Computationally Efficient and Secure Kronecker-based Compressive Sensing

by

Parichehreh Firoozi, **B.Sc.**

A thesis submitted to the
Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

Master of Applied Science in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario
September, 2021

Abstract

We propose an efficient permuted Kronecker-based sparse measurement matrix for compressive sensing applications. We use sub-matrices to create a block-diagonal matrix and multiply it with a deterministic permutation matrix to measure the sparse or compressible signals. Using ECG signals from the MIT-BIH Arrhythmia database, we show that the reconstructed signal quality is comparable to the ones achieved using standard compressive sensing methods. Our methodology results in an overall reduction in storage and computations and can be generalized to other classes of eligible measurement matrices in compressive sensing. We show that with the use of a securely generated one-time sensing matrix, our proposed method is computationally secure against plaintext and ciphertext-only attacks. The proposed one-time sensing matrix is superior to other measurement matrices in the literature in terms of the number of linear feedback shift register bits required for their generation.

Acknowledgments

I would like to express my sincere gratitude to my supervisor's Professor Sreeraman Rajan and Professor Ioannis Lambadaris for their invaluable guidance and continuous support. My deepest appreciation to both of them for their warm presence, kind support, and encouragement during the challenging time I have had throughout my master's program. It was a great honor to work and study under their supervision.

My special thanks to my husband, Nima Palizban, for the time he spent proofreading this thesis.

It is my privilege to thank Carleton University, the Department of Systems and Computer Engineering, for providing a professional academic environment to experience, learn and enjoy.

I would like to acknowledge the National Sciences and Engineering Research Council (NSERC) of Canada for their financial support during this work and Ericsson Canada Inc for offering me an internship opportunity during Summer 2021.

Finally, I would like to thank my family for their love, patience, and lifelong support.

Table of Contents

List of Tables

List of Figures

Nomenclature

Acronym	Meaning
BCH	Bose Chaudhuri Hocquenghem
CLT	Central Limit Theorem
COA	Ciphertext Only Attack
CR	Compression Ratio
CS	Compressive Sensing
CSC	Compressive Sensing Cryptosystem
DCT	Discrete Cosine Transform
ECG	Electrocardiogram
GRNG	Gaussian Random Number Generator
IoT	Internet of Thing
LFSR	Linear Feedback Shift Register
MRI	Magnetic Resonance Imaging
NP	Nondeterministic Polynomial
NSP	Null Space Property

Acronym	Meaning
OTS	One Time Sensing
PA	Plaintext Attack
PRNG	Pseudo Random Number Generator
RADAR	Radio Detecting and Ranging
RIP	Restricted Isometry Propert
SBCM	Sparse Block Circulant Matrix
SG	Shrinking Generator
SNR	Signal to Noise Ratio
SR	Sparsity Ratio
SSG	Self-Shrinking Generator
S-OTS	Sparse One-Time Sensing
TRNG	True Random Number Generator

Chapter 1

Introduction

Compressive sensing (CS) measures and compresses signals that are sparse or compressible in a single step. CS has attracted considerable attention in several areas such as signal processing, audio and video processing, medical imaging, network, communication, information security. This chapter presents a review of CS theory and few fundamentals needed to understand one of the applications of CS, namely Compressive Sensing Cryptosystem (CSC).

1.1 Compressive Sensing

Following the advances in technology, the amount of data generated by sensing systems has increased dramatically. The process of data acquisition may be costly or physically impossible to build devices capable of acquiring samples at the necessary Nyquist rate. Thus, despite extraordinary advances in computational power, the acquisition and processing of signals in application areas such as imaging, video, and remote surveillance continue to pose a tremendous challenge [1]. Conventional approaches to sampling signals follow Shannon's theorem which requires sampling rate to be at least twice the maximum frequency present in the signal (the so-called Nyquist rate) [2]. However, in many applications, most of the sensed data can be thrown away with almost no perceptual loss [3]. Therefore, compression becomes a necessity prior to storage or transmission [4]. Compressive Sensing (CS) has been introduced out of the work of Emmanuel Candès, Justin Romberg, Terence Tao, and David Donoho, who showed that a finite-dimensional signal having a sparse or compressible representation can be simultaneously sensed and compressed. Figure 1.1 compares traditional

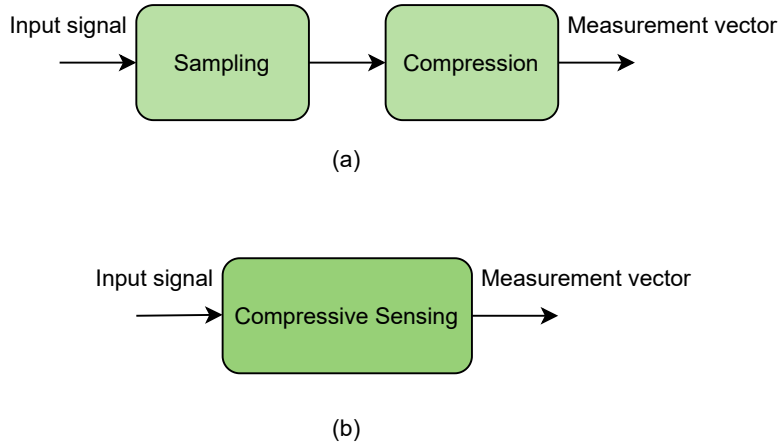

Figure 1.1: (a) Traditional method for sampling and compression, (b) compressive sensing.

method and CS method. The sensed signal can be later recovered from a small set of linear measurements [2], [3].

1.1.1 Sparse Signals

Signal x with length N is k-sparse if it has at most k non-zero elements where $k \ll N$. Mathematically

$$||x||_0 \leq k \tag{1.1}$$

where $||x||_0 := |supp(x)|$ is the number of non-zero elements of signal x, i.e., $supp(x) = \{i : x_i \neq 0\}$.

1.1.2 Compressible Signals

In nature, very rarely signals are truly sparse. Instead, they are compressible, meaning that they can be well-approximated by the first few principal components. In the other word, a signal is compressible if its sorted coefficients' magnitude in a certain domain $\mathbf{\Psi}$ (for instance frequency domain) decay rapidly. Figure 1.2 shows an example of an ECG signal and its sorted coefficients' magnitude in the Discrete Cosine Transform (DCT) domain.

A time domain signal can be transformed to have a sparse representation in the

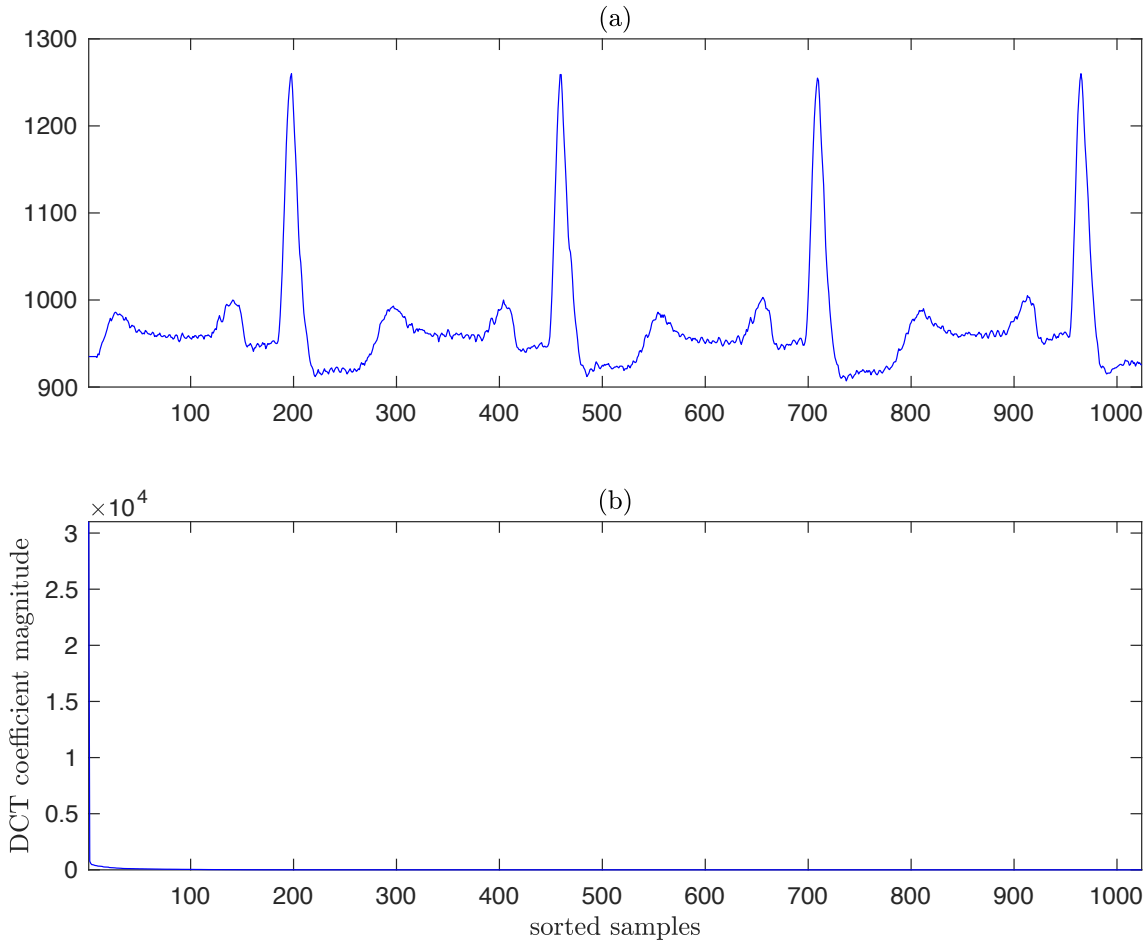

Figure 1.2: (a) An ECG signal of length $N = 1024$, (b) absolute value of sorted coefficients of an ECG signal in DCT domain.

following manner:

$$x = \mathbf{\Psi} s \tag{1.2}$$

where $s \in \mathbb{R}^{N\times 1}$ is the time domain signal, $\mathbf{\Psi} \in \mathbb{R}^{N\times N}$ known as sparsifying/transform basis, and $x \in \mathbb{R}^{N\times 1}$ is the sparse signal. Generally in the area of compressive sensing, $\mathbf{\Psi}$ is orthonormal and is termed as sparsification matrix.

1.1.3 Measurement Matrix

For simplicity, here we assume x is a finite length signals which has been sampled at Nyquist rate. The dimensionality of such a signal is reduced through a linear

transform using measurement matrix $\mathbf{\Phi}$ as follows:

$$y = \mathbf{\Phi}x \tag{1.3}$$

where $y \in \mathbb{R}^{M\times 1}$ represents the compressed measurement vector, $\mathbf{\Phi} \in \mathbb{R}^{M\times N}$ is the measurement matrix. The matrix $\mathbf{\Phi}$ maps $\mathbb{R}^N$ to $\mathbb{R}^M$ where N is generally larger than M, $M < N$. We can define the compression ratio (CR) as

$$CR = 1 - \frac{M}{N}. \tag{1.4}$$

An appropriate design of the measurement matrix ensures that the recovery of x from measurement vector y is possible. To have a perfect recovery, $\mathbf{\Phi}$ should have the Null Space Property (NSP). A stricter condition, Restricted Isometry Property (RIP), is required for recovery in the presence of noise.

Null Space Property

Null space of the measurement matrix $\mathbf{\Phi}$ is denoted by

$$\mathcal{N}(\mathbf{\Phi}) = \{z : \mathbf{\Phi}z = 0\}. \tag{1.5}$$

If we want to recover all k-sparse signals from measurement vectors y, then for any distinct k-sparse signals x and x', we must have $\mathbf{\Phi}x \neq \mathbf{\Phi}x'$. $\mathbf{\Phi}$ uniquely represents all k-sparse signals if and only if $\mathcal{N}(\mathbf{\Phi})$ do not contain signals which are $2k$-sparse. To characterize this property, we can use the definition of spark [1].

Definition 1. The spark of a given matrix $\mathbf{\Phi}$ is the smallest number of columns of $\mathbf{\Phi}$ that are linearly dependent.

Theorem 1. For any vector $y \in \mathbb{R}^M$, there exists at most one k-sparse signal such that $y = \mathbf{\Phi}x$ if and only if spark($\mathbf{\Phi}$) > 2K.

NSP guarantees the recovery of noiseless signal. However, in presence of noise a stronger condition is required.

Restricted Isometry Property

When the measurements are contaminated with noise or have been corrupted by some errors such as quantization. Restricted isometry condition on matrices $\mathbf{\Phi}$ is defined

as follows [1]:

Definition 2. Matrix $\mathbf{\Phi}$ satisfies the restricted isometry property (RIP) of order k if there exists a $\delta_k \in (0, 1)$ such that

$$(1 - \delta_k)\|x\|_2^2 \leq \|\mathbf{\Phi}x\|_2^2 \leq (1 + \delta_k)\|x\|_2^2 \tag{1.6}$$

Types of Measurement Matrices

Matrices that satisfy RIP can be constructed randomly or deterministically.

Random matrix $\mathbf{\Phi}$ can be constructed by choosing the entries ϕ_{ij} as independent and identical (i.i.d) realizations from some probability distribution. Two conditions are required on random distributions to ensure that they satisfy RIP. First, $\mathbf{\Phi}$ should be norm preserving which requires that the variance of distribution be $\frac{1}{M}$, i.e.

$$\mathbb{E}(\phi_{ij}^2) = \frac{1}{M} \tag{1.7}$$

Second, the distribution should be sub-Gaussian, i.e., the tails of distribution decay at least as fast as the tails of a Gaussian distribution. Bernoulli distributions is an example of sub-Gaussian distribution. An advantage of random matrix such as i.i.d Gaussian matrix is that, it satisfies RIP condition with high probability [4]. However, using random matrices have some disadvantages as well. First, generating fully random matrices is sometimes impractical in hardware and in real-world applications. Usually, we use a reduced amount of randomness in our design which also satisfy RIP [5], [6], [7]. Second, random matrices requires a lot of storage. Also, there is no efficient algorithm verifying the RIP condition of a random matrix [8].

Deterministic measurement matrices such as chirp sensing matrix, the second-order Reed-Muller sensing matrix and the Binary BCH-matrix have been proposed in the literature. Deterministic measurement matrices provide several advantages compared to random measurement matrices. For instance, the second order Reed-Muller matrix [9] is easy to construct and deploy as it a bipolar deterministic matrix. The binary BCH-based matrix simplifies the sampling process and reduces the computational complexity [10]. When compressive sensing systems are designed using the chirp sensing matrix [11], low complexity fast reconstruction algorithms are possible.

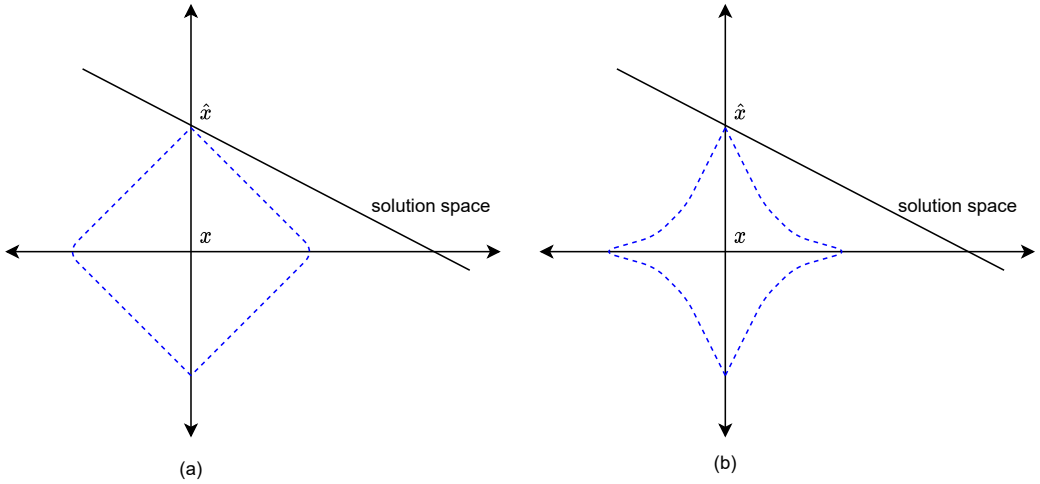

Figure 1.3: Solution point in $\mathbb{R}^2$ by a one-dimensional subspace (a) l_1 norm (b) l_p quasi-norm and $p = 1/2$.

1.1.4 Compresssive Sensing Recovery

After obtaining the compressively sensed signal y, the original sparse signal x can be recovered. Recovery can be formulated as the problem of finding the sparsest signal that satisfies $y = \mathbf{\Phi}x$ and can be formulated as follows:

$$\hat{x} = argmin\|z\|_0 \quad \text{subject } to \quad z \in \{z : \mathbf{\Phi}z = y\}, \tag{1.8}$$

where $\hat{x}$ is the recovered signal, $\|z\|_0$ is l_0 norm or the number of non-zero entries of z, and z is in the set of candidate k-sparse signals [1]. The objective function $\|z\|_0$ is non-convex, and equation (1.8) is difficult to solve. By replacing $\|z\|_0$ with its convex relaxation $\|z\|_1$, a solution may be obtained. The reformulation of the problem is as follows:

$$\hat{x} = argmin\|z\|_1 \quad \text{subject } to \quad z \in \{z : \mathbf{\Phi}z = y\}, \tag{1.9}$$

where $\|z\|_1$ is l_1 norm or summation of absolute values of entries of z. Problem (1.9) can be posed as a linear program. l_1 minimization will provide an accurate method for sparse signal recovery and the solution to (1.9) is the same as (1.8). Figure 1.3 graphically shows the solution point in $\mathbb{R}^2$ by a one-dimensional subspace that satisfies the constraints, for l_1 norm and the l_p quasi-norm where $0 \leq p < 1$ ($p = 1/2$ in this figure) [4].

There exist a variety of algorithms to solve minimization problems given by equation (1.8) and equation (1.9). The design of recovery algorithms are evaluated by various criteria such as the number of measurements, robustness to measurement noise, speed and performance guarantees [12]. Some of the recovery algorithms are convex relaxation algorithm [13–17], non-convex minimization algorithm [18–20], greedy iterative algorithm [21–23], combinatorial algorithm [24–26], and iterative thresholding algorithm [27–29].

In this thesis, we use a method proposed in [30] to convert equation (1.9) to a computationally feasible linear problem by defining two vectors $u = max(z, 0)$ and $v = max(-z, 0)$ and a vector $z = u - v$. Note that $\|z\|_1 = \sum_{i=1}^{N} |z_i|$ and the equation (1.9) is converted to the following problem:

$$\hat{x} = argmin \sum_{i=1}^{N} (u_i + v_i) \quad \text{s.t } \mathbf{\Phi}(u - v) = y \ \& \ u, v > 0.$$

Subsequently, we will use the aforementioned linear problem as the recovery problem to find $\hat{x}$. We use a powerful mathematical programming solver available for linear programming problems called Gurobi Optimizer [31].

1.1.5 Applications of Compressive Sensing

CS has a wide variety of applications which some of them are single pixel camera [32], medical imaging, particularly in Magnetic Resonance Imaging (MRI) [33], Radio Detecting and Ranging (RADAR) [34], analog to information conversion [35], communication systems [36], and applications in security. We discuss security aspects of CS in Section 1.3. In the following sections, we first present a review of the basics of cryptography and then provide a review of CSC.

1.2 Basics of Cryptography

Cryptography is the science of secret writing with the goal of hiding the meaning of a message. The science or the art of breaking a cryptosystem is called cryptanalysis. In the following sections, we first briefly introduce cryptography's terminologies. We then follow it with an introduction to random number generators and linear feedback shift registers which are used in Chapter 4.

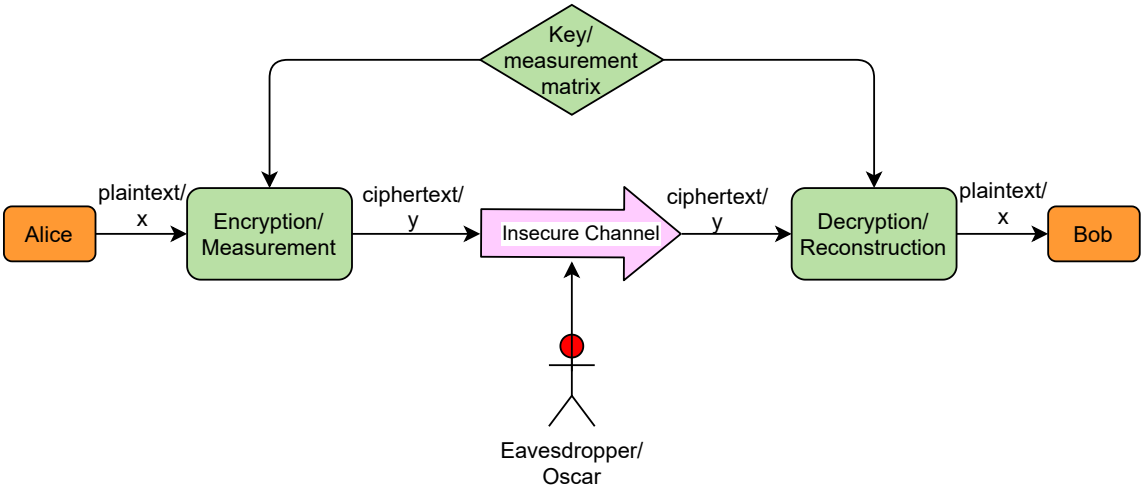

Figure 1.4: Block diagram of a cryptosystem.

1.2.1 Terminology

Suppose there are two users, Alice and Bob, who want to communicate through an insecure channel. An eavesdropper called Oscar has access to an insecure channel and can listen to their communication. Alice and Bob are communicating some important information that should not be revealed to Oscar. In this situation, symmetric cryptography offers a powerful solution. Alice encrypts her message x using a symmetric algorithm and a key, then sends the ciphertext y. Bob receives the ciphertext and since he has the same key as Alice, he is able to decrypt the message. Figure (1.4) summarizes these terminologies. If the encryption algorithm is strong enough, the ciphertext will look like random bits to the eavesdropper and contains no useful information. In symmetric cryptography, we need a secure channel for the distribution of the key between Alice and Bob. In practical systems there exist a public cryptography procedure, prior to data transmission, to establish the secret key. The key has to be transmitted once and then can be used for securing many subsequent communications. Keeping the encryption algorithm secret makes the whole system harder to break. However, the only way to find out whether an encryption method is strong, i.e. can not be broken by an attacker, is to make it public and have it analyzed by other cryptographers. **Kerckhoffs' principle**: A cryptosystem should be secure even if the attacker knows all details about the system, with the exception of the secret key. In particular, the system should be secure when the attacker knows the encryption and decryption algorithms [37]. Two common types of attacks in cryptography are

discussed in the following paragraphs.

A brute force attack (also known as exhaustive key search) is a type of attack in which the attacker, Oscar, has the ciphertext from listening to the insecure channel and may also have a short piece of plaintext like the header of a file that was encrypted. Let (x, y) denote the pair of plaintext and ciphertext and let $K = \{k_1, ..., k_n\}$ be the key space of all possible keys k_i. A brute force attack checks for every $k_i \in K$. If a possible correct key is found then it stops, if not, it proceeds with the next key.

$$\mathbb{D}_{k_i}(y) = x$$

where $\mathbb{D}$ denotes the decryption operation. If the size of key space is large enough, the encryption scheme is resistant to brute force attacks. How many key bits are required? We can check Table 1.1 for an estimate [37].

Table 1.1: Estimated time for successful brute force attacks on symmetric cryptosystem

Key length (bits)	Security estimation
56 - 64	Short term: a few hours or days using regular computers
112 - 128	Long term: several decades in the absence of quantum computers
256	Long term: several decades even with quantum computers

Of course, we cannot predict the future in terms of new technical and theoretical developments with certainty. However, roughly speaking, **Moore's law** states that computing power doubles every 18 months while the costs stay constant. Note that, a large key space alone is not sufficient for a strong encryption function [37].

An analytic cryptographic attack is a mathematical manipulation that attempts to reduce the complexity of the cryptographic algorithm. If this attack is successful, the attacker can quickly deduce how the plaintext is converted to the ciphertext. Good ciphers should hide the statistical properties of the encrypted plaintext. So, it cannot easily be broken by analytical attacks [37].

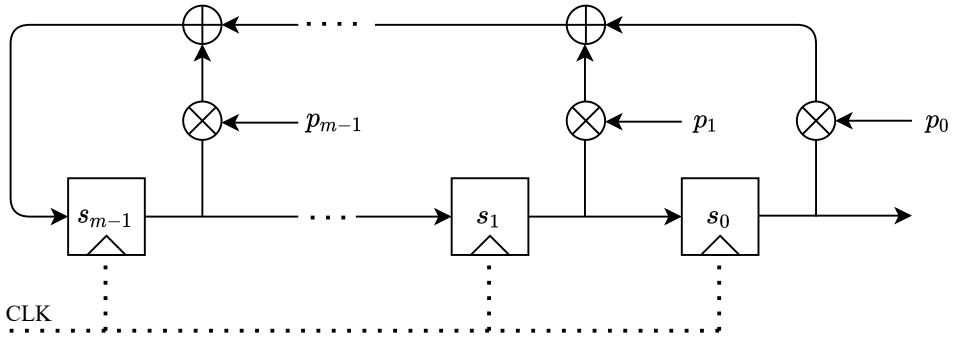

Figure 1.5: An LFSR of degree m with initial values $s_0, ..., s_{m-1}$ and feedback coefficients $p_0, ..., p_{m-1}$.

1.2.2 Random Number Generators

In cryptography, a key stream is a stream of random or pseudo-random characters that are combined with the plaintext to produce the ciphertext. Therefore, the generation of the key stream is the main issue for the security of the cipher. A basic requirement of a key stream bit is that it should appear as a random sequence to an attacker. Otherwise, the attacker can guess the key bits and do the decryption. In this section, we review more about random numbers and how to generate them [37].

True Random Number Generators (TRNGs) generate outputs which can not be reproduced. i.e., in the example of flipping a coin 100 times and recording the resulting sequence of 100 bits, it is practically impossible to generate the same 100 bit sequence. In this case the probability of success is $1/2^{100}$ which is very small.

Pseudo Random Number Generators (PRNGs) generate outputs which are computed recursively from an initial seed in the following way:

$$s_0 = \text{seed}, \qquad s_{i+1} = f(s_i), \qquad i = 0, 1, ... \tag{1.10}$$

In the above equation, s_{i+1} is a function of s_i and f is the function operator. PRNGs posse good statistical properties in the sense that their output approximates a sequence of true random numbers [37]. Shift registers with feedback is an example of PRNGs which can easily be realized in hardware and is discussed in the next section.

Linear Feedback Shift Registers

A practical way of realizing long pseudo random sequences is to use Linear Feedback Shift Registers (LFSRs) which can easily be implemented in hardware. An LFSR consists of clock storage elements (flip-flops) and a feedback path. The number of flip-flops determines the degree of LFSR. Therefore, an LFSR with m flip-flops is of degree m. An example of an LFSR of degree m is shown in Figure 1.5. The possible m feedback of flip-flops which can be active or not, are combined by XOR operation. According to feedback coefficients, $p_0, p_1, ..., p_{m-1}$, a feedback is active (closed switch) if $p_i = 1$ and deactivate (open switch), if $p_i = 0$. Assume the LFSR initial state is $s_0, ..., s_{m-1}$, the next output bit of LFSR is s_m and computed as follows:

$$s_m \equiv (s_{m-1}p_{m-1} + ... + s_1p_1 + s_0p_0) \ mod \ 2 \tag{1.11}$$

Generally, the output sequence is calculated by:

$$s_{i+m} \equiv \sum_{j=0}^{m-1} p_j.s_{i+j} \ mod \ 2; \quad s_i, p_j \in \{0, 1\}; \ i = 0, 1, 2, ... \tag{1.12}$$

The output values are given by a combination of some previous output values. The output sequence of an LFSR repeats periodically, depending on the length of the LFSR. The maximum sequence length generated by LFSR of degree m is $2^m - 1$ [38].

An LFSR can be easily analyzed since its input and output is linearly dependent. LFSRs by themselves are insecure and are prone to analytic attacks. However, they can be used in a secure fashion. For example, combinations of several LFSRs can build strong cryptosystem. The shrinking generator and self-shrinking generator are two secure and efficient pseudo random number generators which are discussed in the following sections.

The Shrinking Generator

Construction of a pseudo random generator based on a simple combination of two LFSRs is proposed in [39]. The construction has attractive properties such as hardware simplicity and security; security conditions such as exponential period, good statistical properties, and resistance to known attacks. The construction is suitable for the practical implementation of efficient stream cipher cryptosystems. This construction uses two sequences of pseudo random bits to create a third sequence of

pseudo random bits of better quality than the original ones. The quality stands for the difficulty of predicting the pseudo random sequence.

Let $(a_0, a_1, ...)$ and $(b_0, b_1, ...)$ denote the first and the second pseudo random bits, respectively. The third sequence of pseudo random bits, $(c_0, c_1, ...)$, which is the result of the Shrinking Generator (SG) is calculated as follows:

$$c_n = \begin{cases} b_n & \text{if } a_n = 1 \\ \text{Discard pair} & \text{if } a_n = 0 \end{cases} \tag{1.13}$$

It means if we have pairs $(a_n, b_n) = (1, 1)$ and $(a_n, b_n) = (1, 0)$ the output bit of SG is $c_n = b_n = 1$ and $c_n = b_n = 0$ respectively. In the case we have pairs $(a_n, b_n) = (0, 1)$ and $(a_n, b_n) = (0, 0)$, the pair will be discarded and SG wait for the next pair.

The shrinking generator has exponential bounds on the period of the produced sequences. The importance of a long period is to avoid the repetition of the sequence after short period of times. Let n and m be the length of the first and the second sequence. Then, the period of the result SG sequence will be $(2^n - 1)2^{m-1}$.

The Self-Shrinking Generator

Motivated by SG, the construction of a pseudo random generator based on a single linear feedback shift register is investigated in [40]. The Self Shrinking Generator (SSG) posses attractive properties; its period, linear complexity and known cryptanalytic attacks allow for efficient practical implementations at a reasonable scale.

SSG has a simpler structure using only one LFSR whose output sequence is shrunken similar to the shrinking generator. Let the output bits of the single LFSR to be $(a_0, a_1, a_2, a_3, ...)$ and consider two consecutive bits of this sequence, (a_n, a_{n+1}). The output bit of SSG, d_n, is generated using the same methodology as SG uses where b_n is replaced by a_{n+1} as follows:

$$d_n = \begin{cases} a_{n+1} & \text{if } a_n = 1 \\ \text{Discard pair} & \text{if } a_n = 0 \end{cases} \tag{1.14}$$

The SSG and the SG are closely related to each other i.e., the self-shrinking generator can be implemented as a shrinking generator, and visa versa. The self-shrinking generator has its main interest in implementing the shrinking principle at lower hardware

costs. For a self-shrinking generator implemented with a maximum length LFSR of length n, the period is lower bounded by $2^{n/2}$. However, a strong evidence is provided that the period is in fact 2^{n-1} for $n > 3$ [40].

Gaussian Random Number Generator

An efficient method to generate high-quality Gaussian random numbers are required in many practical applications. A wide range of Gaussian Random Number Generators (GRNG) has been described in the literature. In [41] an overview of GRNG methods and algorithms including a classification of the various techniques, the performance, and accuracy of the GRNGs is presented. There are many methods to classify GRNGs. One approach in which the GRNGs may be classified is based on the generation method. Using this approach, GRNG may be classified as either based on an exact method or based on an approximate method. Exact methods, such as the Box-Muller method, produce perfect Gaussian random numbers if implemented in an ideal environment. Approximate methods, like the Central Limit Theorem (CLT) method, produce outputs that are approximately Gaussian even if the arithmetic used is perfect.

One of the earliest exact transformation methods to create Gaussian random numbers is the Box-Muller transform. It produces a pair of Gaussian random numbers (a, b) from a pair of uniform numbers (u_1, u_2) as follows:

$$a = \sqrt{-2 \ln u_1}. \sin 2\pi u_2, \qquad b = \sqrt{-2 \ln u_1}. \cos 2\pi u_2 \tag{1.15}$$

The algorithm produces two random numbers each time that is executed. The generation function returns the first value to the user and caches the other value for return on the next function call. It is an optimized algorithm in which the computation of cosine and sine can be performed in a single step [41]. An efficient hardware implementation of high quality Gaussian random number generator using the Box-Muller method is presented in [42]

By the central limit theorem, the sum of k zero-mean uniform random numbers will approximate a Gaussian with zero mean and standard deviation $\sqrt{k/12}$. Larger values of K provide better approximations. The main disadvantage of this approach is that the convergence to the Gaussian distribution is slow with increasing k. This approach has been used in hardware implementations as a way of combining two or

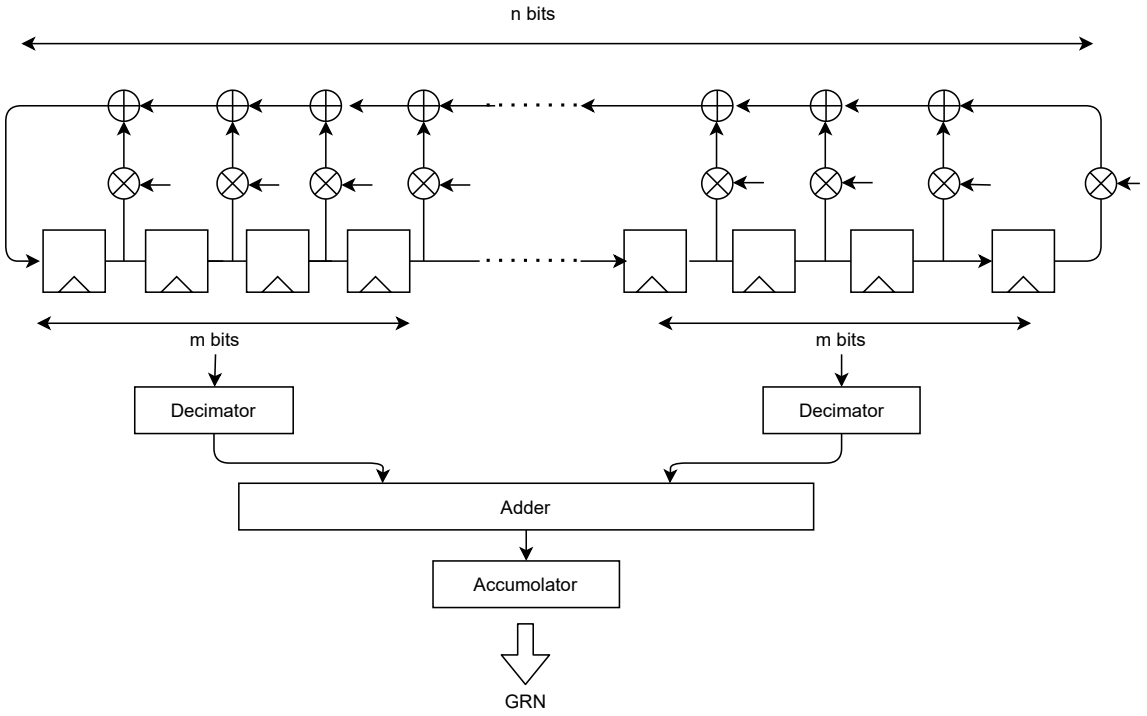

Figure 1.6: GRNG based on adding four m-bits LFSR

more lower quality Gaussian numbers to produce a higher quality one [41].

A simple practical Gaussian-distributed pseudo random number generator based on the central limit theorem is proposed in [43]. A long LFSR is used to extend the period of a random number sequence, binary data in the LFSR are grouped and decimated to make their autocorrelation to become an impulse, and the central limit theorem (CLT) is applied to these binary data to generate a Gaussian-distributed pseudo random number. The n-bit length LFSR is divided into 4 groups, each having a length of m-bits, $n = 4m$. Then, the four m-bits LFSRs are added to create a Gaussian random number based on CLT as shown in Figure 1.6. The period of the Gaussian-distributed random number of this method is $(2^n - 1)/8n$. The proposed design in [43] can be practically used in digital systems which have not enough design space for complicated random number generators.

In [44] the proposal of an algorithm based on linear feedback shift registers that results in a low implementation cost and complexity was presented. The proposed generator followed the same approach as in [43] which was based on CLT and adding 4 m-bits LFSR. The main difference is that all sequences of numbers are generated

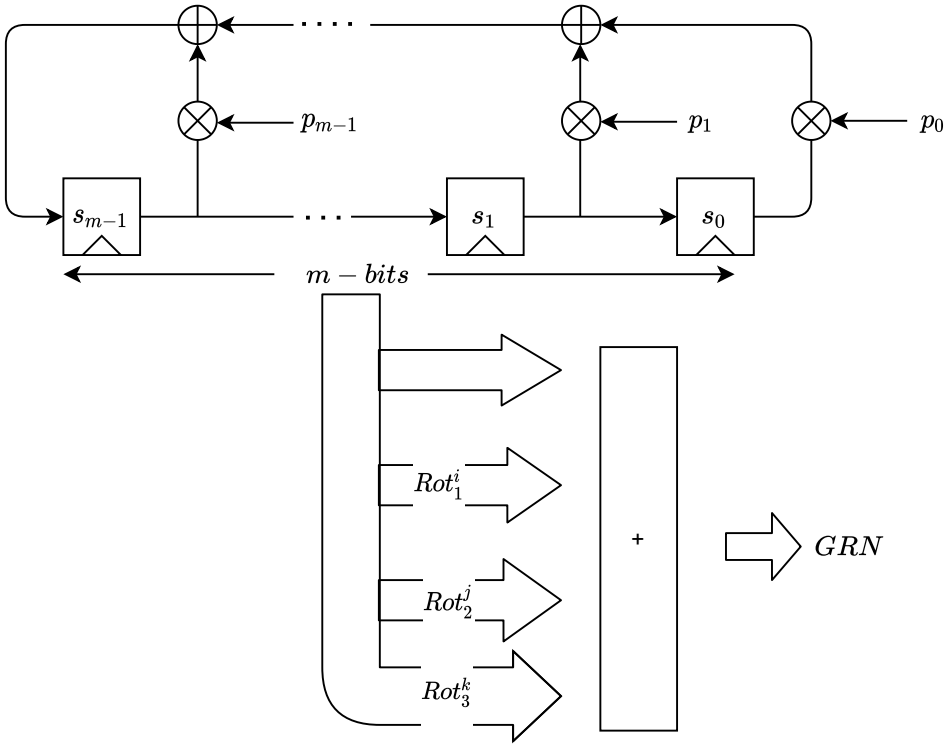

Figure 1.7: GRNG based on cyclic shift.

from the unique LFSR by applying cyclic rotations. The rotation is just a cyclic shift of the content of a given state of the LFSR. Considering the state as a binary vector, a k-rotation implies k single rotations to the right. Finally, the original sequence and its three different rotations are summed up to form a Gaussian random number as shown in Figure 1.7. The computational cost required to implement this algorithm is much lower than the Box-Muller method.

We should note that to have uncorrelated Gaussian random numbers, we need to empty LFSR buffers each time we generate a new number so in case of an m-bit LFSR, each Gaussian random number needs m clocks cycles (each LFSR bit is generated during a single clock cycle by it's structure).

In this section, we presented a basic review on cryptography, two common attacks, different methods to create pseudo random numbers, and Gaussian random number generators which we use later in the construction of our proposed measurement matrix. Now that we have basic knowledge of cryptography, in the next section, we discuss Compressive Sensing Cryptography (CSC).

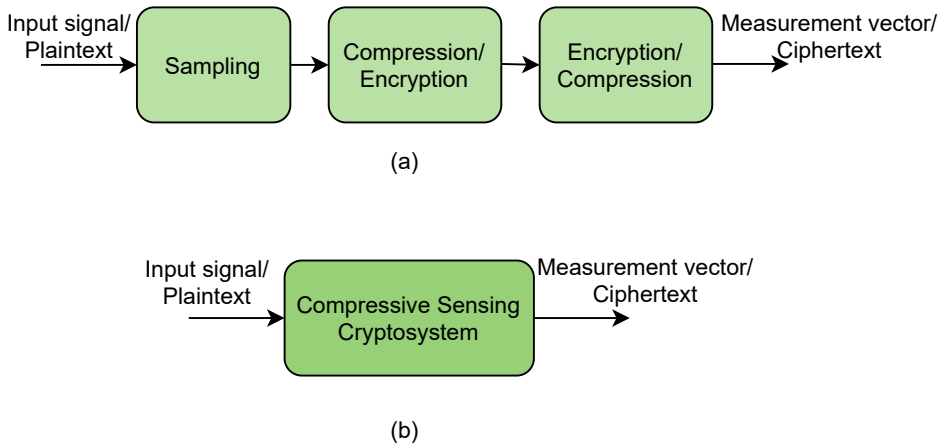

Figure 1.8: (a) Traditional sensing, compression, and encryption, (b) Compressive sensing cryptosystem. The input signal here is assumed to be sparse.

1.3 Compressive Sensing Cryptosystem

The application of CS in the field of information security have captured a great deal of researchers' attention. CS can be regarded as a symmetric-key cryptosystem when the measurement matrix acts as a key. Compressive Sensing Cryptosystem (CSC) has been introduced as an approach to simultaneously sense, compress, and encrypt sparse or compressible signals. Figure 1.8 (a) shows the the traditional method for obtaiing a cryptosystem and Figure 1.8 (b) depcits the CSC. CSC was mainly connected with multimedia and cloud computing scenarios. For instance, image encryption, image, and video watermarking, image hiding, and authentication could use CSC. Under the background of cloud multimedia computing, the CS technique offered privacy-preserving multimedia cloud computing, outsourcing of image reconstruction service, multimedia data storage, healthcare monitoring system, and emergency healthcare system [45].

Figure 1.4 (b) shows a simplified block diagram of a compressive sensing cryptosystem. In the encryption phase, the input signal/plaintext x is measured/encrypted using a secret key, $\mathbf{\Phi}$, which is communicated through a secure channel and known by an authorized person. Then, the resulting ciphertext y is communicated through an insecure channel. An eavesdropper may attack the ciphertext/measurement vector throughout the communication. In the decryption phase, only an authorized person who knows the key can recover the signal.

The security of CS was first considered in [46] where the measurement matrix was the secret key and was randomly generated by sampling independent and identically distributed (i.i.d.) Gaussian numbers. According to to [46], CSC does not achieve Shannon's definition of perfect secrecy. However, CSC provides computational security.

1.3.1 Computational Secrecy

Practical cryptosystems are usually only computationally secure. It means that breaking the cryptosystem is equivalent to solve an NP-hard problem, i.e., a problem whose solution can not be computed in polynomial time. However, the ciphertext contains information about the message, extracting information of original signal x from ciphertext y for an adversary without the appropriate key is assumed to be difficult (e.g., NP-hard). Thus, the statement of secrecy relies on assumptions about the difficulty of the computational problem and the computational resources available to the adversary. The amount of computation required to find the correct key is proportional to the size of key space [46].

Gaussian sensing matrices with i.i.d. entities provide a high degree of computational security in CSC [46]. However, generating and communicating the whole i.i.d. Gaussian matrices for each sensing can be challenging. In practice, a seed/key is communicated via a secure channel and at the receiver part, the measurement matrix is securely and efficiently generated from the initial seed.

1.3.2 Perfect Secrecy

A cryptosystem is information-theoretically secure if it can not be broken even with infinite computational resources. Information-theoretic secrecy relies on the statistical properties of a system. According to Shannon's work [47], an encryption scheme achieves perfect secrecy if $P(X = x|Y = y) = P(X = x)$. Alternatively, this condition can be stated as $I(x; y) = 0$; i.e., the mutual information between x and y is zero.

Lemma 1. Let x be a message, $P_X(x) > 0 \; \forall \; x \in \mathbb{R}^N$ and $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ be a measurement matrix. For $y = \mathbf{\Phi}x$,

$$I(x; y) > 0$$

and therefore perfect secrecy is not achieved [46]

It has been proven in [48] that under specific distributions of the signal, a CS framework that exploits Gaussian i.i.d. sensing matrices can achieve secrecy in an information theoretic sense. The explanation is: the measurements reveal only the energy of the sensed signal, and the energy of the measurements leaks information about the signal. Consequently, a CS framework that uses Gaussian random matrices is, at least in theory, perfectly secure when sensing constant energy signals.

1.3.3 One Time Sensing

The sensing of multiple signals with a similar measurement matrix compromises the confidentiality of CS measurements. Hence, the One Time Sensing (OTS) matrix has been considered to be used in CSCs. In the OTS scenario, each measurement matrix is used only once and different sensing matrices that are statistically independent are generated for each sensing. Storing and transferring these measurement matrices is costly. Therefore, practical systems are based on a shared key stream which is used to generate the OTS sensing matrices [48]. Under this scenario, it is sufficient to consider the confidentiality of a single CS framework $y = \mathbf{\Phi}x$, since measurements of multiple signals will be statistically independent. The OTS scenario with securely generated random matrices seems the most promising one for providing an effective confidentiality layer. When the sensing matrix is used multiple times, some information might leak. If same matrix is used many times, the CSC can be breached. For example, in one of the attack forms, where the plaintext and its corresponding ciphertext are compromised, the knowledge of N linearly independent messages would be sufficient to solve the system of linear equations and construct the measurement matrix [48].

1.3.4 Common Attacks

After the design of the measurement matrix, the security properties of the CSC is evaluated through two common types of attacks: Ciphertext Only Attack (COA) and Plaintext Attack (PA). In COA, an adversary tries to figure out a plaintext by only observing the corresponding ciphertext. In the plaintext attacks (PA) the adversary aims to retrieve the initial state of the key from the pairs of known plaintext and the corresponding ciphertext, (x, y), such that $y = \mathbf{\Phi}x$. In a different version of PA or COA, a partially corrupted encoding matrix $\hat{\mathbf{\Phi}}$, that differs from the original $\mathbf{\Phi}$

in some entries is also known in addition to y (and x in PA) [49]. CS is not secure under PA when the same sensing matrix is used multiple times. Generally, PA is a stronger attack than COA since in PA the attacker may achieve the key and do the decryption.

1.4 Thesis Organization

Chapter 2 provides a relevant literature review on compressive sensing and compressive sensing cryptosystem. We discuss the limitations of the most relevant methods to our proposed method in the literature and present the motivation behind our proposed method presented in this thesis. We formulate the problem addressed in this thesis and summarize our thesis contributions.

Chapter 3 discusses our proposed method for an efficient CS. In this chapter, we first briefly go through basic knowledge of coding theory and its tie to CS. We further explain BCH-based codes in coding theory and how we use that as a measurement matrix in CS. Then we describe our proposed permuted Kronecker-based sparse measurement matrix. Finally, we discuss the performance of the proposed method using three classes of sub-matrices on ECG signals from MIT-BIH Arrhythmia database. We also compare our method with other states of the arts efficient CS techniques.

Chapter 4 discusses our proposed technique for efficient CSC. In this chapter, we first explain our proposed Kronecker-based matrix which is subsequently multiplied by a pseudo-random permutation matrix to construct a sparse one-time sensing (OTS) matrix. Then, we compare sparsity and the number of linear feedback shift register (LFSR) bits of the proposed method to other sparse measurement matrix structures proposed in the literature. Finally, we analyze the security of the proposed scheme against plaintext and ciphertext only attacks. We use the size of solution space in the plain-text attack as a measure to show the number of candidate keys.

Chapter 5 is a summarized version of our work in order to improve efficiency and security in CS. We also provide directions for future research.

Chapter 2

Literature Review

In this chapter, we provide a literature review of CS and CSC. Then, we provide our motivation to propose our measurement matrix under Section 2.3. We also mention our contribution and thesis structure.

2.1 Compressive Sensing

The basic idea of compressive sensing has been proposed by Emmanuel Candès, Justin Romberg, Terence Tao, and David Donoho in [50]. They claim that sparse signals can be reconstructed with fewer samples than Nyquist sampling theorem requires. They tent to recover a vector x from a contaminated observations $y = \mathbf{\Phi}x + e$ by solving the following minimization problem.

$$argmin\|x\|_1 \quad \text{subject to} \quad \|\mathbf{\Phi}x - y\|_{\ell_2} < \epsilon, \tag{2.1}$$

Later the theorey of CS has been discussed more in [2] [3] [4].

Designing the measurement matrix is an important part of the CS which has attracted lots of attention. One classification of CS measurement matrix is based on random and deterministic structure. The CS has been proposed along with random measurement matrices such as matrices drawn from i.i.d. Gaussian with zero mean and Bernoulli distribution of $\pm$1s. It has been proved in the literature that these random matrices are incoherent with any other basis and satisfy the RIP condition. But, storing and reproducing the random matrices at the receiver part is challenging in practice. These random matrices need to be transmitted along with the signal, which is not practical for signal processing applications. So the research interest

has been diverted towards the design of deterministic and structured measurement matrices in CS. Examples of such matrices are circulant, Toeplitz, structured random matrices. The advantages of structured random matrices are faster acquisition, less storage requirement, reproducibility, and reduced transmission overhead, while the drawback is the requirement of a higher number of measurements compared to random matrices [36].

In [51], random filters are proposed for compressive signal acquisition. In this approach, a signal is captured by convolving it with a finite random-tap filter and then down-sampling the filtered signal to obtain a compressed representation. The random filter accelerates sensing and reconstruction algorithms. It is easily implementable in software or hardware and can be used to capture finite length, discrete-time signals.

A structured random measurement matrix is proposed in [52]. In this technique, the first row of the measurement matrix is a random pulse sequence. Then, the next row is obtained by the circular shift of a previous row. This procedure is repeated for all other rows to generate a measurement matrix. This structure has advantages like faster acquisition, easy storage, transmission, and incoherence with any fixed orthogonal basis.

A deterministic $M \times M^2$ matrix is designed with chirp sequences forming its columns in [11]. It has been shown that this type of matrix is valid for compressed sensing measurements. For sufficiently sparse signals, chirp matrices admit a fast reconstruction and recover the signal with computational complexity $O(M \log M)$ for M measurements. However, the CS performance with such matrices is limited when M/N is small.

In [9], a deterministic sensing matrix of size $2^p \times 2^{p(p+1)/2}$ is proposed which has a fast reconstruction algorithm. The matrix construction is based on the second-order Reed-Muller codes. The proposed matrix has ± 1 entries excluding the normalization factor of $1/\sqrt{2^m}$. This matrix does not have RIP uniformly with respect to all k-sparse vectors.

In [10], an RIP fulfilling bipolar ± 1 measurement matrix is introduced. The columns of these matrices are binary BCH code vectors where the zeros are replaced by -1. In addition, they combined the binary and bipolar matrices to form a ternary sensing matrix ($\{-1, 1, 0\}$ elements) that satisfies RIP condition. These matrices, in addition to their deterministic and known structure, simplify the measurement process. Also, they enable the exact recovery of noiseless signals with substantially

reduced computations.

In [53], a simple and efficient measurement matrix called binary permuted block diagonal matrix is proposed. This matrix is binary and highly sparse and has a few ones in each column and each row. Therefore, it can simplify the compressed sensing procedure dramatically. The proposed measurement matrix has the following advantages: easy hardware implementation because of the binary elements, high sensing efficiency because of the highly sparse structure, incoherent with different popular sparsity basis, fast and nearly optimal reconstructions.

A simple deterministic measurement matrix that facilitates the hardware implementation has been proposed in [54]. In [54] thresholds are applied in the discrete cosine transform domain to control the sparsity of the signal. They proposed a deterministic binary block diagonal matrix. The blocks of ones, which make up the diagonal of the matrix, are identical and contain N/M elements. Their simulation and experimental results show that the proposed measurement matrix has a better performance in terms of reconstruction quality compared with random matrices.

The idea of Kronecker CS first introduced in [55] involving compressed sensing and recovery of multidimensional signals. Many important applications of CS involve high dimensional signals. The measurement matrix in Kronecker CS can jointly measure different types of structures present in the signal. In particular, when partitioned measurements are used and the same measurement matrix is applied to each portion of the signal, the resulting measurement matrix can be expressed as the Kronecker product of an identity matrix with the measurement matrix. The Kronecker CS formulation follows the standard CS approach of a single measurement and standard recovery algorithms.

A measurement matrix based on the Kronecker product is found in [56]. Their numerical simulations on 2-D images verify that the proposed measurement matrix has better performance in storage space, construction time, and image reconstruction when compared with commonly used matrices in CS such as Gaussian and Bernoulli matrix. This novel measurement matrix facilitates hardware implementation of compressive sensing in the image and high dimensional signal.

A Kronecker-based recovery technique is proposed for compressed ECG signals in [57]. The proposed modified Kronecker-based CS technique reduces the mutual coherence between the sensing matrix and measurement matrix. So, the quality of the recovered ECG signal improves when compared to the standard Kronecker-based

CS technique.

In [58], a permutation-based sparse measurement matrix is proposed. It is composed of several sub-matrices, each has a block-diagonal structure with Gaussian random variables (as its non-zero entries). Each sub-matrix is subsequently multiplied by a random permutation matrix. The generated measurement matrix is sparse and has specific number of non-zero entries per column. It has a simple and effective recovery algorithm. Also, the complexity of the proposed algorithm is relatively low, which grows linearly with the source signal length N. The entries of the proposed matrix is limited to Gaussian random numbers.

In [59], the authors developed a sparse sensing matrix with a low computational cost while maintaining the quality of the reconstructed signal. The design approach is based on a Sparse Block Circulant Matrix (SBCM) which is defined as below:

$$\mathbf{\Phi}_{SBCM} = \begin{bmatrix} \mathbf{\Phi}_{1,1} & \mathbf{\Phi}_{1,2} & \dots & \mathbf{\Phi}_{1,n} \\ \mathbf{\Phi}_{2,1} & \mathbf{\Phi}_{2,2} & \dots & \mathbf{\Phi}_{2,n} \\ \vdots & \vdots & & \vdots \\ \mathbf{\Phi}_{m,1} & \mathbf{\Phi}_{m,2} & \dots & \mathbf{\Phi}_{m,n} \end{bmatrix}_{M \times N} \tag{2.2}$$

where $\mathbf{\Phi}_{i,j} \in \mathbb{R}^{l \times l}$ is a sparse circulant square matrix and $N = n \times l$, $M = m \times l$. There is only one non-zero element in each row and each column of $\mathbf{\Phi}_{i,j}$ which is chosen from an i.i.d. Gaussian random distribution. The next row is constructed with a single shift to the previous row. The n non-zero elements of $\mathbf{\Phi}_{SBCM}$ in a row are different and so as the m different elements of a column. The measurement of the sparse signal x is done by $y = \mathbf{\Phi}_{SBCM} x$. Their simulations validate that SBCM reduces the computational burden significantly. It also can be easily implemented in a digital system. SBCM is particularly useful for large-scale, real-time, block-based CS applications, such as distributed sensor networks. The signal recovery using SBCM is similar to the one achieved with random Gaussian sensing matrices. We compare our proposed method in Chapter 3 with SBCM.

2.2 Compressive Sensing Cryptosystem

Security is an important criterion in some applications when communicating the information using CS to the external world, for instance, to a computing cloud. The security of CS was first considered in [46] where the measurement matrix was the secret key and was randomly generated by sampling independent and identically distributed (i.i.d.) Gaussian numbers. According to to [46], CSC does not achieve Shannon's definition of perfect secrecy. However, it can provide computational secrecy. It means, extracting information of x from y for an adversary without the appropriate key is equivalent to solving a computational problem that is NP-hard. The number of computations required to find the correct key is proportional to the number of candidate keys. In practice, 2^{64} candidate keys is a sufficiently large random seed, making the evaluation of all keys difficult. In [46], it was also showed that when an adversary tries to do a recovery with key $\mathbf{\Phi}'$ which is different from the true key $\mathbf{\Phi}$, the attacker will recover an M-sparse signal instead of the original K-sparse signal with probability one .

Although Gaussian sensing matrices with i.i.d. entities provide a high degree of security in CSC [46], generating and communicating the i.i.d. Gaussian matrices for OTS are challenging. In practice, a seed/key is communicated via a secure channel and the measurement matrix is securely and efficiently generated from the initial seed.

In [60], the security of the CS is analyzed using the i.i.d. Gaussian One-Time Sensing (OTS) approach. The most significant novelty of this work is the analysis of generic sensing matrices, providing important insights on the confidentiality of non-Gaussian sensing matrices. They also consider two possible kinds of attacks to CS measurements. The first attack aims at estimating the energy of the signal from the energy of the measurements. The second attack aims at distinguishing two different equal-energy signals.

A comprehensive review of secure wireless communications based on CS using different types of random measurement matrices such as Gaussian matrix and circulant matrix was presented in [61]. Applications of secure CS depending on communication scenarios such as wireless wiretap channel, wireless sensor network, internet of things, crowd sensing, smart grid, and wireless body area networks were reviewed.

In [62], circulant matrices were considered for practical CS systems. Circulant matrix consists of a sequence of i.i.d. Gaussian or sub-Gaussian variables in its first row; the next row is generated by a circular shifting of the previous row by one

component. The circulant matrix has a similar performance as the full Gaussian matrix in terms of sensing and recovery. While, the circulant matrices have lower complexity in contrast to Gaussian matrices when generated, transmitted, and used for measuring the signal. Although such matrices enable fast computation, they may leak some information of the sensed signal since their rows are correlated. Results show a circulant matrix can provide a weak encryption layer if the signal is sparse in the sensing domain.

In [63] a CS based encryption scheme, "Kryptein", for cloud based IoT systems has been proposed. This method claims to be fast and energy efficient. The sparsifying basis in this method is constructed according to a learning based approach using a database. In this approach, each different set of database has a specific sparsifying basis. Then, the measurement matrix is constructed from the sparsifying basis utilizing singular value decomposition scheme. The authors showed that reconstruction of the original data from cipher is difficult for attackers and producing large errors. The proposed measurement matrix in [63] is not sparse.

In [64], two main issues namely the power consumption and security of CS in wearable wireless sensors are considered. Both issues are addressed in IoT-based remote health monitoring systems by exploring CS. The presented solution exploits shift registers with sparse sensing matrices to construct an encryption key to implement a lightweight efficient encryption scheme. The initial sensing matrix is selected as a sparse Bernoulli matrix. There are two reasons for this choice of sensing matrix. Firstly, sparse Bernoulli matrices are easier to implement in hardware; thus, they more suitable for real-world CS-based applications. Secondly, Bernoulli matrices are proven to hold both RIP and incoherence with high probability. The results in the paper showed that CS can reduce power consumption by a factor of 35%. In addition, the paper demonstrated that transmission can be kept secure even if the illegitimate attacker can access 95% of the information needed to recover the data.

In [65], the security of a CS-based cryptosystem called Sparse One-Time Sensing (S-OTS) cryptosystem is studied. S-OTS encrypts plaintext with a sparse measurement matrix. To construct the secret matrix and renew it at each encryption, a bipolar keystream and a random permutation pattern are employed, which can be obtained by a keystream generator of stream ciphers. S-OTS cryptosystem employs

a secret measurement matrix as follows:

$$\mathbf{\Phi} = \frac{1}{\sqrt{Mr}}\mathbf{SP} \tag{2.3}$$

where $\mathbf{S} \in \{-1, 0, 1\}^{M\times N}$ is a sparse matrix containing q nonzero elements in each row, $\mathbf{P} \in \{0, 1\}^{N\times N}$ is a matrix for permuting the columns of S, and $r = q/N$ is the row-wise sparsity. For efficient encryption $\frac{N}{M} \leq q \ll \frac{N}{2}$. With a small number of non-zero elements in the measurement matrix, S-OTS cryptosystem achieves efficient CS encryption in terms of memory and computational cost.

In the security analysis, the authors of [65] show that S-OTS cryptosystem can be indistinguishable as long as each plaintext has constant energy, which formalizes computational security against COA. In addition, a chosen PA [49] is considered against the proposed method, which consists of two sequential stages: keystream and key recovery attacks. Against keystream recovery under CPA, S-OTS cryptosystem can be secure with overwhelmingly high probability, as an adversary needs to distinguish a prohibitively large number of candidate key streams. Finally, the authors conduct an information-theoretic analysis to show that the S-OTS cryptosystem can be resistant against key recovery under CPA by guaranteeing the probability of success is extremely low. In conclusion, the S-OTS cryptosystem can be computationally secure against COA and the two-stage CPA, while providing efficiency in CS encryption. We compare our proposed method with S-OTS in Chapter 4.

We choose two most relevant methods proposed in the literature in order to compare with our method described in Chapter 3: SBCM [59] and S-OTS [65]. The main reason behind this choice is that we have control on the sparsity of the measurement matrix. Therefore, we can create measurement matrices with same sparsity as our proposed method and compare other factors such as recovered signal quality, recovery time, and hardware costs. However, one significant limitation of SBCM and S-OTS is that their entries are only Gaussian and ternary, respectively. In our method, we can use any eligible measurement matrix and obtain an outstanding performance in terms of SNR, time, and storage costs.

2.3 Problem Statement and Contributions

Efficiency and confidentiality are the two key requirements in CS-based communication. Efficient CSC attracts more attention especially in devices with limited processing power. For instance, wearable devices for remote health monitoring systems and Internet of Things (IoT)-based wireless sensor networks that need to continuously process the data and securely transfer it to the database need CSC. A major effort in CS is to design a measurement matrix that can be used to encode and compress sparse or compressible signals. The measurement matrix structure has a direct impact on the computational and storage costs as well as the recovered signal quality. In CSC applications, the measurement matrix is the secret key. Driven by the aforementioned requirements (efficiency and confidentiality), in this thesis, we first focus on designing a sparse and efficient measurement matrix for CS applications. We further improve the security of our proposed measurement matrix to be used in CSC applications. Below is the summary of this thesis contributions:

- Propose a permuted Kronecker-based sparse measurement matrix for sensing and data recovery in CS applications. This approach can utilize different types of eligible measurement matrices in CS. Using BCH-based measurement matrices, our approach outperforms existing sparse CS methods.

- Our methodology results in an overall reduction in storage and computations, both during the sensing and recovery process.

- Extend the proposed method with an OTS matrix for compressive sensing cryptosystem. Secure OTS sub-matrcies and random permuation matrix are considered. The proposed method is superior to other OTS sparse measurement matrix structures in the literature in terms of the number of LFSR bits required for its generation. The proposed sparse measurement matrix is computationally secure against plaintext and ciphertext only attacks.

The research done for this thesis has culminated in the following peer reviewed publications:

1. P. Firoozi, S. Rajan and I. Lambadaris, “Efficient Compressive Sensing of Biomedical Signals Using A Permuted Kronecker-based Sparse Measurement Matrix,” 2021 IEEE International Symposium on Medical Measurements and Applications (MeMeA), pp. 1-5, 2021.

2. P. Firoozi, S. Rajan and I. Lambadaris, "Efficient Kronecker-based Sparse One-Time Sensing Matrix For Compressive Sensing Cryptosystem," IEEE International Mediterranean Conference on Communications and Networking (MediterCom), 2021. (Accepted: July 5th 2021.)

Chapter 3

Efficient Compressive Sensing

In devices such as IoT-based sensors where power and computational resources are limited, an efficient way of sensing is required. Efficient compressive sensing may be achieved by an appropriate design of a measurement matrix that is sparse. In this thesis, a sparse measurement matrix is considered efficient if it leads to a reduction in storage, computations and recovery time.

In this chapter, we present the design of an efficient and sparse measurement matrix and evaluate its performance through simulations. We start by choosing a basic measurement matrix and then expand this matrix through a Kronecker product operation to create a block diagonal matrix. This matrix is subsequently rearranged using a specific permutation matrix to sense a sparse signal x. Simulations show that this method outperforms the simple Kronecker CS approach in terms of reconstructed signal quality. Also, our methodology is superior to known CS methods in terms of increased sparsity of the measurement matrix and enhanced recovery time. We use random Gaussian, Bernoulli, and BCH-based sub-matrices to form the block diagonal matrix. However, the proposed method can be generalized to include other measurement matrices as well. The flexibility to apply different sub-matrices is an advantage of our method over other state-of-the-art techniques using block-diagonal sparse measurement matrices For instance, [58] and [59] are restricted to Gaussian entities. In particular, we are able to employ BCH-based sub-matrices as measurement matrices which leads to outstanding performance with respect to recovered signal quality, storage, computations, and signal recovery time. We first briefly describe basic concepts of coding theory, BCH codes, and explain the connection between coding theory and CS. Then, we explain our proposed method for efficient CS. Finally, we present the simulation results on ECG signals.

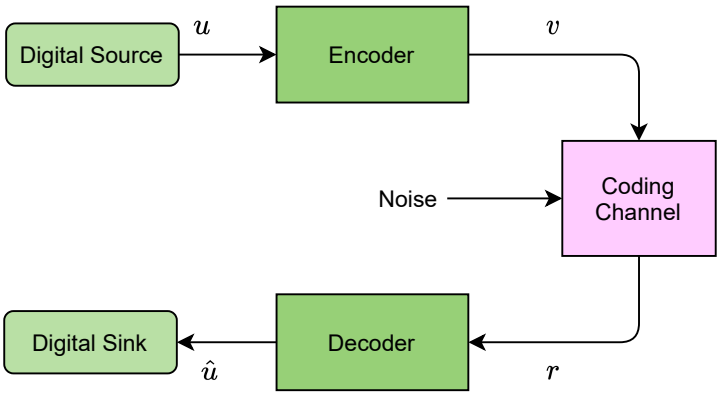

Figure 3.1: A simplified model of a communication system.

3.1 Compressive Sensing and Coding Theory

Currently, the basic research in CS has been driven by contributions from the fields of signal processing, statistics, and computer science. CS has also strong ties to coding and information theory. The connections between coding and CS have been investigated in a limited number of papers such as [10], [66] [67]. One of the important concerns in CS theory is the existence of low-complexity algorithms for compressive sensing and reconstruction. There exist only a small number of practical sensing matrices that satisfy these requirements. Contributions to this growing field have a significant impact on applications such as medical imaging, sensor networks, and cloud-based CS. In this section, we discuss the links between CS and coding theory by first giving a brief introduction to coding theory from [68], then explain their connections. Finally, we explain BCH-based codes as an effective matrix to be used in CS applications.

3.1.1 Useful Coding Theory Terminologies in CS

Reliable data transmission is the ultimate goal of each communication system. A major concern in data transmission is controlling errors. A simplified model of a digital communication system can be represented in Figure 3.1. The channel encoder transforms the source bits u, into a sequence of binary digits v, called a code word. The channel decoder transforms the received sequence r into a binary sequence $\hat{u}$ which is called the recovered sequence. The goal of coding is to design channel encoder and decoder pairs such that information can be transmitted in the noisy environment

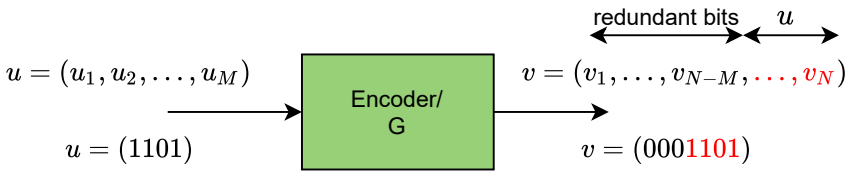

Figure 3.2: A message and a code word.

as fast and reliable as possible. Channel encoder adds some redundant bits to u such that first, it makes the error detection and then, error correction possible [68]. The redundant bits empower codes to combat the channel noise.

There are two different types of codes: block codes and convolution codes; In this thesis, we only consider linear block codes. A linear code is an error-correcting code for which any linear combination of code-words is also a code word [69]. In the block codes, a message block is represented by a binary M-tuple $u = (u_1, u_2, ..., u_M)$. The encoder transforms each message u into an N-tuple $v = (v_1, v_2, ..., v_N)$. The ratio $R = M/N$ is called the code rate. An interesting property of a linear block code is the systematic structure of the code word. It means that each code word has two parts. The first part of the code word is the message itself and the second part is the redundant bits (see Figure 3.2). Each code-word is derived by multiplying the message and a generator matrix, $\mathbf{G}$, as follows:

$$v = u.\mathbf{G} \tag{3.1}$$

where

$$\mathbf{G} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_M \end{bmatrix} = \begin{bmatrix} g_{11} & g_{12} & g_{13} & \cdots & g_{1N} \\ g_{21} & g_{22} & g_{23} & \cdots & g_{2N} \\ \vdots & \vdots & \vdots & & \vdots \\ g_{M1} & g_{M2} & g_{M3} & \cdots & g_{MN} \end{bmatrix}. \tag{3.2}$$

Example 1. a linear code block with $M = 4$ and $N = 7$ is represented in Table 3.1

Table 3.1: Example of messages and their code words.

message	code word	message	code word
0000	0000000	1000	1101000
0100	0110100	1100	1011100
0010	1110010	1010	0011010
0110	1000110	1110	0101110
0001	1010001	1001	0111001
0101	1100101	1101	0001101
0011	0100011	1011	1001011
0111	0010111	1111	1111111

with below generator matrix:

$$\mathbf{G} = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.3}$$

The code word for message (1000) equals to $v = (1101000)$. Note that the addition and multiplication operations are defined over binary field (Mod 2 operation).

For systematic codes it can be shown that there exist a matrix $\mathbf{F}$ such that the generator matrix can be rewritten as:

$$\mathbf{G} = [\mathbf{F}\ \mathbf{I}_M] \tag{3.4}$$

where $\mathbf{I_M}$ is the Identity matrix of order $M \times M$.

There exist another useful matrix for every linear block code called parity check matrix $\mathbf{H}$ such that any vector in rows of $\mathbf{G}$ is orthogonal to rows of $\mathbf{H}$. For systematic codes, it can be shown that

$$\mathbf{H} = [\mathbf{I}_{N-M}\ \mathbf{F}^T] \tag{3.5}$$

The parity check matrix for Example 1 is:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}. \tag{3.6}$$

After transferring the message through a noisy channel, the received signal is

$$r = v + e \tag{3.7}$$

where e is the error vector and defined as

$$e_i = \begin{cases} 1 & if\ r_i \neq v_i \\ 0 & if\ r_i = v_i. \end{cases} \tag{3.8}$$

Syndrome vector s is defined by

$$s = \mathbf{H}.r \tag{3.9}$$

Using the fact that all vectors in the row space of $\mathbf{G}$ are orthogonal to the rows of $\mathbf{H}$ ($\mathbf{H}.v = 0$), then

$$s = \mathbf{H}.(v + e) = \mathbf{H}.e \tag{3.10}$$

The goal of error correction scheme is finding the error vector e. Finally, the code word is obtained by $v = r - e$. The error vector e can be fined by solving (3.10). However, in the linear system defined by the equation (3.10), the number of equations M is less than the number of variables N. Therefore, this linear system does not have a unique solution. In coding theory, the interested error vector is assumed to be the most sparse one.

Another important parameter in the coding theory is the concept of minimum distance. It is used to determine the capability of error correction in a linear coding scheme. The distance between two vectors is defined as the number of different components in them and is shown with d. For example, if $v_1 = (1001)$ and $w_1 = (1010)$, then $d(v_1, w_1) = 2$. Given a block code C the minimum distance is:

$$d_{min} = min\{d(v, w) : v, w \in C, v \neq w\} \tag{3.11}$$

Table 3.2: An analogy between CS and coding theory.

Compressive Sensing	Coding Theory
number of columns in the measurement matrix	code block length: N
number of rows in the measurement matrix	number of parity check bits
compression ratio	$1 - R$
sparse signal: x	error vector: e
measurement vector: y	syndrome: s
measurement matrix: $\mathbf{\Phi}$	parity check matrix: $\mathbf{H}$
spark($\mathbf{\Phi}$)	minimum distance $d_{min}(\mathbf{H})$

A code block with minimum distance d_{min} is capable of detecting and correcting errors less than or equal to $d_{min} - 1$.

The following subsection will present an analogy between CS and coding theory.

3.1.2 Connection between CS and Coding Theory

We summarize the mapping between CS and coding theory terms in Table 3.2. There might be still differences in CS and coding theory terms. For instance, in compressive sensing, the input signal and measurement matrix are in the real domain (they might be binary or not). However, in coding theory, both the error vector and parity check matrix have binary entries. In coding theory, one is interested in finding the most sparse error vector that satisfies equation (3.9) which is similar to the CS minimization recovery problem (1.9). In the following section, we explain how to generate a coding-based measurement matrix, in particular a BCH-based matrix.

3.1.3 BCH-based Measurement Matrix

Bose, Chaudhuri, and Hocquenghem (BCH) code is a powerful and extensively studied class of code in the coding theory. For any positive integers $p \geq 3$ and $k < 2^{p-1}$, there exists a binary BCH code with the properties given below. This means we can design a measurement matrix with a compression ratio equal to $1 - M/N$ that guarantees the recovery of k-sparse signals using BCH-based codes.

Block length	$N = 2^p - 1$
Number of parity-check digits	$M \leq pk$
Minimum distance	$d_{min} \geq 2k + 1$

Table 3.3: Generator polynomial for two BCH codes.

N	M	k	generator polynomial	$\mathbf{H}$
7	3	1	$1 + x + x^3$	$\mathbf{H}_{3,7}$
15	10	3	$1 + x + x^2 + x^4 + x^5 + x^8 + x^{10}$	$\mathbf{H}_{10,15}$

Each BCH matrix is described by a minimal polynomial. The minimal polynomials for different BCH codes with different block lengths and code rates are already derived and presented in the coding theory reference books [68]. The details of generating parity check matrix from the minimal polynomial is out of the scope of this thesis. We use the already existing polynomials to generate BCH codes. Table 3.3 provides the generator polynomial for two BCH codes and their parity check matrix as below.

$$\mathbf{H}_{3,7} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix},$$

$$\mathbf{H}_{10,15} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

There are some limitations in using BCH codes as a measurement matrix in CS. First, the BCH codes are not RIP. There are different approaches to make BCH codes RIP. One way is to normalize the columns of $\mathbf{H}$ [10] and this form of normalization is used in this thesis. Below is an example of a normalized BCH matrix:

$$\mathbf{\Phi}_{3,7} = \begin{bmatrix} 1.0000 & 0 & 0 & 0.7071 & 0.5774 & 0.7071 & 0 \\ 0 & 1.0000 & 0 & 0 & 0.5774 & 0.7071 & 0.7071 \\ 0 & 0 & 1.0000 & 0.7071 & 0.5774 & 0 & 0.7071 \end{bmatrix}.$$

When normalizing the columns, entries are no longer $\in (0, 1)$ so it needs more computational power for arithmetic operations. Second, BCH codes are defined for specific M and N values where $N = 2^p - 1$ and p is an integer. This limitation makes it difficult to adjust the size of the measurement matrix.

The BCH-based code explained in this section is a candidate sub-matrix used in our proposed design when used in CS, the recovery results are outstanding.

3.2 Proposed Method

A crucial design parameter affecting the size of the measurement matrix is the sensing length, N. When N is large, the size of the measurement matrix is also large, thus resulting in increased computational effort and storage. To reduce the size of the measurement matrix, the original signal s can be divided into ℓ segments each having n elements which will be sensed separately (we assume that $N = \ell \times n$); however, the quality of the recovered signal is compromised because the segmentation may adversely affect the sparsity of some segments. Therefore, the overall recovered signal quality may suffer due to signal segmentation and subsequent segments concatenation.

We now consider the sensing of transformed signal x of length N. In order to reduce the number of storage, computations, and recovery time, we propose a measurement matrix, henceforth denoted by $\mathbf{A} \in \mathbb{R}^{M\times N}$ defined by

$$\mathbf{A} = (\mathbf{I} \otimes \tilde{\mathbf{\Phi}})\mathbf{P}, \tag{3.12}$$

where $\mathbf{I} \in \mathbb{R}^{l\times l}$ is the identity matrix, $\tilde{\mathbf{\Phi}} \in \mathbb{R}^{m\times n}$ is an existed measurement matrix in the literature, $\mathbf{P} \in \mathbb{R}^{N\times N}$ is an orthogonal permutation matrix that we will define shortly, and $N = l \times n$ and $M = l \times m$. In this method, $\tilde{\mathbf{\Phi}} \in \mathbb{R}^{m\times n}$ can be chosen from different classes of measurement matrices in CS. For instance, random or deterministic measurement matrices like Gaussian, Bernoulli, BCH-based [70], binary, bipolar, and ternary measurement matrices [10] may be considered.

To further demonstrate the measurement process using $\mathbf{A}$, we can rewrite the general CS equation (1.3) as follows :

$$y = (\mathbf{I} \otimes \tilde{\mathbf{\Phi}})\mathbf{P}x = \begin{bmatrix} \tilde{\mathbf{\Phi}} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{\Phi}} & \dots & \mathbf{0} \\ & & \ddots & \\ \mathbf{0} & \mathbf{0} & \dots & \tilde{\mathbf{\Phi}} \end{bmatrix} \mathbf{P}x. \tag{3.13}$$

To be able to use the Kronecker structure, $\mathbf{P}x$ needs to have "distributed" sparsity in the following sense. Assuming x is a k-sparse signal of length N, then the expected sparsity of $\mathbf{P}x$ in each of its segment of length $n = N/\ell$ should be k/ℓ. Thus, the

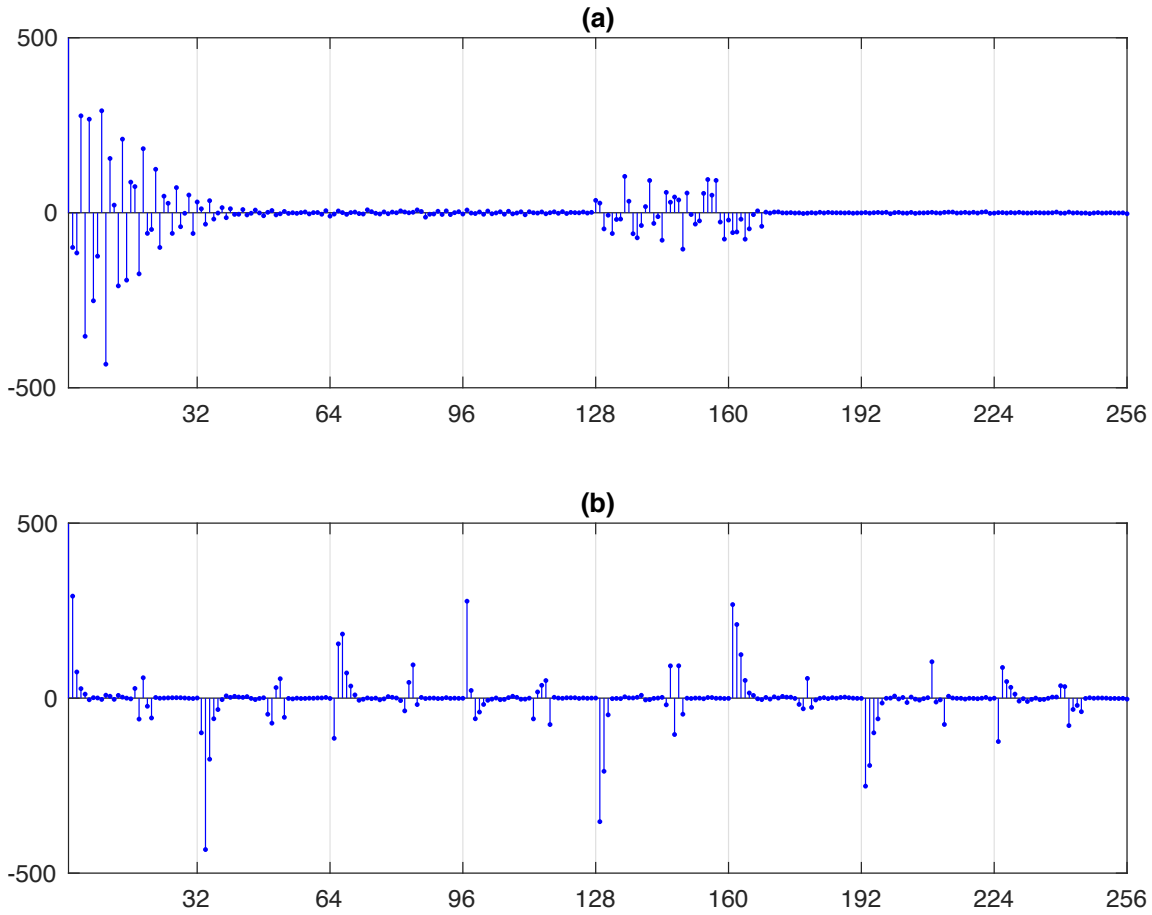

Figure 3.3: Sub-figure (a) is a signal in DCT domain and (b) is its permuted version using proposed permutation matrix for $N = 256$, $\ell = 8$ and $n = 32$. Significant values are spread after interleaving. If original signal has k significant values, the permuted version expected to have k/ℓ significant values per segment.

permutation matrix should be suitably designed to distribute non-zero elements of x uniformly around the spectrum. In our design, each segment of $\mathbf{P}x$ is a down-sampled version of x by ℓ. Since down sampling keeps the characteristics of the signal, each segment is sparse itself and its expected sparsity is k/ℓ. Regardless of the structure of the sparse signal x, the permuted version will have distributed block sparse structure. This is indicated in Figure 3.3. The top subplot shows a sparse signal x while the bottom subplot shows $\mathbf{P}x$ having distributed sparsity. The permutation matrix can be constructed as follows:

$$p_{ij} = \begin{cases} 1 & \text{if} \quad j = (\ell \times i) \bmod N + \lfloor \frac{i}{n} \rfloor \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

where $i = \{0, ..., N-1\}$. Below is a small example for $N = 6$, $M = 3$, $\ell = 3$, and $\tilde{\mathbf{\Phi}} = [\tilde{\phi}_{\mathbf{11}}\ \tilde{\phi}_{\mathbf{12}}]$.

$$\mathbf{A} = (\mathbf{I} \otimes \tilde{\mathbf{\Phi}}) \times \mathbf{P} = \begin{bmatrix} \tilde{\phi}_{11} & \tilde{\phi}_{12} & 0 & 0 & 0 & 0 \\ 0 & 0 & \tilde{\phi}_{11} & \tilde{\phi}_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & \tilde{\phi}_{11} & \tilde{\phi}_{12} \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

It can be shown that the matrix $\mathbf{A}$ satisfies necessary RIP property [55] and therefore it qualifies to be a legitimate measurement matrix for CS. Since $\mathbf{P}x$ is a block sparse signal with sparsity k/ℓ, each segment can be recovered with a sub-matrix, $\tilde{\mathbf{\Phi}}$, which also satisfy RIP itself. Hence, $\tilde{\mathbf{\Phi}}$ will recover a k/ℓ-sparse signal and therefore, $\mathbf{A}$ can sense and recover the entire k-sparse signal.

In the proposed method, measurement matrix $\tilde{\mathbf{\Phi}}$ is the only matrix which should be known by both the decoder and encoder, therefore the storage requirement of our approach is reduced by $\ell^2 = \dfrac{M \times N}{m \times n}$ compared to ordinary CS using a dense measurement matrix of size $M \times N$. For instance, we can compare Gaussian measurement matrix through the proposed method and ordinary CS. Also, sensing with a dense

measurement matrix like Gaussian in ordinary CS needs $M \times N$ multiplications and $M \times (N-1)$ additions while in the proposed method these are reduced to $\ell \times m \times n$ and $\ell \times m \times (n-1)$ respectively, an almost ℓ-fold reduction. Sparsity of the proposed measurement matrix results in storage, computation and recovery time compensation. We define Sparsity Ratio (SR) of the measurement matrix $\mathbf{A}$ as follows:

$$\text{SR} = \frac{\text{number of non-zeros in } \mathbf{A}}{M \times N}. \tag{3.15}$$

3.3 Simulation and Results

Our methodology is applicable to any sparse signal. In this section, we apply our methodology to perform CS on ECG signals from the MIT-BIH Arrhythmia database [71]. We apply the Discrete Cosine Transform (DCT), represented by matrix $\mathbf{\Psi}$, to sparsify each ECG signal of length $N \approx 1024$. We use three different measurement sub-matrices $\tilde{\mathbf{\Phi}}$: a) normalized Gaussian with zero mean and variance $1/m$, b) Bernoulli with probability of success equal to 0.5, and c) normalized BCH based measurement matrix [68]. The signal to noise ratio (SNR) is chosen as a measure of the quality of recovered signal. Here, SNR is defined as signal power divided by error power (power in the reconstruction error) and is defined as below:

$$SNR\ (dB) = 10 \times log_{10} \frac{\|s\|_2}{\|\hat{s} - s\|_2}. \tag{3.16}$$

The compression ratio (CR) achieved by CS is given by

$$CR = (1 - \frac{M}{N}) \times 100. \tag{3.17}$$

We first compare our approach with 3 other methods to show the advantages of our proposed technique. We provide a concise summary of employed methods in Table 3.4. The Ordinary CS approach use a regular measurement matrix $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ discussed in [1]. Kronecker CS method segments the signal and measures each segment separately in the time domain [55]. Non-permuted version is similar to the proposed method but does not use the permutation matrix $\mathbf{P}$. Therefore, $\mathbf{I} \otimes \tilde{\mathbf{\Phi}}$ is used as a measurement matrix. Finally, our proposed method employs $\mathbf{A} = (\mathbf{I} \otimes \tilde{\mathbf{\Phi}})\mathbf{P}$ as its measurement matrix.

Table 3.4: Summary of methods.

Ordinary CS	Kronecker CS
$x_{N\times 1} = \mathbf{\Psi}_{N\times N} s_{N\times 1}$	$x_{N\times 1} = (\mathbf{I}_{l\times l} \otimes \tilde{\mathbf{\Psi}}_{n\times n}) s_{N\times 1}$
$\mathbf{\Phi}_{M\times N}$	$\mathbf{\Phi}_{M\times N} = \mathbf{I}_{l\times l} \otimes \tilde{\mathbf{\Phi}}_{m\times n}$
$y_{M\times 1} = \mathbf{\Phi}_{M\times N} x_{N\times 1}$	$y_{M\times 1} = \mathbf{\Phi}_{M\times N} x_{N\times 1}$
$\hat{x}_{N\times 1} = \Re(y, \mathbf{\Phi})$	$\hat{x}_{N\times 1} = \Re(y, \mathbf{\Phi})$
$\hat{s}_{N\times 1} = \mathbf{\Psi}^{-1}_{N\times N} \hat{x}_{N\times 1}$	$\hat{s}_{N\times 1} = (\mathbf{I}_{l\times l} \otimes \tilde{\mathbf{\Psi}}^{-1}_{n\times n}) \hat{x}_{N\times 1}$
Non-permuted	**Proposed method**
$x_{N\times 1} = \mathbf{\Psi}_{N\times N} s_{N\times 1}$	$x_{N\times 1} = \mathbf{\Psi}_{N\times N} s_{N\times 1}$
$\mathbf{\Phi}_{M\times N} = \mathbf{I}_{l\times l} \otimes \tilde{\mathbf{\Phi}}_{m\times n}$	$A_{M\times N} = (\mathbf{I}_{l\times l} \otimes \tilde{\mathbf{\Phi}}_{m\times n}) \mathbf{P}_{N\times N}$
$y_{M\times 1} = \mathbf{\Phi}_{M\times N} x_{N\times 1}$	$y_{M\times 1} = \mathbf{A}_{M\times N} x_{N\times 1}$
$\hat{x}_{N\times 1} = \Re(y, \mathbf{\Phi})$	$\hat{x}_{N\times 1} = \Re(y, \mathbf{A})$
$\hat{s}_{N\times 1} = \mathbf{\Psi}^{-1}_{N\times N} \hat{x}_{N\times 1}$	$\hat{s}_{N\times 1} = \mathbf{\Psi}^{-1}_{N\times N} \hat{x}_{N}$

The methods in Table 3.4 are tested on 10 different ECG signals, each of length 1024 for different CRs of 25%, 50%, and 75% respectively. The basic measurement matrix is chosen to be normalized Gaussian matrix in this case. The linear programming as introduced in [30] is used to solve the optimization problem in the recovery process with the GUROBI solver [31]. However, other optimization techniques for signal recovery can be used as well. We perform 50 CS trials for each signal using different Gaussian random measurement matrices and we show the average SNR and its standard deviation (σ) for the recovered signals in Table 3.5. We observe that the Ordinary CS, Kronecker CS, and the Proposed method provide comparable results, however, Non-permuted version has a poorer performance. The Proposed method has slightly lower SNR ($\sim 1\ dB$) than Ordinary CS, but our method employs ℓ times sparser measurement matrix using the Kronecker structure resulting in substantially fewer computations, storage, and recovery time. In Non-permuted method, segments that have most of the DCT significant values might not be recovered properly and will have poorer overall SNR. In Kronecker CS, different segments are recovered with variable qualities since each segment is compressively sensed independently. In this case, segments having more non-zero components in the DCT domain experience

Table 3.5: Performance comparison of different methods for 10 ECG signals, each of length 1024, using normalized Gaussian measurement matrix (50 trials). SNR and σ in dB.

	Ordinary CS		Kronecker CS		Non-permuted		Proposed method	
CR	SNR	σ	SNR	σ	SNR	σ	SNR	σ
%25	54.08	2.54	50.65	3.94	31.71	3.05	53.38	2.68
%50	43.64	4.47	40.05	5.09	27.89	2.61	42.86	4.78
%75	32.43	4.04	26.43	2.75	25.23	2.54	31.98	3.80

more distortion in their recovery resulting in lower SNR and less smoothness of the recovered signal. Figure 3.4 shows an ECG signal and its recovered version using four methods explained in Table 3.4.

Figure 3.5 provides the average SNR of our proposed method on an ECG signal (105m) for different segmentation levels when $\ell = 1, 2, 4, 8, 16, 32$. In this simulation, $\tilde{\mathbf{\Phi}}$ is a Gaussian measurement matrix. In the same figure, Sparsity Ratio (SR) as well as the recovery time reported by the GUROBI solver are shown. It can be seen that when the segmentation number increases both SR and recovery time decrease while SNR remains almost the same. For a small change in SNR from 46.72 to 46.03 dB, the recovery time drops from 2.98 to 0.29 seconds (more than 10 times faster) with a sparsity ratio that is 32 times better. So, if the basis measurement matrix satisfy RIP, with larger ℓ, the recovery time will be smaller and the measurement matrix $\mathbf{A}$ becomes ℓ times sparser and needs less storage and computations while changes in SNR are very small.

Finally, we perform a comparison by using a known sparse matrix referred as sparse block circulant matrix (SBCM) [59]. SBCM is constructed from $m \times n$ smaller square blocks of size $\ell \times \ell$. Each such block has one Gaussian random number in its first row and each subsequent row is a single circular shifted version of its previous row. We compare with our proposed method using three different sub-matrices, $\tilde{\mathbf{\Phi}}$, i.e. normalized Gaussian ($\mathbf{A}_G$) and BCH ($\mathbf{A}_{BCH}$) as well as Bernoulli ($\mathbf{A}_{Be}$).

Table 3.6 shows the simulated performance (average SNR, SR, and recovery time)

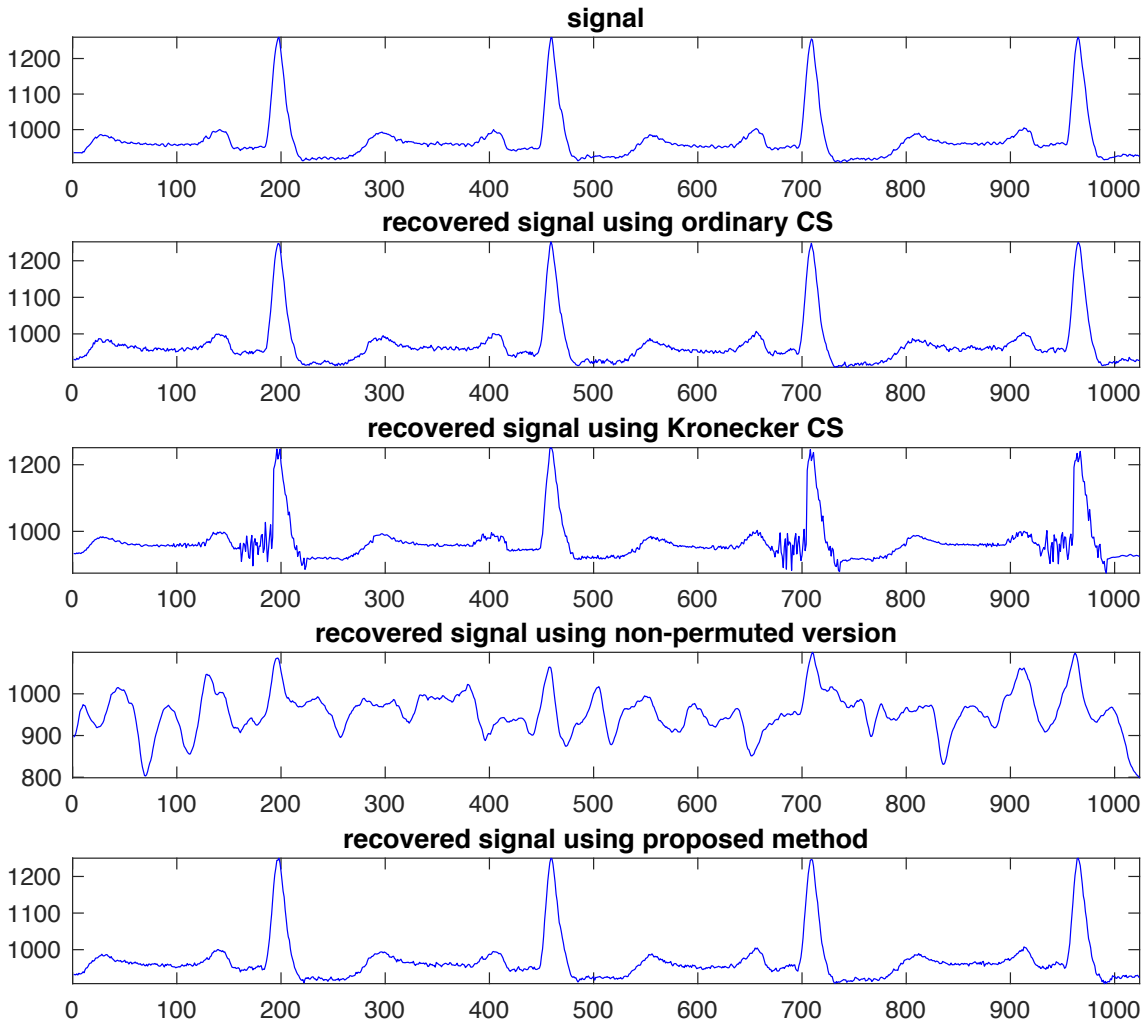

Figure 3.4: An ECG signal (105m) and its recovered version with four methods using Gaussian measurement matrix for $CR = 50\%$ is shown. SNR of the recovered signal is 46.93, 35.39, 25.37, and 49.21 dB respectively.

Table 3.6: Performance comparison of proposed method ($\mathbf{A}$), ordinary CS ($\mathbf{\Phi}$), and SBCM on an ECG signal (105m) for three CRs: 28%, 50%, 72%. Indices, G, Be, and BCH stand for Gaussian, Bernoulli, and BCH-based matrices respectively. Block length (ℓ) is 8.

Method	CR ≈	SNR	sparsity ratio (SR)	recovery time
$\mathbf{A}_G$	28	54.5366	0.1250	0.8034
$\mathbf{A}_{Be}$		54.4804	0.0623	0.8590
$\mathbf{A}_{BCH}$		60.1087	0.0185	0.2031
$\mathbf{\Phi}_G$		54.7297	1.0000	5.5953
$\mathbf{\Phi}_{Be}$		54.7359	0.4997	4.0181
$\mathbf{\Phi}_{BCH}$		60.4028	0.1412	0.4177
$SBCM$		54.6697	0.1250	1.7371
$\mathbf{A}_G$	50	45.3863	0.1250	0.4916
$\mathbf{A}_{Be}$		45.6805	0.0636	0.5130
$\mathbf{A}_{BCH}$		52.2955	0.0311	0.3076
$\mathbf{\Phi}_G$		46.9778	1.0000	2.9406
$\mathbf{\Phi}_{Be}$		47.0447	0.5008	2.2968
$\mathbf{\Phi}_{BCH}$		52.8919	0.2508	0.5010
$SBCM$		46.4975	0.1250	1.0902
$\mathbf{A}_G$	72	30.8473	0.1250	0.3340
$\mathbf{A}_{Be}$		30.9977	0.0645	0.3282
$\mathbf{A}_{BCH}$		42.3825	0.0465	0.2274
$\mathbf{\Phi}_G$		32.9922	1.0000	1.2996
$\mathbf{\Phi}_{Be}$		32.9444	0.5001	1.0876
$\mathbf{\Phi}BCH$		42.8277	0.3600	0.3786
$SBCM$		33.1810	0.1250	0.6767

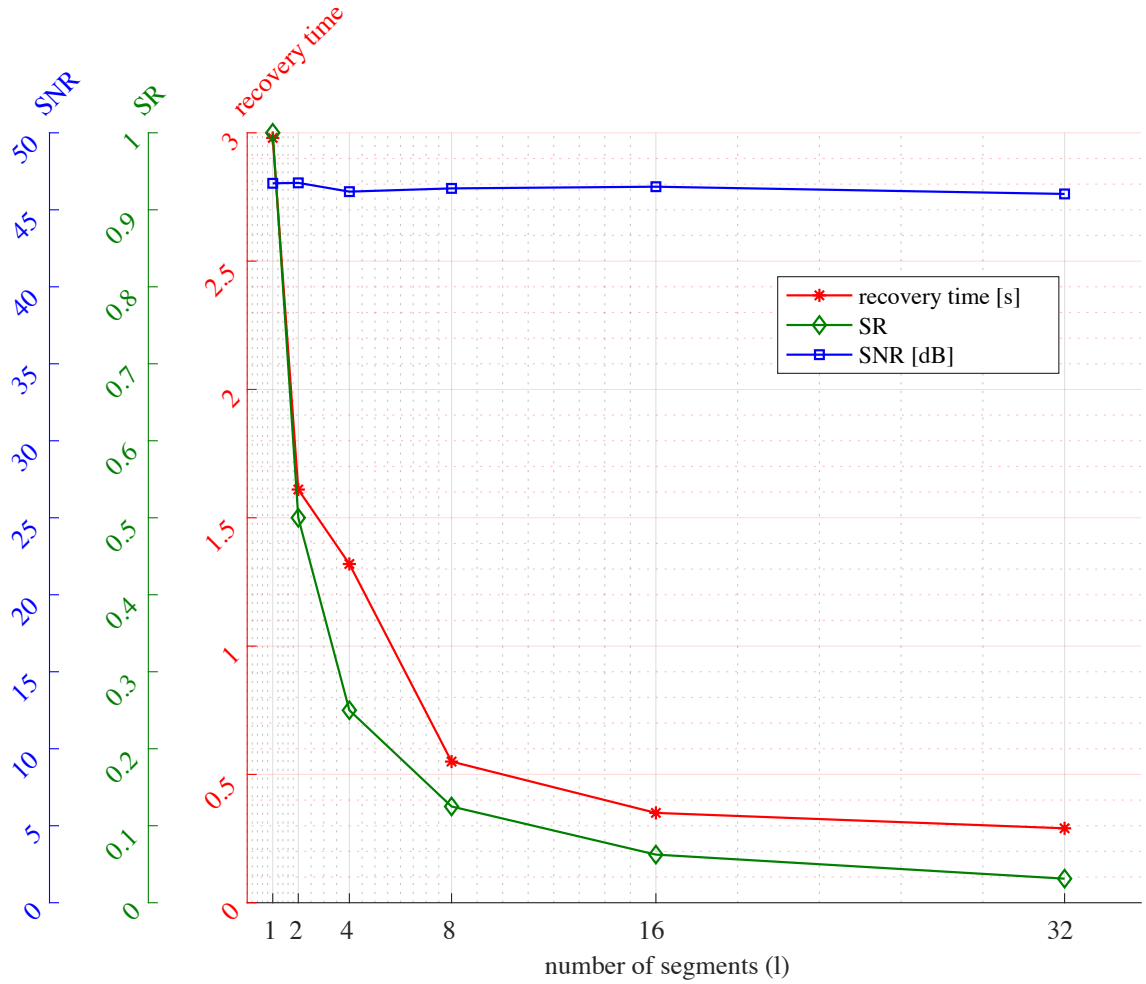

Figure 3.5: Figure shows average SNR, sparsity ratio (SR), and recovery time for an ECG signal (105m) as a function of segmentation with $CR = 50\%$.

for different methods on an ECG signal (105m). In this simulation, $N = 1016$, $\ell = 8$ and three CRs: 28%, 50%, 72% are considered where size of sub-matrices, $\tilde{\mathbf{\Phi}}$, are: (91×127), (63×127), (35×127), respectively. For each method employing the random sensing matrices, we average the results over 100 trials. We observe that the highest SNR is associated with the BCH-based sensing matrix both for our proposed Method and for ordinary CS. Our proposed method $\mathbf{A}_{BCH}$, provides smaller sparsity ratio and recovery time while the SNR performance is practically the same. CS with $SBCM$ has a comparable SNR and sparsity ratio with our method $\mathbf{A}_G$; however, the recovery time using $\mathbf{A}_G$ is substantially less (on average twice smaller). As an additional observation, BCH sub-matrices give better performance but they

are designed with fixed dimensions [70], [68] and therefore, are not adjustable to any arbitrary CR.

In summary, the best combined performance in terms of SNR, sparsity ratio, and recovery time is attributed to $\mathbf{A}_{BCH}$.

Chapter 4

Efficient Compresive Sensing Cryptosystem

Security is an important criterion in many applications in CS-based communication. For example, personal remote health monitoring devices which record vital signals and communicate them to a cloud system for further assessment would not like to disclose the information to a non-authorized party. A literature review on the security aspects of CS has been provided in Section 2.2. We propose a structure for CSC and compare it with a recent most relevant method in the literature called S-OTS [65].

In this chapter, we design an efficient one-time sensing matrix based on the Kronecker product structure which was discussed in the previous chapter to enhance security. The proposed block diagonal sensing matrix is multiplied with a pseudo random permutation matrix for enhancing security. As has been mentioned in the previous chapter, our Kronecker-based method provides several advantages such as lower storage and computational cost. Also, using simulations, we show that the method is secure against Cipher-text-Only Attack (COA). We further show our technique is computationally secure against Plain-text Attacks (PA).

4.1 Proposed Method

We propose a Kronecker-based one-time sensing matrix that provides enhanced computational efficiency and security. The proposed measurement matrix is created from smaller eligible securely generated CS sub-matrices and a permutation matrix as shown below:

$$\mathbf{A} = (\mathbf{I} \otimes \tilde{\mathbf{\Phi}})\mathbf{P} \tag{4.1}$$

where $\mathbf{A} \in \mathbb{R}^{M \times N}$ is the proposed measurement matrix, $\mathbf{P} \in \mathbb{R}^{N \times N}$ is a pseudo random permutation matrix, $\mathbf{I} \in \mathbb{R}^{\ell \times \ell}$ is the identity matrix, $\tilde{\mathbf{\Phi}} \in \mathbb{R}^{m \times n}$ is a securely generated measurement matrices in CS, $\otimes$ is the Kronecker multiplication operation, $M = \ell \times m$, $N = \ell \times n$, and ℓ is the number of segments of the signal. For instance, $\tilde{\mathbf{\Phi}}$ can be either an i.i.d. Gaussian matrix, a Bernoulli, or a Bipolar matrix. The i.i.d. Gaussian sub-matrices are generated using a secure Gaussian Random Number Generator (GRNG). The Bernoulli and bipolar sub-matrices are generated using Self Shrinkage Generator (SSG) [40] which are discussed in detail in Sections 1.2.2 and 1.2.2. All random matrices are generated using an initial seed/key which is communicated through the secure channel in our approach. Permutation matrix $\mathbf{P}$ which is generated using (SSG) [40] serves two purposes:

- Distributes all non-zeroes of a k-sparse signal, x, in a uniform fashion across the entire signal (and within its every l segments of length n), so that $\mathbf{P}x$ has a block-sparse structure. In this way, $\mathbf{P}x$ has ℓ segments/blocks, each with expected sparsity of k/ℓ. Then, if $\tilde{\mathbf{\Phi}}$ recovers each segment individually, the designed block diagonal matrix $(\mathbf{I} \otimes \tilde{\mathbf{\Phi}})$ recovers the whole $\mathbf{P}x$ properly.
- Increases the security of the method with scrambling and adding more randomness to the measurement matrix.

The reduction in the storage and computational costs of this method is similar to the one discussed in the previous chapter since both methods use similar structure (see Section 3.2).

4.2 Security Analysis

In order to enhance security, the sensing matrix is regenerated for each new sensing. For the generation of such matrices, we need to efficiently produce random numbers. In the following paragraphs, we explain a fast and efficient method for generating random numbers [40].

An efficient Pseudo Random Number Generator (PRNG) (called Shrinking Generator (SG)) based on a combination of two LFSRs is presented in [39]. According to this method, we consider two bit streams $a = (a_0, a_1, ...)$ and $b = (b_0, b_1, ...)$ from two LFSRs. The output bit of SG , $c = (c_0, c_1, ...)$, is defined by considering each pair of

(a_n, b_n). We repeat output bit of SG here for convenience.

$$c_n = \begin{cases} b_n & \text{if } \mathrm{a_n} = 1 \\ \text{Discard pair} & \text{if } \mathrm{a_n} = 0. \end{cases} \tag{4.2}$$

Motivated by SG, the self-shrinking generator (SSG) based on a single LFSR was proposed in [40]. In this method, two consecutive bits of a single LFSR, (a_n, a_{n+1}), is considered. The output bit of SSG is generated using the same methodology as SG use where b_n is replaced by a_{n+1} (see Sections 1.2.2 and 1.2.2).

To create $\mathbf{P}$, we start with an identity matrix of size N. Then, we use SSG to create N distinct random numbers. The matrix $\mathbf{P}$ is created by rearranging columns of the identity matrix. We sequentially select columns of identity matrix according to the generated array of N distinct random numbers and place them in $\mathbf{P}$. The number of SSG bits needed to shuffle the identity matrix to generate $\mathbf{P}$ is in the order of $Nlog_2(N)$ [65].

For generating $\tilde{\mathbf{\Phi}}$ when it is a Bernoulli or Bipolar random matrix we need only one random bit for each of its entries. SSG [40] is used to generate random sequence $a = (a_0, a_1, ...)$ which will then represent the entries of Bernoulli matrix. In the case that $\tilde{\mathbf{\Phi}}$ is Bipolar matrix, the output bits of SSG are transformed to -1 and 1 following the mapping $(-1)^{a_n}$. Therefore, for the generation of Bernoulli or Bipolar matrices $\tilde{\mathbf{\Phi}}$, we need $m \times n$ SSG bits.

When $\tilde{\mathbf{\Phi}}$ is an i.i.d. Gaussian measurement matrix, we rely on the application of the central limit theorem in conjunction with a random number generator. A wide range of GRNGs has been investigated in the literature and some are discussed in Section 1.2.2. GRNG based on cyclic rotations proposed in [44] is an efficient method, but it is not secure enough. This means that if an adversary has the generated Gaussian number, the size of solution space to find the initial seeds of the B-bits LFSR is not large enough. Also, the GRNG proposed in [43] needs $4 \times B$ LFSR bits therefore generates an uncorrelated Gaussian number each $4 \times B$ clocks. In this work, we modify the method in [43] and consider four independent B-bits LFSRs as shown in Figure 4.1 which are then added (element-wise) to generate a $(B + 2)$-bits i.i.d Gaussian number. In this way, each B-clock results in an i.i.d Gaussian number. To generate an i.i.d Gaussian measurement matrix with this method, we need $m \times n \times 4 \times B$ bits. Figure 4.2 shows the histogram of 1,000,000 samples of

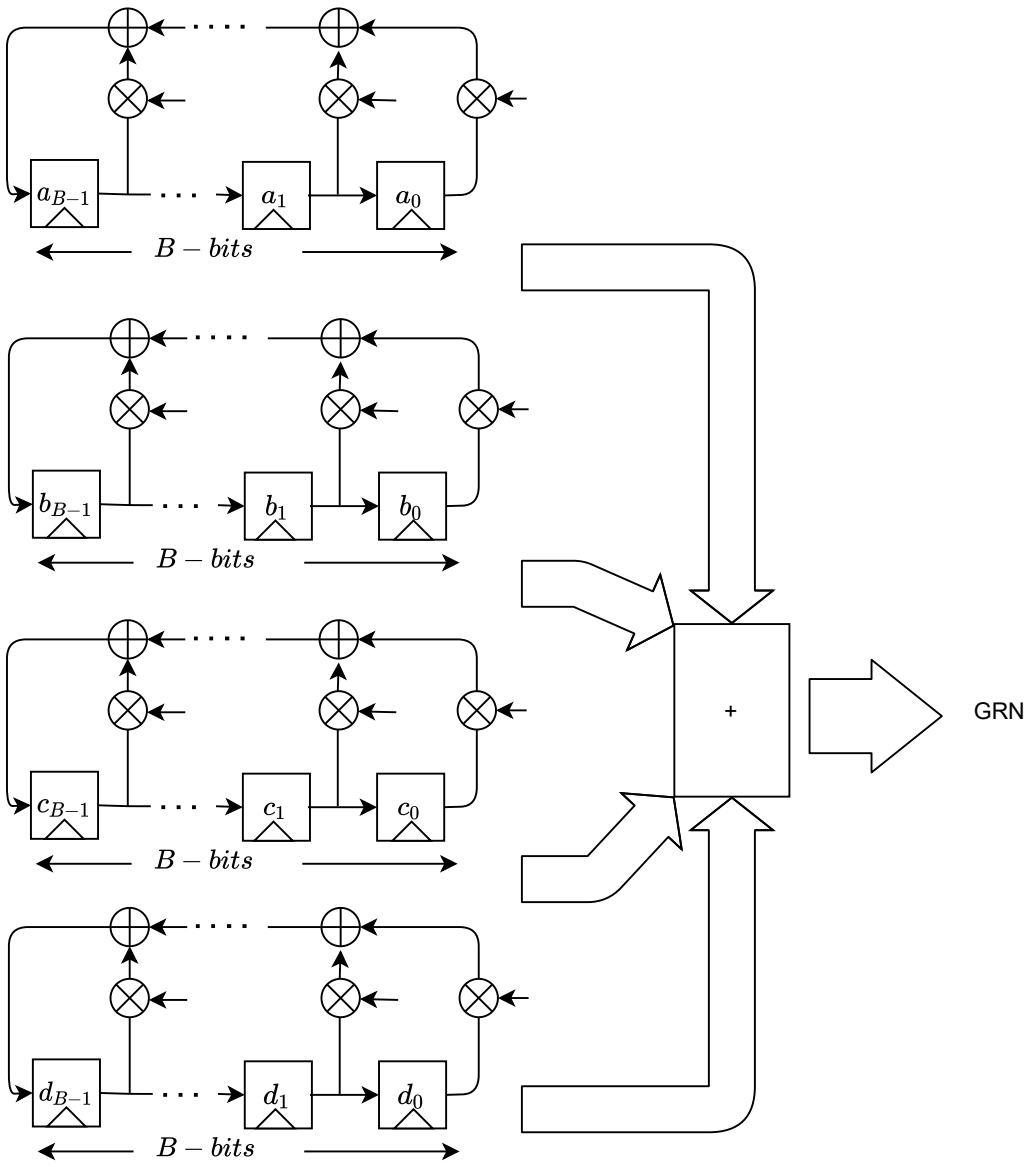

Figure 4.1: GRNG using four B-bits LFSRs

generated random numbers with the proposed method. It can be seen that it has Gaussian distribution and the samples auto-correlation function shows a spike that confirms that the generated Gaussian random numbers are uncorrelated therefore i.i.d (since it is Gaussian).

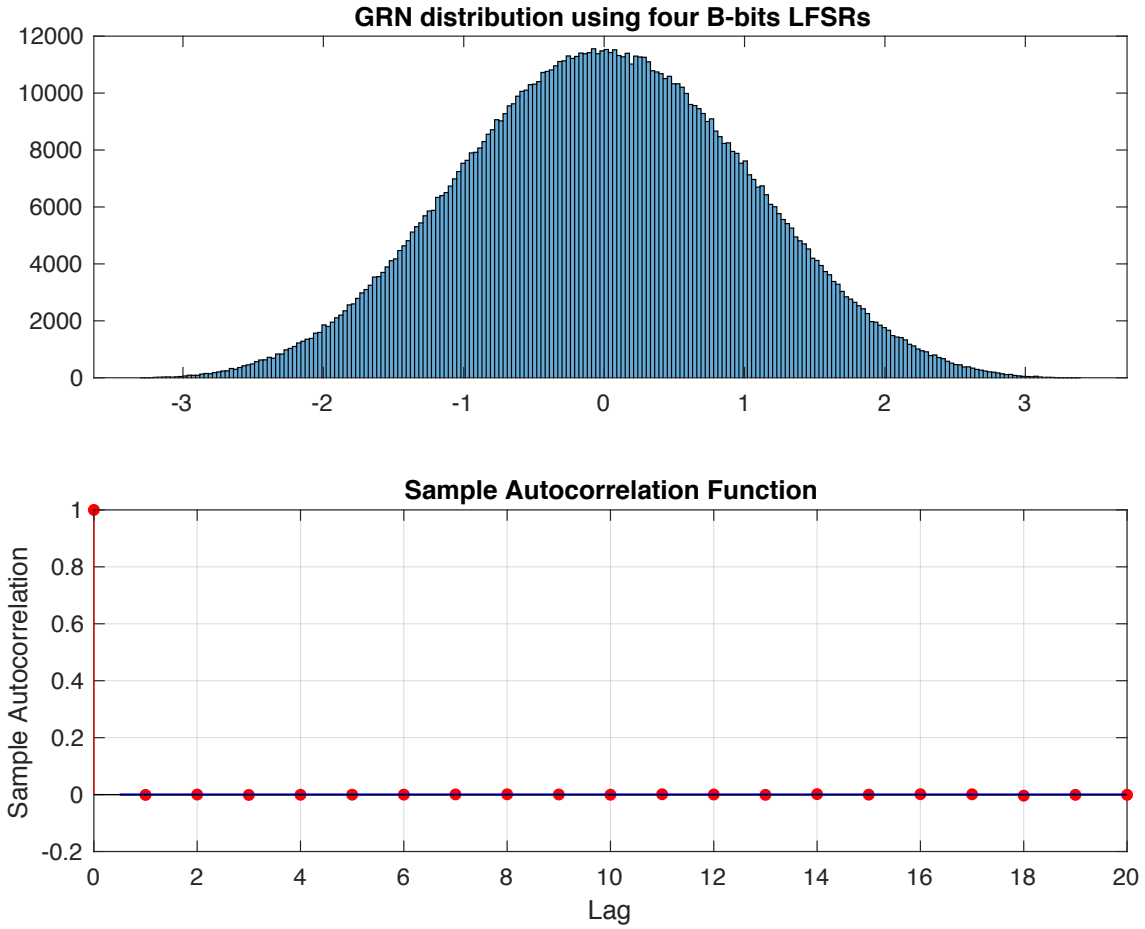

Figure 4.2: Histogram of 1,000,000 Gaussian random number samples using four B-bits LFSRs along with the auto correlation function.

In our security analysis, we have two assumptions. Firstly, the structure of measurement matrix is known to the attacker. It means that the size of measurement matrix A, size of smaller sub-matrices $\tilde{\mathbf{\Phi}}$, and the class of sub-matrices (i.e., Gaussian, Bernoulli, Bipolar) are known to the adversary. Secondly, the permutation matrix is disclosed to the attacker which means no further security layer is added by permutation matrix and the security comes entirely from matrix $\tilde{\mathbf{\Phi}}$. We consider two threats

to this security scheme. The weaker threat is to find the measurement matrix partially and utilize it to decrypt a single transmission (since we use OTS matrix in our method). A stronger threat is to find the secret key used for generating $\tilde{\mathbf{\Phi}}$ and try to construct the measurement matrix for each sensing and decrypt the whole plaintext. In this case, security relies on LFSRs and random number generators which are used to generate the random matrices.

We analyze the security properties of the measurement matrix through two different types of attacks: Cipher-text-Only Attack (COA) and Plain-text Attack (PA). In COA, the adversary has information only about the cipher-text y and tries to find the corresponding plain-text x. Furthermore, we assume that the adversary has knowledge about the structure and the type of measurement matrix. We consider the case that adversary also completely knows the permutation matrix $\mathbf{P}$ but not $\tilde{\mathbf{\Phi}}$. To test the security of our scheme, we generate a matrix $\hat{\mathbf{\Phi}}$ (a modified/corrupted version of $\tilde{\mathbf{\Phi}}$) and create the corresponding Kronecker-based measurement matrix $\hat{\mathbf{A}}$. We perform the measurement and recovery with $\mathbf{A}$ and $\hat{\mathbf{A}}$, respectively. Then, we compute the correlation between the recovered signal and the original one to assess their similarities. Simulation results are presented in Section 4.3.

For the PA evaluation of our method, we assume that a pair of chosen plaintext and ciphertext (x, y) is intercepted by an adversary who tries to guess the measurement matrix $\mathbf{A}$ and ultimately the secret key for its generation. To assess the security, we need to answer the following two questions:

- What is the size of solution space in terms of the elements of $\mathbf{A}$ such that $y = \mathbf{A}x$ for a given pair of (x, y)?
- How difficult is it to determine the key/seed for $\mathbf{A}$, if the adversary has knowledge of samples of $\mathbf{A}$?

We address both the previous questions in the following paragraphs and show that our technique has a large solution space both for the determination of $\mathbf{A}$ and its key/seed. Therefore, our method is computationally secure against PA.

To find the size of solution space of $y = \mathbf{A}x$, let us first assume the following simplified equation where the permutation matrix $\mathbf{P}$ is removed.

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{\Phi}} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{\Phi}} & \dots & \mathbf{0} \\ & & \ddots & \\ \mathbf{0} & \mathbf{0} & \dots & \tilde{\mathbf{\Phi}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}. \tag{4.3}$$

We express the first row of (4.3) as

$$y_1 = \sum_{i=1}^{n} \tilde{\phi}_{1i} x_i, \tag{4.4}$$

where x_i and y_1 are known and we are seeking the size of solution space for coefficients $\tilde{\phi}_{1i}$. Equation (4.4) can be written as follows:

$$y_1 = \sum_{i=1}^{n} q_i \tag{4.5}$$

where $q_i = \tilde{\phi}_{1i} x_i$. Now, by solving for q_i, and having x_i, one can find $\tilde{\phi}_{1i} = q_i / x_i$. Therefore, the size of solution space of problem (4.4) and (4.5) are similar.

We assume each q_i is represented by B bits. So, the number of distinct values for q_i is $L = 2^B$. Since there are L possible values for each q_i and $i \in 1, ..n$, then there are L^n different combinations for the summation in (4.5). On the other hand, summation of q_is can be accommodated in $B + \lceil log_2(n) \rceil$ where $\lceil log_2(n) \rceil$ represents the number of carryout bits and $\lceil \cdot \rceil$ represents the ceiling function that maps a variable to the least integer greater than the variable. Therefore, y_1 has $2^{(B+\lceil log_2(n) \rceil)}$ combinations. Assuming y_1 is uniformly distributed (it would be the same order even if it has Gaussian distribution), the expected number of solutions for (4.5) is

$$S_{\tilde{\mathbf{\Phi}}}(n, L) = \frac{L^n}{2^{(B+\lceil log_2(n) \rceil)}} = \frac{L^{n-1}}{2^{\lceil log_2(n) \rceil}}. \tag{4.6}$$

For the case of the Kronecker structure in (4.3), since with a pair of (x, y), we can have ℓ equations, the size of solution space derived in (4.6) is divided by ℓ and leads to the following solution:

$$S(n, L, \ell) = \frac{L^{n-1}}{\ell \times 2^{\lceil log_2(n) \rceil}} \tag{4.7}$$

where $\ell = N/n$ is the number of signal segments (or blocks). In our study, we consider the case where $\mathbf{P}$ is known. Thus, (4.7) provides the size of solution space for finding $\mathbf{A}$ with a known pair of (x, y). The inclusion of unknown $\mathbf{P}$ results in increased $S(n, L, \ell)$. Next, we discuss the complexity of finding the initial key/seed for the generation of $\tilde{\mathbf{\Phi}}$ when a sample of $\mathbf{A}$ has been intercepted.

When $\tilde{\mathbf{\Phi}}$ is Gaussian, the complexity of finding the key by knowing $\mathbf{A}$ follows a similar analysis for deriving (4.6). In this case, q_i, $i = 1, 2, 3, 4$ form the output bits of the LFSRs and y_1 is the generated Gaussian random number. Adding the outputs of four LFSRs result in $B + \lceil log_2(4) \rceil = B + 2$ bits which has Gaussian distribution. There are L^4 combinations for the sum of q_is and the result has $2^{B+2} = L \times 4$ distinct combinations. Therefore, the expected size of solution space for q_i is:

$$S_q(L) = \frac{L^4}{L \times 4} = \frac{L^3}{4} \tag{4.8}$$

As an example, if we use four 64-bits LFSRs, the size of solution space is $\frac{(2^{64})^3}{4} = 2^{190}$ which is a large number (we assume numbers $> 2^{64}$ are large enough).

When $\tilde{\mathbf{\Phi}}$ is Bernoulli or Bipolar, the generated sequence using SSG has shown to be secure according to [40].

We summarise our analysis in Table 4.1 where we show the sparsity, number of required LFSR bits, and size of the solution space of our proposed method using Gaussian, Bernoulli, and Bipolar sub-matrices (A_G, A_{Be}, A_{Bi}) respectively. We compare against ordinary CS using Gaussian, Bernoulli, and Bipolar measurement matrices (Φ_G, Φ_{Be}, Φ_{Bi}) as well as the recent S-OTS method [65]. Our proposed sensing matrix is ℓ times sparser compared to its ordinary CS counterpart. The sparsity of the proposed method using Bipolar sub-matrices is similar to S-OTS while A_{Bi} requires ℓ times less LFSR bits. It can be seen that the size of the solution space in A_{Bi} and S-OTS is of the same order. However, the required number of LFSR bits to generate A_{Bi} is ℓ times smaller than S-OTS. Furthermore, comparing our method using Gaussian matrix (A_G) against S-OTS, we can see that if $4 \times B = \ell$, then the two methods have the same number of LFSR bits. However, A_G, because of its Gaussian entries, leads to better security.

Table 4.1: Comparison of sparsity, number of LFSR bits, and size of the solution space for different CS methodologies. All methods need an extra $Nlog_2(N)$ LFSR bits for generating $\mathbf{P}$.

Method	sparsity	LFSR bits	S
A_G	$1/\ell$	$m \times n \times 4 \times B$	$\frac{2^{B(n-1)}}{\ell \times 2^{\lceil log_2(n) \rceil}}$
A_{Be}	$1/2\ell$	$m \times n$	$\frac{2^{(n-1)}}{\ell \times 2^{\lceil log_2(n) \rceil}}$
A_{Bi}	$1/\ell$	$m \times n$	$\frac{2^{(n-1)}}{\ell \times 2^{\lceil log_2(n) \rceil}}$
Φ_G	1	$m \times n \times \ell^2 \times 4 \times B$	$\frac{2^{B(N-1)}}{2^{\lceil log_2(N) \rceil}}$
Φ_{Be}	1/2	$m \times n \times \ell^2$	$\frac{2^{(N-1)}}{2^{\lceil log_2(N) \rceil}}$
Φ_{Bi}	1	$m \times n \times \ell^2$	$\frac{2^{(N-1)}}{2^{\lceil log_2(N) \rceil}}$
S-OTS	$1/\ell$	$m \times n \times \ell$	$\frac{2^{(n-1)}}{2^{\lceil log_2(n) \rceil}}$

4.3 Simulation and Results

We apply our proposed technique on a set of ECG signals from the MIT-BIH Arrhythmia database [71]. Linear programming as proposed in [30] is used for the signal recovery using the Gurobi [31] solver. Discrete Cosine Transform (DCT) is chosen to be the sparsifying basis. Signal-to-Noise Ratio (SNR) is used to measure the quality of recovered signal.

Our proposed method is evaluated using three different sub-matrices namely i.i.d. Gaussian, Bernoulli, and Bipolar. The corresponding measurement matrices are denoted by A_G, A_{Be}, and A_{Bi} respectively. First, the recovered signal quality is evaluated on ECG signals with length $N = 1024$. Then, the security performance of the proposed methodology is evaluated and contrasted against the recent work on efficient CSC (S-OTS) [65].

Figure 4.3 depicts the quality of recovered signal using A_G, A_{Be}, A_{Bi}, and S-OTS. The average SNR over 100 trials for 10 ECG signals is shown for different ℓ. It can be seen that performance of our approach using A_G, A_{Be}, and A_{Bi} matrices is comparable to S-OTS. A slight decrease ($< 3dB$ in the worst case) can be observed

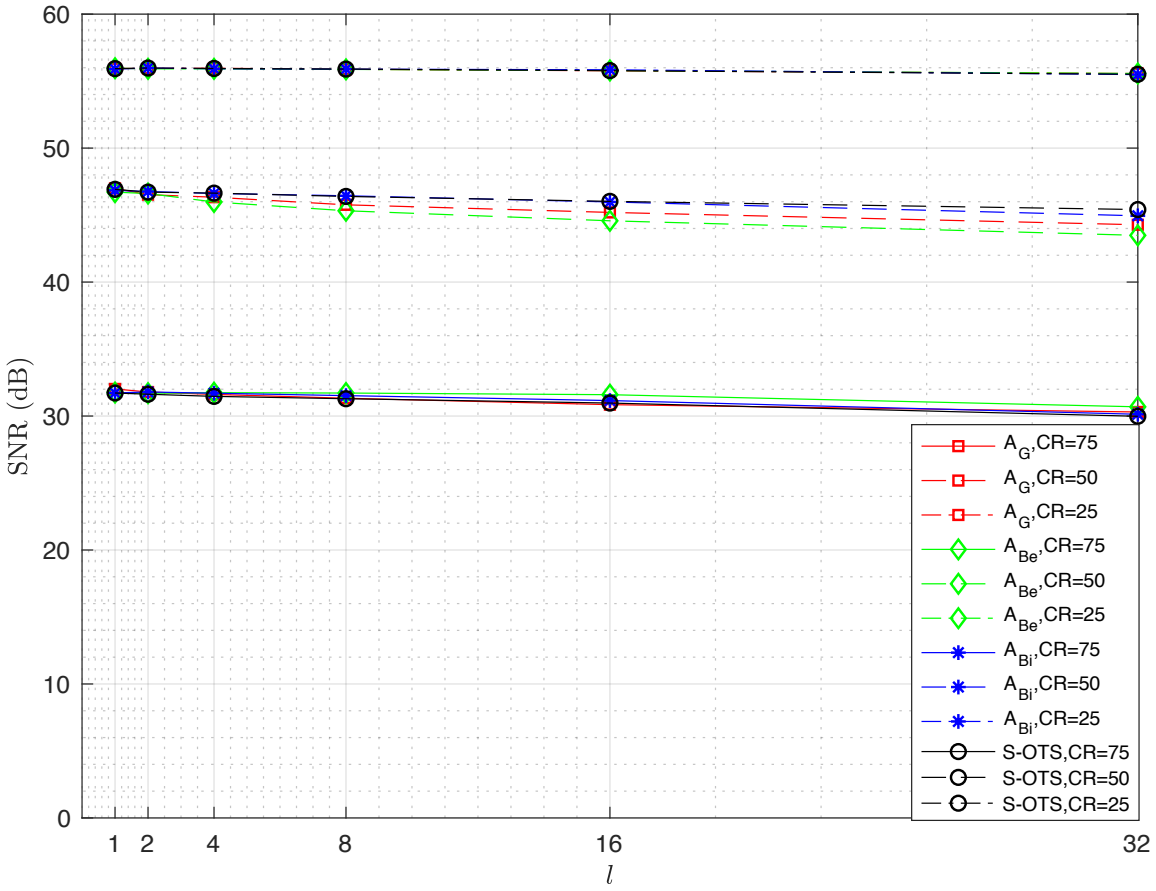

Figure 4.3: Quality of recovered signal using S-OTS and our proposed method with Gaussian (A_G), Bernoulli (A_{Be}), and Bipolar (A_{Bi}) sub-matrices in shown. The average SNR of 10 ECG signals over 100 trials is calculated for different ℓ and $CR = 25, 50, 75\%$.

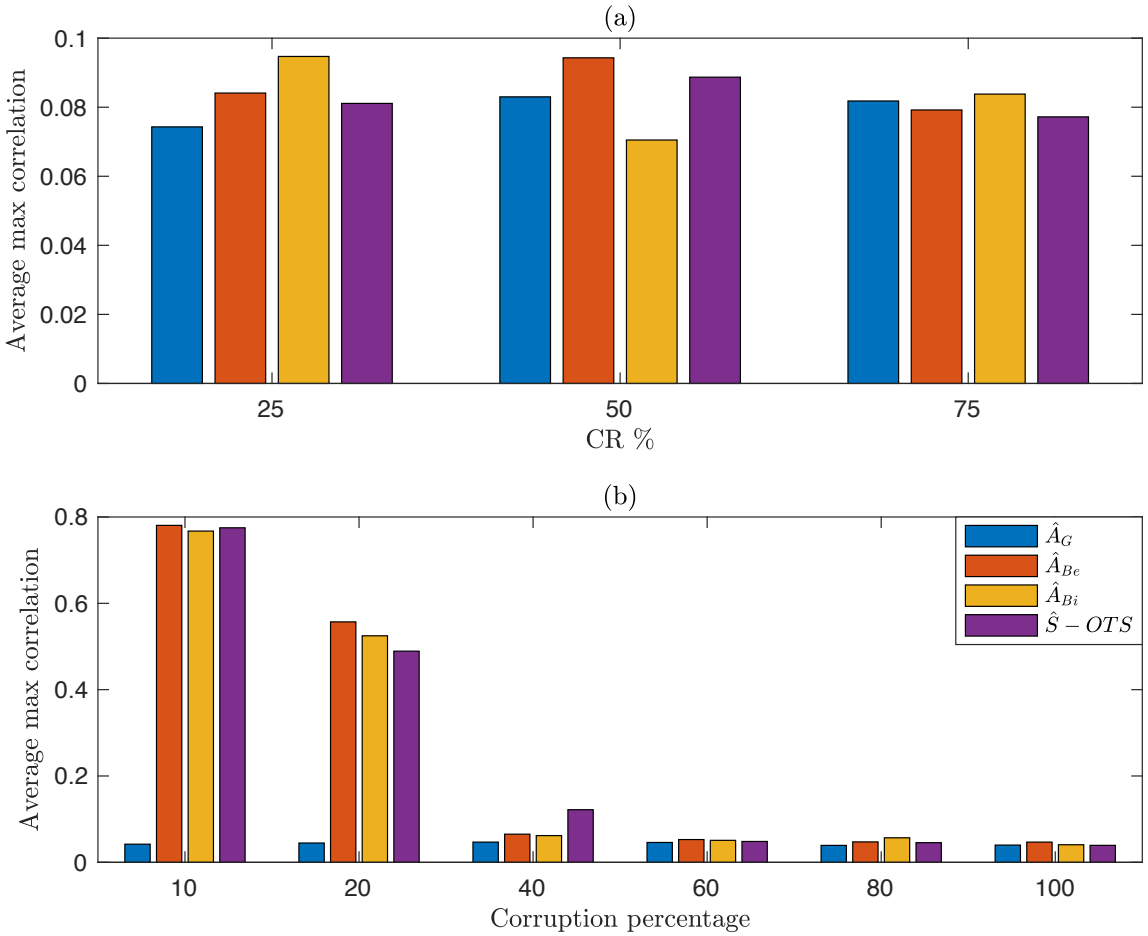

Figure 4.4: Both figures show the average maximum correlation between the recovered signals over 100 trials and the 10 ECG signals. (a) refers to 100% sensing matrix corruption while (b) utilizes $CR = 50\%$.

in the SNR when ℓ gets larger. Choosing larger values of ℓ results in a sparser and more efficient measurement matrix.

We evaluate the security of our proposed method against COA by recovering the signal using $\hat{\mathbf{\Phi}}$ with all of its entries being different compared to $\tilde{\mathbf{\Phi}}$ (i.e., 100% corruption). We let $\hat{\mathbf{A}} = (\mathbf{I} \otimes \hat{\mathbf{\Phi}})\mathbf{P}$ (assume $\mathbf{P}$ is known). We then find the maximum correlation between the original signal and recovered signal using $\hat{\mathbf{A}}$. Furthermore, our proposed approach is compared to S-OTS method. In order to corrupt S-OTS measurement matrix, we corrupt the non-zero entries of the matrix [65].

Figure 4.4(a) displays the average maximum correlation across 10 ECG signals over 100 sensing trials with different measurement matrices: $\hat{A}_G$, $\hat{A}_{Be}$, $\hat{A}_{Bi}$,

and $\hat{S}$-OTS. The non-zero entries of the measurement matrices are 100% corrupted (which means sub-matrices are regenerated) and the results are presented for $CR = 25, 50, 75\%$. The simulation shows the max correlation in all methods is very low (less than 0.1). This indicates that recovery with 100% corruption of the non-zeros of measurement matrix will not provide any information regarding the original signal. So, the methods are computationally secure against COA. Figure 4.4 (b) shows the average maximum correlation for different corruption percentages of the non-zero entries of the measurement matrix when CR = 50%. It can be seen that the average maximum correlation using $\hat{A}_G$ even if 10% of $\tilde{\mathbf{\Phi}}$ is corrupted (i.e., 90% is known to the adversary), the average maximum correlation is very low and is typically less than 0.1. However, the signal recovery with 10% and 20% corruption of $\hat{A}_{Be}$, $\hat{A}_{Bi}$, $\hat{S} - OTS$ results in substantially a higher correlation. It is because of the binary entries of the measurement matrices. Therefore, with 50% chance, the corrupted entries may be equal to the original entries of the measurement matrix. More than 40 % corruption of the measurement matrices A_{Be}, $\hat{A}_{Bi}$, and $\hat{S}$-OTS leads to correlation of 0.1 or less in the recovered signals.

Figure 4.5 displays an original ECG signal (105m) with length 1024 and the recovered signal with 10% corrupted measurement matrix for different methodologies, and CR=50%. For each method, amongst the recovered signals from 100 trials, the one having the maximum correlation with the original ECG signal is shown.

We analyzed the security of our proposed method against PA by evaluating the size of solution space to equation (4.3) when the adversary knows a pair of plain-text x, its corresponding cipher-text y, and $\mathbf{P}$. The adversary will need to do an exhaustive search in the solution space which can become very large. As an example, when $N = 2048$, $l = 16$, $n = 128$, $\tilde{\mathbf{\Phi}}$ is i.i.d. Gaussian matrix and each of Gaussian entries are stored in $B = 64$ bits, $L = 2^B$, the size of solution space is (given by (4.7)) $\frac{(2^{64})^{127}}{16\times128} = 2^{8117}$. However, if $\tilde{\mathbf{\Phi}}$ is Bipolar or Bernoulli matrix, each of its entries can be stored in a single bit ($B = 1$), the size of solution space is $\frac{(2)^{127}}{16\times128} = 2^{116}$ which still is a large number.

The complexity of finding the initial key/seed of $\tilde{\mathbf{\Phi}}$ when a sample of A is eavesdropped is evaluated by the size of solution space driven by (4.8). The security of finding the initial key when $\tilde{\mathbf{\Phi}}$ is Gaussian, Bernoulli, or Bipolar matrix has been discussed in Section 4.2.

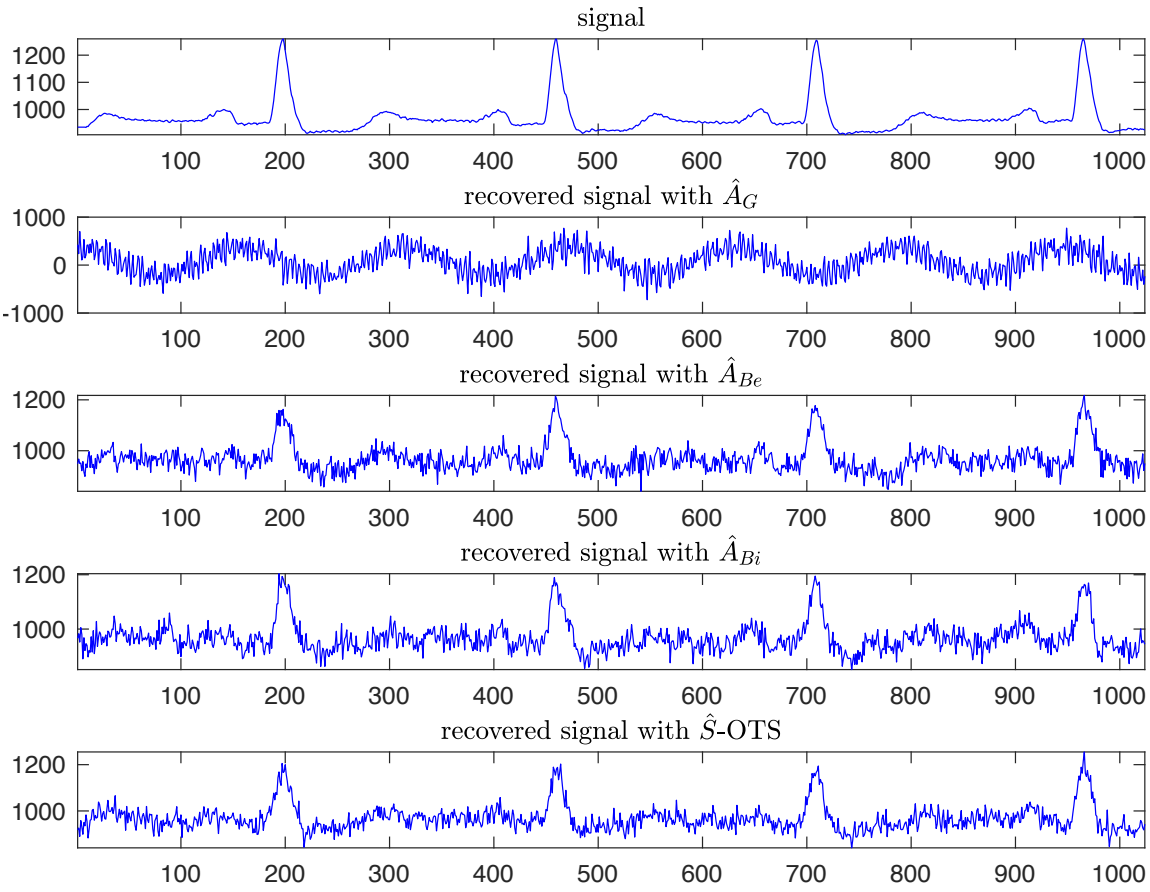

Figure 4.5: An ECG signal (105m) and its recovered version with 10% corrupted measurement matrix is shown. Recovered version shows the maximum correlation over 100 trials for $N = 1024$, $l = 16$, $CR = 50\%$.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

A practical CS method needs to be efficient in terms of time and space as well as the recovered signal quality. In this thesis, we aimed to address these requirements. Furthermore, we improve our proposed method to provide security for CSC applications.

We presented a method for the general design of a measurement matrix for CS and recovery and tested it on real ECG signals (discussed in Chapter 3). We proposed a sparse interleaved Kronecker-based measurement matrix generated by smaller size sub-matrices that existed in CS. Our results show that our method offers comparable performance (in terms of recovered signal quality) with standard CS methodologies that appeared in literature. The main advantage of our proposed technique is l times reduction in the sparsity of the measurement matrix and consequently storage, computations, and recovery time. Furthermore, our method can be easily extended to other random or deterministic measurement matrices that satisfy the requirements of compressive sensing.

We then improved our method and proposed a general sparse one-time sensing matrix for efficient and computationally secure CSC (discussed in Chapter 4) . We utilized a random permutation matrix to uniformly distribute non-zero entities of the sparse signal. Then, we sensed the permuted signal with a block diagonal sensing matrix. Our method is compared against ordinary CS methods using ECG signals chosen from the MIT-BIH Arrhythmia database and demonstrates enhanced performance in terms of computations and sparsity. We also compared our proposed technique to a recent method (S-OTS) in [65]. We showed that S-OTS is similar to our method (when bipolar sub-matrices are used) in terms of sparsity, security, and the recovered

signal quality. However, our approach is superior to S-OTS in terms of computational requirements (ℓ times less LFSR bits in hardware implementation). Finally, the computational aspect of the security against COA and PA was studied. For the security evaluation against COA, we have shown that the recovered signal (using an altered measurement matrix) has a low correlation with the original signal. In the security analysis against PA, we estimated the size of the solution space for finding the sensing matrix using the knowledge of the sensed signal and the corresponding measurement vector. We further showed that the size of solution space is large in practice and our proposed method is computationally secure against PA.

5.2 Future Work

- Further analysis with different sparsifying basis is recommended as better sparsification might improve the recovered signal quality. For instance, different wavelet transforms (such as Haar, Daubechies) may be considered. Alternatively adaptive basis may be an option.

- Other random or deterministic sub-matrices in the CS or CSC literature can be used in our proposed technique. This is one of the advantages of our proposed method. Our proposed methods can be applied to any combination of measurement and sparsification matrices. Some candidate measurement matrices are: LDPC based matrices, chirp matrices, and Reed-Muller code based matrices. Also one can further investigate the connection of CS to coding theory and utilize some available matrices existing in the coding theory literature.

- One can study the joint effect of sparsifying basis and measurement matrices and try to achieve the best combined result in terms of computation costs and recovered signal quality. Also, different size of sub-matrices instead of using same size sub-matrices can be considered. Further study can be done to find the best permutation matrix $\mathbf{P}$ which distribute non-zeros (or most significant values) of a given signal uniformly. Also, multiplying the whole measurement matrix with a second permutation matrix from the left side (to do row-wise permutation) may be considered.

- When using BCH sub-matrices in our method, we found outstanding results compared to other sub-matrices. However, BCH matrices' dimensions are fixed

and can have only certain values. Therefore, BCH matrices are not easily adjustable for different compression ratios. One can investigate to combine different BCH matrices to achieve more available sizes and therefore possibility of design for various compression ratios. The proposed method can be applied to multi-dimensional signals as long as the signals are sparse in some domain. For example, gray scale images, such as MRI images can be considered. Such images may be compressed using BCH-based sub-matrices in conjunction with wavelet transforms as sparsifying basis. Security of such compressed images may also be analyzed.

- The security of a provided encryption method is not always proved nor guaranteed. Our analysis is based on size of the solution space. Security resistance to different analytic attacks can be further investigated. Also, different random number generators can be used to improve both security and efficiency.

List of References

[1] Y. Eldar and G. Kutyniok, *Compressed Sensing: Theory and Applications*. Cambridge, United Kingdom: Cambridge University Press, 2012.

[2] E. J. Candes and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, pp. 21–30, 2008.

[3] D. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, pp. 1289–1306, 2006.

[4] R. G. Baraniuk, "Compressive sensing [lecture notes]," *IEEE Signal Processing Magazine*, vol. 24, pp. 118–121, 2007.

[5] J. A. Tropp, J. N. Laska, M. F. Duarte, J. k. Romberg, and R. G. Baraniuk, "Beyond Nyquist: Efficient sampling of sparse bandlimited signals," *IEEE Transactions on Information Theory*, vol. 56, no. 1, pp. 520–544, 2010.

[6] J. A. Tropp, M. B. Wakin, M. F. Duarte, D. Baron, D., and R. G. Baraniuk, "Random filters for compressive sampling and reconstruction," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 3, pp. 872–875, 2006.

[7] J. Romberg, "Compressive sensing by random convolution," *SIAM Journal on Imaging Sciences*, vol. 2, no. 4, pp. 1098–1128, 2009.

[8] T. L. N. Nguyen and Y. Shin, "Deterministic sensing matrices in compressive sensing: A survey," *The Scientific World Journal*, vol. 2013, pp. 1–6, Nov. 2013.

[9] S. D. Howard, A. R. Calderbank, and S. J. Searle, "A fast reconstruction algorithm for deterministic compressive sensing using second order reed-muller codes," in *the 42nd Annual Conference on Information Sciences and Systems (CISS 2008)*, pp. 11–15, 2008.

[10] A. Amini and F. Marvasti, "Deterministic construction of binary, bipolar, and ternary compressed sensing matrices," *IEEE Transactions on Information Theory*, vol. 57, pp. 2360–2370, 2011.

[11] L. Applebauma, S. S. S. D. Howardb, and R. Calderbank, "Chirp sensing codes: Deterministic compressed sensing measurements for fast recovery," *Applied and Computational Harmonic Analysis*, vol. 26, pp. 283–290, 2009.

[12] S. Qaisar, R. M. Bilal, W. Iqbal, M. Naureen, and S. Lee, "Compressive sensing: From theory to applications, a survey," *Journal of Communications and Networks*, vol. 15, pp. 443–456, 2013.

[13] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[14] S. S. Chen, D. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.

[15] W. Lu and N. Vaswani, "Modified basis pursuit denoising (modified-BPDN) for noisy compressive sensing with partially known support," in *IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings (ICASSP)*, pp. 3926–3929, Apr. 2010.

[16] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *Journal of the Royal Statistical Society*, vol. 58, no. 1, pp. 267–288, 1996.

[17] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Annals of statistics*, vol. 32, no. 2, pp. 407–451, 2004.

[18] R. R. Chartrand, "Exact reconstruction of sparse signals via nonconvex minimization," *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 707–710, 2007.

[19] J. Murray and K. Kreutz-Delgado, "An improved FOCUSS-based learning algorithm for solving sparse linear inverse problems," in *35th Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 347–351, Feb. 2001.

[20] D. P. Wipf and B. D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2153–2164, 2004.

[21] S. Mallat and Z. Zhifeng, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[22] J. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[23] D. Needell and J. Tropp, "Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2009.

[24] G. Cormode and S. Muthukrishnan, "Combinatorial algorithms for compressed sensing," in *40th Annual Conference on Information Sciences and Systems*, pp. 198–201, 2006.

[25] A. Gilbert, S. Muthukrishnan, and M. Strauss, "Improved time bounds for near-optimal sparse fourier representations," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 5914, pp. 1–15, Jan. 2004.

[26] A. C. Gilbert, M. Strauss, J. Tropp, and R. Vershynin, "One sketch for all: Fast algorithms for compressed sensing," in *Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing*, pp. 237–246, 2007.

[27] D. Donoho, "De-noising by soft-thresholding," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 613–627, 1995.

[28] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[29] R. Berinde, P. Indyk, and M. Ruzic, "Practical near-optimal sparse recovery in the L1 norm," in *46th Annual Allerton Conference on Communication, Control, and Computing*, pp. 198–205, Oct. 2008.

[30] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, pp. 33–61, 1998.

[31] L. Gurobi Optimization, "Gurobi optimizer reference manual." http://www.gurobi.com, 2020.

[32] M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[33] M. Lustig, D. Donoho, J. Santos, and J. Pauly, "Compressed sensing MRI," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 72–82, 2008.

[34] Q. Cheng, A. Ihalage, Y. Liu, and Y. Hao, "Compressive sensing radar imaging with convolutional neural networks," *IEEE Access*, vol. 8, pp. 212917–212926, 2020.

[35] S. Kirolos, J. Laska, M. Wakin, M. Duarte, D. Baron, T. Ragheb, Y. Massoud, and R. Baraniuk, "Analog-to-information conversion via random demodulation," in *Proc. IEEE Dallas Circuits and Systems Workshop (DCAS)*, pp. 71–74, 2006.

[36] M. Rani, S. Dhok, and R. B. Deshmukh, "A systematic review of compressive sensing: Concepts, implementations and applications," *IEEE Access*, vol. 6, pp. 4875–4894, 2018.

[37] C. Paar and J. Pelzl, *Understanding Cryptography : A Textbook for Students and Practitioners*. Berlin, London: Springer, 2009.

[38] T. W. Cusick and P. Stanica, "Chapter 2 - Fourier analysis of boolean functions," in *Cryptographic Boolean Functions and Applications (Second Edition)*, pp. 7–29, Academic Press, second edition ed., 2017.

[39] D. Coppersmith, H. Krawczyk, and Y. Mansour, "The shrinking generator," in *Advances in Cryptology — CRYPTO' 93*, pp. 22–39, 1994.

[40] W. Meier and O. Staffelbach, "The self-shrinking generator," in *Advances in Cryptology — EUROCRYPT'94*, pp. 205–214, 1995.

[41] D. B. Thomas, W. Luk, P. Leong, and J. Villasenor, "Gaussian random number generators," *Association for Computing Machinery*, vol. 39, pp. 1–38, Nov. 2007.

[42] J. Malik, J. Malik, A. Hemani, and N. Gohar, "An efficient hardware implementation of high quality awgn generator using box-muller method," in *2011 11th International Symposium on Communications Information Technologies (ISCIT)*, pp. 449–454, 2011.

[43] K. M. Kang, "FPGA implementation of gaussian-distributed pseudo-random number generator," in *6th International Conference on Digital Content, Multimedia Technology and its Applications*, pp. 11–13, 2010.

[44] G. Cotrina, A. Peinado, and A. Ortiz, "Gaussian pseudorandom number generator based on cyclic rotations of linear feedback shift registers," *Sensors*, vol. 20, no. 7, 2020.

[45] Y. Zhang, L. Zhang, J. Zhou, L. Liu, F. Chen, and X. He, "A review of compressive sensing in information security field," *IEEE Access*, vol. 4, pp. 2507–2519, 2016.

[46] Y. Rachlin and D. Baron, "The secrecy of compressed sensing measurements," in *2008 46th Annual Allerton Conference on Communication, Control, and Computing, Urbana-Champaign, IL*, pp. 813–817, Sep. 2008.

[47] C. Shannon, "Communication theory of secrecy systems," *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.

[48] T. Bianchi, V. Bioglio, and E. Magli, "Analysis of one-time random projections for privacy preserving compressed sensing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 313–327, 2016.

[49] V. Cambareri, M. Mangia, F. Pareschi, R. Rovatti, and G. Setti, "On known-plaintext attacks to a compressed sensing-based encryption: A quantitative analysis," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 10, pp. 2182–2195, 2015.

[50] E. Candès, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, Aug. 2006.

[51] J. Tropp, M. Wakin, M. Duarte, D. Baron, and R. Baraniuk, "Random filters for compressive sampling and reconstruction," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 3, pp. 872–875, 2006.

[52] J. Romberg, "Sensing by random convolution," in *2nd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, pp. 137–140, 2007.

[53] Z. He, T. Ogawa, and M. Haseyama, "The simplest measurement matrix for compressed sensing of natural images," in *2010 IEEE International Conference on Image Processing*, pp. 4301–4304, 2010.

[54] A. Ravelomanantsoa, H. Rabah, and A. Rouane, "Compressed sensing: A simple deterministic measurement matrix and a fast recovery algorithm," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 12, pp. 3405–3413, 2015.

[55] M. F. Duarte and R. G. Baraniuk, "Kronecker compressive sensing," *IEEE Transactions on Image Processing*, vol. 21, pp. 494–504, 2012.

[56] Z. Baoju, T. Xiang, W. Wei, and X. Jiazu, "The research of Kronecker product-based measurement matrix of compressive sensing," *EURASIP Journal on Wireless Communications and Networking*, Dec. 2013.

[57] D. Mitra, H. Zanddizari, and S. Rajan, "Investigation of kronecker-based recovery of compressed ECG signal," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, pp. 3642–3653, 2020.

[58] K. Wu and X. Guo, "Compressive sensing with sparse measurement matrices," in *IEEE 73rd Vehicular Technology Conference (VTC Spring)*, pp. 1–5, 2011.

[59] J. Sun, S. Wang, and Y. Dong, "Sparse block circulant matrices for compressed sensing," *IET Communications*, vol. 7, pp. 1412–1418, 2013.

[60] T. Bianchi, V. Bioglio, and E. Magli, "Analysis of one-time random projections for privacy preserving compressed sensing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 313–327, 2016.

[61] Y. Zhang, Y. Xiang, L. Y. Zhang, Y. Rong, and S. Guo, "Secure wireless communications based on compressive sensing: A survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1093–1111, 2019.

[62] T. Bianchi and E. Magli, "Analysis of the security of compressed sensing with circulant matrices," in *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 173–178, 2014.

[63] W. Xue, C. Luo, G. Lan, R. Rana, W. Hu, and A. Seneviratne, "Kryptein: A compressive-sensing-based encryption scheme for the internet of things," in *16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pp. 169–180, 2017.

[64] H. Djelouat, A. Amira, F. Bensaali, and I. Boukhennoufa, "Secure compressive sensing for ECG monitoring," *Computers & Security*, vol. 88, 2020.

[65] W. Cho and N. Y. Yu, "Secure and efficient compressed sensing-based encryption with sparse matrices," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1999–2011, 2020.

[66] W. Lu, K. Kpalma, and J. Ronsin, "Sparse binary matrices of LDPC codes for compressed sensing," in *Data Compression Conference*, pp. 405–405, 2012.

[67] S. Khalid and S. Khan, "Application of compressed sensing on images via BCH measurement matrices," in *International Conference on Robotics and Emerging Allied Technologies in Engineering (iCREATE)*, pp. 78–81, 2014.

[68] S. Lin and D. J. Costello, *Error Control Coding: Fundementals and Appclications*, vol. 2. Prentice Hall series in computed applications in electrical engineering, 2001.

[69] Wikipedia, "Linear codes." https://en.wikipedia.org/wiki/Linear_code.

[70] A. Amini and F. Marvasti, "Deterministic construction of compressed sensing matrices using BCH codes," *arXiv*, Aug. 2009.

[71] P. Plawiak, "ECG signals (1000 fragments). Mendeley data." http://dx.doi.org/10.17632/7dybx7wyfn.3, 2017.