# Hardness of Successive-Cancellation Decoding of Linear Codes

**Arman Fazeli, Alexander Vardy,** and **Hanwen Yao**
University of California San Diego, La Jolla, CA 92093, USA
**{afazelic,avardy,hwyao}@ucsd.edu**

***Abstract*—Successive-cancellation decoding has gained much renewed interest since the advent of polar coding a decade ago. For polar codes, successive-cancellation decoding can be accomplished in time $O(n \log n)$. However, the complexity of successive-cancellation decoding for other families of codes remains largely unexplored. Herein, we prove that successive-cancellation decoding of general binary linear codes is NP-hard. In order to establish this result, we reduce from maximum-likelihood decoding of linear codes, a well-known NP-complete problem. Unlike maximum-likelihood decoding, however, the *successive-cancellation decoding problem* depends on the choice of a generator matrix. Thus we further strengthen our result by showing that there exist codes for which successive-cancellation decoding remains hard for every possible choice of the generator matrix. On the other hand, we also observe that polynomial-time successive-cancellation decoding can be extended from polar codes to many other linear codes. Finally, we show that *every* binary linear code can be *encoded* as a polar code with dynamically frozen bits. This approach makes it possible to use list-decoding of polar codes to approximate the maximum-likelihood decoding performance of arbitrary codes.**

***Index Terms*—coding theory, polar codes, successive-cancellation decoding, computational hardness, maximum-likelihood decoding**

## I. Introduction

The class of *polynomial-time* problems, denoted by P, is defined to be the set of all decision problems that can be solved, *in the worst case*, in a number of steps that is polynomially bounded by the length of their inputs. NP is the class of decision problems which can be solved by a non-deterministic Turing machine in a polynomially-bounded number of steps. A problem is said to be NP-hard if every problem in NP can be reduced to it by a deterministic Turing machine in a polynomial time. In other words, NP-hard problems are at least hard as the hardest problem in NP.

There are many interesting computational problems in the field of coding theory with the hardness of many of which remaining open until today. Two fundamental problems, namely *maximum-likelihood decoding* (MLD) and computation of the weight distribution were shown to be NP-hard for the class of binary linear codes by Berlekamp, McEliece, and van Tilborg in 1978 [4] and was later extended to non-binary linear codes in [3]. The classical hardness results in the coding theory are centered mostly around ML decoding and computation of the minimum distances. The *successive-cancellation decoding* (SCD) has resurfaced behind the polar codes after their discovery by Erdal Arıkan just over a decade ago [1]. But, no hardness results were established for it until today. In this paper, we prove that the SCD problem for linear codes is also NP-hard. The default formulation of the SCD problem depends on the given generator matrix. We show how to modify the structure of polar codes in order to construct a wider family of generator matrices for which SCD can be implemented with a polynomial-time complexity.

The construction of polar codes were later generalized to *large-kernel* polar codes, which are shown to have a superior performance at finite block-lengths under SC decoding [7], [12]. In fact, polar codes with sufficiently large kernels were recently proven to have near-optimal scaling properties [11]. The SC decoding of large-kernel polar codes is based on an efficient scheduling of SC decoding over individual kernels within the code construction. While there are many binary kernels for which low-complexity SC decoding algorithms exist [6], [17], [19], no polynomial-time algorithm was known to perform SC decoding of arbitrary kernels. The NP-hardness of the SCD problem proves that no such algorithm exists in general. The challenge of finding *good* polarization kernels that can be SC decoded efficiently remains *unsolved*.

For the common communication channels, such as *binary additive white-Gaussian-noise* channel (B-AWGN) and *binary symmetric channel* (BSC), the problem of maximum-likelihood decoding is equivalent to the minimum-distance decoding. The decision problem that corresponds to the MLD problem is usually formulated in terms of the distances and stated as follows.

**Problem:** MLD-Linear
**Input:** An $(n-k) \times n$ matrix $H$ over $\mathbb{F}_q$, a target vector $\mathbf{y} \in \mathbb{F}_q^n$, and an integer $d > 0$.
**Question:** Is there a vector $\mathbf{x} \in \mathbb{F}_q^n$ of weight $\leq d$, such that $H\mathbf{x}^t = \mathbf{y}^t$?

Note that the input to the MLD-Linear can be equivalently described with the generator matrix $G$ instead of $H$. Furthermore, the choice of $G$ or $H$ does not change the outcome. But, this is not the case for the SCD problem. In fact, not only the solution but also the hardness of the problem depend on the choice of the generator matrix. The solution to the SCD problem also depends on the channel parameters such as flip probability in BSC($p$), which is why it cannot be described in terms of just the minimum distances. Let $\mathbf{u}$, $\mathbf{x}$, and $\mathbf{y}$ denote

the uncoded information vector, the transmitted codeword, and the received vector respectively, *i.e.* $\mathbf{u} \to \mathbf{x} \to \mathbf{y}$. We leave further details to the next section. For now, let us state the binary SCD problem for a given generator matrix as follows.

**Problem:** SCD-Linear
**Input:** A $k \times n$ matrix $G$ over $\mathbb{F}_2$, an index $i \in [k]$, a target vector $\mathbf{y} \in \mathbb{F}_2^n$, a frozen vector $u_1^{i-1} \in \mathbb{F}_2^{i-1}$, and a flip probability $0 < p < 1/2$.
**Question:** $\mathbb{P}(\mathbf{y}, u_1^{i-1}|u_i = 0) \lessgtr \mathbb{P}(\mathbf{y}, u_1^{i-1}|u_i = 1)$ ?

The generator matrix in the SCD-Linear problem is fixed in the input. However, there are many different generator matrices that can generate the same linear code. It is possible for the SCD-Linear problem to be hard for some choices of $G$ and not so difficult for a different choice of $G$ that generates the same linear code. A natural question follows: How hard is the SCD-Linear problem if we relax the constraint on $G$ and let the solution choose from the *ensemble* of all generator matrices for the given linear code? Such an ensemble can be described by all generator matrices such as $G$ that satisfy $HG^t = \mathbf{0}$ for a given parity-check matrix of the code. Let us formally state this problem as follows.

**Problem:** Ensemble SCD-Linear
**Input:** A $(n-k) \times n$ matrix $H$ over $\mathbb{F}_2$, an index $i \in [k]$, a target vector $\mathbf{y} \in \mathbb{F}_2^n$, a frozen vector $u_1^{i-1} \in \mathbb{F}_2^{i-1}$, and a flip probability $0 < p < 1/2$.
**Output:** A $k \times n$ generator matrix $G$ over $\mathbb{F}_2$ such that $HG^t = \mathbf{0}$ and the corresponding bit: $\mathbb{P}(\mathbf{y}, u_1^{i-1}|u_i = 0) \lessgtr \mathbb{P}(\mathbf{y}, u_1^{i-1}|u_i = 1)$.

The Ensemble SCD-Linear problem is a *computational* problem in contrast to the SCD-linear *decision* problem since the output contains not only the a bit that corresponds to the probability comparison but also a $k \times n$ generator matrix $G$. However, in term of the hardness, the Ensemble SCD-Linear problem is *computationally easier* than the SCD-Linear problem as the machinery is allowed to generator matrix that makes the computation simpler. In other words, the NP-hardness of Ensemble SCD-Linear results in SCD-Linear being NP-hard, but the reverse is not true. We will show that the Ensemble SCD-Linear problem is also NP-hard, which is a *stronger* result than what was discussed earlier.

### A. Our contributions

In this paper, we prove that both SCD-Linear and Ensemble SCD-Linear problems are NP-hard. We describe an *easier* version of the SCD-Linear problem that can be stated in terms of the minimum Hamming distances and no longer depends on the flip probability ($p$) of the underlying communication channel. We show this problem is also NP-hard. Our proofs are based on polynomial reductions of the MLD problem to these problems. We also show that, as a corollary of SCD-Linear being NP-hard, the *bit-wise maximum a posteriori probability* (MAP) decoding problem is also NP-hard.

We also utilize some variations of conventional polar codes, namely large-kernel polar codes and polar codes with dynamically-frozen values, to design multiple families of non-singular $n \times n$ generator matrices for which SCD-Linear algorithms with polynomial-time complexity exist.

### B. Related work

Since the seminal work of Berlekamp et al in 1978, the complexity of MLD of general linear codes has been extensively studied. It was shown in [5], [14] that the problem remains hard even if the code is known and unlimited preprocessing is allowed. In [2], the MLD-Linear was shown to be NP-hard to even approximate within a constant factor. The problem was proven not to be fixed-parameter tractable in [9]. Berlekamp et al also conjectured that the problem of finding the minimum distance (MD-Linear) is also NP-hard. It was later proved to be true in [20] and the results were strengthened in a series of papers including [8], [10]. The maximum-likelihood decoding of a few specific families of error-correction codes such as product codes [3] and Reed-Solomon codes [13] are also shown to be NP-hard. To the best of our knowledge, this question has not been answered for any other common family of error-correction codes. However, in [15] it was observed *through simulations* that the list decoding of polar codes with moderate list sizes can effectively perform ML decoding. It remains open to show if the MLD decoding of polar codes is NP-hard or not.

### C. Paper outline

The channel model is given in Section II. In the same section, we prove the NP-hardness of the SCD-Linear and Ensemble SCD-Linear problems. In Section III, we give a brief overview of the polar codes, which we modify in order to introduce multiple families of generator matrices for which SCD-Linear can be solved in polynomial-time.

## II. Successive-Cancellation Decoding is Hard

### A. Channel model

Let us recall the notation from the previous section, where $\mathbf{u} \in \mathbb{F}_2^k$ denotes the uncoded information vector. The encoded vector is given by $\mathbf{x} = \mathbf{u}G$, where $G$ is a $k \times n$ generator matrix. The received vector $\mathbf{y}$ is generated by passing $\mathbf{x}$ through $n$ independent copies of a BSC($p$). The goal in the successive-cancellation decoding is to decode $u_1, u_2, \ldots, u_k$ one-by-one. Therefore, at step $i$, the values of $u_1, \ldots, u_{i-1}$ are known to the decoder. To be consistent with the polar coding literature, we refer to these values as the frozen values. We can now model the $i$-th bit-channel during SC decoding as depicted in Figure 1. The channel input is a binary variable $u_i \in \mathbb{F}_2$ and the values for $u_j, j \in [k] \setminus \{i\}$ are drawn independently and uniformly at random. The output of the channel is given by $(u_1^{i-1}, y_1^n) \in \mathbb{F}_2^{n+i-1}$. For a given frozen vector $u_1^{i-1}$, define the linear cosets $\mathcal{C}_0^{(i)}$ and $\mathcal{C}_1^{(i)}$ as

$$\mathcal{C}_t^{(i)} \triangleq \{\mathbf{x} | \mathbf{x} = u_1^k G, u_i = t, u_{i+1}^k \in \mathbb{F}_2^{k-i}\}, \forall t \in \{0, 1\}. \quad (1)$$

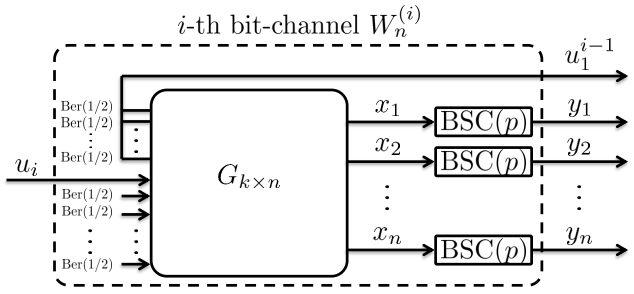


Fig. 1. The channel model for the $i$-th bit-channel during SD decoding of a code generated by a $k \times n$ generator matrix $G$, transmitted over $n$ *i.i.d.* copies of a BSC($p$). The values of $u_1, \ldots, u_{i-1}$ are known to the decoder.

Then, the decision problem in SCD-Linear becomes equivalent to

$$\sum_{\mathbf{x}\in\mathcal{C}_0^{(i)}} \mathbb{P}(\mathbf{x})\mathbb{P}(\mathbf{y}|\mathbf{x}) \lesseqgtr \sum_{\mathbf{x}\in\mathcal{C}_1^{(i)}} \mathbb{P}(\mathbf{x})\mathbb{P}(\mathbf{y}|\mathbf{x}). \tag{2}$$

However, all of the codewords in $\mathcal{C}_1^{(i)}$ (for $t = 0, 1$) are transmitted with equal probabilities. So,

$$\mathbb{P}(\mathbf{x})\mathbb{P}(\mathbf{y}|\mathbf{x}) = \frac{1}{2^{k-1}} p^{d_H(\mathbf{y},\mathbf{x})}(1-p)^{n-d_H(\mathbf{y},\mathbf{x})}, \tag{3}$$

where $d_H(\mathbf{y}, \mathbf{x})$ denotes the Hamming distance between $\mathbf{y}$ and $\mathbf{x}$. This allows us to further simplify the decision problem in SCD-Linear as

$$\sum_{\mathbf{x}\in\mathcal{C}_0^{(i)}} q^{d_H(\mathbf{y},\mathbf{x})} \lesseqgtr \sum_{\mathbf{x}\in\mathcal{C}_1^{(i)}} q^{d_H(\mathbf{y},\mathbf{x})}, \tag{4}$$

where $q \triangleq p/(1-p) < 2p < 1$. It is clear that if $p \to 0$, then $q \to 0$ as well. This becomes important in the next section, where we prove the hardness of the SCD-Linear problem.

### *B. Hardness proofs*

Let us first clarify the range for which $p$ takes values from since it is an input to both of the SCD-Linear and Ensemble SCD-Linear problems. Note that the size of the other elements in the input is given by $poly(n)$. In order to keep the asymptotic order of the input size, it suffices to limit $p$ to the set of fractional numbers given by

$$\{\frac{\nu}{2^m} | \nu \in \mathbb{N}, \nu < 2^m, \text{ for some } m = poly(n)\}. \tag{5}$$

Next, we point out that for small values of $p$ such as $p = 2^{-poly(n)}$, both sides of (4) become dominated by the terms with the smallest powers since

$$|\mathcal{C}_t^{(i)}| \leq 2^{n-1} \Rightarrow q^{d_{\min}} > |\mathcal{C}_t^{(i)}| q^{d_{\min}+1}, \text{ for all } q \leq 2^{-n}. \tag{6}$$

This fact allows us to focus on a special case of the SCD-Linear problem, which is equivalent to a decision problem that no longer depends on the flip probabilities and can be stated as the following.

**Problem:** SC-Distance
**Input:** A $k \times n$ matrix $G$ over $\mathbb{F}_2$, an index $i \in [k]$, a target vector $\mathbf{y} \in \mathbb{F}_2^n$, and a frozen vector $u_1^{i-1} \in \mathbb{F}_2^{i-1}$.
**Question:** $d_H(y, \mathcal{C}_0^{(i)}) \lesseqgtr d_H(y, \mathcal{C}_1^{(i)})$ ?

The notation $d_H(\mathbf{y}, \mathcal{C})$ denotes the minimum Hamming distance between $\mathbf{y}$ and the elements in $\mathcal{C}$. While leaving the details to the full version of this paper, we point out that the SC-Distance problem is also an special case of the Nearest-Coset problem, where the two cosets are a linear shift of each other. It is possible to show that the SC-Distance problem is equally as hard as the Nearest-Coset problem and at least as hard as the Nearest-Codeword problem. Here, we take a simpler route. According to (1), we have $\mathcal{C}_1^{(i)} = g_i + \mathcal{C}_0^{(i)}$, where $g_i$ is the $i$-th row in $G$. Given that $G$ is an arbitrary binary matrix, we deduce that $\mathcal{C}_0^{(i)}$ and $\mathcal{C}_1^{(i)}$ can respectively be an arbitrary linear coset and an arbitrary linear shift of it. In the following we prove that the MLD-Linear problem can be polynomially reduced to the SC-Distance problem which establishes the latter to be NP-hard. Given that the SC-Distance problem is a special case of the SCD-Linear problem, our proof also establishes the NP-hardness of the SCD-Linear problem.

*Proof of SC-Distance $\in$ NP-hard.* Let us start with a binary linear code $\mathcal{C}$ that is described by a $k \times n$ generator matrix in $\mathbb{F}_2^n$ denoted by $G$. Let $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_k$ denote the rows in $G$ and define $\mathcal{C}^{(i)}$ as

$$\mathcal{C}^{(i)} \triangleq \text{span}\langle \mathbf{g}_i, \mathbf{g}_{i+1}, \ldots, \mathbf{g}_k \rangle, \ \forall i \in [k]. \tag{7}$$

Further, let $\mathbf{y}$ be the received vector and denote the set of its nearest codewords in $\mathcal{C}$ by

$$\mathcal{S}_\mathbf{y} \triangleq \{\mathbf{x} | \mathbf{x} \in \mathcal{C}, d_H(\mathbf{y}, \mathbf{x}) = d_H(\mathbf{y}, \mathcal{C})\}. \tag{8}$$

In the following, we show how one can find an element in $\mathcal{S}_\mathbf{y}$ by employing the algorithmic solution to the SC-Distance problem (oracle) in $k$ steps. In step 1, we ask the SC-Distance oracle to find out about

$$d_H(y, \mathcal{C}^{(2)}) \lesseqgtr d_H(y, \mathbf{g}_1 + \mathcal{C}^{(2)}), \tag{9}$$

where a tie is broken with a coin flip. For simplicity, assume

$$d_H(y, \mathcal{C}^{(2)}) \geq d_H(y, g_1 + \mathcal{C}^{(2)}). \tag{10}$$

This means that at least one of the nearest codewords to $\mathbf{y}$ belongs to the linear coset $\mathbf{g}_1 + \mathcal{C}^{(2)}$. In other words,

$$\mathcal{S}_\mathbf{y} \cap (\mathbf{g}_1 + \mathcal{C}^{(2)}) \neq \emptyset. \tag{11}$$

Note that we have so far reduced the size of our search for elements in $\mathcal{S}_\mathbf{y}$ in half. We can repeat the same method in each step: In step $i$, we start from a linear shift of $\mathcal{C}^{(i)}$ denoted by $\mathbf{a} + \mathcal{C}^{(i)}$ for which we know that $\mathcal{S}_\mathbf{y} \cap (\mathbf{a} + \mathcal{C}^{(i)}) \neq \emptyset$. Then, we ask the SC-Distance oracle about

$$d_H(y, \mathbf{a} + \mathcal{C}^{(i+1)}) \lesseqgtr d_H(y, \mathbf{a} + \mathbf{g}_i + \mathcal{C}^{(i+1)}), \tag{12}$$

which allows us to cut the search size for an element in $\mathcal{C}_\mathbf{y}$ once again. We continue by updating the *shift* vector $\mathbf{a}$ as

$$\mathbf{a}_{\text{new}} \longleftarrow \mathbf{a}_{\text{old}} \quad \text{or} \quad \mathbf{a}_{\text{new}} \longleftarrow \mathbf{a}_{\text{old}} + \mathbf{g}_i, \tag{13}$$

according to the inequality direction in (12). Upon finishing the $k$-th step, the search size is reduced to only one codeword,

which gives a solution to the MLD-Linear problem for the linear code $\mathcal{C}$. Given that the number of steps is upper bounded by $n$, we proved that the MLD-Linear problem, which is known to be NP-hard [4], can be polynomially reduced to the SC-Distance problem. Therefore, the SC-Distance problem is also NP-hard. □

The importance of each step is to reduce the search size in half. The choice of the generator matrix has no significant role. In fact, we can achieve the same if the SC-Distance problem is relaxed to make the comparison for a generator matrix of its choice. This is similar to the relaxation that transformed the SCD-Linear problem to the Ensemble SCD-Linear problem. Let us state this computational problem as the following.

**Problem:** Ensemble SC-Distance
**Input:** A $(n-k)\times n$ matrix $H$ over $\mathbb{F}_2$, an index $i \in [k]$, a target vector $\mathbf{y} \in \mathbb{F}_2^n$, and a frozen vector $u_1^{i-1} \in \mathbb{F}_2^{i-1}$.
**Output:** A $k \times n$ generator matrix $G$ over $\mathbb{F}_2$ such that $HG^t = \mathbf{0}$ and the corresponding bit: $d_H(y, \mathcal{C}_0^{(i)}) \leqslant d_H(y, \mathcal{C}_1^{(i)})$.

It is easy to observe that the Ensemble SC-Distance problem is equivalent to a special case of the Ensemble SCD-Linear problem that corresponds to the flip probability $p$ being sufficiently small. So, in order to prove that Ensemble SCD-Linear is NP-hard it suffices to show that Ensemble SC-Distance is NP-hard. Fortunately, the Ensemble SC-Distance problem is NP-hard for the same reason that SC-Distance was: given a received vector $\mathbf{y}$ and a code $\mathcal{C}$ of dimension $k$, one can find a nearest codeword of $\mathbf{y}$ in $\mathcal{C}$ by employing the Ensemble SC-Distance oracle in $k$ many steps, where each step reduces the search size in half.

**Corollary 1.** *The Ensemble SCD-Linear problem is also NP-hard.*

The bit-wise MAP decoding problem is similar to the SC decoding problem in the sense that the outputs in both depend on the choice of the generator matrix for the given linear code. Let us state the MAP decoding problem as

**Problem:** MAP-Linear
**Input:** A $k\times n$ matrix $G$ over $\mathbb{F}_2$ and a target vector $\mathbf{y} \in \mathbb{F}_2^n$.
**Output:** $\mathbb{P}(\mathbf{y}|u_i = 0) \leqslant \mathbb{P}(\mathbf{y}|u_i = 1)$ for all $i \in [k]$

Note that for any $i \in [k]$, we can use column permutations to transform the MAP-Linear problem to a special case of the SCD-Linear problem where $i = 1$. However, the SCD-Linear problem for $i = 1$ is as hard as the general case where $i \in [k]$, which is shown to be NP-hard. Hence, we have the following.

**Corollary 2.** *The MAP-Linear problem is also NP-hard.*

## III. Connections to Polar Codes

So far, we showed that one cannot devise an SC decoding algorithm that runs in polynomial-time for arbitrary generator matrices. Note that it is possible to add $n-k$ additional rows on the top of any $k\times n$ generator matrix and make it an square matrix and the SC decoding of the original generator matrix will be contained within the SC decoding of the extended $n\times n$ matrix. However, this statement does not remain true if any of added $n-k$ rows are placed after or in between the rows of the original generator $k \times n$ generator matrix, which is ironically the case for polar codes. With that in mind, we limit the scope of our investigation from this point forward only to the $n \times n$ non-singular binary matrices and introduce a few families of $n\times n$ generator matrices, for which some polynomial-time SC decoding algorithms exist.

### A. A primer on polar Codes

The polarization theory is based on a simple linear transformation of a binary-input channel that includes many Kronecker products of a binary matrix $K$, called the *polarization kernel*, with itself. The generator matrix of conventional polar codes, as introduced by Arıkan in [1], is given by

$$G = K_2^{\otimes(\log n)}, \text{ where } K_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}. \tag{14}$$

As a result, most of the polar bit-channels, denoted by $W_n^{(i)}$ for $i \in [n]$ become either really *good* or really *bad*, where the goodness can be described in terms of the capacity, bit-error probability, or the Bhattacharyya parameter. Moreover, the ratio of the good channels tend to the symmetric capacity of the underlying communication channel as $n$ becomes large. The encoder equation of polar codes is given by $\mathbf{x} = \mathbf{u}G$, where $\mathbf{u}$ is an uncoded length-$n$ binary vector. The $k$ raw information bits are placed on those coordinates of $\mathbf{u}$ that correspond to the *best* $k$ bit-channels. The values of the remaining $u_i$'s are frozen and known to the decoder. Let $\mathcal{F}$ denote the set of indices that corresponds to the frozen values. At the decoder, the values of $u_i$ for $i \in [n] \setminus \mathcal{F}$ are revealed one-by-one according to the following rule:

$$u_i = \arg\max_t \mathbb{P}(\mathbf{y}, u_1^{i-1}|u_i = t). \tag{15}$$

The butterfly-like structure of polar encoder allows one to compute the probabilities in (15) with $O(n \log n)$ computational complexity. We refer the interested readers to [1] for more details.

### B. Large-kernel polar codes

Soon after discovery of polar codes, Korada, Şaşoğlu, and Urbanke [7] showed that the $2 \times 2$ kernel in the conventional polar codes can be replaced with any binary non-singular $\ell \times \ell$ matrix and the capacity achieving theorems will hold as long as the $\ell \times \ell$ kernel cannot be transformed into an upper-triangular matrix via column permutations.

Since then, many articles has been published in pursuit of finding alternative polarization kernels with better error performance. However, near all of these large-kernel polar codes are far from practical implementation due to an increased decoding complexity that attributes to the complexity of SC decoding within each kernel. In fact, the computational complexity of the only known SC decoding algorithm for

large-kernel polar codes with arbitrary $\ell \times \ell$ kernels is given by $O(2^\ell n \log n)$. On the other hand, the NP-hardness of SCD-Linear, as demonstrated earlier, shows that there is no *efficient* SC decoding algorithm for when $\ell$ grows faster than $\Theta(\log n)$. The polar coding diagram for large kernels consists of $\log_\ell n$ polarization layers, where each layer contains $n/\ell$ kernels. The capacity-achieving results hold true mainly for the case where all these $(n/\ell)\log_\ell n$ kernels are the same. But, they do not have to be the same for the SC decoding to be efficient. Hence, the total number of $n \times n$ generator matrices with polynomial-time SC decoding complexity that are constructed through this method is asymptotically lower-bounded by $\exp\left(c\frac{n(\log n)^2}{\log\log n}\right)$, for any fixed $c > 0$.

### C. Polar codes with dynamically frozen values

The frozen values among $u_i, i = 1, 2, \ldots, n$ are known to the decoder. For symmetric channels, the performance of polar codes under SC decoding does not depend on these values and hence are usually set to be all-zero. However, it was shown in [16], that this is not the case for other decoding methods such as the list decoder in [15], which essentially matches the ML decoding performance for sufficiently large list sizes.

Let $i \in \mathcal{F}$ correspond to a frozen index. [16] suggested to set the value of $u_i$ as a linear combination of bits in $\{u_j\}_{j=0}^{i-1}$ instead of a pre-determined fixed value, *i.e.*

$$\forall i \in \mathcal{F} : u_i = \sum_{j<i} \alpha_{i,j} u_j, \tag{16}$$

and hence comes the name *dynamically frozen values*. This definition was later extended in [18] to include uncoded information bits as well. Note that by precoding the length-$n$ vector $\mathbf{u}$ with a non-singular upper-triangular matrix neither the performance of the code under SC decoding nor its computational complexity would change. To understand why, let $G$ be an $n \times n$ non-singular matrix. Further, let $A_n$ be the upper-triangular precoding matrix, where

$$A_n^{-1} = \begin{bmatrix} 1 & b_{1,2} & \cdots & b_{1,n-1} & b_{1,n} \\ 0 & 1 & \cdots & b_{2,n-1} & b_{2,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & b_{n-1,n} \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}. \tag{17}$$

Then, the overall encoding relation would be given by $\mathbf{x} = \mathbf{u}A_nG$. Let $\mathbf{v} = \mathbf{u}A_n$ denote the precoded vector that serves as the input to the polar encoder and $\mathbf{y}$ be the received vector. The SC decoder for matrix $G$ is capable of computing

$$\mathbb{P}(\mathbf{y}, v_1^{i-1}|v_i = 0) \lesseqgtr \mathbb{P}(\mathbf{y}, v_1^{i-1}|v_i = 1) \tag{18}$$

for any index $i \in [n]$ and any length-$(i-1)$ vector $v_1^{i-1} \in \mathbb{F}_2^{i-1}$ corresponding to the values of $\mathbf{v}$ that are decoded so far. However, there exists a one-to-one map between $\{u_1, \ldots, u_i\}$ and $\{v_1, \ldots, v_i\}$ for any $i \in [n]$ that is given by

$$u_i = v_i + \sum_{j=1}^{i-1} b_{j,i} v_j, \tag{19}$$

which allows us to simply map the probabilities in (18) already computed by the SC decoder of $G$ to the $\mathbb{P}(\mathbf{y}, v_1^{i-1}|v_i = t)$ for all values of $i \in [n]$ and $\in \mathbb{F}_2$. This is equivalent to SC decoding of the overall generator matrix, $A_nG$. The total number of generator matrices with polynomial-time SC decoding complexity that can be generated according to this transformation is hence given by $2^{\frac{n(n-1)}{2}}$.

So far, we modified polar codes to construct generator matrices for which we can implement SC decoding with polynomial-time complexity. But, what can be said about arbitrary generator matrices? Can we use polar codes to reduce the complexity of SC decoding in general? Let $G$ be a $k \times n$ generator matrix of a $(n, k)$ linear code $\mathcal{C}$. Considering that the generator matrix of polar codes is non-singular, there exists a unique $k \times n$ matrix $M$ such that

$$G = MK_2^{\otimes \log n}. \tag{20}$$

The matrix $M$ is not necessarily upper-triangular. In fact, it is not even necessarily an square matrix. However, we can apply a set of elementary row operations on both sides of (20) to arrive at $G' = M'K_2^{\otimes \log n}$, where $M'$ is in the *reduced row echelon* form. Note that $G'$ is also a valid generator matrix for $\mathcal{C}$. Let $\mathcal{I}$ denote the set of coordinates of those columns in $M'$ that contain the leading 1s (and zeros everywhere else). As an example, we have $\mathcal{I} = \{1, 2, 4\}$ for

$$M'_{\text{example}} = \begin{bmatrix} 1 & 0 & \times & 0 & \times & \times & \times & \times \\ 0 & 1 & \times & 0 & \times & \times & \times & \times \\ 0 & 0 & \times & 1 & \times & \times & \times & \times \end{bmatrix}. \tag{21}$$

The interesting observation in here is that the encoding relation for $G'$, which is given by $\mathbf{x} = \mathbf{u}G'$ can be emulated by a polar encoder for $K_2^{\otimes \log n}$ with dynamically frozen values, where the uncoded information bits are placed on coordinates in $\mathcal{I}$. In other words, *every* linear code can be *encoded* as a polar code with dynamically frozen bits. On the contrary, the successive-cancellation decoding problem of an arbitrary code with a $k \times n$ generator matrix is not necessarily equivalent to the successive-cancellation decoding of its corresponding $n \times n$ generator matrix, $G'$, due to the reasons explained in the beginning of this section. But it is expected; otherwise, we would have had P=NP.

On the other hand, the list decoding of polar codes [15] is shown, through simulations, to be able to *approximate* the maximum-likelihood decoding of conventional polar codes with reasonable list sizes. Moreover, the list decoding algorithm of polar codes can be easily extended to the polar codes with dynamically frozen bits. Based on the fact that every linear code can be encoded as a polar code with dynamically frozen bits, we can utilize the list decoding algorithm to approximate the maximum-likelihood decoding performance of arbitrary codes. We can also use the same method to generate a *lower-bound* for the maximum-likelihood decoding performance of arbitrary codes, which would improve as the list size increases. We leave the investigation on computational complexity of these methods for future studies.

## REFERENCES


[1] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–73, June 2009.

[2] S. Arora, L. Babai, J. Stern, and Z. Sweedyk, "The hardness of approximate optima in lattices, codes, and systems of linear equations," *Journal of Computer and System Sciences*, vol. 54, no. 2, pp. 317–31, April 1997.

[3] A. Barg, "Some new NP-complete coding problems," *Problemy Peredachi Informatsii*, vol. 30, pp. 23–28, 1994, (in Russian).

[4] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, "On the inherent intractability of certain coding problems," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, May 1978.

[5] J. Bruck and M. Naor, "The hardness of decoding linear codes with preprocessing," *IEEE Transactions on Information Theory*, vol. 36, no. 2, pp. 381–385, March 1990.

[6] S. Buzaglo, A. Fazeli, P. H. Siegel, V. Taranalli, and A. Vardy, "Permuted successive cancellation decoding for polar codes," *In Proceedings of IEEE International Symposium on Information Theory (ISIT)*, pp. 2618–22, June 2017.

[7] S. B. Korada, E. Şaşoğlu, and Rüdiger Urbanke, "Polar codes: Characterization of exponent, bounds, and constructions," *IEEE Transactions on Information Theory*, vol. 56, no. 12, pp. 6253–64, November 2010.

[8] Q. Cheng and D. Wan, "A deterministic reduction for the gap minimum distance problem," *In Proceedings of the annual ACM Symposium on Theory of Computing (SODA)*, pp. 33–38, May 2009.

[9] R. G. Downey, M. Fellows, A. Vardy, and G. Whittle, "The parametrized complexity of some fundamental problems in coding theory," *SIAM Journal on Computing*, vol. 29, no. 2, pp. 545–570, April 2000.

[10] I. Dumer, D. Micciancio, and M. Sudan, "Hardness of approximating the minimum distance of a linear code," *IEEE Transactions on Information Theory*, vol. 49, no. 1, pp. 22–37, January 2003.

[11] A. Fazeli, H. Hassani, M. Mondelli, and A. Vardy, "Binary linear codes with optimal scaling: Polar codes with large kernels," *In Proceedings of IEEE Information Theory Workshop (ITW)*, pp. 1–5, November 2018.

[12] A. Fazeli and A, Vardy, "On the scaling exponent of binary polarization kernels," *In Proceedings of IEEE Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 797–804, September 2014.

[13] V. Guruswami and A. Vardy, "Maximum-likelihood decoding of Reed-Solomon codes is NP-hard," *In Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 470–478, January 2005.

[14] A. Lobstein, "The hardness of solving subset sum with preprocessing," *IEEE Transactions on Information Theory*, vol. 36, no. 4, pp. 943–6, July 1990.

[15] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–26, March 2015.

[16] P. Trifonov and V Miloslavskaya, "Polar codes with dynamic frozen symbols and their decoding by directed search," *In Proceedings of IEEE Information Theory Workshop (ITW)*, pp. 1–5, September 2013.

[17] P. Trifonov, V. Miloslavskaya, C. Chen, and Y. Wang, "Fast encoding of polar codes with Reed–Solomon kernel," *IEEE Transactions on Communications*, vol. 64, no. 7, pp. 2746–52, June 2016.

[18] P. Trifonov and G. Trofimiuk, "A randomized construction of polar subcodes," *In Proceedings of IEEE International Symposium on Information Theory (ISIT)*, pp. 1863–1867, June 2017.

[19] G. Trofimiuk and P. Trifonov, "Efficient decoding of polar codes with some $16 \times 16$ kernels," *In Proceedings of IEEE Information Theory Workshop (ITW)*, pp. 1–5, November 2018.

[20] A. Vardy, "The Intractability of Computing the Minimum Distance of a Code," *Transactions on Information Theory,* vol. 43, no. 6, pp. 1757–1766, November 1997.