (a) shows that the δ_1 heuristic (computed using CS-OPT) detected the spectral shifting SSDF in approximately 92.7% of the trials with a false alarm rate of about 3.3%. The empirical ROC in Subfigure (b) reflects the matching performance results, and also shows the CS-OPTNOI case, whose performance lies between that of CS-OPT and CS-RAND. The CS-OPTNOI case reflects the most perturbed version of the effective dictionary. The δ_2 SSDF detection heuristic showed a probability of detection of about 94.2% with a 3.3% false alarm rate. As might be expected, and consonant with the spectral spoofing SSDF case, looking at only the correlation sign in Subfigure (d) proved to not be an effective detection heuristic.

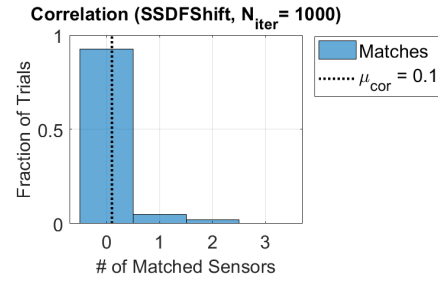
Fig.35 also shows the empirical CDFs for each of the four CS-OPTNOI perturbations, as well as the associated KS distances computed from the CS-OPT case. As is expected, the KS distance increases with increasing effective dictionary perturbation.

7.3.4.3 Support Estimation vs Difference Operator

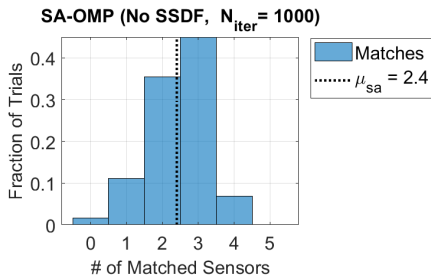
Fig. 34 shows the three difference operators computed for the sensing matrix objective function,



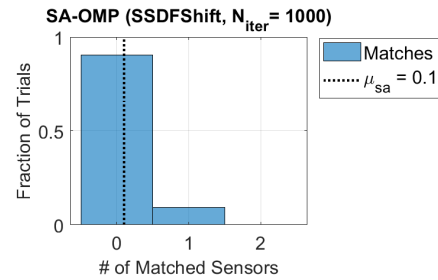
(a) No SSDF with correlation



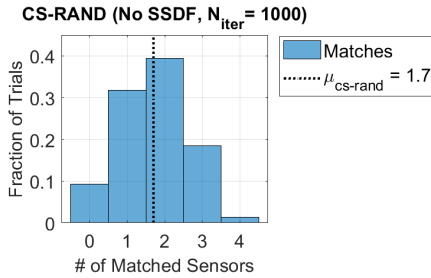
(b) Shifting with correlation



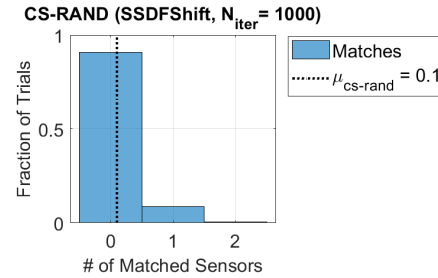
(c) No SSDF with SA



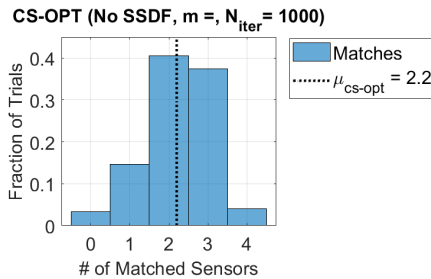
(d) Shifting with SA



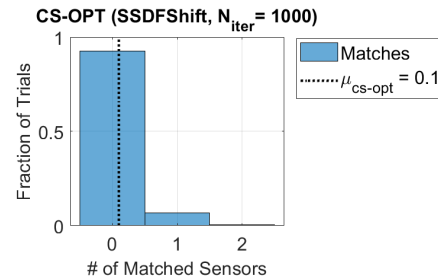
(e) No SSDF with CSRAND



(f) Shifting with CSRAND

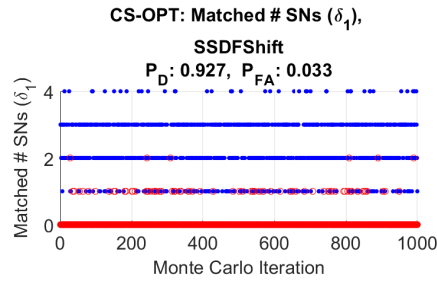


(g) No SSDF with CSOPT

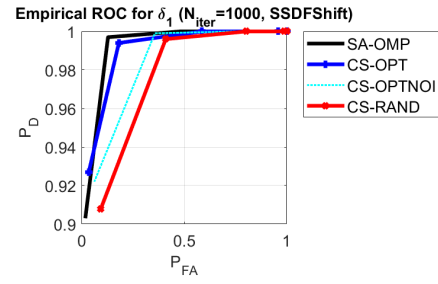


(h) Shifting with CSOPT

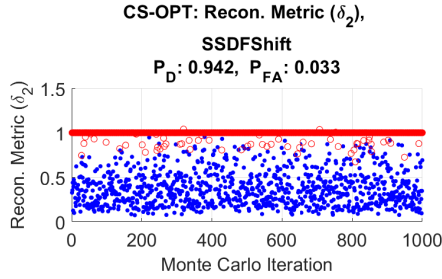
Figure 32: Simulation results. The case with no SSDF results is compared to the case with spectral shifting SSDF.



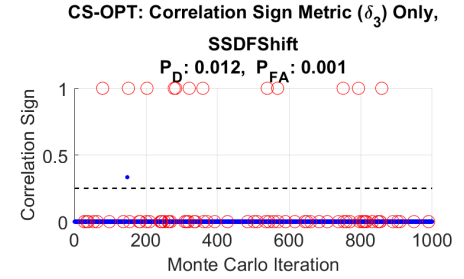
(a) δ_1 detection metric



(b) δ_1 ROC



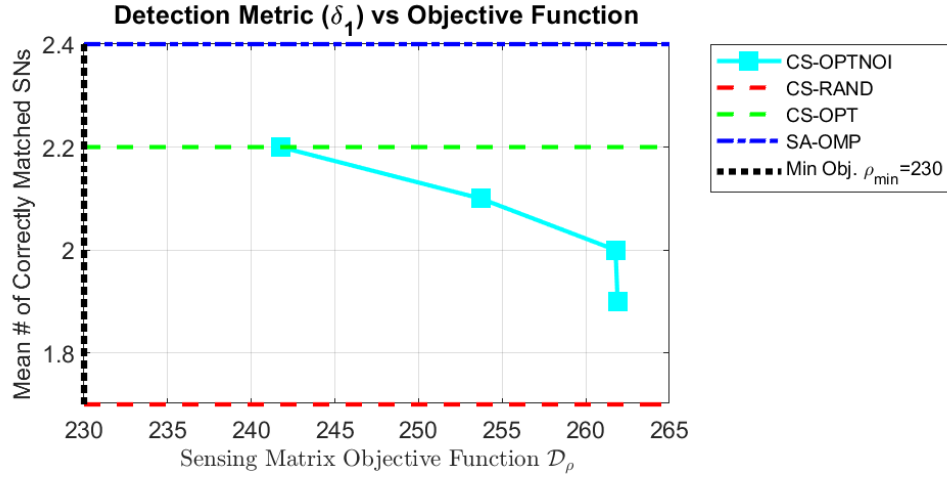
(c) δ_2 detection metric



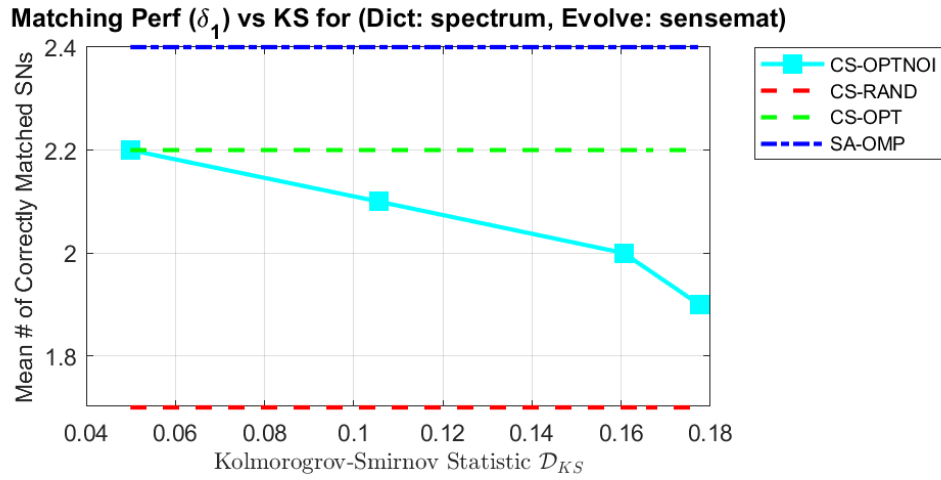
(d) δ_3 only detection metric

Figure 33: Simulation results. Detection heuristics for the spectral shifting SSDF. Note that while Subfigure (b) shows the four main approximation configurations, Subfigures (a,c, and d) reflect only the CS-OPT configuration. The CS-OPTNOI case in Subfigure (b) reflects the most perturbed version of the effective dictionary.

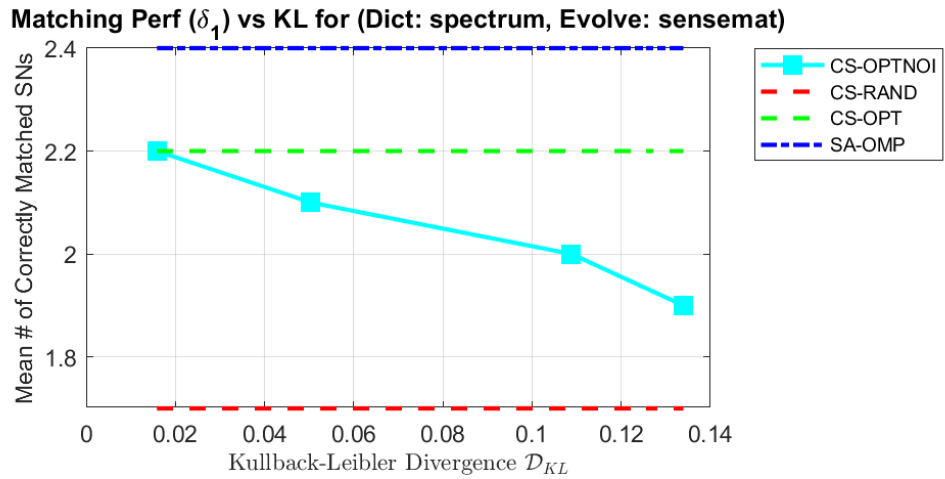
KS statistic, and the KL divergence in Subfigures (a), (b), and (c) respectively. Each operator shows the same trend as each other, namely the increase in distance from the CS-OPT case towards the CS-RAND case for each subsequently increasingly perturbed effective dictionary. The trend is also consistent with the difference operator results in Section 22 and Section 25 for the Gaussian dictionary. Fig. 35 shows the empirical distribution functions of the Gram matrix entries for each of the four CS-OPTNOI cases compared to that of the CS-OPT case. In addition, the KS-statistic is the vertical distance between each CS-OPTNOI curve and the CS-OPT curve, and shows how it increases with increasing effective dictionary perturbation.



(a) Objective Function

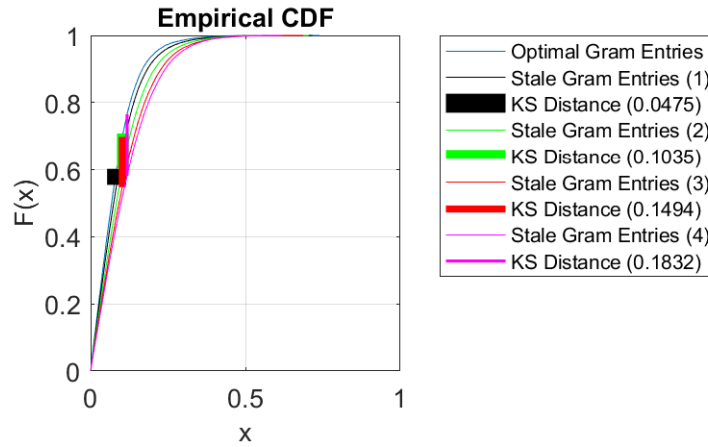


(b) KS Statistic

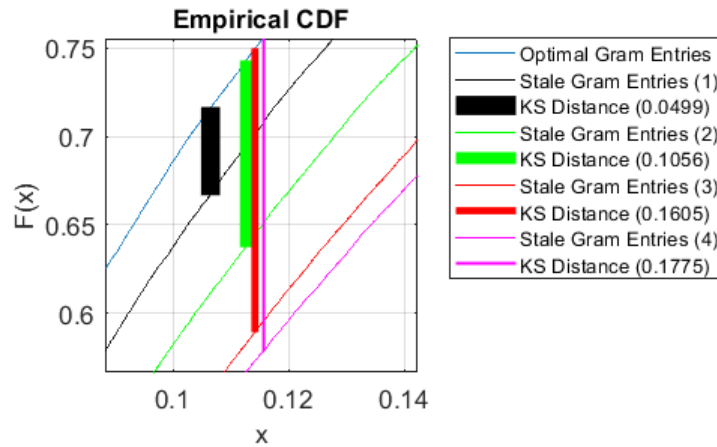


(c) KL Divergence

Figure 34: Support estimation versus difference operator for spectrum dictionary with sensing matrix evolution



(a) EDF of Gram Matrix Entries with KS Distances



(b) Expanded Portion of Subfigure (a)

Figure 35: Empirical CDF of Gram matrix entries formed from the CS-OPT and CS-OPTNOI sensing matrix cases. The lengths of the vertical bars represent the distances between the CS-OPT case and each of the four perturbed cases. Subfigure (b) is an expanded version of Subfigure (a).

7.3.5 Discussion on Spectrum Dictionary Results

Comparing the simulation results depicting matched SN IDs in Fig.30a with Fig.30c, the improvement of orthogonal decomposition via OMP is clearly evident over inner product correlation (when compared to SA-OMP, CS-OPT, and to a lesser extent CS-RAND). Inner product correlation does a good job of identifying the most salient components of the received signal, but identifies significantly fewer of the less prominent signal components. In contrast, OMP identifies more of the higher-order components than correlation, and the failure of OMP to match any sensor IDs is about half that of correlation. This appears to be a direct result of the orthogonal subtraction step of OMP, where already identified signal components are removed from the signal, so as to more strongly correlate with remaining signal components (within the residual).

In every test case with SSDF absent, in comparing SA-OMP with CS-RAND it is seen that the average number of correctly matched SN IDs for CS-RAND is less than that of SA-OMP. This is presumably as a result of the compression from 200 samples down to 70, and also as result of the random sub-optimal sensing matrix used in CS-RAND. Also every test case with SSDF absent, the results show that the average number of correctly matched SN IDs for CS-RAND is less than that of CS-OPT. It is hypothesized that this is due to the use of an optimized sensing matrix based on the dictionary, as both CS-RAND and CS-OPT share the same compression ratio.

As mentioned earlier, in non-orthogonal dictionaries, such as the GDB, a randomly selected sensing matrix is in general not optimal. Looking at the CS-OPT simulation results, the average number of matched SN IDs is higher when compared to CS-RAND, but CS-OPT has less matches than that of SA-OMP. It appears that the compression in CS-RAND results in a decrease in sensor ID matches as compared with SA-OMP. With the optimized sensing matrix, performance is driven higher as compared to CS-RAND, as the optimal sensing matrix helps to compensate for the information loss resulting from compression. SA-OMP generally results in the highest number of overall SN ID matches, as well as more higher-order matches (i.e., matching 3, 4, and 5 SNs). It is proposed that this is a result of the effectiveness of the OMP process, without any lossy compression.

7.4 Compression and Computational Complexity (TC-07)

One of the reasons for performing compression and in particular compressive sensing is to reduce the computational complexity of signal processing tasks. In this dissertation specifically, compression can be used to reduce the computational complexity associated with the cost of the orthogonal matching pursuit (OMP) algorithm. As shown, sensing matrix optimization results in increased matching performance (support identification of the atoms used to estimate a given vector), but comes at the computational cost of performing a singular value decomposition. The computational complexity of the naive implementation of OMP using a dictionary matrix of size $M \times N$ to recover a k - *sparse* vector is estimated to be [36]

$$C_{CS-OPT} = C_{CS-RAND} + C_{SVD} = Nm + mk + mk^2 + k^3. \quad (7.2)$$

If a random sensing matrix of size $m \times N$, $m < M$ is to compress the dictionary and used to recover a k - sparse vector, the complexity reduces to $C_{CS-RAND} = Nm + mk + mk^2 + k^3$. If an optimized sensing matrix is used, then the computational cost of an SVD (M^2 , [68]) must be added in

$$C_{CS-OPT} = C_{CS-RAND} + C_{SVD} = Nm + mk + mk^2 + k^3 + M^2 \quad (7.3)$$

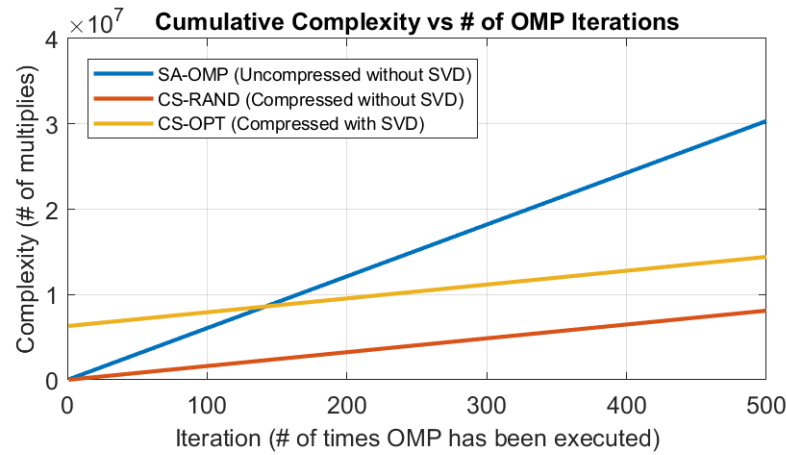


Figure 36: Computational complexity for sparse approximation, compressive sensing with a random matrix, and compressive sensing with an optimized matrix.

Looking at Fig. 36, if the sensing matrix does not need to be updated for a certain number of OMP iterations (approximately 150 iterations), then the cost of performing the SVD (on the dictionary) for CS-OPT becomes dominated by the computational cost of repeatedly performing OMP without compression (SA-OMP) in the full number of dimensions (N) instead of the compressed dimension m . CS-RAND is seen to be the lowest computational complexity (since it requires no SVD and has compression), but as shown in earlier sections, performs worse than SA-OMP and CS-OPT in support estimation. The graph in Fig. 36 was used the following values: $m = 70$, $N = 200$, $M = 300$.

8 Conclusion and Further Research

8.1 Sensing Matrix Evaluation Considerations

On the basis of the simulations in Sections 7.2 and 7.4, the following conclusions are drawn:

- SMO appears to be an effective tool to generate sensing matrices, which separate the atoms of a dictionary better than randomly constructed sensing matrices.
- For *pseudo-randomly generated dictionaries*, the difference operators involving the SMO objective function and statistical distances appears adequate to detect a change in the geometry of effective dictionaries formed from Gram matrices of optimized versus sub-optimal sensing matrices.
- Maximally perturbed effective dictionaries used in the most perturbed cases of CS-OPTNOI (those effective dictionaries whose sensing matrices were optimized on an entirely different, but identically distributed dictionary) were shown to perform worse than the optimized case CS-OPT, but *always better* than the random case CS-RAND.
- Operating on a compressed vector reconstruction complexity, and reduce aggregate network traffic load while simultaneously providing some minor obfuscation of the dictionary.

Some additional possible issues to address regarding SMO include:

- Designing an extremely poor sensing matrix
- Formulating an equiangular tight frame (ETF) using sensing matrices is an active area of research [12, 64, 64, 65]. Many of the methods found use single parameter metrics of the effective dictionary to ascertain conformity to such frames. It would be interesting to apply some of the difference operators developed in this dissertation to optimizing a sensing matrix in order to form an ETF.
- Experimenting with various types and combinations of quantization or conditioning of the optimized sensing matrix (or the dictionary prior to SMO), e.g.
 - Performing QR decomposition, then perform quantization
 - Performing binary quantization, then perform QR decomposition

- Incrementing the severity of quantization and study the effects on support recovery
- Finding the sensing matrix that is closest to the optimized matrix, but whose entries consist only of -1,0,+1 (see Section 3.3.2 and [38,41])

8.2 Additional Greedy Spectrum Approximation and Verification Considerations

SMO was also applied to the case of spectrum dictionaries (GDBs), and showed performance trends analogous to that of the Gaussian dictionaries. Regarding the simulations of Section 7.3 involving spectrum dictionaries (GDBs), the following conclusions are made:

- Practical spectrum approximation using GSA appears to be feasible
- The three SSDF attacks appear to be detectable at a relatively low false alarm rate
- Changes in the GDB are reflected in the three difference operators, and appear suitable to use as a heuristic to update the sensing matrix
- Matching and difference operator performance were shown to be consistent with that of the pseudorandom Gaussian dictionary case
- Using the sign of correlation for matched SNs appears to be effective to detect spectrum inversion SSDF
- SA-OMP and CS-OPT appear to mitigate the effect of dictionary mis-match and noise in the GDB better than CS-RAND

Areas of possible future research regarding GSA and GSV include:

- Additional work on other types of spectrum attacks beyond inverted, spoofing, and shifting is necessary to counteract such threats.
- Addressing scalability: How does GSA and GSV perform as the GDB grows larger?
- The analysis presented studied how effectively the GDB can be compressed when SUs are randomly scattered throughout the CA with random frequency assignments. Continued research on GSA and GSV involving empirically-based and more accurate frequency-usage models would be beneficial.
- Identifying more specific use-cases for spectrum dictionaries would be helpful, and may also help determine how and when spectrum dictionaries can be expected to change.
- Studying the effects of sensor and SU density on the performance of the proposed method.

In addition, GSA utilizes the fact that each trusted SN constitutes an overlapping portion of the candidate secondary users (CSU) honest spectrum report. However if it is known that sensors are very close to each other (i.e. an ultra-dense sensor network), then the spectrum reported by each of the SNs tends to converge. In that case multi-sensor compressive sensing (CS) algorithms can be applied, e.g. simultaneous orthogonal matching pursuit, to account for the jointly-sparsity of the multiply reported SN reports. In the CS literature, this situation is known as the multiple measurement vector (MMV) CS scenario [23]. Further research could aim to identify the different types of MMV recovery algorithms in such a an ultra-dense network.

9 Appendices

9.1 Symbols and Operators

Table 7: Table of Symbols

Symbol	Meaning
$ s $	Absolute value of the scalar or vector s , or cardinality of the set s
0_M	Matrix of Zeros, size $M \times M$
A	Effective Dictionary (general form)
A^T	Transpose of Matrix A
A^{-1}	Inverse of Matrix A
A^+	Moore-Penrose Pseudo-Inverse of Matrix A
$\tilde{A}$	Column-Normalized version of Matrix A
$\mathcal{D}$	General Distance Operator
$\mathcal{D}_{\mathcal{KL}\mathcal{D}}$	Kullback-Leibler Distance Operator
$\mathcal{D}_{\mathcal{KS}}$	Kolmogorov-Smirnov Distance Operator
$\mathcal{D}_\rho$	Sensing Matrix Objective Function Distance Operator
δ_i	Spectrum Falsification Detection Heuristic for $i \in \{1, 2, 3\}$
δ_1	Spectrum Falsification Detection Heuristic Corresponding to Number of Matched Sensor IDs
δ_2	Spectrum Falsification Detection Heuristic Corresponding to Reconstruction Error
δ_3	Spectrum Falsification Detection Heuristic Corresponding to Correlation Sign
$\ A\ _F$	Frobenius Norm of Matrix A
$G(A)$	Gram Matrix of the Matrix A
$G_h(A)$	Hollow Gram Matrix of Effective Dictionary A
$\mathbf{I}(\mathbf{f})$	Indicator Function of f
I_M	Identity Matrix, size $M \times M$

k	Sparsity of a Signal Vector
M	Number of Columns in Dictionary Matrix or Total Number of Sensor Nodes
m	Number of Rows in Sensing or Number of Compressed Measurements
$\mu(A)$	Coherence of the Matrix A
N	Number of Rows in Dictionary Matrix or Total Number of Frequency Bins
Φ	Sensing Matrix (general form)
Φ_t	Sensing Matrix Optimized for a Dictionary that was updated at time t
Ψ	Dictionary
ψ_i	i^{th} Dictionary Column (Atom), Spectrum Report from i^{th} SN
Ψ_t	Dictionary Updated at time t
Σ	Diagonal Matrix of Singular Values
S_{CR}	Set of Index Elements Correlated with Reported Cognitive Radio
S_{MATCH}	Set of Index Elements in Intersection of S_{CR} and S_{SN}
$\bar{S}_{MATCH}$	Mean Number of Matched Sensor Node Indexes
S_{SN}	Set of Index Elements Derived from Sensor Nodes
s	$k - sparse$ Vector
U	Arbitrary Orthonormal Matrix
V	Arbitrary Orthonormal Matrix
V_{11}	Upper Left Sub-Matrix of an Arbitrary Orthonormal Matrix (used in CS matrix optimization)
x	Input Signal Vector (to be approximated)
x_{CR}	Spectrum Report from Candidate Cognitive Radio
y	Measurement Vector

9.2 Compressive Sensing Matrix Measures

One commonly cited set of sensing matrix properties is [16, 22] the

- Spark,
- Null Space Property,
- Restricted Isometry Property, and
- Coherence.

While these four properties have sprouted many off-shoots in the research literature, studying these four properties gives useful insight into the geometry of the CS problem. Below, each of these properties is defined and briefly reviewed in the context of their relevance to CS. While these properties are common in the literature, there also exist many additional properties that are derived from these main properties, and adapted for particular applications, e.g. dictionary RIP, block-RIP, block-coherence, etc. As can be seen below, much of the material presented in this section can be found in [16, 22].

9.2.1 Sparse Rank (SPARK)

The sparse rank (a.k.a. *spark*) of a matrix A is defined as the smallest number of columns of A that are *linearly dependent*. For an $m \times M$ matrix A , the spark is bounded by $\text{spark}(A) \in \{2, 3, \dots, M + 1\}$ [22].

There exists an important relationship between the unique recoverability of sparse vectors and the effective dictionary A . For any vector $y \in \mathbb{R}^M$, there exists at most one sparse vector $x \in \Sigma_k$ such that $y = Ax$ if and only if $\text{spark}(A) > 2k$ [22].

9.2.2 Null Space Property

The null space of the matrix A is tightly coupled with the ability of a linear system to recover sparse signals. The *null space property* (NSP) of a matrix A captures the degree to which the null space of A contains sparse vectors, i.e., NSP(A) measures the amount of overlap between the vector space that contains k -sparse vectors and the null space of A . In sparse signal processing methods (e.g., CS) the goal is to operate on or approximate sparse vectors. As a result, the more sparse vectors that reside in the null space of the sensing matrix (or effective dictionary), the less likely it is that a sparse signal can be recovered or approximated.

A matrix $A \in \mathbb{R}^{m \times M}$ satisfies the null space property of order k if there exists a constant $C > 0$ such that

$$\|h_\Lambda\|_2 \leq C \frac{\|h_{\Lambda^c}\|_1}{\sqrt{k}} \quad (9.1)$$

$\forall h, \Lambda$, where $h \in \text{Nullspace}(A)$, $\Lambda \leq k$, and $\text{card}(\Lambda) \subset \{1, 2, \dots, N\}$ [22].

9.2.3 Cumulative Coherence

The ℓ_1 coherence function (μ_1) of the matrix A [30] is defined as

$$\mu_1(s) \equiv \max_{i \in [1, N]} \max \left\{ \sum_{i \in S} |\langle a_i, a_j \rangle|, S \subset [1, N], \text{card}(S) = s, i \notin S \right\}, \quad (9.2)$$

The ℓ_1 -coherence function also goes by the names *cumulative coherence function*, or additionally the *Babel function*.

Both the coherence and the ℓ_1 coherence function are invariant under left-multiplication by an orthogonal matrix, U . The columns of UA are the ℓ_2 normalized vectors $Ua_1, Ua_2, \dots, Ua_N$ and the inner products of the columns remain unchanged by U since $\langle Ua_i, Ua_j \rangle = \langle a_i, a_j \rangle$. Also, the unity upper bound on the coherence can be shown to be a consequence of the Cauch-Schwarz inequality $|\langle a_i, a_j \rangle| \leq \|a_i\|_2 \|a_j\|_2$. Both the coherence and ℓ_1 -coherence are related to the conditioning of A . Let A_S be a submatrix of A formed from a subset of its columns, and indexed by the subset $S \subset [1, N]$. Both the coherence $\mu(A)$ and the ℓ_1 -coherence function $\mu_1(A)$, are measures of the condition number of submatrix A_S [22].

9.2.4 t-Averaged Coherence

In [45], in order to compute a sensing matrix that results in an effective dictionary with low coherence, the authors propose starting with a randomly generated sensing matrix. Then utilizing an algorithm to iteratively reduce the rank of a target Gram matrix, they solve an equation for the sensing matrix that minimizes the difference between the effective dictionary of the target Gram matrix and the current Gram matrix. They propose a quantity they name *t-averaged coherence* in order to more accurately reflect the performance of a CS system, as compared to using the worst-case coherence. Reference [45] notes that the coherence is a pessimistic predictor of CS performance. Interestingly, the algorithm proposed in [45] does not directly minimize the t-averaged coherence. Instead, it applies the algorithm, and then through simulation shows that the t-averaged coherence is lowered at each iteration. For a Gram matrix G , the t-averaged coherence is defined as:

$$\mu_t(A) = \frac{\sum_{1 \leq i, j \leq M, i \neq j} |g_{ij}| \cdot \mathbf{I}(|g_{ij}| \geq t)}{\sum_{1 \leq i, j \leq M, i \neq j} \mathbf{I}(|g_{ij}| \geq t)}, \quad (9.3)$$

where $\mathbf{I}()$ is the *indicator function*, t is the threshold parameter, and g_{ij} is the i^{th}, j^{th} element of the Gram matrix G .

9.2.5 Restricted Isometry Property

While the coherence measure can be applied to CS, it again provides very conservative results and does not account for signal noise. Instead, another measure has been developed to aid in sensing matrix design. In CS with orthonormal dictionaries, in order to ensure incoherence with the dictionary, sensing matrices are frequently chosen at random from a class of certain sub-Gaussian distributions [38]. Matrices constructed by randomly selecting entries from a sub-Gaussian distribution will, with high probability, possess a *concentration of measure* property which in the CS literature is typically embodied in the *restricted isometry property* (RIP) [38, 88]

$$(1 - \delta_k) \|x\|_2^2 \leq \Phi x \leq (1 + \delta_k) \|x\|_2^2. \quad (9.4)$$

The RIP is equivalent to the condition that the eigenvalues of the Gram matrix of the effective dictionary be restricted to the range $[1 - \delta, 1 + \delta]$ [16, 38]. Therefore the intent behind designing sensing matrices which obey the RIP is to minimize the extent to which the sensing matrix perturbs the orthogonality of the dictionary.

One relatively large class of sensing matrices that satisfies the RIP for orthonormal dictionaries are matrices that are constructed by randomly selecting its entries from a sub-Gaussian probability distribution. Two sub-Gaussian probability distributions popular in the literature are the standard normal distribution, and the symmetric Bernoulli distribution [16, 38]. Sensing matrices constructed from the symmetric Bernoulli distribution have a computational advantage in that the matrices themselves require very little storage and matrix operations involving the sensing matrix, e.g. multiplication are reduced [41]. In addition, [41] showed that theoretically, up to 2/3 of the entries in the sensing matrix can be zero, i.e. the sensing matrix is able to be sparsified to a significant degree.

Early CS research strives to generate sensing matrices (both random and deterministic) that perform well in terms of signal reconstruction for ONB dictionaries [89]. However, the traditional

RIP is sub-optimal for CS problems involving overcomplete dictionaries, necessitating the need for constructions such as the dictionary RIP [39]. Additionally, testing if a matrix satisfies the RIP is NP-hard [90].

9.3 Sensing Matrix Optimization Derivations

9.3.1 Derivation of Optimal Effective Dictionary

The derivations in this section are based on [12]. The effective (or projected) dictionary is the compressed dictionary given by $A = \Phi\Psi$. The Gram matrix of the normalized effective dictionary is

$$G(A) = A^T A. \quad (9.5)$$

The general objective function for the problem of finding an optimal effective matrix, in the sense that the the Frobenius norm of the Gram is close to that of a target Gram matrix $G_T \in \mathbb{R}^{M \times M}$ is given by

$$A_{opt} = \underset{A \in \mathbb{R}^{m \times M}}{\operatorname{argmin}} \|G_T - A^T A\|_F^2. \quad (9.6)$$

The focus of this work specifies G_T to be the identity matrix I . Consequently, the objective function to be optimized is given by $G_T = I_M$ making (9.6)

$$A_{opt} = \underset{A \in \mathbb{R}^{m \times M}}{\operatorname{argmin}} \|I_M - A^T A\|_F^2 = \underset{A \in \mathbb{R}^{m \times M}}{\operatorname{argmin}} \rho(A), \quad (9.7)$$

where $\rho(A) \equiv \|I_M - A^T A\|_F^2$ is the objective function residual. The primary theorem concerning the necessary and sufficient form for the optimal effective dictionary (on which the theorem specifying the form of the optimal sensing matrix is based) is now presented.

Theorem 1: Optimal Effective Dictionary Matrix Form

Let $A \in \mathbb{R}^{m \times M}$, with $m < M$. Also, let $\operatorname{rank}(A) = m$. Then, the solution set to (9.7) is given by the class of effective dictionaries of the form

$$A_{opt} = U \begin{bmatrix} I_m & 0 \end{bmatrix} V^T \quad (9.8)$$

where $U \in \mathbb{R}^{m \times m}$, $V^T \in \mathbb{R}^{M \times M}$ are arbitrary orthonormal matrices. The resulting residual of the objective function $\rho \equiv \|I_M - A^T A\|_F^2$ is minimized and is given by $\rho = \rho_{min} = M - m$.

Proof of Theorem 1:

From Lemma 2 of [12], the partial derivative of the objective function residual WRT effective matrix A is given by

$$\frac{\partial \rho}{\partial A} = -4A (I_M - A^T A) \quad (9.9)$$

Optimal solutions to the objective function are stationary points that make the above derivative vanish, so to find optimal solutions of A , set Eq. (9.9) to zero

$$\frac{\partial \rho}{\partial A} = -4A (I_M - A^T A) = 0_{M \times M} \quad (9.10)$$

where $0_{M \times M}$ is an $M \times M$ matrix of all zeros. Dividing each side by the constant (-4), the solutions to this equation are the same as that of

$$A (I_M - A^T A) = 0_{M \times M} \quad (9.11)$$

Distributing the effective matrix A gives

$$A - AA^T A = 0_{M \times M}. \quad (9.12)$$

Equating each term results in

$$A_{opt} = A = AA^T A. \quad (9.13)$$

Now taking the SVD of the effective matrix $A \in \mathbb{R}^{m \times M}$ results in

$$SVD(A) = U \Sigma V^T, \quad (9.14)$$

where $U \in \mathbb{R}^{m \times m}$, $V^T \in \mathbb{R}^{M \times M}$ are orthonormal matrices, and $\Sigma \in \mathbb{R}^{m \times M}$.

Since $rank(A) = m$, $\Sigma = \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix}$ the matrix of singular values has $M - m$ zero vectors and can be written as:

$$SVD(A) = U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T. \quad (9.15)$$

Substituting the $SVD(A)$ from Eq. (9.14) in for A within Eq. (9.13) results in

$$\begin{aligned} & U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T \\ &= \left(U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T \right) \left(U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T \right)^T \left(U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T \right). \end{aligned} \quad (9.16)$$

Carrying out the transpose operation on the middle factor (in parentheses on the right hand side) gives

$$U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T = \left(U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T \right) \left(V \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} U^T \right) \left(U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T \right). \quad (9.17)$$

Since U and V are matrices composed of the singular vectors of the matrix A , both U and V are orthogonal so that $UU^T = U^TU = I$ and $VV^T = V^TV = I$. Applying these simplifying relations once to the RHS reduces it to

$$U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T = \left(U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} \right) \left(\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} \right) \left(\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T \right). \quad (9.18)$$

Pre-multiplying each side by U^T , then post-multiplying each side by V

$$U^T U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T V = \left(U^T U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} \right) \left(\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} \right) \left(\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T V \right), \quad (9.19)$$

and again applying the orthonormality of U and V gives

$$\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} = \left(\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} \right) \left(\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} \right) \left(\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} \right). \quad (9.20)$$

The three factors on the RHS can be combined as

$$\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} = \left(\begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} \right)^3. \quad (9.21)$$

Since each off-diagonal element of Σ_m is zero, and each diagonal element (singular value) repre-

sented by σ_j is real and positive, i.e., $\sigma_j > 0$, it can be shown that

$$\Sigma_m = I_m. \quad (9.22)$$

Recall that for any matrix A of rank m , $SVD(A) = U \begin{bmatrix} \Sigma_m & 0_{M-m} \end{bmatrix} V^T$. Substituting Eq. (9.22) into the $SVD(A)$, the optimal effective dictionary will take the form

$$A_{opt} = U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T, \quad (9.23)$$

where as before $U \in \mathbb{R}^m$ and $V \in \mathbb{R}^M$ are arbitrary orthonormal matrices, representing the left and right singular vectors of the optimal effective matrix.

9.3.2 Derivation of Optimal Gram Matrix

The result in Eq. (9.23) necessarily implies that the Gram of the effective dictionary has the form $G(A) = W_o$, where W_o is defined as:

$$\begin{aligned} W_o &\equiv A_{opt}^T A_{opt} \\ &= \left[U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T \right]^T \left[U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T \right] \\ &= \left[V \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix}^T U^T \right] \left[U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T \right]. \end{aligned} \quad (9.24)$$

Noting that U is an orthonormal matrix such that $U^T U = I$ reduces the above to:

$$\begin{aligned} W_o &= \left[V \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix}^T U^T \right] \left[U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T \right] \\ &= V \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix}^T \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T. \end{aligned} \quad (9.25)$$

Multiplying the inner matrices gives:

$$W_o = V \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} V^T. \quad (9.26)$$

Therefore, in order for the minimum objective function to be attained, the Gram matrix of the effective dictionary $G(A_{opt})$ must be of the form defined by W_o .

9.3.3 Derivation of Minimum Objective Function Error

Next is to find the minimum value of the objective function residual (ρ_{min}) that is achieved when the optimal effective dictionary is utilized. This is done by taking the form that was just derived for the optimal effective dictionary, and substituting it into the original objective function, Eq. (9.7)

$$\begin{aligned}
 \rho_{min} \equiv \rho(A_{opt}) &= \|I_M - A_{opt}^T A_{opt}\|_F^2 \\
 &= \left\| I_M - \left(U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T \right)^T \left(U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T \right) \right\|_F^2 \\
 &= \left\| I_M - \left(V \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix}^T U^T \right) \left(U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T \right) \right\|_F^2 \\
 &= \left\| I_M - \left(V \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix}^T \right) \left(\begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T \right) \right\|_F^2 \\
 &= \left\| I_M - V \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} V^T \right\|_F^2.
 \end{aligned} \tag{9.27}$$

Next, apply an identity involving the Frobenius product

$$\begin{aligned}
 \rho_{min} \equiv \rho(A_{opt}) &= \left\| I_M - V \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} V^T \right\|_F^2 \\
 &= \|I_M\|_F^2 + \left\| V \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} V^T \right\|_F^2 \\
 &\quad - 2 \left\langle I_M, V \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} V^T \right\rangle_F.
 \end{aligned} \tag{9.28}$$

In the last term of the last line, since one of the factors is the identity matrix, the only elements of

concern are along the diagonal. The Frobenius inner product is $2m$ (since $m < M$)

$$\begin{aligned}
\rho_{min} &= \|I_M\|_F^2 + \left\| V \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} V^T \right\|_F^2 \\
&\quad - 2 \underbrace{\left\langle I_M, \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} \right\rangle_F}_{=m} \\
&= \|I_M\|_F^2 + \left\| \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} \right\|_F^2 - 2m.
\end{aligned} \tag{9.29}$$

At this point, within the middle term, the orthogonal matrix V does not change the Frobenius norm, i.e. the Frobenius norm is invariant to orthogonal matrix multiplication (which equates to a rigid rotation), and so multiplication by V and V^T can be removed

$$\begin{aligned}
\rho_{min} &= \|I_M\|_F^2 + \left\| \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} \right\|_F^2 - 2m \\
&= M + m - 2m \\
&= M - m.
\end{aligned} \tag{9.30}$$

The minimum objective function error for sensing matrix optimization is thus the difference in the number of rows of the sensing matrix (i.e., measurements m) and the number of columns of the dictionary (i.e., atoms M).

9.3.4 Derivation of Optimal Sensing Matrix

In addition to restricting $G_T = I_M$, this work is also concerned with finding an optimal sensing matrix Φ WRT a given dictionary Ψ . As a result, the objective function (9.7) can be re-cast using the definition of the effective dictionary in (9.58), such that the objective is narrowed to finding Φ_{opt} instead of A_{opt} . Enforcing this condition, the objective function in (9.23) becomes a search for the optimal sensing matrix WRT to the dictionary matrix, as measured by the distance to the identity matrix where distance is derived from the Frobenius norm

$$\Phi_{opt} = \underset{\Phi \in \mathbb{R}^{m \times N}}{\operatorname{argmin}} \quad \|I_M - \Psi^T \Phi^T \Phi \Psi\|_F^2. \tag{9.31}$$

Theorem 2: Optimal Sensing Matrix Form

Let $A \in \mathbb{R}^{m \times N}$, with $m < N$. Also, let $\text{rank}(A) = m$. Then, the solution set to (9.31) is given by the class of sensing matrices of the form

$$\Phi_{opt} = U \begin{bmatrix} I_m & 0 \end{bmatrix} \begin{bmatrix} V_{11}^T \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T, \quad (9.32)$$

where $U \in \mathbb{R}^{m \times m}$, $V_{11}^T \in \mathbb{R}^{\tilde{N} \times \tilde{N}}$ are arbitrary orthonormal matrices and the singular value decomposition of the dictionary is given by $\Psi = U_d \Lambda_d V_d^T$. The sensing matrix Φ_{opt} is optimal in the sense that, for the given dictionary Ψ , the sensing matrix Φ_{opt} minimizes the Frobenius norm of the off-diagonal entries of the effective Gram matrix.

PROOF:

Define the SVD of the dictionary Ψ to be $SVD(\Psi) = U_d \Sigma_d V_d^T$. Also, define $\Psi^\dagger = V_d \Sigma_d^{-1} U_d^T$ to be the Moore-Penrose pseudoinverse of Ψ . Let the matrix W_o to have the following form

$$W_o \equiv V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T \quad (9.33)$$

Theorem 1 states that the objective function $\|I_L - A^T A\|_F$ is minimized when the effective dictionary A has the form

$$A_{opt} = U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T.$$

This necessarily stipulates that the Gram matrix of the effective dictionary is equal to W_o , as in Eq. (9.33)

$$W_o = V \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} V^T, \quad (9.34)$$

where V is orthonormal with dimension M .

By definition, the effective matrix is the product of the sensing matrix and the dictionary, so

$$A_{opt} = U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T = \Phi \Psi,$$

and

$$\Phi = U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V_d^T \Psi^+. \quad (9.35)$$

For the SVD of the matrix Ψ , since $m < M$, the last $(M-m)$ columns of Σ_d will be zeros. Also, since $m < \bar{N}$, the last $\bar{N} - m$ rows of Σ_d will be zero,

$$SVD(\Psi) = U_d \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} V_d^T. \quad (9.36)$$

Correspondingly, the Moore-Penrose pseudoinverse of Ψ can be written as

$$\Psi^+ = V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T. \quad (9.37)$$

Substituting this in for the pseudoinverse of Ψ in (9.35) gives

$$\Phi = U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T. \quad (9.38)$$

Now, compute the Gram matrix of the effective dictionary $A = \Phi\Psi$

$$\begin{aligned} G(A) &= (\Phi\Psi)^T(\Phi\Psi) \\ &= \Psi^T \Phi^T \Phi \Psi. \end{aligned} \quad (9.39)$$

Begin by simplifying the inner matrix product $\Phi^T\Phi$

$$\Phi^T\Phi = \left(U_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix}^T V_d^T V \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix}^T U^T \right) \left(U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T \right) \quad (9.40)$$

Noting that U, V, V_D are each orthogonal matrices, this can be simplified as:

$$\Phi^T\Phi = U_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T \quad (9.41)$$

Substituting this into Eq. (9.39)

$$\begin{aligned}
 G(A) &= \Psi^T \Phi^T \Phi \Psi \\
 &= \Psi^T U_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T \Psi
 \end{aligned} \tag{9.42}$$

Replacing Ψ with its SVD from Eq. (9.36)

$$\begin{aligned}
 G(A) &= \Psi^T U_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T \Psi \\
 &= \left(U_d \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} V_d^T \right)^T U_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T \left(U_d \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} V_d^T \right) \\
 &= V_d \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} U_d^T U_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T U_d \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} V_d^T \\
 &= V_d \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Sigma_d & 0 \\ 0 & 0 \end{bmatrix} V_d^T \\
 &= V_d \begin{bmatrix} I_{\tilde{N}} & 0 \\ 0 & 0 \end{bmatrix} V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d \begin{bmatrix} I_{\tilde{N}} & 0 \\ 0 & 0 \end{bmatrix} V_d^T.
 \end{aligned} \tag{9.43}$$

Define the matrix Q as

$$Q \equiv V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d. \tag{9.44}$$

Then, $G(A)$ can be rewritten using Q as

$$\begin{aligned}
 G(A) &= V_d \begin{bmatrix} I_{\tilde{N}} & 0 \\ 0 & 0 \end{bmatrix} \overbrace{V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d}^Q \begin{bmatrix} I_{\tilde{N}} & 0 \\ 0 & 0 \end{bmatrix} V_d^T \\
 &= V_d \begin{bmatrix} I_{\tilde{N}} & 0 \\ 0 & 0 \end{bmatrix} Q \begin{bmatrix} I_{\tilde{N}} & 0 \\ 0 & 0 \end{bmatrix} V_d^T.
 \end{aligned} \tag{9.45}$$

Recall from Eq. (9.33), the definition of W_o

$$W_o = V \begin{bmatrix} I_m & 0_{m \times (M-m)} \\ 0_{(M-m) \times m} & 0_{(M-m) \times (M-m)} \end{bmatrix} V^T, \quad (9.46)$$

where V is orthonormal. The objective function is minimized if and only if $G(A) = W_o$. What constraint does this place on the matrix Q ? Note that $G(A)$ will be of the form of W_o if

$$Q = \begin{bmatrix} I_{\bar{N}} & 0 \\ 0 & 0 \end{bmatrix} Q \begin{bmatrix} I_{\bar{N}} & 0 \\ 0 & 0 \end{bmatrix}, \quad (9.47)$$

i.e., if Q is unchanged by pre- and post-matrix multiplication by the upper left $\bar{N} \times \bar{N}$ identity matrix, in which case

$$\begin{aligned} G(A) &= V_d \begin{bmatrix} I_{\bar{N}} & 0 \\ 0 & 0 \end{bmatrix} Q \begin{bmatrix} I_{\bar{N}} & 0 \\ 0 & 0 \end{bmatrix} V_d^T \\ &= V_d Q V_d^T. \end{aligned} \quad (9.48)$$

Eq (9.47) will be satisfied if Q takes the following form

$$Q = \begin{bmatrix} Q_{11} & 0 \\ 0 & 0 \end{bmatrix}, \quad (9.49)$$

where Q_{11} is of dimension $\bar{N}$. Enforcing this structure onto the definition of Q gives

$$Q \equiv V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d = \begin{bmatrix} Q_{11} & 0 \\ 0 & 0 \end{bmatrix}. \quad (9.50)$$

In order for this to be true, the matrix product $V_d^T V$ and $V^T V_d$ must each be *two-block diagonal*,

where the dimension of each block is $V_{11} \in \mathbb{R}^{\bar{N} \times \bar{N}}$ and $V_{22} \in \mathbb{R}^{(M-\bar{N}) \times (M-\bar{N})}$, respectively:

$$\begin{aligned}
Q &\equiv V_d^T V \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} V^T V_d \\
&= \begin{bmatrix} Q_{11} & 0 \\ 0 & 0 \end{bmatrix} \\
&= \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix} \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix}^T.
\end{aligned} \tag{9.51}$$

Equating the first and third lines in the previous equation, it is noted that

$$\begin{aligned}
V_d^T V &= \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix} \\
V^T V_d &= \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix}^T.
\end{aligned} \tag{9.52}$$

Lastly, to obtain the sensing matrix form required for the above Gram matrix, substitute $V^T V_d$ from Eq. (9.52) into Eq.(9.38)

$$\begin{aligned}
\Phi_{opt} &= U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} V^T V_d \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T \\
&= U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix} \begin{bmatrix} \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T \\
&= U \begin{bmatrix} I_m & 0_{M-m} \end{bmatrix} \begin{bmatrix} V_{11}^T \Sigma_d^{-1} & 0 \\ 0 & 0 \end{bmatrix} U_d^T,
\end{aligned} \tag{9.53}$$

where V_{11} and Σ_d are each of size $\bar{N} \times \bar{N}$. In addition, a general form for the Gram of the optimized effective dictionary can be found by making the substitutions for $V_d^T V$ and $V^T V_d$ into the last line of Eq. (9.43)

$$\begin{aligned}
G(A_{opt}) &= A_{opt}^T A_{opt} = (\Phi_{opt} \Psi)^T (\Phi_{opt} \Psi) \\
&= V_d \begin{bmatrix} I_{\bar{N}} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix} \underbrace{V_d^T V}_{V_d^T V_d} \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix}^T \begin{bmatrix} I_{\bar{N}} & 0 \\ 0 & 0 \end{bmatrix} V_d^T \\
&= V_d \begin{bmatrix} I_{\bar{N}} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix} \begin{bmatrix} I_m & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_{11} & 0 \\ 0 & V_{22} \end{bmatrix}^T \begin{bmatrix} I_{\bar{N}} & 0 \\ 0 & 0 \end{bmatrix} V_d^T.
\end{aligned} \tag{9.54}$$

9.4 Additional Information on Linear Algebra Topics

This section contains some additional notes on linear algebra-related tools that are relevant to this dissertation.

9.4.1 Symmetric Matrices

The set of symmetric $M \times M$ matrices will be denoted S^M and defined as

$$S^M = \{A \in \mathbb{R}^{M \times M} | A = A^T\}. \tag{9.55}$$

A *symmetric matrix* A is a square matrix for which $A^T = A$, i.e., $a_{ij} = a_{ji}$ [19, 29]. The following statements are all equivalent for a symmetric $n \times n$ matrix A [29]:

- All n pivots are positive.
- All n upper left determinants are positive.
- All n eigenvalues are positive.
- The scalar $x^T A x$ is positive except at possibly $x = 0$.
- The matrix A can be Cholesky-decomposed as $A = R^T R$.

9.4.2 Positive Semi-Definite Matrices

This section contains some useful results on symmetric positive semi-definite (PSD) matrices. Note that only PSD matrices with real entries are considered in this dissertation. The set of symmetric

positive semidefinite matrices is represented as:

$$S_+^M = \{X \in S^M | X \succeq 0\}, \quad (9.56)$$

where $\succeq$ represents the matrix inequality [52]. A matrix $A \in S_+^M$, i.e., is a symmetric positive semidefinite matrix if

$$x^T A x \geq 0, \quad (9.57)$$

$\forall x > 0$ [29].

The $M \times M$ Gram matrix of an $m \times M$ matrix A is found by multiplying the transpose of A by A , and is defined as

$$G(A) = A^T A \quad (9.58)$$

The Gram matrix is symmetric and *positive semidefinite* (PSD) [52].

9.4.3 Affine Sets, Convex Sets and Cones

A set C is *affine* if the line (extending to infinity in both directions) that passes through any two different points in C , also lies within C . This can be formulated as:

$$\theta x_1 + (1 - \theta) x_2 \in C, \quad (9.59)$$

$\forall x_1, x_2 \in C$, and $\theta \in \mathbb{R}$. The definition of an affine set can be equivalently stated as the set C that contains the linear combination of any two points in C for which the coefficients in the linear combination sum to one [52]. The solution set of a system of linear equations is an affine set. Alternatively stated, the converse is also true: every affine set can be expressed as the solution set of a system of linear equations.

A set C is *convex* if the line **segment** between any two points in C lies within C . Mathematically, the set C is convex if

$$\theta x_1 + (1 - \theta) x_2 \in C, \quad (9.60)$$

$\forall x_1, x_2 \in C$, and $0 \leq \theta \leq 1$ [52]. The main difference between an affine set and a convex set lies in the distinction that the affine set consists of a line of infinite length, and the convex set consists of a line segment. Every affine set is also a convex set, since the affine set contains the entire line and

therefore any two distinct points in it. As a result, each pair of distinct points forms a line segment and so the affine set also contains the convex set formed by the points on the line segment [52].

A set C is a *cone* (a.k.a. nonnegative homogeneous) if for every element in C , all multiples of the elements of C formed with nonnegative coefficients are also in C :

$$\theta x \in C. \quad (9.61)$$

If a set is both convex and a cone, then the set is referred to as a *convex cone*. Mathematically a convex cone is defined as a set such that

$$\theta_1 x_1 + \theta_2 x_2 \in C, \quad (9.62)$$

$\forall x_1, x_2 \in C$, and $\theta_1, \theta_2 \geq 0$ [52]. A convex cone is closed under taking linear combinations of points (vectors) from the set C , using nonnegative coefficients. An example of a cone that is not convex is the set formed from the union of the first and third quadrant [91].

An important example of a convex cone, in relation to this dissertation, is the set (space) of symmetric positive semidefinite matrices, i.e., S_+^M forms a *convex cone* (see Section 9.4.4).

9.4.4 The Set of Positive Semi-Definite Matrices form a Cone

PSD matrices arise as [52, 57, 70]:

- Kernels in machine learning
- Covariance matrices within statistics
- Search space in convex and semidefinite programming
- Diffusion tensors in medical imaging applications

The set of all PSD matrices forms a convex cone [52, 70]. To show that the set of positive semidefinite matrices S_+^M forms a convex cone, start by showing that the space of positive semidefinite matrices forms a cone [52, 55]. That is, show that $\theta A \in S_+^M, \forall A \in S_+^M$, where θ is a real scalar. Since $A \in S_+^M$, by the definition of S_+^M in Eq. (9.57), $x^T A x \geq 0, \forall x > 0$. To show that $\theta A \in S_+^M, \forall x > 0, \forall \theta > 0$, it suffices to show that θA is a positive semidefinite matrix. Since the matrix A is positive semidefinite,

$$x^T A x \geq 0. \quad (9.63)$$

Multiplying each side by the scalar θ , and applying the linearity of vector and matrix multiplication with a scalar

$$\begin{aligned}\theta x^T A x &\geq 0 \\ x^T \theta A x &\geq 0.\end{aligned}\tag{9.64}$$

The last line indicates that θA is a positive semidefinite matrix, and consequently that the space of positive semidefinite matrices forms a cone.

Now to show that the space of positive semidefinite matrices forms a *convex* cone, by Eq. (9.62) for $C = S_+^M$, it suffices to show that the linear combination of any two positive semidefinite matrices formed with positive coefficients is a positive semidefinite matrix. Matrices A and B are each assumed positive semidefinite, and by Eq. (9.64), $\theta_1 A$ and $\theta_2 B$ are each positive semidefinite matrices for $\theta_1, \theta_2 \geq 0$

$$\begin{aligned}x^T \theta_1 A x &\geq 0 \\ x^T \theta_2 B x &\geq 0\end{aligned}\tag{9.65}$$

$\forall A, B \in S_+^M$, and $\forall \theta_1, \theta_2 \geq 0$. Adding these two equations gives

$$x^T \theta_1 A x + x^T \theta_2 B x \geq 0.\tag{9.66}$$

Now factoring out the x^T and the x from each term

$$x^T (\theta_1 A + \theta_2 B) x \geq 0.\tag{9.67}$$

Therefore, $\theta_1 A + \theta_2 B$, is a positive semidefinite matrix. Since $\theta_1 A + \theta_2 B$ is a linear combination of positive semidefinite matrices with positive coefficients, and belongs to the set of positive semidefinite matrices, it is concluded that Eq. (9.62) is satisfied for $x_1, x_2 \in C = S_+^M$, the set of positive semidefinite matrices.

Alternatively put, the linear combination of any two points within a convex cone (formed from positive coefficients) reside within the cone. Note that the boundary of the cone is formed from lines of infinite length. Therefore, it was shown that the linear combination of any two positive semidefinite matrices (formed from positive coefficients) is also a positive semidefinite matrix, and therefore the space (set) of positive semidefinite matrices forms a convex cone [91], and and sections

2.1.5 and 8.3.1 of [52].

9.4.5 Geometric Interpretation of Gram Matrix Entries

The geometry of the dictionary, or specifically, the geometric relationship among pairs of dictionary atoms can be computed from the Gram matrix entries [52]. It is noted that much of the material in this section follows from [52]. Restating Eq. (9.58) for the Gram matrix of a dictionary A

$$G(A) = A^T A,$$

the i^{th} column vector of A is a_i

$$A = [a_1 \ a_2 \ \cdots \ a_M],$$

so that the i^{th}, j^{th} entry of the Gram is $G_{ij}(A) = a_i^T a_j$. The diagonal entries of the Gram matrix G are defined as

$$G_{ii} = a_i^T a_i,$$

To relate the distance between atoms to the Gram matrix entries, start by defining the Euclidean length of the i^{th} atom as $l_i = \|a_i\|_2$. Next, recall the definition of the scalar (dot-product) of two vectors a_i, a_j and show that the i^{th}, j^{th} Gram entry is the scalar product of the i^{th} and j^{th} atoms

$$\begin{aligned} a_i \cdot a_j &= \sum_{n=1}^m a_i(n) a_j(n) \\ &= l_i l_j \cos(\theta) \\ &= a_i^T a_j \\ &= G_{i,j} \end{aligned} \tag{9.68}$$

where θ is the angle subtended between the atom vectors a_i and a_j . Next, define the distance between the a_i and a_j atom vector to be

$$d_{ij} = \|a_i - a_j\|_2. \tag{9.69}$$

Looking at Figure 37 for reference, apply the law of cosines to the triangle

$$d_{ij}^2 = l_i^2 + l_j^2 - 2l_i l_j \cos(\theta). \tag{9.70}$$

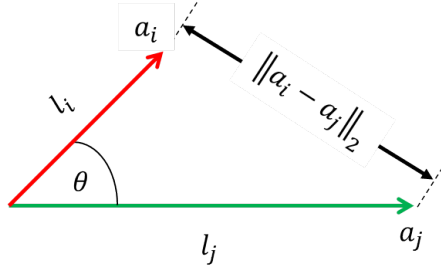


Figure 37: A pair of atom vectors from an un-normalized dictionary.

Substituting from Eq. (9.68) this can be written as

$$\begin{aligned}
 d_{ij}^2 &= l_i^2 + l_j^2 - 2l_i l_j \cos(\theta) \\
 &= l_i^2 + l_j^2 - 2a_i^T a_j \\
 &= l_i^2 + l_j^2 - 2G_{i,j}.
 \end{aligned} \tag{9.71}$$

Taking the square root of each side gives

$$\begin{aligned}
 d_{ij}^2 &= l_i^2 + l_j^2 - 2G_{i,j} \\
 d_{ij} &= (l_i^2 + l_j^2 - 2G_{i,j})^{1/2}.
 \end{aligned} \tag{9.72}$$

Now, if it is assumed that the columns of $A = [a_1, a_2, a_3, \dots, a_M]$ are normalized, i.e. $l_i = \|a_i\|_2 = 1$, then

$$\begin{aligned}
 d_{ij} &= (l_i^2 + l_j^2 - 2G_{i,j})^{1/2} \\
 &= (1 + 1 - 2G_{i,j})^{1/2} \\
 &= (2 - 2G_{i,j})^{1/2}
 \end{aligned} \tag{9.73}$$

and

$$\begin{aligned}
 G_{i,j} &= a_i^T a_j \\
 &= l_i l_j \cos(\theta) \\
 &= \cos(\theta)
 \end{aligned} \tag{9.74}$$

9.4.6 Geometric Perspectives on the Covariance and Gram Spaces

Comparing Gram matrices has much in common with comparing covariance matrices. The set of all S_{++}^M symmetric positive definite matrices does not qualify as a Euclidean vector space under the standard matrix addition and scalar multiplication operations [57]. The set of all S_{++}^M symmetric positive definite matrices is an open convex cone, and is closed only under positive scalar multiplication [57]. However, the space defined by S_{++}^M admits a smooth manifold structure and can be endowed with a Riemannian metric. As a result, [57] suggests that an adequate measure of closeness between pairs of covariance matrices might not be the Euclidean distance. Instead, a distance measure that captures the intrinsic geometry exhibited by the space S_{++}^M would be more suitable [57]. The text [57] presents three different perspectives on comparing (or measuring the similarity between) symmetric positive definite covariance matrices (specifically matrices that have been made positive definite through some type of regularization process, e.g., *diagonal loading*):

- **Perspective 1:** S_{++}^M as a subset of the Euclidean space $S_{++}^M \subset \mathbb{R}^{M^2}$. This subspace then has the Euclidean distance induced on it and a given SPD matrix is treated as a vector.
- **Perspective 2:** S_{++}^M as a smooth manifold. The manifold then has the affine-invariant Riemannian and the log-Euclidean geodesic distances that act as Riemannian metrics on this manifold.
- **Perspective 3:** S_{++}^M as an open convex cone. Then the geometry of the cone induces Bregman divergences that can be defined as a distance measure.

This dissertation will mainly take the perspective (1) above, by analyzing and comparing Gram matrices via distributions of their entries, without regard for the matrix structure. However, the log-Euclidean geodesic distance from perspective (2) will also be studied. In addition to [57], discussions regarding Riemannian metrics, Bregman divergences and topics relating to *f-divergences* can be found in [73].

9.4.7 Metrics, Norms and Matrix Norms

A vector space endowed with an inner product is referred to as a normed vector space ([22], p.516).

Let X be a set. Then a *metric* (d) is defined as function $X \times X \rightarrow [0, \infty)$ that measures distance between elements in the set X , where d possess the following properties for all $x, y, z \in X$:

- $d(x, y) = d(y, x)$
- $d(x, y) \geq 0$
- $d(x, y) = 0$ if and only if $x = y$
- $d(x, z) \leq d(x, y) + d(y, z)$

For additional information on metrics, see [52], Section 8.3.

The definition of a **norm** presented below is based on that which is found in [18]. Let S define a vector space, where $x \in S$. A real-valued function of, denoted $\|x\|$ is a norm of x if $\|x\|$ satisfies the following properties:

- $\|x\| \geq 0, \forall x \in S$ (Nonnegativity)
- $\|x\| = 0, IFF x = 0$
- $\|\alpha x\| = |\alpha| \|x\|$
- $\|\alpha x + y\| \leq \|\alpha x\| + \|y\|$ (Triangle Inequality)

9.5 Cognitive Radio Networks

9.5.1 Section Overview

This section introduces dynamic spectrum access (DSA) concepts such as the cognitive radio network (CRN) architecture. First, some motivations for DSA are explored. Next, general CRN design considerations and potential uses are presented and described. Specific aspects of two CRN design issues are then addressed. The first aspect includes methods and approaches to centralized (database assisted) cooperative spectrum sensing (CSS). Lastly, CRN security issues are briefly reviewed, along with a focus on spectrum sensing data falsification (SSDF) attack models and mitigation strategies. The intent of this section is lay the groundwork for the spectrum approximation and verification applications in the subsequent sections by reviewing some aspects of CRNs found in the literature.

9.5.2 Dynamic Spectrum Access

As noted by [92], DSA has been defined as real-time adjustment of spectrum utilization in response to changing circumstances and objectives [93]. In addition to enabling mobile networking, RF spectrum also enables potentially significant economic growth [92]. As a result, wireless service providers are working hard to campaign for more efficient use of spectrum in an effort to address the

artificial spectrum shortage. DSA is also an important consideration for dense RF networks. Dense RF networks are defined as a drastic increase in the number of users and/or the number of sensors over a geographic area [1, 2, 94–96].

There are two basic approaches to CRN networks: *underlay* and *overlay* networks [1, 97]. In overlay networks, users dynamically use spectrum that is un-occupied with respect to the noise floor. Underlay networks use the concept of *interference temperature*. A secondary user (SU) CR can use spectrum that is occupied by the primary user (PU), as long as the SU maintains its transmitter power low enough so that it doesn't interfere with the operations of the PU [1, 97].

9.5.3 Cognitive Radio Networks

As pointed out in [92] the International Telecommunications Union has defined CR as “an approach to wireless engineering wherein the radio, radio network, or wireless system is endowed with awareness, reason, and agency to intelligently adapt operational aspects of the radio, radio network, or wireless system” [98].

SDRs perform radio functions adaptively in software as opposed to hardware (e.g., adaptive channel selection). CRs are anticipated to be a type of SDR that will “sense and analyze their local radio environment, negotiate optimal sharing arrangements, and then adapt their radio operating parameters (i.e., frequency, power, modulation, transmission timing, direction of transmission, and other waveform characteristics) to maximize shared use of local spectrum resources [92].”

SDRs, smart protocols, and machine learning techniques are three classes of tools being developed and researched to implement the CRN concept by facilitating sharing of the spectrum on a non-interference basis. The process of finding and using available spectrum is referred to the *cognitive cycle* and includes [77]:

- **Sensing:** Observing and recording the spectrum
- **Analysis:** Reasoning to identify optimal operating parameters
- **Adaptation:** Quickly changing the parameters
- **Acting:** Communicating with optimal parameters

In this dissertation, the focus is on the *Analysis* phase of the cognitive cycle. Specifically, performing cognitive reasoning on the sensed spectrum. Observing and recording spectrum usage/availability over the geographic area serviced by the CRN falls under the *Sensing Phase* of

the cognitive radio cycle. This is known as *spectrum sensing*. Numerous publications have developed collaborative or cooperative methods to perform spectrum sensing, and some published methods of spectrum sensing include (but are not limited to): energy-based detection, matched-filtering, covariance-based detection, waveform-based, wavelet, and cyclostationary detection [78]. Different methods perform better in some contexts, and the choice of spectrum sensing method may influence the selection of security and reasoning methods that are subsequently applied.

In this work, the focus application is to perform spectrum sensing data falsification (SSDF) detection. It is assumed that the trusted sensor nodes (SNs) and the candidate secondary user (SU) have already performed energy detection-based spectrum sensing. Subsequent to spectrum sensing, the application of interest in this dissertation is aimed at providing the CRN a security tool to decide if the reported spectrum from a candidate SU is to be trusted. The task of determining trust or authenticity of a users spectrum falls under the *Analysis* phase of the RF cognitive cycle.

9.5.4 Cooperative Spectrum Sensing

9.5.4.1 Types of CSS

Cooperative spectrum sensing (CSS) describes the process of *crowdsourcing* the task of spectrum sensing [99]. CSS is a spatially-diverse approach to spectrum sensing, i.e. CSS helps to compensate for SUs that may be blocked by obstructions and are not able sense some transmitters (a.k.a. the *hidden node* problem). In CSS, each user in the CRN coverage area can potentially contribute its locally sensed channel usage and availability. Some publications advocate sharing this spectrum knowledge directly with other SUs; this is referred to as *de-centralized fusion*. Others propose a fused, centralized database or aggregator at which the cooperatively shared spectrum is stored (known as centralized fusion). Still others propose a set of fixed trusted sensor nodes (SNs) that are not SUs per se, but provide means for evaluating assessing spectrum usage and/or collaboratively forwarding SU spectrum reports to the centralized database. All of these techniques and their derivatives are intended to provide an organized way to ascertain and manage spectrum over the area.

Each SU senses and digitizes the signals within the *spectral search space*. The *spectral search space* is the pool of possible frequency bins which an SU may potentially use. The spectral search space contains N elements or channels that need to be monitored. Each SU reports it's environmental spectrum information in the form of a *spectrum report*. The spectrum report is computationally

realized as a column vector $y \in \mathbb{R}^N$ and contains the estimated power at each of the N channels in the spectrum search space.

9.5.4.2 Centralized Spectrum Access System

In this dissertation, CSS is assumed to be performed, and then aggregated at a single center known as the *spectrum access system* (SAS) [92]. Other publications describing centralized management of these processes refers to the SAS as the *fusion center* (FC). In this dissertation, the term SAS will be taken to be synonymous with the term FC, and from this point forward, the term SAS will be exclusively used to avoid confusion. SNs re-transmit (forward) SUs local spectrum reports to the SAS, so as to obtain a more accurate estimate of the local spectrum use, as opposed to using individual a single sensing report. Multiple sensing reports are collected together in one location (at the SAS) to form the GDB in the hopes that a reliable single sensing report for a particular area can be obtained. However, the issue of whether to trust the incoming spectrum sensing report arises. One method suggested in the literature for doing so is through the use of trusted sensor nodes (SNs). In this dissertation, in addition to utilizing spectrum from SUs, the system also has access to a set of geographically separated trusted sensor nodes (SNs), which the system uses to ascertain the reliability of the spectrum reported from the SUs.

To summarize, the components of the database-assisted CSS CRN (utilized in this dissertation) are:

- CRN Users: Primary Users (PUs), Secondary Users (SUs)
- Candidate Secondary Users (CSUs)
- Trusted Sensor Nodes (SNs)
- Spectrum Access System (SAS)
- Geographic Database (GDB)

9.5.4.3 The Geographic Database

A geographic database (GDB) in its simplest instantiation, is simply a database of channels that are used in a given location, at a given time, and is maintained at the SAS. The GDB goes by various names, e.g. radio environment map or spectrum database. In some publications, there is no distinction made between terms SAS and GDB. A GDB is only one part of the SAS which acts to manage the spectrum, incorporate and enforce policy enabling effective coexistence of RF devices

over an area. Note that in some publications, the concept of the radio environment map (REM) is introduced to capture the essence of all of these tasks [100, 101].

Many publications address the design and usage of GDBs for CRN [102–107]. The GDB considered in this present work contains channel indexes and associated spectral power as estimated at trusted sensor nodes (SNs), which are scattered in a region surrounding the CR. Specifically the GDB considered here contains:

- SN index
- Occupied channel index
- Estimated signal energy or power

Each of the M SNs is indexed by $i \in [1, M]$ and its estimated power at each of the N channels is recorded in the form of a column vector $\psi_i \in \mathbb{R}^N$, and is referred to as a *sensor node spectrum report*. At present it appears unclear to what extent and which data in the GDB will be shared and be publicly available [100].

A confluence of factors has enabled the geo-location database (GDB) to evolve, including internet connectivity, cloud-based computing, low-cost geo-location technology, low-cost sensing technology, efficient compression algorithms and communications protocols [100]. With the advent of dense and ultra-dense networks, GDB design requirements will be driven largely by high-throughput, with regards to computation, storage, querying, and communication throughput [1, 2].

The primary responsibility of a SAS is to protect the incumbent or PUs from RF interference [92] while aiding SU devices by supplying available channel capacity. In order to achieve this responsibility, it is helpful for the SAS to know the location and operating parameters of the RF devices within its geographic area. There must be a means for the SAS to translate policy into allowable operational parameters in order to be able to effectively adapt the SU devices. SAS operators will likely be empowered by regulatory agencies to protect incumbents and ensure the fair distribution of available spectrum to authorized SUs. While the actual enforcement of regulation violations will likely be carried out by the regulators, the GDB needs to be able to provide the regulators information required to determine the extent of unauthorized transmission or interferences. The GDB will be required to warehouse historical data it collects for some specified amount of time to facilitate posterior regulation investigation and enforcement.

Another entity which may be involved in the spectrum request process is a *spectrum broker*, who may purchase spectrum from users with more priority and lease the spectrum to users residing at a

lower priority [108]. In addition the other duties of spectrum management and policy enforcement, it may be possible for the SAS to perform the duties of a spectrum broker as well.

In this dissertation, it is assumed that the SAS will mainly process spectrum requests (queries) from candidate SUs. In order to do so, the SAS will need to perform the following tasks, relevant to this dissertation:

- Receive spectrum reports generated by trusted SNs and aggregate the reports into the GDB
- Receive spectrum reports generated by candidate SUs
- Be able to identify which SNs first received the candidate SU spectrum request
- Perform verification of the candidate SUs spectrum report with respect to the SN reports in the GDB

When a potential SU desires to use a channel, it will issue a *spectrum request*. The spectrum request will be received by the SNs in its local vicinity and will be collaboratively forwarded to the SAS, along with the SN IDs corresponding to the SNs that first received the spectrum request. Candidate SU spectrum requests may also include queries such as [100]

- What channels are available and for what time period(s)?
- Which available channels are optimal?
- How much spectrum is required to perform a specific task or use a specific application?

For frequently used applications, the GDB may be able to inform the SU regarding the amount or location of potential spectrum. In addition, the CSU may also choose to provide information concerning its current and anticipated location(s), and anticipated time during which spectrum is expected to be used.

9.5.5 Additional Aspects of the CRN Signal Model

The coverage area of the cognitive radio network (CRN) is assumed to contain N_E RF transmitters. Each transmitter emits a strictly bandlimited signal of bandwidth B . The received signal model is a multiband signal, i.e., the received signal is assumed to consist of a finite sum of such bandlimited signals. At each of the receivers, the received signal s is bandpass filtered, shifted down in frequency, sampled at a rate of F_s , and low-pass filtered. The baseband received signal is corrupted with additive white Gaussian noise (AWGN) and is represented as

$$s[n] = s_o[n] + \eta, \quad (9.75)$$

where $s_o, s \in \mathbb{R}^{N_s}$, and $\eta \in \mathbb{R}^{N_s}$ is a vector whose elements are drawn IID from $\mathcal{N}(0, \sigma^2)$, a zero-mean normal distribution with variance σ^2 set so as to result in a desired signal to noise ratio. In the spectrum approximation the PSD of the received signal $s[n]$ is estimated at both a candidate secondary user (CSU) and at a trusted sensor node (SN). The power spectral density (PSD) estimate P of the signal s is computed via the periodogram, defined as

$$P(\omega) = \frac{1}{N_s} \left| \sum_{n=0}^{N_s-1} s[n] e^{-j\omega n} \right|^2, \quad (9.76)$$

where N_s is the number of signal samples in one observation window [79, 80]. Next, the PSD is divided in frequency (channelized) into N subbands such that a single subband contains the maximum PSD estimate from a block of $\Delta = N_s/N$ contiguous frequencies:

$$x_c[m] = \max_{i=m\Delta:(m+1)\Delta} \{P(\omega_i)\}, \quad (9.77)$$

for $m = 0, 1, \dots, N - \Delta$, so that x_c is of size $N \times 1$.

After the PSD is channelized, it is multiplied by a constant gain factor g , and thresholded so that any entry of $x_c[m]$ below P_{min} is quantized to zero, and any entry of $x_c[m]$ above P_{max} is set to P_{max} :

$$x[m] = \begin{cases} gx_c[m] & P_{min} \leq gx_c[m] \leq P_{max} \\ P_{max} & gx_c[m] \geq P_{max} \\ 0 & gx_c[m] \leq P_{min}. \end{cases} \quad (9.78)$$

The vector $x[m] \in \mathbb{R}^N$ is referred to as a *spectrum report* and can be generated either at a CSU or any trusted SN in the coverage area. The spectrum report generated at the i^{th} trusted SN will be denoted ψ_i and the spectrum report generated by the CSU will simply be denoted by x . In addition to the periodogram method, there are a variety of non-parametric and parametric methods available to estimate PSD [79, 80].

9.5.6 CRN Security

9.5.6.1 Overview

Underlying all of the phases of the cognition cycle is the requirement of security in the CRN. Operational security is a critical issue for successful CRN. All four of the above cognition phases possess security vulnerabilities. Precisely where the enforcement and policing should occur is still a debated issue [100]. In addition to conventional RF network security threats, CR technology introduces entirely new classes of security threats and challenges, over and above traditional RF networks.

CRN threats are mainly dependent on two main capabilities: Cognition and reconfigurability. Cognitive capability threats include adversaries mimicking PU behaviors, transmitting a false report of spectrum sensing, and jamming channels for legitimate SUs and PUs during sensing and acting phases in cognitive cycle. The CRs reconfigurability can be manipulated by attackers during analysis and/or adaptation phases of the CRN cycle via installation of malware on SU devices [77]. Three main areas in which attacks can occur are:

- Primary User Emulation
- Spectrum Sensing Data Falsification (SSDF)
- Location Privacy

In primary user emulation (PUE), a candidate CR pretends to behave in some manner like a PU, with the intent of fooling the system into thinking that it has priority of other SUs in terms of spectrum use [109]. While PUE is not the focus of this work, it is mentioned here for completeness. In this work, the focus is specifically on the SSDF attack. Dynamically sharing spectrum presents security concerns, both for the incumbents, as well as for the SUs who are looking utilize the spectrum. Through spoofing, jamming, and data alteration, attackers strive to exploit the spectral information gained to degrade performance, with respect to privacy, integrity, and availability [110, 111]. Secure CRN operation will involve trying to protect the incumbent users or PUs from SUs that may intentionally or unintentionally interfering with existing PU communications [112].

DSA in CSS CRNs can present security concerns through selfish behavior such as hoarding spectrum opportunities, possibly by reporting false sensing information [113]. A selfish SU itself can initiate an attack by attempting to occupy all or part of the available radio resources, prohibiting

other PUs or SUs from accessing those, and significantly degrading the overall performance of a CRN.

Traditional security approaches are largely intended for static spectrum usage, and are not directly applicable to DSA-based CRNs. Since users in distributed CRNs are geographically separated, locating malicious actors can be difficult due to privacy requirements of the SU and the ability of the SU to dynamically change its occupied spectrum [114].

When a fusion center with GDB is utilized within a CRN, database attacks become a concern. DB threats are present in various layers of the communications protocol. In this work, the focus is on the physical (PHY) layer of CRN DB threats. Reference [115] defines three classes of PHY-layer threats including sensory manipulation attacks, belief manipulation attacks against learning radios, and self-propagating behavior potentially leading to cognitive radio viruses. The focus of this dissertation is on sensor manipulation attacks that will be outlined below.

9.5.6.2 Spectrum Sensing Data Falsification

During the course of CSS, when using a centralized SAS, measurements from multiple distributed SUs are fused/integrated to produce an accurate appraisal of the spectrum usage throughout the CRN coverage area, the CRN needs a way to ensure that the reported spectrum is accurate [109]. Spectrum reports sent to the SAS may contain inaccurate or falsified information, thereby reporting that occupied channels are free, and available channels are occupied. This would deny legitimate users from accessing those channels that are available, and also result in interference for those users who are using the occupied channels. When a CR falsely reports spectrum information it is referred to as *spectrum sensing data falsification* (SSDF), a.k.a. a *Byzantine attack*. Since SSDF attacks may prevent legitimate users from accessing spectrum, SSDF is also considered a denial of service attack [116].

Multiple methods have been developed to perform CSS. They include [109]:

- Walds Sequential Probability Test
- Weighted Sequential Probability Ratio Test [117]
- Various Forms of the Neyman-Pearson

However, these early techniques do not provide a means of detecting SSDF. The survey [110] provides a review of multiple types of *Byzantine* attacks and defense in CR, and lays out a taxonomy of different types of CR attacks:

- Attack Scenario (Centralized/Decentralized)
- Attack Basis (Independent/Dependent)
- Attack Opportunity (Probabilistic/Non-Probabilistic)
- Attack Population (Small-Scale/Massive)

Furthermore, they divide CR Byzantine defense techniques into two different sensing scenarios: homogeneous and heterogeneous defense. In homogeneous defense, all relevant sensors are expected to report consistent results, whereas heterogeneous defense techniques presume that all relevant sensors can be expected to report different sensing results. In this dissertation, the focus is on the *heterogeneous* CR sensing scenario. Even when two SNs reside in relatively close proximity, differences between their sensing reports are expected, due to obstructions in the signal path and propagation effects (free-space path loss differences, multipath, etc.) [110]. In [110], two classes of Byzantine defense strategies for heterogeneous scenarios are described:

- *Propagation Model-Based Defense*: Uses characteristics of any known local propagation characteristics to ascertain the credibility of a sensing report
- *Likelihood Detection-Based Defense*: Uses known detection characteristics of the SUs to ascertain the sensing report credibility

The SSDF detection method proposed in this dissertation is different from both the propagation model and likelihood-based defense methods. In propagation model-based defenses ([118–120]), the sensing report generated by an SU is directly related to the known channel characteristics between the SU and PU. As a result, any differences between the sensing results of a pair of honest SUs (e.g. SU_1 and SU_2) should be due to the differences in the each of the PU-SU propagation channels $PU - SU_1$ and $PU - SU_2$. If each PU-SU channel is known, then any differences between SU_1 and SU_2 that cannot be explained by the differences in their respective channels can be analyzed for possible falsification. In this dissertation, prior known propagation characteristics of the PU-SU RF channels are not applied. Instead, the proposed method, each SN sector local to an honest CSU is used to construct a linear combination of sensing reports that approximates the spectrum reported by the CSU. However, some propagation-based detection concepts are similar the approach proposed in this dissertation, e.g. in [99] it is proposed to geographically group SUs based on the similarity of their sensing report. However in [99], the sensing report is a real-valued measurement of PU power, instead of a vector of spectrum measurements.

The SSDF detection method proposed in this dissertation is different from the likelihood detection-based defense as well. In likelihood-based defenses, the idea appears to be to find a relationship between an SU's ability to detect spectrum (e.g. detection performance) and the SU's sensing report. Deviations between the expected sensing performance of an SU and the SU's actual sensing performance can be used as a basis to detect when it's falsifying its sensing results. The method proposed in this dissertation uses no such information regarding the detection performance of a CSU.

A reputation-based SSDF detection scheme using trusted nodes is presented in [121]. The method uses a static threshold at the SAS to determine whether the spectrum report from an SU is in one of three trust states. Similarly, in [122], the authors divide CSS SSDF detection strategies into two very broad categories: outlier detection and reputation-based methods. They view SUs with bad trust metrics as outliers. In this method, using a dynamic (as opposed to static) threshold, the SAS (or fusion center) decides whether the PU channel is vacant. The adaptable threshold varies dynamically to capture changes in the spectrum that occur with time.

Another method of reputation-based SSDF detection is presented in [123] with trusted node assistance, and is referred to as *reliable spectrum sensing (RSS)*. The method also includes a novel *node-selection procedure*. The RSS method involves 1) Authenticating a trusted node, 2) Authenticating the cell for which the node is responsible for, and 3) Determining if the trusted node report agrees with the sensor report from the SU.

In [116], a regression-based method of SSDF detection is presented that uses a method called *maximum matched filtering (MMF)*. The MMF method appears to make what it calls *local sensing decisions*, but does not appear to define exactly what such a decision actually is, except that a set of decisions made by a given sensor is stored in a binary vector, so that each spectrum report consists of a binary vector. The distance between two sensing reports is taken to be the Hamming distance (i.e., the number of positions in which each sensing vector differs). The idea appears to be to fit a regression plane to the sensing reports. Sensing reports that are consistent with each other will be close to the plane; those reports that are inaccurate will be far from the plane. The residual from the fitted plane becomes an SSDF detection metric. The method presented in this dissertation also attempts to fit a hyperplane to the measured sensing reports at a centralized center. However, the present research in this dissertation is different in a number of aspects. First, the measurements in the present dissertation are not quantized to binary values. Second, the hyperplane fitting of [116] simply correlates sensing reports. In contrast, the present dissertation uses iterative approximation

(orthogonal matching pursuit) to de-mix the vectors that participate in the spectrum measurement, and so counteracts linear dependencies that may be present among the measurements. Third, in this work, the application of optimal compressive sensing matrices are explored to reduce the network overhead of having to transmit sensing reports over the CRN. Lastly, the current method applies to densified networks, dense both in users and in sensors. None of these four aspects is mentioned in [116]. The SSDF detection method is heavily based on a spectrum approximation method, and as such could be used to inform reputation-based methods.

9.5.6.3 SSDF Cases Addressed by the Proposed Method

One possible application of the greedy spectrum approximation method studied in this dissertation is as a secure way to perform centralized CSS by detecting when a SU is supplying inaccurate spectrum usage information (i.e. engaging in an SSDF attack.) The CRN architecture presumed in this dissertation is one that uses centralized CSS, where a set of trusted SNs scattered throughout the coverage area and spectrum is managed by a SAS with a GDB. These SNs can be dedicated hardware devices or other SUs whom the CRN has deemed trusted. A tangent area of security issues associated with database-centric CRNs is known as *secure database (DB) querying*. Secure DB querying is an active area of research [124, 125]. There has been interest in CRN DB security and in particular, within the area of CRN location privacy and spectrum verification, e.g., [4] provides a helpful survey on location privacy in CRNs.

The main type of SSDF attacks addressed by the tools developed in this work are known as *Byzantine attacks*, and in particular three types of Byzantine SSDF attacks are of interest:

- Spectrum Spoofing (sending spectrum sensed from a different node),
- Spectrum Inversion (inverting sensed spectrum reports)
- Spectrum Shifting (sending a shifted version of spectrum sensed)

Spectrum spoofing and spectrum inversion attacks have been identified in the literature [81, 126]. In spectrum spoofing, a malicious CR uses some technique (e.g., a high-gain antenna) to intentionally obfuscate its position to the FC, reporting its position to the FC as being in one location, when it actually resides in a different location. The issue of location verification is important when a CR queries the FC for available channels in its area. The FC must verify the location of the CR prior to granting it access to channels. If the candidate CR is deemed trusted, then the FC can also choose to

use its reported spectrum to update the GDB. The CR desires the requested bandwidth and the FC requires the CR location (at least to some level of precision/anonymity). Shifting the spectrum from another node is a more subtle attack in that the shifted spectrum shares similar overall characteristics (e.g., energy, spectral usage sparsity, bandwidth, etc.) with spectrum from other locations. The spectral shifting attack has not been found to date in the literature.

9.6 Additional Applications of Gram Matrix Distances

Aside from sensing matrix evaluation, Gram matrix distances are useful in other applications. One such application is for image classification [57], when each image is represented as a covariance matrix. As described in [57] and the references therein, each image consists of pixels and one can associate to each pixel (or group of pixels) a vector of features (e.g., intensity). A given set of pixels within an image then gives rise to a set of feature vectors, which can be stored within a matrix. The covariance of the data matrix provides a rich representation of the associated pixel set. The covariance is a symmetric, positive semi-definite matrix, and comparisons between sets of pixels can be performed by comparing the associated pair of covariance matrices.

The set of Gram matrices is also related via a linear operator, to the set of Euclidean distance matrices, which have numerous applications, including the sensor network localization problem, the matrix completion problem, and problems in molecular conformation [54, 127].

9.7 SSDF Prototyping Algorithms

This section includes a high-level description of the algorithms used in the MATLAB simulations.

9.7.1 Build CR Network GDB

Algorithm 2 Build CR Network Algorithm

Input: $A = [a_1, a_2, \dots, a_M] : mxM, y : mx1, k$
Randomly Place SUs Throughout Service Area
Randomly Assign Center Frequencies to Each SU
Build Cognitive Radio Database (Redundant Dictionary)
 for all $i=1$:Number of Tower Sites **do**
 for all $j=1$:Number of Active Transmitters (SUs) in Service Area **do**
 Compute geographic distance from j^{th} transmitter to i^{th} tower site
 Compute received power at i^{th} tower site due to j^{th} transmitter
 end for
 Find the set ind_{cur} of active Txs whose power measured at i^{th} tower site is above p_{thresh}
 For each transmitter in ind_{cur} , identify the sector in which it resides
 Convert received power for each transmitter from dBm to linear power scale
 Construct received signal and power spectral density estimate at each sector (1,2, and 3)
 of current (i^{th}) tower site
 Store the power spectral density of all three sectors of current tower site
 end for
Channelize to N spectral bins
Threshold any power below D_{thresh} to zero
Save to File (*.mat)

9.7.2 Call CRN Spectrum Analysis Simulation

Algorithm 3 Simulation Algorithm

Input: $A = [a_1, a_2, \dots, a_M] : mxM, y : mx1, k$
Initialize Configuration
Load Dictionary
Perturb Dictionary
Construct Sensing Matrices
Optionally Condition Sensing Matrices After Optimization
Compare Updated and Stale Effective Dictionaries
Call Monte Carlo Test
Compute Simulation Statistics
Plot Spectrum Approximation and Spectrum Verification Results

9.7.3 Construct Candidate SU Spectrum Report

Algorithm 4 Build Candidate SU Spectrum Report

Randomly select geographic location of candidate SU within service area
Find all other SUs in the service area whose power received at the CSU is greater than p_{thresh}
Compute composite received signal
Compute power spectral density of received signal
Channelize and threshold PSD to form spectrum report
Find cell tower sites (and sectors) closest to candidate SU
Output candidate CR spectrum report and truth atoms

9.7.4 Monte Carlo SSDF Test

Algorithm 5 Monte Carlo SSDF Test

Input: $A = [a_1, a_2, \dots, a_M] : mxM, y : mx1, k$
for all $i=1$:Number of Monte Carlo Iterations **do**
 Build candidate CR spectrum report
 Test for SSDF using updated optimized sensing matrix
 Test for SSDF using stale sensing matrix # 1
 Test for SSDF using stale sensing matrix # 2
 Test for SSDF using stale sensing matrix # 3
 Test for SSDF using stale sensing matrix # 4
end for
Output detection heuristics:
 Number of matched SNs (δ_{MATCH})
 Reconstruction Metric (δ_{RECON})
 Correlation Sign Metric (δ_{SIGN})

9.7.5 Compute SSDF Detection Simulation Statistics

Algorithm 6 Compute Simulation Statistics

Find average number of Correctly Matched Sensor Node IDs
 Compute probability of detection and probability of false alarm at various thresholds for:
 Number of matched SNs (δ_{MATCH})
 Reconstruction Metric (δ_{RECON})
 Correlation Sign Metric (δ_{SIGN})
Output candidate CR spectrum report and truth atoms

9.8 Core Prototyping Code

9.8.1 Overview

A selection of what is deemed to be the core functions of simulation code is provided in this section. It is hoped that by doing so, including a subset of the code aids in understanding the theory developed, verifying some concepts of the theory, as well as reproducing similar results upon which to build. The software environment used to develop the prototyping code is MATLAB, version R2017b.

9.8.2 Orthogonal Matching Pursuit

```
function [Supp, C, negCorr, Xr, resnorm, resvssolnorm, dk, lambda, yhat] = ...
    ompNative(Y,A,config, NumIters)

% OMP algorithm
% Adapted from Eldar & Misali's Modulated Wideband Converter Code
% INPUTS:
%       Y                : Measurements (m x 1)
%       A                : Sampling Matrix (m x M)
%       NumIters         : Number of Iterations (Sparsity Estimate)

%% Input Parsing
resvssolnorm = inf;
Xr=0; dk=0; lambda=0;
xhat = 0;

%% Initialization
% [mA,nA] = size(A);
% [mY,nY] = size(Y);

residual = Y;          % Initialize Residual to Measurements
Supp = [];             % Initialize Support to the Null Set
iter = 1;              % Set Iteration Counter to 1
resnorm = norm(Y);     %

if ~exist('plotFlag')
    plotFlag = 0;
end

% Compute 2-Norm of Each Column of A
NormACols = sqrt(diag(A'*A)); % L X 1
negCorr = zeros(NumIters,1);
while ((iter <= NumIters))
    %% Match
    % Find Which Column of Measurement Matrix A Most 'Participates' in the
    % Measurement I.e., Find that Column of the Measurement Matrix
    % that is Most Strongly Correlated with the Residual
    Z_1 = transpose(A)*residual;% L x 1 (L X m X 1)

    % Take ABS() Divide Each Element of Z_1 by the 2-norm^2
    % of it's associated Column in A
```

```

%    Z = abs(Z_1)./NormACols;% L x 1
%    Z = abs(Z_1);% L x 1
% Sometimes Previous Statement is Written as
Z = sum(abs(Z_1).^2,2)./(NormACols.^2);

%% Identify
[maxCor(iter), maxPos] = max(Z);% Index of Maximum Row Norm
BestLoc = maxPos(1);% If a tie occurs for multiple rows, select the lowest row
if Z_1(BestLoc)<0
    negCorr(iter) = 1;
end

%% Update
% Update Support (Index Set) with Maximum Row
Supp = [Supp BestLoc];

% Update Sub-Matrix of Selected 'Atoms'
As = A(:, Supp);

% Orthogonalize: Find LS Solution
xhat = pinv(As)*Y;

% Approximate Measurement Vector
yhat = As*xhat;

% Update Residual Between Measurements and Approximation
residual = Y-yhat;

% Update Norms
resnorm = norm(residual);
resvssolnorm = resnorm/norm(yhat);

% Increment iter index
iter = iter+1;

% Optionally Plot Original and Approximated Signal
if plotFlag
    figure(4); subplot(2,1,1)
    plot(Y); hold on; plot(yhat); legend('Measured', 'Approximated'); hold off
    subplot(2,1,2); plot(residual); ylim([-0.2 0.2])
    grid on; disp('Hit Any Key to Step Through'); pause
end

end

%% Optionally Use Unity Coefficients
if config.useUnityCoeff == 0
    C = xhat;%C=pinv(As)*Y;

elseif config.useUnityCoeff == 1
    C = ones(NumIters,1);
end

%% Reconstruction (Optional)
Xr = zeros(size(A,1),1);
XR = [];

if exist('As')

```

```

C = pinv(As)*Y;
for atom = 1:length(C)
    Xr = Xr + C(atom).*A(:,atom);
end

else
    Xr(Supp,:) = 0;
end

```

9.8.3 Inner Product Correlation

```

function [corrInd, C, negCorr] = correlate(D,x, maxNumAtoms)
% Inner Product Correlation

% Returns the 'maxNumAtoms' of atoms that most correlate
% with columns of the dictionary of 'D'

% INPUTS:
% D: Dictionary
% x: Input Vector
% maxNumAtoms: Sparsity Estimate

z1 = D'*x;
z = abs(z1).^2./(diag(D'*D));% Divide each element of z1 by 2-norm of D
[~,corrInd] = sort(z, 'descend');
corrInd = corrInd(1:maxNumAtoms);
negCorr = sign(z1(corrInd));
C = ones(size(corrInd));
end

```

9.8.4 Sensing Matrix Optimization

```

function [PHIo, Gh, Go, Gn] = optSensingMatLi(PSI, M, config)
% Based on
% G. Li, Z. Zhu, D. Yang, L. Chang, and H. Bai,
% On projection matrix optimization for compressive
% sensing systems, IEEE Transactions on Signal Processing,
% vol. 61, pp. 2887-2898, June 2013.

% INPUT:
% M: Number of measurements (m)
% PSI: Dictionary
% config: Configuration structure

%% Input Argument Parsing and Checking
% Parse Input Configuration Structure
if nargin>2
    Nbar = config.Nbar;
    normalize = config.normalize;
    V11 = config.V11;
    U = config.U;
end

[N,L] = size(PSI);

% Randomly Construct Two Orthonormal Sub-Matrices
% V11
if isempty(V11) || strcmp(V11, 'eye')
    V11 = eye(Nbar,Nbar);
elseif strcmp(V11, 'rand')
    V11 = orth(randn(Nbar));%
elseif strcmp(V11, 'V')

```

Approved for Public Release; Distribution Unlimited. PA
Case Number: 88ABW-2018-5393. Date Cleared: 25 OCT 2018

```

        V11 = V(1:Nbar,1:Nbar);
    end

    % U
    % Also could try experiment with quantizing U
    % U = rand(M); U(U<0.5) = -1; U(U>=0.5) = 1;
    if isempty(U) || strcmp(U, 'eye')
        U = eye(M,M);
    elseif strcmp(U, 'rand')
        U = orth(randn(M));
    end

    %% Find Left Singular Vectors and Singular Values of Dictionary
    [Ud, s, Vd] = svd(PSI);

    %% Invert Singular Value Diagonal Matrix
    Lambda = (s(1:Nbar,1:Nbar)).^-1;
    Lambda(Lambda==Inf) = 0;

    %% Construct Optimal Sampling Matrix
    A = [eye(M,M) zeros(M,N-M)];
    B = [transpose(V11)*Lambda zeros(Nbar,N-Nbar); ...
        zeros(N-Nbar,Nbar) zeros(N-Nbar,N-Nbar)];
    PHIo = U*A*B*Ud';

    %% Optionally Normalize
    if normalize
        PHIo = sqrt(L/M)*PHIo;
    end

    %% Analysis (for analysis & development only)
    if nargout == 2
        % Compute Optimized Gram Explicitly

        % [Ud, s, Vd] = svd(PSI);
        % G = A'*A = (PSI'*PHI')*(PHI*PSI)
        % G = (Ud*s*Vd')' * (U*A*B*Ud')' * (U*A*B*Ud') * (Ud*s*Vd');
        % G = Vd*s'*Ud' * Ud*B'*A'*U' * (U*A*B*Ud') * Ud*s*Vd'

        T1 = (Ud')*(Ud);
        % G = Vd*s'*T1*B'*A'*U' * U*A*B*T1*s*Vd'

        T2 = U'*U;
        % G = Vd*s'*T1*B'*A'*T2*A*B*T1*s*Vd'

        T3 = A'*T2*A;
        % G = Vd*s'*T1*B'*T3*B*T1*s*Vd'

        T4 = B'*T3*B;
        % G = Vd*s'*T1*T4*B*T1*s*Vd'

        T5 = T1*T4*T1;
        % G = Vd*s'*T5*s*Vd'

        T6 = s'*T5*s;

        G = Vd*T6*Vd';

        Gh = G - diag(diag(G));
    else
        Gh=0;
    end

```

```

end

% figure ; image (Gh)

% Compute Min Error Using Computed G
% Unnormalized:
A = PHIo*PSI;
Go = A'*A;% Should Equal 'G' from above
Predicted_unnorm = (L-M);
Computed_unnorm = norm(eye(L)-Go, 'fro')^2;
m = size(A,1);
r_A = rank(A);
% For A to be full rank , m and r_A should be equal.

% Normalized:
normFact = sqrt(L/M);
An = normFact*A;% Popt*PSI);
Gn = An'*An;
Predicted_norm = Predicted_unnorm + ((L-M)^2)/M;
Computed_norm = norm(eye(L)-Gn, 'fro')^2;

```

9.8.5 Simple Demo of Sensing Matrix Optimization

```

function optSampMatrixIntroDemo
close all
numiter = 1;
% Demonstrates the effect of a projection matrix on the
% Frobenius norm of the resulting effective Gramian

% Construct Dictionary with Some Linear Dependency
PSI = rand(3,4);
%PSI(:,1) = PSI(:,2)+0.5*rand(3,1);

m = 2;[N,M] = size(PSI);Nbar = 2;

[Gorig,froNormOrig] = calcGram(PSI)

PSIn = normc(PSI);

col = ['r','y','g','b'];
for iter3 = 1:M

    figure(1);% subplot(2,2,1)
    vectarrow(zeros(N,1),PSIn(:,iter3), col(iter3))
    hold on;
end

title(['Original (Normalized) Dictionary Atoms of \Psi , '...
' || I - G(\Psi) ||^2_F = ' num2str(round(froNormOrig,1))])
set(gcf, 'Position', [100, 100, 650, 300])
set(findall(gcf,'-property','FontSize'),'FontSize',11)
saveas(gcf,'proj_vector_ex_li_orig','png')
end

for iter = 1:numiter
%     delta = 1;
%     iter = iter+1;

% Random
PHI1 = randmat(m,N,'stdnorm');
Arand = normc(PHI1*PSI);
[G1,froNorm1(iter)] = calcGram(Arand);
%     froNorm1(iter) = norm(eye(size(G1))-Arand'*Arand)^2

```

```

% Optimal
%PHI2 = randmat(m,N,'stdnorm');
config = sense_matrix_opt_config(m);% Configuration
config.V11='rand';
[PHI2,~,G2] = optSensingMatLi(PSI, m, config);
A2 = (PHI2*PSI);
%[G2,froNorm2] = calcGram(A2)
%
    froNorm2(iter) = norm(eye(size(G2))-G2, 'fro')^2
%
    V11=orth(randn(Nbar));

config = sense_matrix_opt_config(m);% Configuration f
config.V11='rand';
[PHIopt,~,Gopt] = optSensingMatLi(PSI, m, config);
froNormOpt(iter) = norm(eye(size(Gopt))-Gopt, 'fro')^2
Aopt = (PHIopt*PSI);
%[Gopt,froNormOpt] = calcGram(Aopt)

A1n = normc(Arand);
A2n = normc(A2);
Aoptn = normc(Aopt);

%figure(1);subplot(2,2,2);cla;subplot(2,2,3);cla;subplot(2,2,4);cla;

for iter2 = 1:M

    figure(2);% subplot(2,2,2)
    hold on
    vectarrow(zeros(m,1),A1n(:,iter2), col(iter2))
    %
        axis([-maxComp maxComp -maxComp maxComp])
    title(['CS-RAND Effective Dictionary, A_{rand} = '...
        '\Phi_{rand}\Psi, || I - G(A_{rand}) ||^2_F = '...
        num2str(round(froNorm1,1))])
    axis square;xlim([-1.5 1.5]);ylim([-1.5 1.5])

    figure(3);% subplot(2,2,3)
    hold on
    vectarrow(zeros(m,1),A2n(:,iter2), col(iter2))
    %
        axis([-maxComp maxComp -maxComp maxComp])
    title(['CS-OPT 1 Effective Dictionary, A_{opt1} = '...
        '\Phi_{opt1}\Psi, || I - G(A_{opt1}) ||^2_F = '...
        num2str(round(froNorm2,1))])
    axis square;xlim([-1.5 1.5]);ylim([-1.5 1.5])

    figure(4);% subplot(2,2,4)
    hold on
    vectarrow(zeros(m,1),Aoptn(:,iter2), col(iter2))
    %
        axis([-maxComp maxComp -maxComp maxComp])
    title(['CS-OPT 2 Effective Dictionary, A_{opt2} = '...
        '\Phi_{opt2}\Psi, || I - G(A_{opt2}) ||^2_F = '...
        num2str(round(froNormOpt,1))])
    axis square;xlim([-1.5 1.5]);ylim([-1.5 1.5])

end
%figure(1);subplot(2,2,2)
figure(2)

set(gcf, 'Position', [100, 100, 650, 300])

```

```

set(findall(gcf,'-property','FontSize'),'FontSize',11)
saveas(gcf,'proj_vector_ex_li_rand1','png')

figure(3)
set(gcf,'Position',[100,100,650,300])
set(findall(gcf,'-property','FontSize'),'FontSize',11)
saveas(gcf,'proj_vector_ex_li_opt1','png')

figure(4)
set(gcf,'Position',[100,100,650,300])
set(findall(gcf,'-property','FontSize'),'FontSize',11)
saveas(gcf,'proj_vector_ex_li_opt2','png')

end

```

```

function [Gn,froNorm] = calcGram(A)
%      G = normc(A'*A);
An = normc(A);
Gn = abs(An'*An);
G = An'*An;
% G=G-diag(diag(G));
froNorm=norm(eye(size(G))-G,'fro');
end

```

9.8.6 Gaussian Dictionary Simulation Code

```

function optmatdemo(dictType, evolve)

%Demonstration of Optimized Sensing Matrix and Difference Operators
%
% DESCRIPTION:
%
% The purpose of this function is to study greedy support estimation
% using generic randomly generated dictionaries with respect to:
% 1) Effectiveness of optimized sensing matrices and effect of stale
%
%      dictionaries/sensing matrices
% 2) The utility of measuring the distance between an initial effective
%      dictionary and an altered or perturbed effective dictionary using
%      - Sensing matrix objective function
%      - Kolmogorov-Smirnov statistic between distributions of Gram
%      entries
%      - Kullback-Leibler divergences between distributions of Gram
%      entries

% This function performs the following tasks:
% 1) Randomly construct one of three types of *GENERIC RANDOM* dictionaries
% 2) Construct pertrub versions of the generic dictionary
% 3) Construct a random sensing matrix
% 4) Compute an optimized sensing matrix (based on Li (2013)) using the
%      original dictionary
% 5) Compute a set of 'optimized, yet stale' sensing matrices based on the
%      perturbed dictionaries
% 6) Prototype the naive implementation of the orthogonal matching pursuit
%      (OMP) algorithm for support estimation using the following sensing
%      matrices:
%      - Identity sensing matrix (sparse approximation, SA-OMP)
%      - Randomly generated compressed sensing matrix (CS-RAND)
%      - Optimized sensing matrix (CS-OPT)
%      - Stale sensing matrix (CS-OPTNOI)

```

```

% 7) Compute Gram matrices for effective dictionaries of:
%      - Original dictionary
%      - Randomly compressed dictionary
%      - Optimized compressed dictionary
%      - Stale compressed dictionary
% 8) Display plots of support estimation results and Gram matrix entries
% 9) Display plots of KS, KLD, and sensing matrix objective distances as a
%      function of support estimation performance

close all force

%% -----
% Configuration
% -----
N = 200;           % Dimension of Each Atom
M = 300;           % Number of Atoms
m = 70;200;40;     % Number of Compressed Measurements
snr_in = 2;5;8;    % Input SNR in dB
numMcIter = 1000;  % Number of Monte Carlo Iterations
numHistBins = 100; % Number of Histogram Bins
binWidth = 0.005;  % Histogram bin widths
saveFlag = 1;

% Perturbed Sensing Matrix/Dictionary Default
config_perturb = phi_psi_perturb_config;

% dictType = 'gauss'; % Generic dictionary type

% Specify Whether to Evolve the Dictionary or the Sensing Matrix Over Time
config_perturb.evolve = evolve;% 'sensing_matrix';

% Config Per Scenario Input
%eval(scenario)

%% -----
% Construct Dictionaries
% -----
if strcmp(dictType, 'gauss')
    % Gaussian Distribution—very different because it contains negative numbers
    % PSI = 1.5*randn(N,M);
    % PSI2_full = 1.5*randn(N,M);

    PSI = randn(N,M);
    PSI2_full = randn(N,M);

elseif strcmp(dictType, 'weib')
    % Weibull Distribution
    pd = makedist('Weibull');
    PSI = random(pd,N,M,1); % 10*rand(N,M);
    PSI2_full = random(pd,N,M,1);

elseif strcmp(dictType, 'unif')
    % Uniform Distribution
    PSI = rand(N,M);
    PSI2_full = rand(N,M);

```

```

elseif strcmp(dictType , 'unifquant ')
    % Quantized Uniform Distribution
    PSI = round(rand(N,M));
    PSI(PSI<0.5) = 0;
    PSI(PSI>0) = 1;

    PSI2_full = round(rand(N,M));
    PSI2_full(PSI2_full<0.5) = 0;
    PSI2_full(PSI2_full>0) = 1;

end

%%% -----
% Construct Random Sensing Matrix using Sub-Gaussian Model
% -----
PHI = randmat(m,N,'stdnorm ');
Pall.P = PHI;

%%% -----
% Optimize Sensing Matrix based on Updated Dictionary
% -----
config = sense_matrix_opt_config(m);% Configuration for Optimized Sensing Matrix

[PHIo, ~] = optSensingMatLi(PSI, m, config);
Pall.Popt = PHIo;

%%% -----
% Compute Gram of Original Dictionary
% -----
[~, ~, GAorig] = mutual_coherence(PSI);

%%% -----
% Compute Gram of Optimized Dictionary
% -----
Arand = PHI*PSI;
[~, ~, GARanh] = mutual_coherence(Arand);

%%% -----
% Compute Gram of Optimized Dictionary
% -----
Ao = PHIo*PSI;
[~, ~, GAoph] = mutual_coherence(Ao);

%%% -----
% Perturb Dictionary
% -----
[Dall] = perturbDict(PSI, config_perturb, 0, PSI2_full);

```

```

%% -----
% Optionally Evolve Sensing Matrix
% -----
if strcmp(config_perturb.evolve, 'sensing_matrix')

    [Pall.PoptNoi1, ~] = optSensingMatLi(Dall.Dnoi1, m, config);
    [Pall.PoptNoi2, ~] = optSensingMatLi(Dall.Dnoi2, m, config);
    [Pall.PoptNoi3, ~] = optSensingMatLi(Dall.Dnoi3, m, config);
    [Pall.PoptNoi4, ~] = optSensingMatLi(Dall.Dnoi4, m, config);

elseif strcmp(evolve, 'sensing_matrix_same')
    % Construct Optimized Sensing Matrices for the Correct Dictionary,
    % but vary U and V11 submatrices to show that each sensing
    % matrix is still optimal
    config.U = orth(randn(m)); config.V11 = orth(randn(config.Nbar));
    [Pall.PoptNoi1, ~] = optSensingMatLi(Dall.D, m, config);

    config.U = orth(randn(m)); config.V11 = orth(randn(config.Nbar));
    [Pall.PoptNoi2, ~] = optSensingMatLi(Dall.D, m, config);

    config.U = orth(randn(m)); config.V11 = orth(randn(config.Nbar));
    [Pall.PoptNoi3, ~] = optSensingMatLi(Dall.D, m, config);

    config.U = orth(randn(m)); config.V11 = orth(randn(config.Nbar));
    [Pall.PoptNoi4, ~] = optSensingMatLi(Dall.D, m, config);
end

%% -----
% Compute Gram and Coherence
% of Effective Dictionary Formed from the Stalest Effective Dictionary
% -----
Anoi4 = PHIo*Dall.Dnoi4;
[, ~, GAnoi4] = mutual_coherence(Anoi4);

%% Plot CDF Histogram of Stale Gram Entries
figure
subplot(2,1,1)
hcdf_noi = histogram(abs(GAnoi4(:)), numHistBins, 'Normalization', 'cdf');
title('CDF of Gram Entries for Stale Effective Dictionary')

%% Plot CDF Histogram of Optimized Gram Entries
subplot(2,1,2)
hcdf_opt = histogram(abs(GAopth(:)), numHistBins, 'Normalization', 'cdf');
title('CDF of Gram Entries for Optimized Effective Dictionary')

%% Plot CDF Line Plot of Stale & Optimized Gram Entries
figure;
subplot(2,1,1)
plot(hcdf_noi.BinEdges(2:end), hcdf_noi.Values);
xlim([0 1])
hold on;
plot(hcdf_opt.BinEdges(2:end), hcdf_opt.Values, 'g')

```

```

title('CDF of Gram Entries ')
xlim([0 1])
legend('Stale \Phi', 'Optimized \Phi')

%% Plot Survival Function Line Plot of Stale & Optimized Gram Entries
subplot(2,1,2)
plot(hcdf_noi.BinEdges(2:end), 1-hcdf_noi.Values);
hold on;
plot(hcdf_opt.BinEdges(2:end), 1-hcdf_opt.Values, 'g')
title('Survival Function (1-CDF) of Gram Entries ')
xlim([0 1])
legend('Random \Phi', 'Optimized \Phi')

%% Display Area Under Survival Functions to Command Window
area_under_surv_funct_rand = sum(1-hcdf_noi.Values)
area_under_surv_funct_opt = sum(1-hcdf_opt.Values)

%% -----
% Compute Statistics and Distances
% Between Stale and Optimized Effective Dictionaries
% -----
[gramdist] = callCompSampMat(Pall, Dall, config_perturb.evolve);

% Compute Minimum Objective Function Value
rho_min = M-m;

%% -----
% Support Estimation
% -----

% Set OMP Configuration
config_omp = set_omp_config;

% IPC
[~, ~, muCorr, numCorrectCorr] = testcorr(M, PSI, eye(N), numMcIter, ...
    config_omp, snr_in);

% SA
[~, ~, muSa, numCorrectSa] = testomp(M, PSI, eye(N), numMcIter, ...
    config_omp, snr_in);

% CS-RAND
[~, ~, muRand, numCorrectRand] = testomp(M, PSI, PHI, numMcIter, ...
    config_omp, snr_in);

% CS-OPT
[~, ~, muOpt, numCorrectOpt] = testomp(M, PSI, PHIo, numMcIter, ...
    config_omp, snr_in);

```

```

% CS-OPTNOI1
if strcmp(config_perturb.evolve,'sensing_matrix')|| ...
    strcmp(config_perturb.evolve,'sensing_matrix_same')
    [percCorrectNoi1,~,muNoi1,~]=testomp(M,PSI,Pall.PoptNoi1,...
        numMcIter,config_omp,snr_in);
elseif strcmp(config_perturb.evolve,'dictionary')
    [percCorrectNoi1,~,muNoi1,~]=testomp(M,Dall.Dnoi1,PHIo,...
        numMcIter,config_omp,snr_in);
end
percCorrectNoi(1) = percCorrectNoi1;

```

```

% CS-OPTNOI2
if strcmp(config_perturb.evolve,'sensing_matrix')|| ...
    strcmp(config_perturb.evolve,'sensing_matrix_same')
    [percCorrectNoi2,~,muNoi2,~]=testomp(M,PSI,Pall.PoptNoi2,...
        numMcIter,config_omp,snr_in);
elseif strcmp(config_perturb.evolve,'dictionary')
    [percCorrectNoi2,~,muNoi2,~]=testomp(M,Dall.Dnoi2,PHIo,...
        numMcIter,config_omp,snr_in);
end
percCorrectNoi(2) = percCorrectNoi2;

```

```

% CS-OPTNOI3
if strcmp(config_perturb.evolve,'sensing_matrix')||...
    strcmp(config_perturb.evolve,'sensing_matrix_same')
    [percCorrectNoi3,~,muNoi3,~]=testomp(M,PSI,Pall.PoptNoi3,...
        numMcIter,config_omp,snr_in);
elseif strcmp(config_perturb.evolve,'dictionary')
    [percCorrectNoi3,~,muNoi3,~]=testomp(M,Dall.Dnoi3,PHIo,...
        numMcIter,config_omp,snr_in);
end
percCorrectNoi(3) = percCorrectNoi3;

```

```

% CS-OPTNOI4
if strcmp(config_perturb.evolve,'sensing_matrix')|| ...
    strcmp(config_perturb.evolve,'sensing_matrix_same')
    [percCorrectNoi4,~,muNoi4,numCorrectNoi4]=...
        testomp(M,PSI,Pall.PoptNoi4,numMcIter,config_omp,snr_in);
elseif strcmp(config_perturb.evolve,'dictionary')
    [percCorrectNoi4,~,muNoi4,numCorrectNoi4]=...
        testomp(M,Dall.Dnoi4,PHIo,numMcIter,config_omp,snr_in);
end
percCorrectNoi(4) = percCorrectNoi4;

```

```

%%
% Plot Support Estimation Results for Generic Dictionary
%
plot_generic_dict_results(gramdist,rho_min,numCorrectCorr,...
    numCorrectSa,numCorrectRand,numCorrectOpt,numCorrectNoi4,...
    muCorr,muRand,muOpt,muSa,muNoi1,muNoi2,muNoi3,muNoi4,...
    numMcIter,m,dictType,saveFlag,config_perturb.evolve...
    ,GAorig,GAranh,GAoph,GAnoi4,binWidth);

```

```

end

```

%% ————— LOCAL FUNCTION DECLARATIONS —————

```
function [percCorrect,snr_of_x, mu, numCorrect] = testomp(M,PSI,PHI,...
    niter, config, snr_in)
% config = config.omp;
k = config.maxNumAtoms;
A = PHI*PSI;
numCorrCum=0;
N = size(PSI,1);

snr_of_x = zeros(niter,1);
for iter = 1:niter

    % Construct Input Vector to be Approximated
    % as Linear Combination of Random Atoms from PSI
    ind = randperm(M,k);
    s = zeros(M,1);
    s(ind) = rand;
    xo = PSI*s;

    % Add Noise to Input Vector
    noimag = std(xo)/(10^(snr_in/20));
    noi = noimag*randn(N,1);
    x = (xo + noi);
    snr_of_x(iter) = snr(xo,noi);

    % Take Compressive Measurement
    Y = PHI*x;

    % Recover Support via Native OMP Algorithm
    [Supp] = ompNative(Y,A,config, k);
    %[Supp] = correlate(A, Y, k);

    % Store Results
    numCorrect(iter) = length(intersect(ind,Supp));
    numCorrCum = numCorrCum + numCorrect(iter);

end
percCorrect = numCorrCum/(k*niter);
mu = round(mean(numCorrect),1);
end % testomp
```

```
function [percCorrect,snr_of_x, mu, numCorrect] =...
    testcorr(M,PSI,PHI, niter, config, snr_in)
% config = config.omp;
k = config.maxNumAtoms;
A = PHI*PSI;
numCorrCum=0;
N = size(PSI,1);

snr_of_x = zeros(niter,1);
for iter = 1:niter
```

```
    % Construct Input Vector to be Approximated
    % as Linear Combination of Random Atoms from PSI
```

```

ind = randperm(M,k);
s = zeros(M,1);
s(ind) = rand;
xo = PSI*s;

% Add Noise to Input Vector
noimag = std(xo)/(10^(snr_in/20));
noi = noimag*randn(N,1);
x = (xo + noi);
snr_of_x(iter) = snr(xo,noi);

% Take Compressive Measurement
Y = PHI*x;

% Recover Support via Inner Product Correlation Algorithm
[Supp] = correlate(A, Y, k);

% Store Results
numCorrect(iter) = length(intersect(ind, Supp));
numCorrCum = numCorrCum + numCorrect(iter);

end
percCorrect = numCorrCum/(k*niter);
mu = mean(numCorrect);
end % testomp

```

9.8.7 Sensing Matrix Evaluation

The Kolmogorov-Smirnov distance was implemented using MATLAB's *kstest2* function. The Kullback-Leibler divergence was implemented with a function that was downloaded from the Math-Works file exchange [128].

10 References

- [1] M. Kamel, W. Hamouda, and A. Youssef, "Ultra-dense networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 18, pp. 2522–2545, Fourth quarter 2016.
- [2] J. Zhu, X. She, and P. Chen, "Ultra dense networks: General introduction and design overview," in *Signal Processing for 5G: Algorithms and Implementations, First Ed.* (F. Luo and C. Zhang, eds.), ch. 20, pp. 483–508, John Wiley and Sons Press, 2016.
- [3] A. Rawat, P. Anand, H. Chen, and P. Varshney, "Collaborative spectrum sensing in the presence of byzantine attacks in cognitive radio networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 774–786, 2011.
- [4] M. Grissa, B. Hamdaoui, and A. A. Yavuz, "Location privacy in cognitive radio networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1726–1760, 2017.
- [5] M. A. Davenport, P. T. Boufounos, M. B. Wakin, and R. G. Baraniuk, "Signal processing with compressive measurements," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, pp. 445–460, April 2010.
- [6] S. K. Sharma, E. Lagunas, S. Chatzinotas, and B. Ottersten, "Application of compressive sensing in cognitive radio communications: A survey," *IEEE Communications Surveys Tutorials*, vol. 18, pp. 1838–1860, thirdquarter 2016.
- [7] B. A. Jayawickrama, E. Dutkiewicz, I. Oppermann, G. Fang, and J. Ding, "Improved performance of spectrum cartography based on compressive sensing in cognitive radio networks," in *2013 IEEE International Conference on Communications (ICC)*, pp. 5657–5661, June 2013.
- [8] B. A. Jayawickrama, E. Dutkiewicz, I. Oppermann, and M. Mueck, "Iteratively reweighted compressive sensing based algorithm for spectrum cartography in cognitive radio networks," in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 719–724, April 2014.
- [9] J. A. Bazerque and G. B. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," *IEEE Transactions on Signal Processing*, vol. 58, pp. 1847–1862, March 2010.
- [10] S. Y. Shih and K. C. Chen, "Compressed sensing construction of spectrum map for routing in cognitive radio networks," in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*, pp. 1–5, May 2011.
- [11] C. Feng, S. Valaee, and Z. Tan, "Multiple target localization using compressive sensing," in *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, pp. 1–6, Nov 2009.
- [12] G. Li, Z. Zhu, D. Yang, L. Chang, and H. Bai, "On projection matrix optimization for compressive sensing systems," *IEEE Transactions on Signal Processing*, vol. 61, pp. 2887–2898, June 2013.

- [13] J. Kelly and J. Ashdown, "Spectrum sensing falsification detection in dense cognitive radio networks using a greedy method," *IEEE National Aerospace and Electronics Conference (NAECON)*, 2018.
- [14] J. Kelly and J. Ashdown, "Method of spectrum sensing falsification detection in dense cognitive radio networks," Patent Application No. 16/157,738, 2018.
- [15] R. Baraniuk, M. A. Davenport, M. F. Duarte, and C. Hegde, "An introduction to compressive sensing," *OpenStax CNX*, Aug 2014 <http://cnx.org/> Accessed, Aug 2018.
- [16] M. Davenport, M. Duarte, Y. Eldar, and G. Kutyniok, "Introduction to compressed sensing," in *Compressed Sensing : Theory and Applications* (Y. Eldar and G. Kutyniok, eds.), ch. 1, pp. 1–64, Cambridge University Press, 2012.
- [17] M. Vetterli, J. Kovacevic, and V. Goyal, *Foundations of Signal Processing*. Cambridge University Press, 2014.
- [18] T. Moon and W. Stirling, eds., *Mathematical Methods and Algorithms for Signal Processing*. Prentice-Hall, Inc., 2000.
- [19] A. Tucker, *Linear Algebra: An Introduction to the Theory and Use of Matrices*. Macmillan Pub Co, 1993.
- [20] M. Fowler, "Lecture notes in signal transform theory, set 19 (overcomplete representations)," 2017.
- [21] J. Kovacevic and A. Chebira, "Life beyond bases: The advent of frames (part i)," *IEEE Signal Processing Magazine*, vol. 24, pp. 86–104, July 2007.
- [22] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Birkhauser Publishers, 2013.
- [23] M. Mishali and Y. C. Eldar, "From theory to practice: Sub-nyquist sampling of sparse wide-band analog signals," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, pp. 375–391, April 2010.
- [24] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proceedings of the IEEE*, vol. 98, pp. 1045–1057, June 2010.
- [25] J. M. Duarte-Carvajalino and G. Sapiro, "Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization," *IEEE Transactions on Image Processing*, vol. 18, pp. 1395–1408, July 2009.
- [26] K. Skretting and K. Engan, "Learned dictionaries for sparse image representation: properties and results," *Proc.SPIE*, vol. 14, pp. 8138 – 8138, 2011.
- [27] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society*, vol. 58, no. 1, pp. 267–268, 1996.
- [28] Z. Zhang and M. Fowler, "Lecture notes in linear algebra and engineering programming, set 16b (summary of cases for $ax=b$)," 2016.
- [29] G. Strang, *Introduction to Linear Algebra, 4th ed.* Wellesly Cambridge Press, 2009.
- [30] J. A. Tropp, "Greed is good: algorithmic results for sparse approximation," *IEEE Transactions on Information Theory*, vol. 50, pp. 2231–2242, Oct 2004.

- [31] J. Tropp, *Topics in Sparse Approximation*. PhD dissertation, University of Texas at Austin, 2004.
- [32] A. Gilbert, “Algorithms for sparse analysis lecture 1: Background on sparse approximation,” 2011 <https://www.math.ias.edu/files/wam/2011/AGlect1.pdf> (accessed 11-JAN2018).
- [33] A. Gilbert, “Algorithms for sparse analysis lecture 2: Hardness results for sparse approximation problems,” 2011 <https://www.math.ias.edu/files/wam/2011/AGlect2.pdf> (accessed 28-FEB2018).
- [34] A. Cohen, W. Dahmen, and R. DeVore, “Compressed sensing and best k-term approximation,” *J. Amer. Math. Soc.*, vol. 22, no. 1, p. 211231, 2009.
- [35] T. Blumensath and G. R. M. Davies, “Greedy algorithms for compressed sensing,” in *Compressed Sensing : Theory and Applications* (Y. Eldar and G. Kutyniok, eds.), ch. 8, pp. 348–393, Cambridge University Press, 2012.
- [36] B. Sturmfels and M. G. Christensen, “Comparison of orthogonal matching pursuit implementations,” *20th European Signal Processing Conference (EURSIPCO 2012)*, pp. 220–224, August 2012.
- [37] R. Baraniuk, S. Foucart, D. Needell, Y. Plan, and M. Wootters, “One-bit compressive sensing of dictionary-sparse signals,” *Information and Inference*, 2016.
- [38] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, “A simple proof of the restricted isometry property for random matrices,” *Journal of Constructive Approximation*, vol. 28, no. 3, pp. 253–263, 2008.
- [39] E. J. Candes, Y. C. Eldar, D. Needell, and P. Randall, “Compressed sensing with coherent and redundant dictionaries,” *Appl. Comput. Harmon. Anal.*, vol. 31, pp. 59–73, July 2011.
- [40] M. A. Davenport and M. B. Wakin, “Analysis of orthogonal matching pursuit using the restricted isometry property,” *IEEE Transactions on Information Theory*, vol. 56, pp. 4395–4401, Sept 2010.
- [41] D. Achlioptas, “Database-friendly random projections,” *Proc. ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems*, p. 274281, 2001.
- [42] Y. Chi and R. Calderbank, “Coherence-based performance guarantees of orthogonal matching pursuit,” in *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 2003–2009, Oct 2012.
- [43] M. Rosenfeld, “In praise of the gram matrix,” in *The Mathematics of Paul Erdos II* (R. Graham, J. Neetel, and S. Butler, eds.), pp. 318–323, Springer, Berlin, Germany, 1996.
- [44] L. Welch, “Lower bounds on the maximum cross correlation of signals (corresp.),” *IEEE Transactions on Information Theory*, vol. 20, pp. 397–399, May 1974.
- [45] M. Elad, “Optimized projections for compressed sensing,” *IEEE Transactions on Signal Processing*, vol. 55, pp. 5695–5702, Dec 2007.
- [46] D. Donoho and M. Elad, “Optimally sparse representation in general (non-orthogonal) dictionaries via ℓ_1 minimization,” *Proc. Nat. Acad. Sci.*, vol. 100, pp. 2197–2202, 2002.
- [47] R. Gribonval and M. Nielsen, “Sparse representations in unions of bases,” *IEEE Transactions on Information Theory*, vol. 49, pp. 3320–3325, Dec 2003.

- [48] K. Schnass and P. Vandergheynst, "Dictionary preconditioning for greedy algorithms," *IEEE Transactions on Signal Processing*, vol. 56, pp. 1994–2002, May 2008.
- [49] L. Zelnik-Manor, K. Rosenblum, and Y. C. Eldar, "Sensing matrix optimization for block-sparse decoding," *IEEE Transactions on Signal Processing*, vol. 59, pp. 4300–4312, Sept 2011.
- [50] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, pp. 3397–3415, Dec 1993.
- [51] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Transactions on Information Theory*, vol. 47, pp. 2845–2862, Nov 2001.
- [52] S. Boyd and L. Vendenberghe, *Convex Optimization*, 7th ed. Cambridge University Press, 2009.
- [53] D. Skillicorn, *Understanding Complex Datasets: Data Mining with Matrix Decompositions*. Chapman and Hall/CRC, Taylor and Francis Group, 2008.
- [54] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli, "Euclidean distance matrices: Essential theory, algorithms, and applications," *IEEE Signal Processing Magazine*, vol. 32, pp. 12–30, Nov 2015.
- [55] J. Dattoro, *Convex Optimization Euclidean Distance Geometry*, 2nd ed., Moo, v2017.09.11. Meboo Publishing USA, <http://meboo.convexoptimization.com/access.html> (accessed 19-MAR2018), 2017.
- [56] W. U. Bajwa, R. Calderbank, and S. Jafarpour, "Model selection: Two fundamental measures of coherence and their algorithmic significance," in *2010 IEEE International Symposium on Information Theory*, pp. 1568–1572, June 2010.
- [57] H. Q. Minh and V. Murino, "Covariances in computer vision and machine learning," in *Synthesis Lectures on Computer Vision* (G. Medioni and S. Dickinson, eds.), Morgan and Claypool Publishers, 2018.
- [58] D. D. S. Chen, "Basis pursuit," *Stanford University, Statistics Departement*, 1994.
- [59] MathWorks, "Matching pursuit algorithms," *Matlab R2017a Documentation Product Notes*, 2017.
- [60] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, pp. 4655–4666, Dec 2007.
- [61] S. Maymon and Y. C. Eldar, "The viterbi algorithm for subset selection," *IEEE Signal Processing Letters*, vol. 22, pp. 524–528, May 2015.
- [62] T. T. Cai and L. Wang, "Orthogonal matching pursuit for sparse signal recovery with noise," *IEEE Transactions on Information Theory*, vol. 57, pp. 4680–4688, July 2011.
- [63] D. Simon, *Optimal State Estimation*. John Wiley and Sons, Inc., 2006.
- [64] J. A. Tropp, I. S. Dhillon, R. W. Heath, and T. Strohmer, "Designing structured tight frames via an alternating projection method," *IEEE Transactions on Information Theory*, vol. 51, pp. 188–209, Jan 2005.

- [65] W. Chen, M. R. D. Rodrigues, and I. J. Wassell, "Projection design for statistical compressive sensing: A tight frame based approach," *IEEE Transactions on Signal Processing*, vol. 61, pp. 2016–2029, April 2013.
- [66] J. Kovacevic and A. Chebira, "Life beyond bases: The advent of frames (part ii)," *IEEE Signal Processing Magazine*, vol. 24, pp. 115–125, Sept 2007.
- [67] V. Abolghasemi, S. Ferdowsi, and S. Sanei, "A gradient-based alternating minimization approach for optimization of the measurement matrix in compressive sensing," *Signal Processing*, vol. 92, pp. 999–1009, 2012.
- [68] G. Golub and C. V. Loan, eds., *Matrix Computations, Third Ed.* Johns Hopkins University Press, 1996.
- [69] P. Kvam and B. Vidakovic, eds., *Nonparametric Statistics with Applications to Science and Engineering.* John Wiley and Sons, Inc., 2007.
- [70] S. Bonnabel and R. Sepulchre, "Riemannian Metric and Geometric Mean for Positive Semidefinite Matrices of Fixed Rank," *ArXiv e-prints*, July 2008.
- [71] C. Giusti, E. Pastalkova, C. Curto, and V. Itskov, "Cliques topology reveals intrinsic geometric structure in neural correlations," *Proceedings of the National Academy of Sciences*, vol. 112, no. 44, pp. 13455–13460, 2015.
- [72] C. Giusti, R. Ghrist, and D. Bassett, "Twos company, three (or more) is a simplex: Algebraic-topological tools for understanding higher-order structure in neural data," *Journal of Computational Neuroscience*, vol. 41, no. 1, pp. 1–14, 2016.
- [73] A. Basu, H. Shioya, and C. Park, *Statistical Inference: The Minimum Distance Approach.* CRC Press, Taylor and Francis Group, 2011.
- [74] T. Cover and J. Thomas, *Elements of Information Theory.* John Wiley and Sons, Inc., 2006.
- [75] F. Luo and C. Zhang, eds., *Signal Processing for 5G: Algorithms and Implementations, First Ed.* John Wiley and Sons Press, 2016.
- [76] S. S. Company, "General survey of radio frequency bands 30 mhz to 3ghz, version 2.0," 2010 <http://www.sharespectrum.com/> Accessed 12-APR2018.
- [77] R. K. Sharma and D. B. Rawat, "Advances on security threats and countermeasures for cognitive radio networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 17, pp. 1023–1043, Secondquarter 2015.
- [78] D. Rawat, M. Song, and S. Shetty, *Dynamic Spectrum Access for Wireless Networks.* Spring Publishing, 2015.
- [79] M. Fowler, "Lecture notes in digital signal processing, set vi-1 (introduction and periodogram)," 2018.
- [80] J. Proakis and D. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications.* Pearson Education, Inc., 2007.
- [81] Z. Gao, H. Zhu, Y. Liu, M. Li, and Z. Cao, "Location privacy in database-driven cognitive radio networks: Attacks and countermeasures," in *2013 Proceedings IEEE INFOCOM*, pp. 2751–2759, April 2013.

- [82] R. Campos and L. Lovisolo, "Rf fingerprinting location techniques," in *Handbook of Position Location: Theory, Practice, and Advances* (S. Zekavat and R. Buehrer, eds.), ch. 15, pp. 487–520, John Wiley and Sons Publishers, 2012.
- [83] K. J. and W. Yeo, "A coherent data filtering method for large scale rf fingerprint wi-fi positioning systems," *EURASIP Journal on Wireless Communications and Networking*, vol. 1, no. 13, 2014.
- [84] S. U. Rehman, K. W. Sowerby, and C. Coghill, "Radio-frequency fingerprinting for mitigating primary user emulation attack in low-end cognitive radios," *IET Communications*, vol. 8, pp. 1274–1284, May 2014.
- [85] E. Arias-Castro and Y. Eldar, "Noise folding in compressed sensing," *IEEE Signal Processing Letters*, vol. 18, no. 8, 2011.
- [86] S. Peter, M. Artina, and M. Fornasier, "Damping noise-folding and enhanced support recovery in compressed sensing," *IEEE Transactions on Signal Processing*, vol. 63, pp. 5990–6002, Nov 2015.
- [87] M. A. Davenport, J. N. Laska, J. R. Treichler, and R. G. Baraniuk, "The pros and cons of compressive sensing for wideband signal acquisition: Noise folding versus dynamic range," *IEEE Transactions on Signal Processing*, vol. 60, pp. 4628–4642, Sept 2012.
- [88] E. J. Candes, "The restricted isometry property and its implications for compressed sensing," *Comptes Rendus Mathematique*, vol. 346, pp. 589–592, May 2008.
- [89] S. Li and G. Ge, "Deterministic sensing matrices arising from near orthogonal systems," *IEEE Transactions on Information Theory*, vol. 60, pp. 2291–2302, April 2014.
- [90] A. S. Bandeira, E. Dobriban, D. G. Mixon, and W. F. Sawin, "Certifying the restricted isometry property is hard," *IEEE Transactions on Information Theory*, vol. 59, pp. 3448–3450, June 2013.
- [91] C. Haohuang, "What is an example of non-convex cone?," <https://math.stackexchange.com/questions/345862/what-is-an-example-of-non-convex-cone> (accessed 19-MAR2018).
- [92] S. Bhattarai, J. M. J. Park, B. Gao, K. Bian, and W. Lehr, "An overview of dynamic spectrum sharing: Ongoing initiatives, challenges, and a roadmap for future research," *IEEE Transactions on Cognitive Communications and Networking*, vol. 2, pp. 110–128, June 2016.
- [93] P. Tenhula, "Regulatory framework(s) for facilitating new spectrum sharing schemes," *Proc. Int. Symp. Adv. Radio Technology*, pp. 142–145, 2012.
- [94] I. Nokia, "Ultra Dense Network White Paper," tech. rep., Nokia, Inc., 2016.
- [95] A. A. S. Stefanatos, A. Gotsis, "Effects of densification and randomness of infrastructure deployment in cellular networks," in *5G Wireless Technologies* (A. Alexiou, ed.), ch. 5, p. 147, The Institution of Engineering and Technology, London, United Kingdom, 2013.
- [96] X. Ge, S. Tu, G. Mao, C. X. Wang, and T. Han, "5g ultra-dense cellular networks," *IEEE Wireless Communications*, vol. 23, pp. 72–79, February 2016.
- [97] Q. Zhao and B. M. Sadler, "A survey of dynamic spectrum access," *IEEE Signal Processing Magazine*, vol. 24, pp. 79–89, May 2007.

- [98] I. T. U. (ITU), “Definitions of software defined radio (sdr) and cognitive radio systems (crs), geneva, switzerland,” 2009.
- [99] O. Fatemieh, R. Chandra, and C. A. Gunter, “Secure collaborative sensing for crowd sourcing spectrum data in white space networks,” in *2010 IEEE Symposium on New Frontiers in Dynamic Spectrum (DySPAN)*, pp. 1–12, April 2010.
- [100] J. Schmidt and P. Stanforth, “Spectrum sharing using geo-location databases,” in *Opportunistic Spectrum Sharing and White Space Access: The Practical Reality* (O. Holland, H. Bogucka, and A. Medeisis, eds.), ch. 15, pp. 339–367, John Wiley and Sons, Inc., 2015.
- [101] Y. Zhao, B. Le, and J. Reed, “Network support: The radio environment,” in *Cognitive Radio Technology* (B. Fette, ed.), ch. 11, pp. 337–364, Elseiver, Inc., 2006.
- [102] R. Murty, R. Chandra, T. Moscibroda, and P. Bahl, “Senseless: A database-driven white spaces network,” *IEEE Transactions on Mobile Computing*, vol. 11, pp. 189–203, Feb 2012.
- [103] N. Wang, Y. Gao, and B. Evans, “Database-augmented spectrum sensing algorithm for cognitive radio,” in *2015 IEEE International Conference on Communications (ICC)*, pp. 7468–7473, June 2015.
- [104] E. Troja and S. Bakiras, “Efficient location privacy for moving clients in database-driven dynamic spectrum access,” in *2015 24th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–8, Aug 2015.
- [105] L. Zhang, C. Fang, Y. Li, H. Zhu, and M. Dong, “Optimal strategies for defending location inference attack in database-driven crns,” in *2015 IEEE International Conference on Communications (ICC)*, pp. 7640–7645, June 2015.
- [106] B. Bahrak, S. Bhattarai, A. Ullah, J. M. J. Park, J. Reed, and D. Gurney, “Protecting the primary users’ operational privacy in spectrum sharing,” in *2014 IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN)*, pp. 236–247, April 2014.
- [107] M. Barrie, S. Delaere, G. Sukareviiien, J. Gesquiere, and I. Moerman, “Geolocation database beyond tv white spaces? matching applications with database requirements,” in *2012 IEEE International Symposium on Dynamic Spectrum Access Networks*, pp. 467–478, Oct 2012.
- [108] Y. Luo, L. Gao, and J. Huang, “Spectrum broker by geo-location database,” in *2012 IEEE Global Communications Conference (GLOBECOM)*, pp. 5427–5432, Dec 2012.
- [109] M. Fitch, “Security,” in *Spectrum Sharing in Wireless Networks : Fairness, Efficiency, and Security* (F. H. J. Matyjas, S. Kumar, ed.), ch. 24, pp. 583–604, CRC Press, Inc., 2016.
- [110] L. Zhang, Q. Wu, Z. Han, and J. Wang, “Byzantine attack and defense in cognitive radio networks: A survey,” *IEEE Communications Surveys and Tutorials*, vol. 17, no. 3, pp. 1342–1363, 2015.
- [111] R. Pietro and J. Domingo-Ferrer, “Security in wireless mobile ad hoc networks,” in *Mobile Ad Hoc Networks: Cutting Edge Directions, 2nd Ed.* (S. Basangi, M. Conti, S. Giordano, and I. Stojmenovic, eds.), ch. 4, pp. 106–153, John Wiley and Sons Publishers, 2013.
- [112] S. Bhattarai, P. R. Vaka, and J. M. J. Park, “Thwarting location inference attacks in database-driven spectrum sharing,” *IEEE Transactions on Cognitive Communications and Networking–In Review*, 2017.

- [113] G. Baldini, T. Sturman, A. Biswas, R. Leschhorn, G. Godor, and M. Street, "Security aspects in software defined radio and cognitive radio networks: A survey and a way ahead," *IEEE Communications Surveys Tutorials*, vol. 14, pp. 355–379, Feb 2012.
- [114] Z. Zhang, H. Zhang, S. He, and P. Cheng, "Achieving bilateral utility maximization and location privacy preservation in database-driven cognitive radio networks," in *2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems*, pp. 181–189, Oct 2015.
- [115] T. C. Clancy and N. Goergen, "Security in cognitive radio networks: Threats and mitigation," in *2008 3rd International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CrownCom 2008)*, pp. 1–8, May 2008.
- [116] P. S. Chatterjee and M. Roy, "A regression based spectrum-sensing data - falsification attack detection technique in cwsn," in *2015 International Conference on Information Technology (ICIT)*, pp. 48–53, Dec 2015.
- [117] R. Chen, J. M. Park, and K. Bian, "Robust distributed spectrum sensing in cognitive radio networks," in *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, April 2008.
- [118] S. Jana, K. Zeng, W. Cheng, and P. Mohapatra, "Trusted collaborative spectrum sensing for mobile cognitive radio networks," *IEEE Transactions on Information Forensics and Security*, vol. 8, pp. 1497–1507, Sept 2013.
- [119] A. W. Min, K. G. Shin, and X. Hu, "Attack-tolerant distributed sensing for dynamic spectrum access networks," in *2009 17th IEEE International Conference on Network Protocols*, pp. 294–303, Oct 2009.
- [120] A. W. Min, K. G. Shin, and X. Hu, "Secure cooperative sensing in ieee 802.22 wrans using shadow fading correlation," *IEEE Transactions on Mobile Computing*, vol. 10, pp. 1434–1447, Oct 2011.
- [121] K. Zeng, P. Pawelczak, and D. Cabric, "Reputation-based cooperative spectrum sensing with trusted nodes assistance," *IEEE Communications Letters*, vol. 14, pp. 226–228, March 2010.
- [122] L. Wang, L. Zhang, and X. Chen, "A dynamic threshold strategy against ssdf attack for cooperative spectrum sensing in cognitive radio networks," in *2015 International Conference on Wireless Communications Signal Processing (WCSP)*, pp. 1–5, Oct 2015.
- [123] Y. Al-Mathehaji, S. Boussakta, M. Johnston, and H. Fakhrey, "Defeating ssdf attacks with trusted nodes assistance in cognitive radio networks," *IEEE Sensors Letters*, vol. 1, pp. 1–4, Aug 2017.
- [124] Z. Chen, L. Chen, and H. Zhong, "Towards secure and verifiable database-driven spectrum sharing," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 285–296, June 2017.
- [125] M. Guarnieri, S. Marinovic, and D. Basin, "Securing databases from probabilistic inference," in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pp. 343–359, Aug 2017.
- [126] J. Lee and R. Buehrer, "Security issues for position location," in *Handbook of Position Location: Theory, Practice, and Advances* (S. Zekavat and R. Buehrer, eds.), ch. 3, pp. 67–104, John Wiley and Sons Publishers, 2012.

- [127] N. Krislock and H. Wolkowicz, "Euclidean distance matrices and applications," *Handbook on Semidefinite, Cone and Polynomial Optimization, in International Series in Operations Research and Management Science*, no. 2009-06, pp. 879–914, 2011.
- [128] D. Fass, "Kullback-leibler divergence matlab code," 2016 <https://www.mathworks.com/matlabcentral/fileexchange/13089-kldiv> (accessed JAN2018).

List of Acronyms

Acronyms	Meaning
AWGN	Additive White Gaussian Noise
BP	Basis Pursuit
CA	Coverage Area (of an RF Network)
CS	Compressive Sensing
CS-RAND	Compressive Sensing using a Randomly Generated Sensing Matrix
CS-OPT	Compressive Sensing using an Optimized Sensing Matrix
CS-OPTNOI	Compressive Sensing using a Stale Sensing Matrix
CR	Cognitive Radio
CRN	Cognitive Radio Network
CSRD	Compressive Sensing in a Redundant Dictionary
CSU	Candidate Secondary User
CSS	Collaborative Spectrum Sensing
DSA	Dynamic Spectrum Access
GDB	Geographic Database
GSA	Greedy Spectrum Approximation
GSV	Greedy Spectrum Verification
IPC	Inner Product Correlation
KLD	Kullback-Leibler Divergence
KS	Kolmogorov-Smirnov Statistic
LS	Least Squares
MP	Matching Pursuit
NSP	Null Space Property
OLS	Ordinary Least Squares
OMP	Orthogonal Matching Pursuit

PSD	Power Spectral Density or Positive Semi-Definite (Matrix)
REM	Radio Environment Map
RF	Radio Frequency
RIP	Restricted Isometry Property
SA	Sparse Approximation
SA-OMP	Sparse Approximation using Orthogonal Matching Pursuit
SAS	Spectrum Access System
SME	Sensing Matrix Evaluation
SMO	Sensing Matrix Optimization
SN	Sensor Node
SNR	Signal to Noise Ratio
SU	Secondary User
SSDF	Spectrum Sensing Data Falsification
SVD	Singular Value Decomposition
TC-xx	Test Case xx