Decoding Polar Codes for a Generalized Gilbert-Elliott Channel With Unknown Parameter

Yong Fang and Jun Chen, *Senior Member, IEEE*

***Abstract*— Decoding of polar codes, a class of capacity-achieving channel codes, typically requires the perfect knowledge of channel parameter in advance. This paper aims to investigate how to decode polar codes when channel parameter is unknown. Specifically, we study a generalized Gilbert-Elliott channel model, which assumes that the channel switches between a finite number of states. On the platform of Soft CANcellation (SCAN), which is a low-complexity iterative decoding algorithm of polar codes superior to the widely-used Successive Cancellation (SC) decoder, we propose three adaptive algorithms, *i.e.*, Sliding-Window SCAN (SWSCAN), Weighted-Window SCAN (W^{2}SCAN), and Linear-Weighting SCAN (LWSCAN). These adaptive SCAN decoders are seeded with a coarse estimate of channel state, and after each SCAN iteration, the decoders progressively refine the estimate of channel state. Experimental results demonstrate that the proposed adaptive SCAN decoders outperform the original SCAN decoder and other competitors.**

***Index Terms*— Polar codes, channel with memory, soft cancellation, channel estimation.**

I. Introduction

In 2009, Arıkan proposed a novel channel coding technique based on a simple finding [1]: After an *eXclusive OR* (XOR) operation on the inputs, two *independent* and *identical* **physical** channels can form two *different* **virtual** channels. Let W_1 and W_2 be two physical channels and from them, two virtual channels V_1 and V_2 are formed. If W_1 and W_2 are mutually independent and $W_1 = W_2 = W$, where W stands for a symmetric binary-input channel, we have $I(V_1) \leq I(W) \leq I(V_2)$ and $I(V_1) + I(V_2) = 2I(W)$, where $I(\cdot)$ denotes the capacity of a given channel. By repeating this operation n times, $N = 2^n$ independent and identical physical channels W_i's can form N different virtual channels V_i's, and $\sum_{i=1}^{N} I(V_i) = NI(W)$. It is proved in [1] that, as $N \to \infty$, degraded virtual channels will become useless, *i.e.*, their capacities tend to 0 while upgraded virtual channels will become perfect, *i.e.*, their capacities tend to 1. In addition, the fraction of perfect virtual channels will approach $I(W)$ while the fraction of useless virtual channels will approach $1-I(W)$ [1]. This phenomenon is called *channel polarization*. Thus, by sorting all virtual channels and assigning $K < NI(W)$ best virtual channels to carry information bits while leaving the other $(N-K)$ inferior virtual channels idle, we will get a class of *capacity-achieving* channel codes—polar codes [1]. Theoretically, the inputs of idle virtual channels can be arbitrarily set, but they are usually fixed to zeros in practice and thus called *frozen bits*. Due to the regular layered structure of polarization, the encoding complexity of polar codes is $O(N \log_2 N)$. As for decoding complexity, if the virtual channels are highly polarized, there is a simple and efficient *hard* decoding algorithm—*Successive Cancellation* (SC), whose complexity is $O(N \log_2 N)$ [1]. Possessing so many merits (near-linear complexity and capacity-achieving performance), polar codes immediately arouse keen interest from both academia and industry, and quickly find an application in 5G [2].

Though polar codes are capacity-achieving as $N \to \infty$, the performance is poor for short to moderate code length. There are mainly two reasons: *imperfect polarization* and *small minimum distance*. The problem of imperfect polarization can be resolved by using large kernels or adopting better decoding algorithms. Notice that SC decoding is actually designed based on the assumption of perfect polarization, so it must be carefully tuned if virtual channels are imperfectly polarized. There are two ways to improve SC decoding: One is maintaining multiple branches during decoding, *e.g.*, list decoding [3] and stack decoding [4]; the other is using iterative *soft* decoding, *e.g.*, *Soft CANcellation* (SCAN) decoding [5]. More recently, SCAN decoding is combined with the *SC List* (SCL) algorithm in [6]. For the problem of small minimum distance, there are two solutions: One is serially concatenating polar codes with some outer codes, *e.g.*, *Cyclic Redundancy Check* (CRC) codes [4]; the other is using *dynamic* frozen bits [7], *i.e.*, every frozen bit is allowed to be a function of previous information bits instead of zero.

As shown in [1], the construction of polar codes for memoryless channels consists of three steps: (a) Calculating the capacities of virtual channels induced by physical channels; (b) sorting virtual channels according to their capacities; (c) assigning best virtual channels to carry information bits. It is clear that a polar code constructed in this way is

Manuscript received January 10, 2021; revised April 24, 2021; accepted June 19, 2021. Date of publication July 6, 2021; date of current version October 18, 2021. The work was supported by the National Key Research and Development Program of China (grant no. 2020YFB0505602), the Open Research Fund of the Shaanxi Province Key Laboratory of Information Communication Network and Security, China (grant no. ICNS202002), and the Natural Science Basic Research Program of Shaanxi Province, China (grant no. 2021JM-188). The associate editor coordinating the review of this article and approving it for publication was P. Trifonov. *(Corresponding author: Yong Fang.)*

Yong Fang is with the School of Information Engineering, Chang'an University, Xi'an, Shaanxi 710064, China, and also with the Shaanxi Key Laboratory of Information Communication Network and Security, Xi'an University of Posts and Telecommunications, Xi'an, Shaanxi 710121, China (e-mail: fy@chd.edu.cn).

Jun Chen is with the Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON L8S 4K1, Canada (e-mail: chenjun@mcmaster.ca).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2021.3095195.

Digital Object Identifier 10.1109/TCOMM.2021.3095195

customized for a specific channel and may not be suitable for other channels. To address this issue, a universal channel polarization method is proposed in [8], which includes several layers of slow transforms that combine different channels followed by several layers of Arıkan's fast transforms that combine identical channels. For *universal* polar codes, the locations of "perfect" and "useless" virtual channels are almost independent of the given physical channels. It means that *polar codes can be designed without the knowledge of channels*. To decode universal polar codes effectively, the SCL algorithm is extended in [9].

Polar codes were originally designed for *stationary* channels. It is shown in [10] that Arıkan's construction polarizes *non-stationary* channels in the same way as it polarizes stationary channels. Further in [11], to speed up the polarization of non-stationary channels, Arıkan's channel polarization transform is combined with certain permutations at each polarization level as well as certain skipped operations.

All the aforementioned works on polar codes involve only *memoryless* channels. In practice, channels with *memory* widely exist in communication and storage systems, so it is important to study the applications of polar codes to such channels. In [12], a special kind of channels with memory are considered. It is proved that, if any two physical channels W_i and W_{i+k} tend to be independent as k increases, the virtual channels V_i's will be polarized by Arıkan's construction just as in the case of memoryless channels (**Theorem** 1 of [12]), and the polarization speed of upgraded virtual channels is the same as that of memoryless channels (**Theorem** 2 of [12]). In [12], it is conjectured that the polarization speed of degraded virtual channels is the same as that of memoryless channels (**Conjecture** 3 of [12]), which is proved for a special class of channels with memory in [13]. Specifically, if the inputs/outputs/states of physical channels W_i's form a *Finite-state Aperiodic Irreducible Markov* (FAIM) process, the polarization speed of degraded virtual channels is the same as that of memoryless channels (**Theorem** 13 of [13]). In [14], universal polar codes are extended to channels with memory.

A very important model for channel memory is *Inter-Symbol Interference* (ISI). In [15], a practical decoding algorithm of polar codes for ISI channels, *i.e.*, *SC Trellis* (SCT), was proposed, which makes use of the trellis structure of channel state transitions. This algorithm is generalized to arbitrary *finite-state* channels in [16]. In addition, following [17], which extends polar codes to asymmetric channels, [16] puts forward two encoding methods for polar codes, which are able to generate the desired capacity-achieving input distribution. A merit of the SCT algorithm is that it requires no knowledge of channel state.[1] As many other channel codes, the decoder of polar codes typically needs to be seeded with channel state, and accurate estimate of channel state tends to benefit decoding performance. For stationary memoryless channels, the constant state can be easily estimated at the decoder by various methods, *e.g.*, inserting additional pilot symbols [18].

[1]Note that channel *state* is different from channel *parameter*. For example, if channel state is Markovianly-varying, the state transition probability matrix is called channel *parameter*. Though the SCT algorithm requires no channel *state*, it still needs channel *parameter*.

However, for finite-state channels, since the state is time-varying, it is hardly possible to exactly trace the local state at each time. Fortunately, the SCT algorithm succeeds in handling the problem. To achieve better results, the SCL algorithm can be combined with the SCT algorithm to handle finite-state channels for (universal) polar codes [9], which will be referred to as *SC Trellis List* (SCTL) algorithm in this paper.

In this paper, we consider a special kind of finite-state channels different from ISI channels. In our model, there is no ISI, but the *state* of additive channel noise is Markovianly-varying. This model is actually a generalized form of the Gilbert-Elliott channel, which has many applications in practice. For example, it can be used to approximate the patterns of burst error, packet loss, and channel attenuation. We assume that the *parameter* of additive channel noise is known in advance at *neither* encoder *nor* decoder. Note that this setting is different from that of the SCTL algorithm. In [15] and [16], channel *parameter* is assumed to be known, even though exact channel *state* at each time is unavailable, and the SCTL algorithm seeds the decoder with channel parameter instead of channel state. In our prior works [19], [20], the problem of transmitting *Low-Density Parity-Check* (LDPC) codes over smoothly-varying channels with unknown parameter has been studied. A variant of *Belief Propagation* (BP) algorithm, *i.e.*, the so-called *Sliding-Window BP* (SWBP) algorithm, is developed, which can exactly estimate the local state of smoothly-varying channel at each time during decoding, even though channel parameter is unknown. The SWBP is a pilot-free method for joint *channel decoding* and *state estimation* that possesses many merits, *e.g.*, low complexity, low error rate, and strong robustness. So the following questions naturally arise: Can SWBP be used to decode polar codes over Gilbert-Elliott channels, and if yes, how to implement it and how well it performs?

The above questions will be answered in the present paper. We show that on the platform of SCAN decoding, it is possible to adapt the core idea of SWBP to polar codes over Gilbert-Elliott channels. The reason why we choose SCAN as the platform to extend SWBP from LDPC codes to polar codes is that SCAN decoding is an *iterative* algorithm with *low complexity*. We develop a counterpart of SWBP for polar codes, called *Sliding-Window* (SW) *SCAN* (SWSCAN). Like SWBP, the SWSCAN is also a pilot-free method for joint *channel decoding* and *state estimation*, and the role of state estimation is to improve decoding performance. Further, we enhance SWSCAN by introducing unequal tap weights and get a variant called *Weighted-Window* (WW) *SCAN* (W^{2}SCAN). Both SWSCAN and W^{2}SCAN are low-pass filters in essence, so they may be more suited to *smoothly-varying* channel. To handle Gilbert-Elliott channels with abrupt state changes, this paper proposes a method named *Linear-Weighting* (LW) *SCAN* (LWSCAN). These three adaptive SCAN decoders for Gilbert-Elliott channels are the main contributions of this paper. However, it is worth mentioning that these three adaptive SCAN decoders, in their present forms, are not yet able to cope with non-additive or non-symmetric channels.

The rest of this paper is arranged as below. Sect. II defines a generalized Gilbert-Elliott channel model. Sect. III describes the SCAN algorithm in detail. Sect. IV, Sect. V, and Sect. VI present the SWSCAN, W^{2}SCAN, and LWSCAN algorithms, respectively. Sect. VII reports simulation results. Finally, Sect. VIII concludes this paper.

II. Generalized Gilbert-Elliott Channel

A *Gilbert-Elliott* channel is a *Binary Symmetric Channel* (BSC) with time-varying crossover probability determined by a two-state Markov process. The Gilbert-Elliott channel model can be generalized from two aspects: (a) The channel may include more than two states; (b) the channel may not be a BSC. Let X_i and Y_i be the input and the output of a generalized Gilbert-Elliott channel at time i. We set $Y_i = X_i \oplus Z_i$ (for the BSC) or $Y_i = (1-2X_i) + Z_i$ (for the *Additive White Gaussian Noise* (AWGN) channel, if the *Binary Phase Shift Keying* (BPSK) modulation is used), where Z_i is the channel noise independent of X_i. It can be seen that a generalized Gilbert-Elliott channel is an additive noise channel without ISI. Assume that $X_{1:N} \triangleq (X_1, \dots, X_N)$ is an *independent and uniformly-distributed* (i.u.d.) binary random process, and $Z_{1:N}$ is a *Hidden Markov Model* (HMM) random process. Let S_i be the hidden state of Z_i. Let $\mathcal{S} = \{1, \dots, \kappa\}$ be the space of channel state. The state sequence $S_{1:N}$ is a 1st-order stationary Markov process with $\kappa \times \kappa$ transition matrix $\mathbf{A} = (\alpha_{s,t})_{\kappa\times\kappa}$, where $s, t \in \mathcal{S}$ and

$$\alpha_{s,t} \triangleq \Pr(S_{i+1} = t | S_i = s). \tag{1}$$

Further, assume that the channel $W(y_{1:N}|x_{1:N}, s_{1:N})$ is *conditionally-memoryless given state*:

$$W(y_{1:N}|x_{1:N}, s_{1:N}) = \prod_{i=1}^{N} W_i(y_i|x_i, s_i), \tag{2}$$

where $W_i(y_i|x_i, s_i)$ denotes the i-th sub-channel. For the BSC model, let $\epsilon(s)$ denote the conditional crossover probability given state $s \in \mathcal{S}$. Then given $S_i = s_i$, the *local* crossover probability at time i is $\epsilon(s_i)$ and Z_i is a Bernoulli random variable with bias probability $\epsilon(s_i)$: $Z_i \sim \mathcal{B}(\epsilon(s_i))$. For the AWGN channel model, let $\sigma^2(s)$ denote the conditional noise variance given state $s \in \mathcal{S}$. Then given $S_i = s_i$, the *local* noise variance at time i is $\sigma^2(s_i)$ and Z_i is a zero-mean Gaussian random variable with variance $\sigma^2(s_i)$: $Z_i \sim \mathcal{N}(0, \sigma^2(s_i))$. For conciseness, let $\epsilon_i \triangleq \epsilon(s_i)$ and $\sigma_i^2 \triangleq \sigma^2(s_i)$. We call $\bar{\epsilon} = \frac{1}{N}\sum_{i=1}^{N} \epsilon_i$ the *global* crossover probability and $\bar{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N} \sigma_i^2$ the *global* noise variance.

III. An Overview of Polar Codes

This section will first briefly review encoding of polar codes and then describe SCAN decoding of polar codes in detail, which is extremely important for the reader to understand the algorithms proposed in the next sections. This section ends with a discussion of decoding correctness check and a comparison of three decoders in different fields.

A. Encoding of Polar Codes

The message $u_{1:N} \in \mathbb{B}^N$ is encoded into codeword $x_{1:N} = u_{1:N}G_N = u_{1:N}B_N F^{\otimes n}$, where B_N permutes $u_{1:N}$ in the bit-reversed order, $F = \left(\begin{smallmatrix}1,0\\1,1\end{smallmatrix}\right)$, and $\otimes$ denotes the Kronecker product. Then $x_{1:N}$ is transmitted over a noisy physical channel $W_{1:N}$. If $W_{1:N}$ is a generalized Gilbert-Elliott channel, the memory between adjacent physical sub-channels will make the polarization effect of virtual sub-channels unsatisfying. To achieve better polarization effect for virtual sub-channels, we decorrelate adjacent physical sub-channels by permuting $x_{1:N}$ before transmission, *i.e.*, $x_{\pi(i)}$ is transmitted over W_i, where $(\pi(1), \dots, \pi(N))$ is a random permutation of $(1, \dots, N)$ [21], [22]. After permutation, better polarization effect can be obtained for virtual sub-channels (see the example to be given in sub-Sect. VII-A).

For polar codes, N virtual sub-channels can be constructed from N physical sub-channels by n levels of polarization. Let $V_{l,i}$ denote the i-th virtual sub-channel after l levels of polarization, where $0 \le l \le n$ and $1 \le i \le N$. Initially, we set $V_{0,\pi(i)} = W_i$. Then as l increases from 1 to n, we have $(V_{l,i}, V_{l,i+2^{n-l}}) \leftarrow (V_{l-1,i}, V_{l-1,i+2^{n-l}})$, where $1 \le (i - k2^{n-l+1}) \le 2^{n-l}$ for $0 \le k < 2^{l-1}$. Finally, N virtual sub-channels $V_{n,1:N}$ after n levels of polarization are sorted according to their capacities. For an (N, K) polar code, only K virtual sub-channels with the largest capacities are used to deliver information bits, while the others are left idle. Let $\mathcal{A}$ be the set of indices of top K virtual sub-channels and $\mathcal{A}^c = \{1, \dots, N\} \setminus \mathcal{A}$. For any $i \in \mathcal{A}^c$, u_i is called a frozen bit and is often set to 0. Recent works indicate that dynamically-set frozen bits can bring gains [7].

B. Soft CANcellation (SCAN) Decoding

Compared with LDPC codes, an advantage of polar codes is that they can be decoded at very low complexity by the SC algorithm. However, the performance of SC decoding is seriously impacted by the degree of channel polarization. For short to medium code length, which is relevant to the current main application of polar codes in 5G [2], virtual sub-channels are far from perfectly polarized, so SC decoding may perform poorly. To achieve better performance, SCL decoding [3] may be used instead. However, both SC decoding and SCL decoding are non-iterative in nature, while the algorithms to be developed in the following sections require an iterative decoder. Hence we will use SCAN decoding, which is an iterative algorithm. As shown by Fig. 1, to implement SCAN decoding, we define two $(n+1) \times N$ matrices: $(L_{l,i})_{(n+1)\times N}$ and $(R_{l,i})_{(n+1)\times N}$. The former stores the messages passed from x-nodes to u-nodes, while the latter stores the messages passed from u-nodes to x-nodes. For u_i, $R_{n,i}$ is its *intrinsic* message and $L_{n,i}$ is its *extrinsic* message, while for x_i, $L_{0,i}$ is its *intrinsic* message and $R_{0,i}$ is its *extrinsic* message. In this way, SCAN decoding can be iterated. Like SC decoding, SCAN decoding can be implemented in the *Likelihood* field (L-field), the *Likelihood Ratio* (LR) field (LR-field), or the *Log-LR* (LLR) field (LLR-field).

1) Initialization of SCAN Decoding: Let $W_i(y|x, s)$ be the i-th physical sub-channel and let $\hat{s}_i$ be the estimate of s_i.

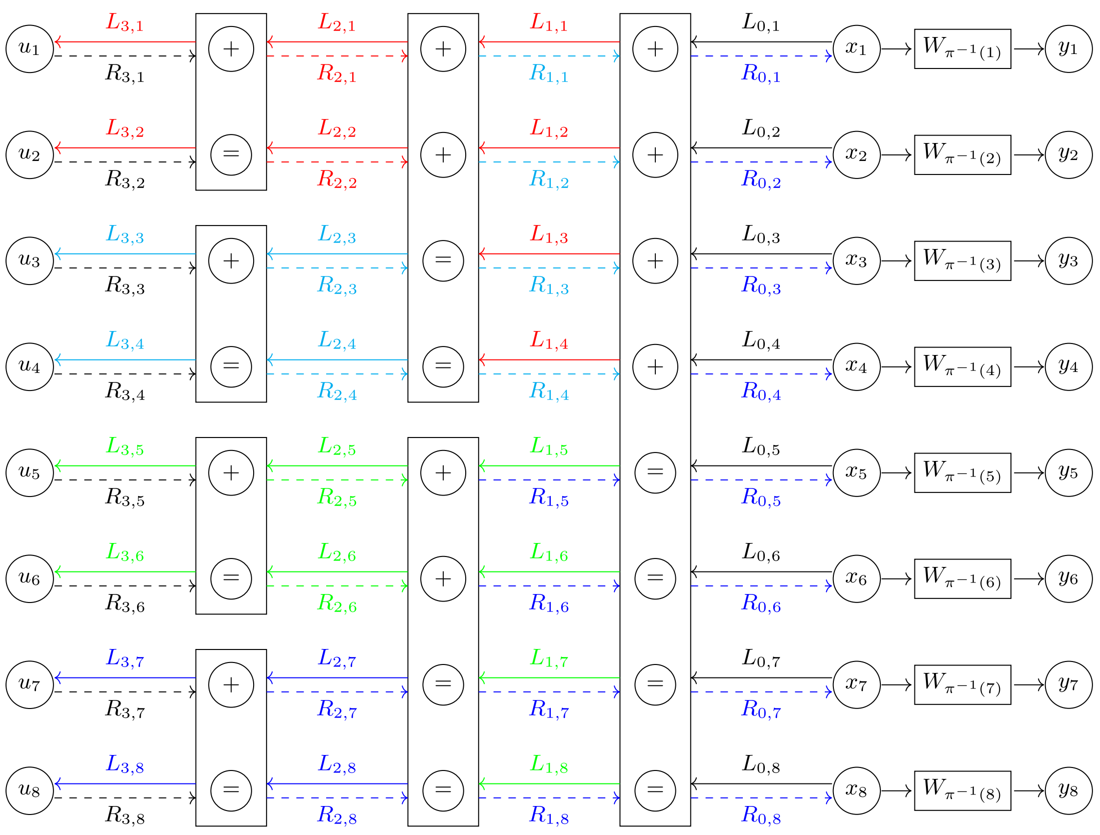

$L_{3,1}$ … $L_{3,8}$, $R_{3,1}$ … $R_{3,8}$
$L_{2,1}$ … $L_{2,8}$, $R_{2,1}$ … $R_{2,8}$
$L_{1,1}$ … $L_{1,8}$, $R_{1,1}$ … $R_{1,8}$
$L_{0,1}$ … $L_{0,8}$, $R_{0,1}$ … $R_{0,8}$
u_1 … u_8
x_1 … x_8
$W_{\pi^{-1}(1)}$ … $W_{\pi^{-1}(8)}$
y_1 … y_8

Fig. 1. An example of SCAN decoding for polar codes, where $L_{l,i}$'s refer to the messages propagated from x-nodes to u-nodes, and $R_{l,i}$'s refer to the messages propagated from u-nodes to x-nodes. The symbol $\oplus$ denotes channel degradation and the symbol $\circleddash$ denotes channel upgradation.

Depending on the running field of SCAN decoding, $L_{l,i}$ and $R_{l,i}$ should be initialized in different ways (assume that frozen bits are set to 0):

- **L-Field:** $L_{0,\pi(i)} = W_i(y_i|1,\hat{s}_i) \in [0,1]$ for $i \in [1:N]$ and $R_{n,i} = 0$ for $i \in \mathcal{A}^c$. For other l and i, we set $L_{l,i} = R_{l,i} = 0.5$.
- **LR-Field:** $L_{0,\pi(i)} = \frac{W_i(y_i|0,\hat{s}_i)}{W_i(y_i|1,\hat{s}_i)} \in \mathbb{R}^+$ for $i \in [1:N]$ and $R_{n,i} = +\infty$ for $i \in \mathcal{A}^c$. For other l and i, we set $L_{l,i} = R_{l,i} = 1$.
- **LLR-Field:** $L_{0,\pi(i)} = \log \frac{W_i(y_i|0,\hat{s}_i)}{W_i(y_i|1,\hat{s}_i)} \in \mathbb{R}$ for $i \in [1:N]$ and $R_{n,i} = +\infty$ for $i \in \mathcal{A}^c$. For other l and i, we set $L_{l,i} = R_{l,i} = 0$.

It will be seen in sub-Sect. VII-D that with our methods to be proposed in the next sections, the estimate of s_i can be very coarse while not affecting the result significantly. More details about how to set $\hat{s}_i$ can be found in the first paragraph of Sect. IV.

2) *Schedule of SCAN Decoding:* SCAN decoding includes one or more *iterations*, each of which contains $N/2$ *rounds*. Each round includes an x-to-u *pass* calculating $L_{n,1:N}$, the extrinsic messages of $u_{1:N}$, followed by a u-to-x *pass* calculating $R_{0,1:N}$, the extrinsic messages of $x_{1:N}$. It is instructive to illustrate SCAN decoding via an example. In Fig. 1, the solid lines with left arrows form the x-to-u message flow, and the dashed lines with right arrows form the u-to-x message flow. The black solid/dashed lines with left/right arrows denote the intrinsic messages of x-nodes/u-nodes. For $N = 8$, each SCAN iteration includes four rounds, marked with different colors, red for the first, cyan for the second, green for the third, and blue for the fourth. It can be seen from Fig. 1 that after each round, two elements of $L_{n,1:N}$ will be calculated, but only after the last round of an iteration, all elements of $R_{0,1:N}$ will be calculated as a whole. The schedule of SCAN decoding when $N \neq 8$ can be deduced by analogy. For more details, please refer to the released source code (polar_dec_fp.m and polar_dec_bp.m) [31].

3) *Notation for SCAN Decoding:* In [5], SCAN decoding was proposed in the LLR-field, which can be easily extended to the L-field and the LR-field. To begin with, let us define some equivalent operations in different fields. For $a, b \in [0,1]$, we define

$$a \oplus b \triangleq \frac{1-(1-2a)(1-2b)}{2} \in [0,1]. \tag{3}$$

It is interesting to note that for $a, b \in \mathbb{B}$, $a \oplus b$ defined by (3) is just the XOR of a and b. For $a, b \in \mathbb{R}^+$, we define

$$a \boxtimes b \triangleq \frac{ab+1}{a+b} \in \mathbb{R}^+. \tag{4}$$

For $x \in \mathbb{R}^+$, we define

$$\Psi(x) \triangleq -\log\tanh(x/2) \in \mathbb{R}^+. \tag{5}$$

It is easy to verify $\Psi^{-1}(x) = \Psi(x)$. We define the sign function as

$$\mathrm{sgn}(x) \triangleq \begin{cases} 0, & x \geq 0 \\ 1, & x < 0. \end{cases} \tag{6}$$

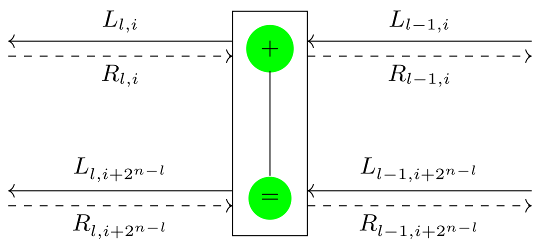


Fig. 2. Message-passing kernel for SCAN decoding, where $1 \leq l \leq n$, $0 \leq k < 2^{l-1}$, and $1 \leq (i - k2^{n-l+1}) \leq 2^{n-l}$.

With the help of $\Psi(x)$ and $\operatorname{sgn}(x)$, for $a, b \in \mathbb{R}$, we define

$$a \boxplus b \triangleq \begin{cases} |a \boxplus b| = \Psi(\Psi(|a|) + \Psi(|b|)) \\ \operatorname{sgn}(a \boxplus b) = \operatorname{sgn}(a) \oplus \operatorname{sgn}(b), \end{cases} \tag{7}$$

where $|\cdot|$ denotes the absolute value of a real number. Let $a' = \frac{1-a}{a}$, $b' = \frac{1-b}{b}$, and $c' = \frac{1-c}{c}$. Let $a'' = \log(a')$, $b'' = \log(b')$, and $c'' = \log(c')$. Given $c = a \oplus b$, it is easy to verify $c' = a' \boxtimes b'$ and $c'' = a'' \boxplus b''$. It is obvious that $a \oplus b$, $a \boxtimes b$, and $a \boxplus b$ are equivalent operations in the L-field, the LR-field, and the LLR-field, respectively.

Finally, for $a, b \in [0, 1]$, we define

$$a \otimes b \triangleq \frac{ab}{ab + (1-a)(1-b)} \in [0, 1]. \tag{8}$$

It is easy to verify that $a \otimes b$, $a \times b$, and $a + b$ are equivalent operations in the L-field, the-LR field, and the LLR-field, respectively.

4) Message-Passing Kernel of SCAN Decoding: Fig. 2 shows the message-passing kernel. Let $1 \leq (i - k2^{n-l+1}) \leq 2^{n-l}$, where $1 \leq l \leq n$ and $0 \leq k < 2^{l-1}$. With the notations defined in the previous subsection, we can get the message-passing kernels in different fields. For example, in LLR-field, the kernel for the x-to-u message passing is

$$\begin{cases} L_{l,i} = L_{l-1,i} \boxplus (L_{l-1,i+2^{n-l}} + R_{l,i+2^{n-l}}) \\ L_{l,i+2^{n-l}} = (L_{l-1,i} \boxplus R_{l,i}) + L_{l-1,i+2^{n-l}}, \end{cases} \tag{9}$$

and the kernel for the u-to-x message passing is

$$\begin{cases} R_{l-1,i} = R_{l,i} \boxplus (L_{l-1,i+2^{n-l}} + R_{l,i+2^{n-l}}) \\ R_{l-1,i+2^{n-l}} = (L_{l-1,i} \boxplus R_{l,i}) + R_{l,i+2^{n-l}}. \end{cases} \tag{10}$$

In L-field, the message-passing kernels can be obtained by replacing $\boxplus$ with $\oplus$ and replacing $+$ with $\otimes$. In LR-field, the message-passing kernels can be obtained by replacing $\boxplus$ with $\boxtimes$ and replacing $+$ with $\times$. For more details, please refer to the source code (channel_upgrade.m and channel_degrade.m) released in [31].

5) Termination of SCAN Decoding: Let β_i be the overall message of u_i, *i.e.*,

$$\beta_i = \begin{cases} L_{n,i} \otimes R_{n,i}, & \text{in L-field} \\ L_{n,i} \times R_{n,i}, & \text{in LR-field} \\ L_{n,i} + R_{n,i}, & \text{in LLR-field.} \end{cases} \tag{11}$$

After each SCAN iteration, for every $i \in \mathcal{A}$, we make a hard decision: $\hat{u}_i = \mathbf{1}_{\mathcal{I}}(\beta_i)$, where

$$\mathbf{1}_{\mathcal{I}}(x) \triangleq \begin{cases} 1, & x \in \mathcal{I} \\ 0, & x \notin \mathcal{I} \end{cases} \tag{12}$$

and $\mathcal{I}$ takes different forms depending on the implementation field:

$$\mathcal{I} = \begin{cases} (0.5, 1], & \text{in L-field} \\ [0, 1), & \text{in LR-field} \\ (-\infty, 0), & \text{in LLR-field.} \end{cases} \tag{13}$$

Usually, decoding correctness of polar codes is verified by CRC checksums. If frozen bits are fixed, we can define a set $\mathcal{C} \subset \mathcal{A}$ and let $u_{\mathcal{C}}$ be the CRC checksum of $u_{\mathcal{A}\setminus\mathcal{C}}$. If $\hat{u}_{\mathcal{A}\setminus\mathcal{C}}$ and $\hat{u}_{\mathcal{C}}$ satisfy the CRC constraint, $\hat{u}_{\mathcal{A}\setminus\mathcal{C}}$ is deemed correct and the decoding will be ceased [4]. If frozen bits are dynamic, $u_{\mathcal{A}^c}$ is the CRC checksum of $u_{\mathcal{A}}$. If $\hat{u}_{\mathcal{A}}$ and $\hat{u}_{\mathcal{A}^c}$ satisfy the CRC constraint, $\hat{u}_{\mathcal{A}}$ is deemed correct and the decoding will be ceased [7].

However, with SCAN, the decoding correctness of polar codes can be verified in the absence of CRC checksums [6], [23]. After each SCAN iteration, besides $\hat{u}_{\mathcal{A}}$, we also estimate $x_{1:N}$. Let γ_i be the overall message of x_i, *i.e.*,

$$\gamma_i = \begin{cases} L_{0,i} \otimes R_{0,i}, & \text{in L-field} \\ L_{0,i} \times R_{0,i}, & \text{in LR-field} \\ L_{0,i} + R_{0,i}, & \text{in LLR-field.} \end{cases} \tag{14}$$

For every $i \in [1 : N]$, we make a hard decision: $\hat{x}_i = \mathbf{1}_{\mathcal{I}}(\gamma_i)$, where $\mathcal{I}$ is defined by (13). If $\hat{u}_{1:N} G_N = \hat{x}_{1:N}$, the decoding is ceased; otherwise, one more iteration is run. Just as LDPC codes, to avoid endless loop, a threshold is set as the upper bound of the iteration number. If the iteration number is greater than the threshold, the decoding will be forcedly terminated. Now it can be seen that decoding correctness can be verified without resorting to additional information not contained in polar codes.

C. Comparison of SCAN Decoding in Different Fields

1) Stability: It is well known that decoding channel codes in the L-field and the LR-field is usually unstable due to the *Not-a-Number* (NaN) issue that occurs if $0 \cdot \infty$, $0/0$, or ∞/∞. By convention, we define $\infty/\infty = 0/0 = 0 \cdot \infty = 1$. Let us analyze the stability of SCAN decoding in different fields:

- **L-Field:** There are two kinds of operations: $\oplus$ and $\otimes$. As shown by (3), since two operands of $\oplus$ are defined over $[0, 1]$, the $\oplus$ operation has no NaN issue. For the $\otimes$ operation, as shown by (8), the NaN issue will occur if one operand is 0 and the other is 1. We define $(0 \otimes 1) = (1 \otimes 0) = 0.5$.
- **LR-Field:** There are two kinds of operations: $\boxtimes$ and $\times$. At first glance, the $\boxtimes$ operation defined by (4) has the NaN issue. However, if we rewrite (4) as

$$\frac{ab + 1}{a + b} = \frac{1}{1/a + 1/b} + \frac{1}{a + b} \in \mathbb{R}^+, \tag{15}$$

  the NaN issue is eliminated automatically. It is easy to get $(0 \boxtimes 0) = (\infty \boxtimes \infty) = \infty$ and $(0 \boxtimes \infty) = (\infty \boxtimes 0) = 0$. For the $\times$ operation, the NaN issue is unavoidable.
- **LLR-Field:** There are two kinds of operations: $\boxplus$ and $+$. Then according to (7), the definition of $\boxplus$, it is obvious that SCAN decoding in the LLR-field has neither multiplication nor division, so there is no NaN issue.

With the NaN issue properly handled, SCAN decoding in different fields will produce similar results, as shown in sub-Sect. VII-B.

2) Complexity: It is well known that the arithmetic operations have an increasing order of hardware implementation complexities as follows: bitwise operation, addition/subtraction, multiplication, and division. Let us make a comparison as below:

- **L-Field:** As shown by (3), $a \oplus b$ needs only one multiplication of $(1-2a)$ and $(1-2b)$ (note that multiplying or dividing a variable by 2 can be implemented by bitwise shifting). As shown by (8), $a \otimes b$ needs two multiplications and one division. Overall, three multiplications and one division are needed.
- **LR-Field:** As shown by (15), $a \boxtimes b$ needs four divisions. Overall, one multiplication and four divisions are needed.
- **LLR-Field:** As shown by (7), the key component in the calculation of $a \boxplus b$ is the function $\Psi(\cdot)$, which is usually implemented by a *Look-Up Table* (LUT). Hence, neither multiplication nor division is needed.

From the above analysis, we can find that the complexity of SCAN decoding decreases in the order of LR-field, L-field, and LLR-field. However, if the LUT technique is not used, SCAN decoding in the LLR-field will be of the highest complexity, because the $\log$ and $\tanh$ operations are very time-consuming, as shown in sub-Sect. VII-B.

IV. State Estimation via Sliding-Window Algorithm

Now we focus on $L_{0,1:N}$, the intrinsic messages of x-nodes. To trigger SCAN decoding, we seed $L_{0,1:N}$ with estimated local states of physical sub-channels:

- **BSC model**: $X_{\pi(i)}$ is sent and $Y_{\pi(i)} = X_{\pi(i)} \oplus Z_i$ is received. Let ϵ_i be the bias probability of Z_i and $\hat{\epsilon}_i$ be the estimate of ϵ_i. Then

$$L_{0,\pi(i)} = \begin{cases} (1-\hat{\epsilon}_i)y_{\pi(i)} + \hat{\epsilon}_i(1-y_{\pi(i)}), & \text{in L-field} \\ (\frac{1-\hat{\epsilon}_i}{\hat{\epsilon}_i})^{1-2y_{\pi(i)}}, & \text{in LR-field} \\ (1-2y_{\pi(i)})\log\frac{1-\hat{\epsilon}_i}{\hat{\epsilon}_i}, & \text{in LLR-field.} \end{cases} \tag{16}$$

- **AWGN channel model**: $(1-2X_{\pi(i)})$ is sent and $Y_{\pi(i)} = (1-2X_{\pi(i)}) + Z_i$ is received. Let σ_i^2 be the variance of Z_i and $\hat{\sigma}_i^2$ be the estimate of σ_i^2. Then

$$L_{0,\pi(i)} = \begin{cases} \dfrac{1}{1+\exp(2y_{\pi(i)}/\hat{\sigma}_i^2)}, & \text{in L-field} \\ \exp(2y_{\pi(i)}/\hat{\sigma}_i^2), & \text{in LR-field} \\ 2y_{\pi(i)}/\hat{\sigma}_i^2, & \text{in LLR-field.} \end{cases} \tag{17}$$

Once the intrinsic messages $L_{0,1:N}$ of x-nodes are initialized, they will remain unchanged during decoding. However, the decoder may be less effective if the channel is mis-estimated. For LDPC codes, if the intrinsic messages of variable nodes are coarsely seeded, a significant gain can be achieved by elaborately refining the intrinsic messages of variable nodes after each BP iteration. There are many schemes that can achieve this goal. Among them, SWBP decoding [19], [20] is particularly attractive due to its low error rate, low complexity, and strong robustness. Since SCAN decoding is also an iterative algorithm, it is natural to expect that the SWBP algorithm can be extended from LDPC codes to polar codes on the platform of SCAN decoding. The principle is as below. First, we initialize $\hat{\epsilon}_i$ or $\hat{\sigma}_i^2$ arbitrarily. For example, it is usually not difficult to estimate the global crossover probability $\bar{\epsilon} = \frac{1}{N}\sum_{i=1}^{N}\epsilon_i$ (for the BSC model) or the global noise variance $\bar{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}\sigma_i^2$ (for the AWGN channel model), so we can initially set $\hat{\epsilon}_i = \hat{\epsilon}$ or $\hat{\sigma}_i^2 = \hat{\sigma}^2$ for all $i \in [1:N]$, where $\hat{\epsilon}$ is the estimate of $\bar{\epsilon}$ and $\hat{\sigma}^2$ is the estimate of $\bar{\sigma}^2$. Then after each SCAN iteration, we update $\hat{\epsilon}_i$ or $\hat{\sigma}_i^2$ by some means.

A. Problem Formulation

At the decoder, depending on the channel model, either $y_{\pi(1:N)} = x_{\pi(1:N)} \oplus z_{1:N}$ or $y_{\pi(1:N)} = (1-2x_{\pi(1:N)}) + z_{1:N}$ is received. After each SCAN iteration, we get γ_i, the overall message of x_i, by (14), which can be used to calculate the *a posteriori* probability that $X_i = 1$:

$$p_i = \begin{cases} \gamma_i, & \text{in L-field} \\ \frac{1}{1+\gamma_i}, & \text{in LR-field} \\ \frac{1}{1+\exp(\gamma_i)}, & \text{in LLR-field.} \end{cases} \tag{18}$$

From y_i and p_i, depending on the channel model, we either calculate (for the BSC model)

$$\hat{z}_i \triangleq |p_{\pi(i)} - y_{\pi(i)}| \tag{19}$$

or calculate (for the AWGN channel model)

$$\hat{z}_i^2 \triangleq p_{\pi(i)}(y_{\pi(i)}+1)^2 + (1-p_{\pi(i)})(y_{\pi(i)}-1)^2. \tag{20}$$

We can take $\hat{z}_i$ as the *soft estimate* of z_i. Now our problem is how to estimate $\epsilon_{1:N}$ from $\hat{z}_{1:N}$ or estimate $\sigma_{1:N}^2 \triangleq (\sigma_1^2, \ldots, \sigma_N^2)$ from $\hat{z}_{1:N}^2 \triangleq (\hat{z}_1^2, \ldots, \hat{z}_N^2)$.

B. Sliding-Window SCAN

1) Basic Principle: Let us consider the BSC model first. To avoid out-of-bound accesses, for window size $(2m+1)$, we pad $\hat{z}_{1:N}$ to get

$$\hat{z}_{(1-m):(N+m)} = (\hat{z}_m, \ldots, \hat{z}_1, \hat{z}_{1:N}, \hat{z}_N, \ldots, \hat{z}_{N-m+1}), \tag{21}$$

i.e., $\hat{z}_i = \hat{z}_{1-i}$ for $(1-m) \le i \le 0$ and $\hat{z}_i = \hat{z}_{2N+1-i}$ for $(N+1) \le i \le (N+m)$. This is actually the symmetric padding. Note that (21) is not an ad hoc formula and we will find its use in sub-Sect. V-B. Then a simple way to estimate ϵ_i is

$$\hat{\epsilon}_i = \frac{1}{2m+1}\sum_{i'=-m}^{m}\hat{z}_{i+i'} = \hat{\epsilon}_{i-1} + \frac{1}{2m+1}(\hat{z}_{i+m} - \hat{z}_{i-1-m}). \tag{22}$$

Similarly, a simple way to estimate σ_i^2 is

$$\begin{aligned} \hat{\sigma}_i^2 &= \frac{1}{2m+1}\sum_{i'=-m}^{m}\hat{z}_{i+i'}^2 \\ &= \hat{\sigma}_{i-1}^2 + \frac{1}{2m+1}(\hat{z}_{i+m}^2 - \hat{z}_{i-1-m}^2). \end{aligned} \tag{23}$$

It is also shown by (22) and (23) that $\hat{\epsilon}_i$ or $\hat{\sigma}_i^2$ can be deduced recursively by *sliding window*, so given $\hat{\epsilon}_{i-1}$ or $\hat{\sigma}_{i-1}^2$, the order of complexity to calculate $\hat{\epsilon}_i$ or $\hat{\sigma}_i^2$ is $O(1)$, irrespective of half window size m. We refer to this scheme as *Sliding-Window* (SW) *SCAN* (SWSCAN). A similar scheme based on BP decoding of LDPC codes is called SWBP in [19], [20].

2) Window Size: As shown by (22) and (23), half window size m plays a key role in SWSCAN. To find the optimal m, we need to exploit the *conditionally-memoryless* property. Let us take the BSC model for example. We slightly modify (22) to get

$$\hat{\epsilon}_i(m) = \frac{1}{2m}\sum_{i'=1}^{m}(\hat{z}_{i-i'} + \hat{z}_{i+i'}) \tag{24}$$

and define the *cross entropy* between $\hat{z}_{1:N}$ and $\hat{\epsilon}_{1:N}$ as

$$\eta(m) \triangleq -\sum_{i=1}^{N}(\hat{z}_i \log\hat{\epsilon}_i(m) + (1-\hat{z}_i)\log(1-\hat{\epsilon}_i(m))). \tag{25}$$

By comparing (24) with (22), one can find that $\hat{z}_i$ is excluded from the estimate of ϵ_i, which makes $\hat{\epsilon}_i(m)$ and $\hat{z}_i$ approximately *conditionally independent* of each other given ϵ_i. In other words, $\hat{\epsilon}_i(m)$ and $\hat{z}_i$ can be approximately taken as two *independent observations* of ϵ_i, *i.e.*, $\hat{z}_i \leftrightarrow \epsilon_i \leftrightarrow \hat{\epsilon}_i(m)$. Hence, the cross entropy $\eta(m)$ can be used as a criterion to evaluate different half window sizes m. Similarly, for the AWGN channel model, we slightly modify (23) to get

$$\hat{\sigma}_i^2(m) = \frac{1}{2m}\sum_{i'=1}^{m}(\hat{z}_{i-i'}^2 + \hat{z}_{i+i'}^2) \tag{26}$$

and define the *cross entropy* between $\hat{z}_{1:N}^2$ and $\hat{\sigma}_{1:N}^2$ as

$$\begin{aligned}\eta(m) &\triangleq -\textstyle\sum_{i=1}^{N}\log\left(\frac{1}{\sqrt{2\pi\hat{\sigma}_i^2(m)}}\exp(-\frac{\hat{z}_i^2}{2\hat{\sigma}_i^2(m)})\right)\\ &\propto \textstyle\sum_{i=1}^{N}\left(\frac{\hat{z}_i^2}{\hat{\sigma}_i^2(m)} + \log\hat{\sigma}_i^2(m)\right).\end{aligned} \tag{27}$$

Now the optimal m can be obtained by

$$\dot{m} = \arg\min_{m}\eta(m). \tag{28}$$

As shown by (25) and (27), the order of complexity to compute $\eta(m)$ is $O(N)$. In view of the low complexity of $\eta(m)$, we suggest implementing (28) simply by a full search over $\{1,\ldots,N/2\}$. If so, the order of complexity of SWSCAN is $O(N^2)$, the same as that of SWBP [19], [20].

C. Underflow and Overflow Handling

As shown by (24), for small m, it is likely that $\hat{\epsilon}_i(m)$ is equal to 0 or 1, which will cause illegal log operation in (25). In the SWBP algorithm [19], [20], this problem is solved by setting lower and upper bounds for $\hat{\epsilon}_i(m)$. However, such a solution is not very desirable because it not only increases the complexity but also induces a new problem, namely, how to set lower and upper bounds. Inspired by the *Krichevsky-Trofimov* (KT) formula [24], we propose a better solution to this problem. For the BSC model, we modify (24) to get

$$\hat{\epsilon}_i(m) = \frac{1}{2m+1}\left(\tilde{\epsilon} + \sum_{i'=1}^{m}(\hat{z}_{i-i'} + \hat{z}_{i+i'})\right), \tag{29}$$

where $0 < \tilde{\epsilon} < 1$. The standard KT formula sets $\tilde{\epsilon}$ to 0.5, but we set $\tilde{\epsilon}$ to $\frac{1}{N}\sum_{i=1}^{N}\hat{z}_i$ in (29). It is because the KT formula was originally designed for arithmetic coding that is sequential in nature. Hence, the arithmetic codec has no *a priori* knowledge of $z_{1:N}$ before encoding or decoding. On the contrary, most channel codes, *e.g.*, polar codes and LDPC codes, are block codes. With the knowledge of $\hat{z}_{1:N}$, it is conceivably better to set $\tilde{\epsilon}$ to the mean of $\hat{z}_{1:N}$ rather than 0.5. Similarly, for the AWGN channel model, we set $\tilde{\sigma}^2 \triangleq \frac{1}{N}\sum_{i=1}^{N}\hat{z}_i^2$ and refine (26) to

$$\hat{\sigma}_i^2(m) = \frac{1}{2m+1}\left(\tilde{\sigma}^2 + \sum_{i'=1}^{m}(\hat{z}_{i-i'}^2 + \hat{z}_{i+i'}^2)\right). \tag{30}$$

V. State Estimation via Weighted-Window Algorithm

As shown by (29) and (30), SWSCAN decoding (and SWBP decoding for LDPC codes) is actually equivalent to a low-pass filter with *equal* tap weights $\frac{1}{2m+1}$. It is natural to attempt *unequal* tap weights. That is the topic of this section.

A. Weighted-Window SCAN

1) BSC Model: Let $(w_{-m},\ldots,w_{-1},w_0,w_1,\ldots,w_m)$ be the tap weight vector, where w_0 is the coefficient of $\tilde{\epsilon}$ in (29) or the coefficient of $\tilde{\sigma}^2$ in (30). It is very reasonable to set w_0 to the average of all $(2m+1)$ weights, *i.e.*, $w_0 \equiv \frac{1}{2m+1}$. If so, when $m=0$, we have $w_0 = 1$ and thus $\hat{\epsilon}_i = \tilde{\epsilon}$ or $\hat{\sigma}_i^2 = \tilde{\sigma}^2$, which is the desired result, and as m increases, the effect of $\tilde{\epsilon}$ or $\tilde{\sigma}^2$ will be diluted. Moreover, for simplicity, it is very natural to use symmetric taps, *i.e.*, $w_{i'} = w_{-i'}$ for $1 \le i' \le m$. Let $a_{i+i'} \triangleq \hat{z}_{i-i'} + \hat{z}_{i+i'}$ and $\mathbf{a}_i \triangleq (a_{i+1},\ldots,a_{i+m})^\top \in \mathbb{R}^{m\times 1}$ for $1 \le i \le N$. Let $\mathbf{w} \triangleq (w_1,\ldots,w_m)^\top \in \mathbb{R}^{m\times 1}$. For the BSC model, each term of (29) is unequally weighted:

$$\hat{\epsilon}_i(\mathbf{w}) \triangleq w_0\tilde{\epsilon} + \sum_{i'=1}^{m} w_{i'}(\hat{z}_{i-i'} + \hat{z}_{i+i'}) = \mathbf{a}_i^\top\mathbf{w} + w_0\tilde{\epsilon}. \tag{31}$$

We approximate the cross entropy between $\hat{z}_{1:N}$ and $\hat{\epsilon}_{1:N}$ with the ℓ_2-distance to reduce the complexity. It is easy to get

$$\begin{aligned}(\hat{\epsilon}_i(\mathbf{w}) - \hat{z}_i)^2 &\propto \left(\mathbf{w}^\top\mathbf{a}_i\mathbf{a}_i^\top\mathbf{w} - 2(\hat{z}_i - w_0\tilde{\epsilon})\mathbf{a}_i^\top\mathbf{w}\right)\\ &= \left(\mathbf{w}^\top\mathbf{H}_i\mathbf{w} - 2\mathbf{f}_i^\top\mathbf{w}\right),\end{aligned} \tag{32}$$

where $\mathbf{H}_i = \mathbf{a}_i\mathbf{a}_i^\top \in \mathbb{R}^{m\times m}$ and $\mathbf{f}_i = (\hat{z}_i - w_0\tilde{\epsilon})\mathbf{a}_i \in \mathbb{R}^{m\times 1}$. Let $\mathbf{H} = \sum_{i=1}^{N}\mathbf{H}_i$ and $\mathbf{f} = \sum_{i=1}^{N}\mathbf{f}_i$. Then $\eta(m)$ defined by (25) can be approximated with

$$\eta(\mathbf{w}) \triangleq \sum_{i=1}^{N}(\hat{\epsilon}_i(\mathbf{w}) - \hat{z}_i)^2 \propto \left(\mathbf{w}^\top\mathbf{H}\mathbf{w} - 2\mathbf{f}^\top\mathbf{w}\right). \tag{33}$$

This is a standard least-square problem with the optimal solution

$$\dot{\mathbf{w}} = \arg\min_{\mathbf{w}}\eta(\mathbf{w}) = \mathbf{H}^{-1}\mathbf{f}. \tag{34}$$

However, the optimal $\mathbf{w}$ obtained by (34) performs very poorly in practice, because there is no constraint on $\mathbf{w}$. In many cases, the autocorrelation function of a random process looks like the geometric distribution. Hence, considering its physical meanings, the following constraints on $\mathbf{w}$ are reasonable:

- **Non-negativity** and **Monotonicity**: $w_1 \geq \cdots \geq w_m \geq 0$. This is because the correlation between Z_i and Z_{i+k} should be weaker as k increases.
- **Convexity**: $(w_{i'-1} - w_{i'}) \geq (w_{i'} - w_{i'+1})$, *i.e.*, $w_{i'} \leq \frac{w_{i'-1}+w_{i'+1}}{2}$. This is due to the convexity of geometric distribution.
- **Normality**: $w_0 + \sum_{i'=1}^{m} 2w_{i'} = 1$ and thus $\sum_{i'=1}^{m} w_{i'} = \frac{m}{2m+1}$.

With the above constraints imposed, the problem is now formatted as

$$\dot{\mathbf{w}} = \arg\min_{\mathbf{w}} \eta(\mathbf{w}) \quad \text{s.t.} \quad w_1 \geq \cdots \geq w_m \geq 0, \quad w_{i'} \leq \frac{w_{i'-1}+w_{i'+1}}{2}, \quad \sum_{i'=1}^{m} w_{i'} = \frac{m}{2m+1}. \tag{35}$$

It is a standard quadratic programming problem that can be easily solved. We refer to this scheme as *Weighted-Window* (WW) *SCAN* (W^{2}SCAN) decoding. It is easy to verify that the constraint set in (35) is not empty, so the optimization problem defined by (35) has a solution. Let us give an example to show that the optimal solution to (35) may not be an equal-weight vector. Assume $m = 2$ and $\mathbf{f} = \mathbf{0}$. If $\mathbf{H} = \left(\begin{smallmatrix}1,0\\0,1\end{smallmatrix}\right)$, it is easy to get that the solution to (35) is $\dot{\mathbf{w}} = (1/5, 1/5)^\top$, which is an equal-weight vector. However, if $\mathbf{H} = \left(\begin{smallmatrix}1,0\\0,2\end{smallmatrix}\right)$, the solution to (35) is $\dot{\mathbf{w}} = (4/15, 2/15)^\top$, which is not an equal-weight vector.

2) AWGN Channel Model: We define $a_{i+i'} \triangleq \hat{z}_{i-i'}^2 + \hat{z}_{i+i'}^2$ and $\mathbf{a}_i \triangleq (a_{i+1}, \ldots, a_{i+m})^\top \in \mathbb{R}^m$ for $1 \leq i \leq N$. Then it is easy to get

$$\hat{\sigma}_i^2(\mathbf{w}) = \mathbf{a}_i^\top \mathbf{w} + w_0 \tilde{\sigma}^2 \tag{36}$$

and

$$(\hat{\sigma}_i^2(\mathbf{w}) - \hat{z}_i^2)^2 \propto \left(\mathbf{w}^\top \mathbf{H}_i \mathbf{w} - 2\mathbf{f}_i^\top \mathbf{w}\right), \tag{37}$$

where $\mathbf{H}_i = \mathbf{a}_i \mathbf{a}_i^\top \in \mathbb{R}^{m\times m}$ and $\mathbf{f}_i = (\hat{z}_i^2 - w_0\tilde{\sigma}^2)\mathbf{a}_i \in \mathbb{R}^m$. Still let $\mathbf{H} = \sum_{i=1}^{N} \mathbf{H}_i$ and $\mathbf{f} = \sum_{i=1}^{N} \mathbf{f}_i$. We approximate (27) with

$$\eta(\mathbf{w}) \triangleq \sum_{i=1}^{N} (\hat{\sigma}_i^2(\mathbf{w}) - \hat{z}_i^2)^2 \propto \left(\mathbf{w}^\top \mathbf{H} \mathbf{w} - 2\mathbf{f}^\top \mathbf{w}\right). \tag{38}$$

Since (38) and (33) have exactly the same form, they can be solved using the same method (35). The only difference is that $\eta(\mathbf{w})$ in (35) is now defined by (38).

B. Complexity of W^{2}SCAN

Now we consider how to calculate $\mathbf{H}$ and $\mathbf{f}$. Let $\mathbf{H} = (h_{k,l})_{m\times m}$ and $\mathbf{f} = (f_k)_{m\times 1}$, where $1 \leq k, l \leq m$. For the BSC model, we have

$$h_{k,l} = h_{l,k} = \sum_{i=1}^{N} a_{i+k} a_{i+l} = \sum_{i=1}^{N} (\hat{z}_{i-k} + \hat{z}_{i+k})(\hat{z}_{i-l} + \hat{z}_{i+l}) \tag{39}$$

and

$$f_k = \left(\sum_{i=1}^{N} \hat{z}_i a_{i+k}\right) - w_0 \tilde{\epsilon} \left(\sum_{i=1}^{N} a_{i+k}\right). \tag{40}$$

According to the definition of a_{i+k}, we have

$$g_k \triangleq \sum_{i=1}^{N} \hat{z}_i a_{i+k} = \sum_{i=1}^{N} \hat{z}_i (\hat{z}_{i-k} + \hat{z}_{i+k}) \tag{41}$$

and

$$\sum_{i=1}^{N} a_{i+k} = \sum_{i=1}^{N} \hat{z}_{i-k} + \sum_{i=1}^{N} \hat{z}_{i+k}. \tag{42}$$

Following the padding formula (21), it is easy to get

$$\begin{cases} \sum_{i=1}^{N} \hat{z}_{i-k} = 2\sum_{i'=1}^{k} \hat{z}_{i'} + \sum_{i'=k+1}^{N-k} \hat{z}_{i'} \\ \sum_{i=1}^{N} \hat{z}_{i+k} = \sum_{i'=k+1}^{N-k} \hat{z}_{i'} + 2\sum_{i'=N-k+1}^{N} \hat{z}_{i'} \end{cases}. \tag{43}$$

Substituting (43) into (42), we get

$$\sum_{i=1}^{N} a_{i+k} = 2\sum_{i'=1}^{N} \hat{z}_{i'} = 2N\tilde{\epsilon}, \tag{44}$$

where $\tilde{\epsilon} = \frac{1}{N}\sum_{i=1}^{N} \hat{z}_i$. Note that $\sum_{i=1}^{N} a_{i+k}$ is a constant for all $1 \leq k \leq m$. Finally, we get

$$f_k = g_k - \frac{2N\tilde{\epsilon}^2}{2m+1}. \tag{45}$$

Let $\mathbf{\Phi} \triangleq (\phi_{k,l})_{(2m+1)\times(2m+1)} \in \mathbb{R}^{(2m+1)\times(2m+1)}$, where $-m \leq k, l \leq m$, and

$$\phi_{k,l} = \phi_{l,k} \triangleq \sum_{i=1}^{N} \hat{z}_{i+k} \hat{z}_{i+l}. \tag{46}$$

As shown by (46), $\phi_{k,l}$ is to some extent similar to the autocorrelation of $\hat{z}_{1:N}$. Then according to (39) and (41), we have

$$\begin{cases} h_{k,l} = \phi_{-k,-l} + \phi_{-k,l} + \phi_{k,-l} + \phi_{k,l} \\ g_k = \phi_{k,0} + \phi_{-k,0} \end{cases}. \tag{47}$$

Thus, if only $\mathbf{\Phi}$ is known, $\mathbf{H}$ and $\mathbf{f}$ can be easily calculated. According to (46), the order of complexity to calculate $\mathbf{\Phi}$ is $O(Nm^2)$. Note that $\mathbf{H}$ is a *covariance* matrix, so it is a positive semi-definite Hessian matrix and (35) can be solved in polynomial time [25]. As $\mathbf{H} \in \mathbb{R}^{m\times m}$, the polynomial complexity of (35) is in m, not N. More precisely, the order of complexity of solving (35) is $O(m^3)$ (see Sect. 5.3.2 of [26]). The complexity of W^{2}SCAN is dominated by two steps: calculating $\mathbf{\Phi}$, whose order of complexity is $O(Nm^2)$, and solving (35), whose order of complexity is $O(m^3)$. Since $m \leq N/2$, the order of complexity of W^{2}SCAN is $O(Nm^2)$.

The above analysis for the BSC model can be easily extended to the AWGN channel model by replacing (45) with

$$f_k = g_k - \frac{2N\tilde{\sigma}^4}{2m+1}, \tag{48}$$

where $\tilde{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N} \hat{z}_i^2$, and replacing (46) with

$$\phi_{k,l} = \phi_{l,k} \triangleq \sum_{i=1}^{N} \hat{z}_{i+k}^2 \hat{z}_{i+l}^2. \tag{49}$$

C. Window Size

The half window size m is a factor that has significant impact on the performance of W^{2}SCAN. Varying m leads to a tradeoff between error rate and complexity: Larger m will lower error rate but increase computational complexity. One choice is setting the half window size m of W^{2}SCAN to $\dot{m}$ obtained by (28) of SWSCAN. This is actually a mixture of SWSCAN and W^{2}SCAN, and its order of complexity is $O(N^3)$ in the worst case of $m = N/2$. For simplicity, in all experiments of this paper, we set the half window size m of W^{2}SCAN to $\sqrt{N}$, and thus the order of complexity is $O(N^2)$, the same as that of SWSCAN. Hence, we can make a fair comparison between SWSCAN and W^{2}SCAN.

VI. State Estimation via Linear-Weighting Algorithm

As shown in Sect. IV and Sect. V, both SWSCAN and W^{2}SCAN are low-pass filters in essence, so they may not work well for *piecewise-stationary* channel with *abrupt* state changes. This section will make a different attempt to handle this issue.

The *Linear-Weighting* (LW) algorithm, also known as Willems' LW algorithm named after the inventor Willems [27], is a classical method to estimate the local state of piecewise-stationary binary sources. Combining arithmetic coding with the LW algorithm can compress piecewise-stationary binary sources at very low redundancy [27]. Below, we will first introduce the standard form of LW algorithm, then apply it to state estimation for time-varying channels. Finally, we will compare SWSCAN, W^{2}SCAN, and LWSCAN.

A. Basic Principle of LW Algorithm

Let $x_{1:N}$ be a piecewise-stationary binary source with unknown parameter. The LW algorithm is designed for sequential codec. To encode x_i, source bias probability at time i is needed. If there is an abrupt state change between $x_{i'-1}$ and $x_{i'}$, where $1 \le i' \le i$, and the piece $x_{i':i}$ is stationary, then the bias probability of x_i can be estimated by $x_{i':(i-1)}$ with the KT formula [24]. Let T_i, where $1 \le T_i \le i$, denote the time of the last abrupt state change before x_i. More specifically, if $T_i = i'$, where $1 \le i' \le i$, there is an abrupt state change between $x_{i'-1}$ and $x_{i'}$, and the piece $x_{i':i}$ is stationary. Initially, $\Pr(T_1 = 1) = 1$. Let us define

$$q_{i,i'} \triangleq \Pr(X_{1:i} = x_{1:i}, T_i = i'), \tag{50}$$

where $1 \le i \le N$ and $1 \le i' \le i$. Initially, $q_{1,1} = \Pr(X_1 = x_1, T_1 = 1) = \Pr(X_1 = x_1) \cdot \Pr(T_1 = 1) = 0.5$. For simplicity, we will abbreviate $q_{i,i'}$ as $q_{i'}$. Let $\acute{\epsilon}_i$ be the estimated bias probability of x_i. We sequentially encode $x_{1:N}$ with $\acute{\epsilon}_{1:N}$. Initially, $\acute{\epsilon}_1 = 0.5$. Then for $2 \le i \le N$, the following steps are carried out to estimate ϵ_i.

1) Calculate **State Weight**: $w_{i'} = q_{i'}(1-\alpha_{i'})$ for $1 \le i' \le (i-1)$ and $w_i = \sum_{i'=1}^{i-1} q_{i'}\alpha_{i'}$, where $\alpha_{i'} = \frac{1}{2(i-i')}$ is the transition probability from state i' to state i.
2) Estimate the **Conditional Bias Probability** at time i given state i' with the KT formula:

$$\varepsilon_{i'} = \frac{0.5 + \sum_{i''=i'}^{i-1} x_{i''}}{i - i' + 1}, \tag{51}$$

where $1 \le i' \le i$. In particular, if $i = i'$, then $\varepsilon_{i'} = 0.5$.
3) Estimate the **Bias Probability** at time i:

$$\acute{\epsilon}_i = \frac{\sum_{i'=1}^{i} w_{i'}\varepsilon_{i'}}{\sum_{i'=1}^{i} w_{i'}}. \tag{52}$$

4) $q_{i'}$ is updated through

$$q_{i'} = w_{i'}(x_i\varepsilon_{i'} + (1-x_i)(1-\varepsilon_{i'})). \tag{53}$$

B. Adapt LW Algorithm for Channel Coding

Consider the following communication problem: $x_{1:N} \in \mathbb{B}^N$ is sent and $y_{1:N} = x_{1:N} \oplus z_{1:N} \in \mathbb{B}^N$ is received. Assume that $x_{1:N}$ is memoryless and uniformly distributed, and $x_{1:N}$ and $z_{1:N}$ are mutually independent. The noise sequence $z_{1:N}$ can be viewed as a binary source whose bias probability vector $\epsilon_{1:N}$ is just the crossover probability vector between $x_{1:N}$ and $y_{1:N}$. Though $z_{1:N}$ is unknown during decoding, its soft estimate $\hat{z}_{1:N}$ can be obtained by (19) after each SCAN iteration. So with $x_{1:N}$ of (51) and (53) replaced by $\hat{z}_{1:N}$, the LW algorithm in sub-Sect. VI-A can be leveraged to estimate the crossover probability vector $\epsilon_{1:N}$ between $x_{1:N}$ and $y_{1:N}$. This approach is most suitable when the noise sequence is approximately piecewise-stationary.[2]

However, different from source coding which usually proceeds sequentially, channel coding is typically implemented in a blockwise manner. Hence the LW algorithm can be further improved. First, as with SWSCAN and W^{2}SCAN, we can set $\acute{\epsilon}_1 = \tilde{\epsilon} \triangleq \frac{1}{N}\sum_{i=1}^{N} \hat{z}_i$ and

$$\varepsilon_{i'} = \frac{\tilde{\epsilon} + \sum_{i''=i'}^{i-1} \hat{z}_{i''}}{i - i' + 1}, \tag{54}$$

i.e., the bias 0.5 in (51) is replaced with $\tilde{\epsilon}$. Second, the LW algorithm can perform both forward and backward iterations in the block channel coding setting as opposed to forward iterations only in the sequential source coding setting. Let $\grave{\epsilon}_i$ be the estimate of ϵ_i by the backward LW algorithm. Better performance can be achieved if we use $\hat{\epsilon}_i = \frac{\acute{\epsilon}_i + \grave{\epsilon}_i}{2}$ to update the intrinsic messages $L_{0,1:N}$ for SCAN decoding of polar codes. We refer to this scheme as *Linear-Weighting SCAN* (LWSCAN). Note that different from the forward-backward algorithm for HMM, $\hat{\epsilon}_i$ is equal to the arithmetic average rather than geometric average of $\acute{\epsilon}_i$ and $\grave{\epsilon}_i$, because the KT formula is itself an arithmetic average.

C. Extension to AWGN Channel

For the AWGN channel model, $x_{1:N} \in \mathbb{B}^N$ is sent and $y_{1:N} = (1 - 2x_{1:N}) + z_{1:N}$ is received. Let $\hat{z}^2_{1:N}$ be the soft estimate of $z^2_{1:N}$ obtained by (20) after each SCAN iteration. Let $\sigma^2_{1:N}$ be the variance vector of $z_{1:N}$ and $\acute{\sigma}^2_{1:N}$ be the forward estimate of $\sigma^2_{1:N}$. Initially, $\acute{\sigma}^2_1 = \tilde{\sigma}^2 \triangleq \frac{1}{N}\sum_{i=1}^{N} \hat{z}^2_i$.

[2]For example, this is the case when $\alpha_{s,s} \approx 1$ for $s \in \mathcal{S}$.

Then σ_i^2 for $2 \leq i \leq N$ can be estimated by the following steps.

1) Calculate **State Weight**: Transition probabilities are purely subject to the investigated state model, no matter whether the channel is BSC or AWGN. So this step is completely the same as in sub-Sect. VI-A.
2) Estimate the **Conditional Noise Variance** at time i given state i': The variance of a normal random variable can be estimated by the arithmetic average of squared samples, so this step is a generalization of the KT formula:
$$\varsigma_{i'}^2 = \frac{\tilde{\sigma}^2 + \sum_{i''=i'}^{i-1} \hat{z}_{i''}^2}{i - i' + 1}, \tag{55}$$
where $1 \leq i' \leq i$. In particular, if $i = i'$, then $\varsigma_{i'}^2 = \tilde{\sigma}^2$.
3) Estimate the **Noise Variance** at time i:
$$\acute{\sigma}_i^2 = \frac{\sum_{i'=1}^{i} w_{i'} \varsigma_{i'}^2}{\sum_{i'=1}^{i} w_{i'}}. \tag{56}$$
4) As shown by (53), $q_{i'}$ is actually the product of $w_{i'}$ and $\Pr(X_i = x_i | T_i = i')$. It is easy to deduce by analogy that $q_{i'}$ should be updated through
$$q_{i'} = \frac{w_{i'}}{\sqrt{2\pi \varsigma_{i'}^2}} \cdot \exp\left(-\frac{\hat{z}_i^2}{2\varsigma_{i'}^2}\right). \tag{57}$$

Of course, the above steps can be iterated backward to obtain $\grave{\sigma}_{1:N}^2$, the backward estimate of $\sigma_{1:N}^2$. Finally, we use $\hat{\sigma}_{1:N}^2 = \frac{\acute{\sigma}_{1:N}^2 + \grave{\sigma}_{1:N}^2}{2}$ to update the intrinsic messages $L_{0,1:N}$ and then run the next SCAN iteration. This is the AWGN-version of LWSCAN.

D. Comparison Between SWSCAN, W^{2}SCAN, and LWSCAN

1) Complexity: As analyzed in [27], the order of complexity of the LW algorithm is $O(N^2)$, so the order of complexity of LWSCAN is also $O(N^2)$, the same as that of SWSCAN. Since the two competitors have the same order of complexity, they can be fairly compared. However, it should be noticed that the LW algorithm runs sequentially, so it is hard to be parallelized. On the contrary, the SW algorithm can be easily parallelized because the metrics of different half window sizes are independent of each other and can be calculated in parallel. As for W^{2}SCAN, the complexity is on the order of $O(Nm^2)$, so if m is comparable to N, *e.g.*, $m = N/2$, the complexity of W^{2}SCAN is higher than SWSCAN and LWSCAN.

2) Memory: It is easy to find that for both SWSCAN and LWSCAN, the additional memory usage for state estimation is on the order of $O(N)$. As for W^{2}SCAN, if m is comparable to N, the additional memory usage for state estimation is on the order of $O(m^2)$, higher than that of SWSCAN and LWSCAN, because $\mathbf{\Phi}$ is the largest matrix of size $(2m+1) \times (2m+1)$. For more detail, the reader may refer to the source code released in [31].

VII. Experimental Results

On the platform of SCAN decoder, we have realized three adaptive decoders, *i.e.*, SWSCAN, W^{2}SCAN, and LWSCAN. Below we will: (a) give a simple example to illustrate the advantage of permutation for polar codes over channels with memory; (b) compare the implementations of SCAN decoder in different fields; (c) compare SWSCAN decoder with SCTL decoder; and (d) compare three adaptive SCAN decoders. We use *Frame-Error-Rate* (FER) and *Bit-Error-Rate* (BER) as the indicator of error-correcting capability and use runtime as the indicator of decoding complexity.

A. Advantage of Permutation

In sub-Sect. III-A, it is pointed out that for channels with memory, the permutation step after polar encoding is advantage for the polarization of virtual sub-channels. Now we give a simple example to verify this assertion. Consider a two-state AWGN channel with transition probability $\alpha_{1,2} = \alpha_{2,1} = 1/16$. The AWGN variance is 0.5 at the good state and 2 at the bad state. The intrinsic messages of x-nodes, $L_{0,1:N}$, where code length $N = 2^{10}$, are initialized with global AWGN variance $\bar{\sigma}^2 = 1.25$. We run 10^4 trials and count the Bhattacharyya parameter for each virtual sub-channel. The result is shown in Fig. 3. Note that the two sub-figures have different ranges along the $Z(W)$-direction. As shown by sub-Fig. 3(a), if there is no permutation after encoding, the maximal value of $Z(W)$ is smaller than 0.5, *i.e.*, virtual sub-channels are insufficiently polarized. In contrast, as shown by sub-Fig. 3(b), if the output bits of polar encoder are permuted, the maximal value of $Z(W)$ is 0.5, *i.e.*, virtual sub-channels are deeply polarized. This simple example clearly confirms the advantage of the permutation step for polar codes over channels with memory.

B. SCAN Decoders in Different Fields

SCAN decoding was originally carried in the LLR-field [5]. In Sect. III, we also implement SCAN decoding in the L-field and the LR-field. Now we give a simple example to compare SCAN decoders in different fields. We sort $N = 2^{10}$ virtual sub-channels according to the 5G reliability file [2] and set the code rate to 0.5, *i.e.*, only $K = N/2 = 2^9$ best virtual sub-channels are dedicated to information bits. Five *memoryless* AWGN channels with different noise variances are evaluated. When $E_b/N_0 = 1$dB, the decoding is ceased if 256 erroneous frames are detected; similarly, 128 for $E_b/N_0 = 1.5$dB, 64 for $E_b/N_0 = 2$dB, 32 for $E_b/N_0 = 2.5$dB, and 16 for $E_b/N_0 = 3$dB. For each trial, at most 10 iterations are attempted. As here we are more concerned with error rates, no LUT technique is used for the LLR-SCAN. It can be seen from Fig. 4 that the implementations of SCAN decoding in different fields yield very similar performance, but as E_b/N_0 increases, there is a trend that the LLR-SCAN wins out, while the LR-SCAN becomes worse [sub-Fig. 4(a)]. In sub-Fig. 4(b), we show the ratio of running time of LR-SCAN and LLR-SCAN to L-SCAN. Note that both LR-SCAN and LLR-SCAN are slower than L-SCAN, and the LLR-SCAN is the slowest, if no LUT technique is used.

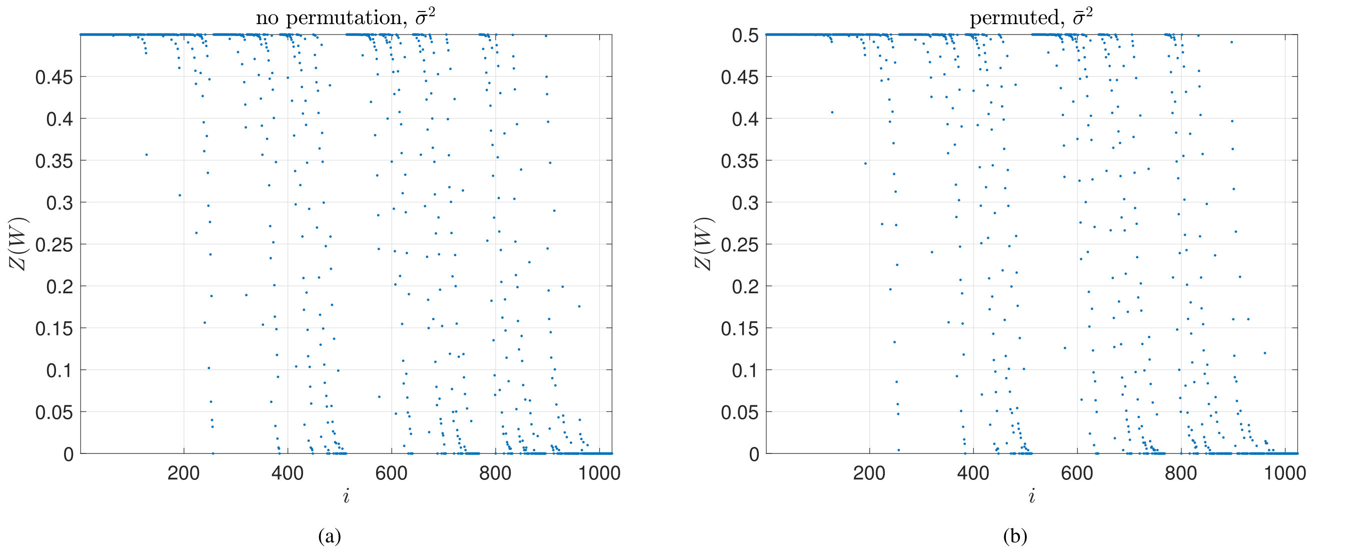


Fig. 3. A simple example of Bhattacharyya parameter to illustrate the advantage of permutation after encoding for polar codes over channels with memory. (a) No permutation. (b) With permutation.

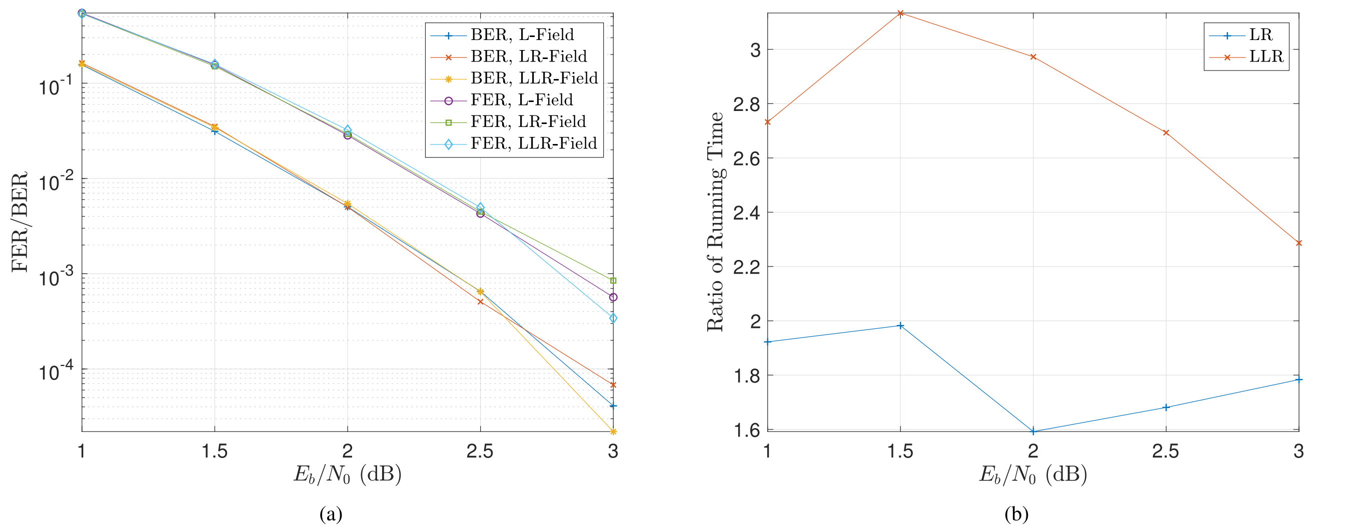
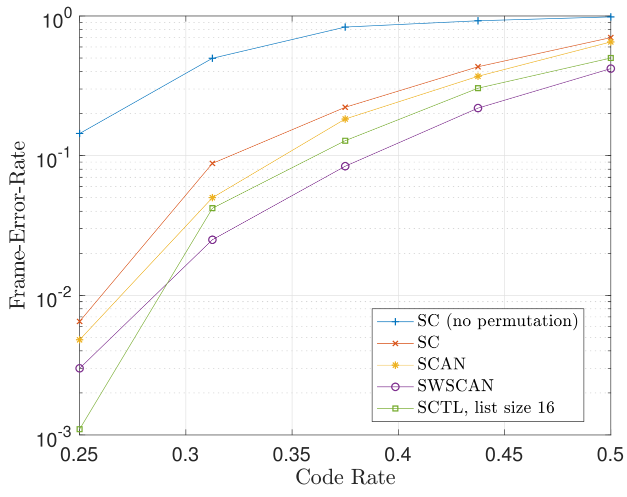


Fig. 4. Comparison of the implementations of SCAN decoding in different fields. (a) Error rates. (b) Ratio of running time of LR-SCAN and LLR-SCAN to L-SCAN.

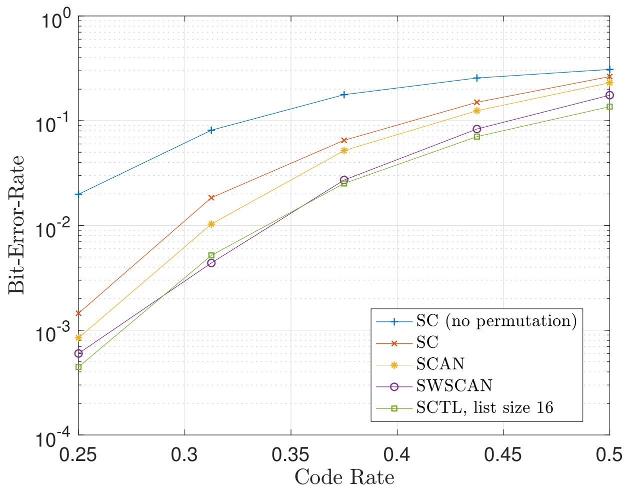
C. Comparison With SCTL Decoder

The SCT algorithm was proposed in [15] for decoding polar codes over ISI channels, and was extended to general finite-state channels in [16]. The SCTL algorithm is the combination of the SCT algorithm with the SCL algorithm [9]. Its order of complexity is $O(|\mathcal{S}|^3\mathcal{L}N\log_2 N)$ and its order of memory usage is $O(|\mathcal{S}|^2\mathcal{L}N\log_2 N)$, where $|\mathcal{S}|$ is the cardinality of state space, $\mathcal{L}$ is the list size, and N is the code length [9]. It is interesting to make a comparison between SCTL and SWSCAN. Consider a Gilbert-Elliott channel with state transition probability $\alpha_{1,2} = \alpha_{2,1} = 0.01$. The crossover probability is 0 at the good state and 0.2 at the bad state, hence the global crossover probability is $\bar{\epsilon} = 0.1$. For SCTL, we run the algorithm in [16] to obtain sorted virtual sub-channels, and the list size is set to 16. For SWSCAN, we still use the 5G reliability file [2], and the maximum iteration number is set to 10. We test five code rates from 0.25 to 0.5. For code rate 0.25, we run 10^4 trials, while for other rates, we run 10^3 trials. In Fig. 5, the parameter set of the Gilbert-Elliott channel is *known* at the receiver for the SCTL decoder, while for other decoders, the parameter set is *unknown* and the intrinsic messages of x-nodes, $L_{0,1:N}$, are initialized with $\bar{\epsilon}$. Except SCTL, all decoders are implemented in the L-field. From Fig. 5, we can make some interesting observations. First, for channels with memory, it is very important to permute the output bits of polar encoder (cf. Fig. 3). Second, SCAN improves SC, and further, SWSCAN improves SCAN. Third, in general, SWSCAN is *significantly* better than SCTL in term of FER, but *slightly* worse in term of BER. Finally, as code rate decreases, there is a trend that SCTL will win out.

Fig. 5 reveals that for hidden Markov channels, the methods based on state transition trellis, *e.g.*, the SCTL algorithm, may not yield satisfactory results. It is not beyond our expectation because a similar phenomenon has been found for the problem



(a)



(b)

Fig. 5. Comparison of SWSCAN decoder with SCTL decoder. (a) FER. (b) BER.

of Slepian-Wolf coding [28], *i.e.*, lossless distributed source coding, which is in essence a problem of channel coding. This is because polar codes are a class of block codes, and the hidden Markov model may be de-synchronized for any blockwise coding scheme based on state transition trellis. For example, it is found in [29] that the forward-backward algorithm is not enough for Slepian-Wolf coding of HMM-correlated binary sources due to asynchronization. To re-synchronize the hidden Markov model, a small proportion of the original source bits, which are equi-spaced, must be transmitted accompanying the compressed bitstream [29]. The asynchronization problem of hidden Markov model can be solved with sequential coding schemes, *e.g.*, distributed arithmetic coding [30]. If one still sticks to block codes, then the coding scheme should be based on state estimation, *e.g.*, SWBP, SWSCAN, *etc.*, rather than state transition trellis.

D. Comparison of Adaptive SCAN Decoders

1) Preparation: We use a κ-state HMM-AWGN channel to test the proposed algorithms. The space of AWGN variance is $\{0, \frac{2\bar{\sigma}^2}{\kappa-1}, \ldots, \frac{2(\kappa-1)\bar{\sigma}^2}{\kappa-1}\}$, and the state transition probability is $\alpha_{s,t} \equiv \lambda^{-1}$ for all $s \neq t$. The 5G reliability file [2] is used to sort $N = 2^{10}$ virtual sub-channels and code rate is fixed to 0.5. We define $E_b/N_0 = -10\log_{10}\bar{\sigma}^2$, where $\bar{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}\sigma_i^2$. Information bits $u_{\mathcal{A}}$ are uniformly generated and the forbidden bits $u_{\mathcal{A}^c}$ are set to zero. After encoding, $x_{1:N}$ is *randomly* permuted. For SCAN and its variants, the maximum iteration number is set to 10. All decoders are implemented in the LR-field. The following experiments are run on a six-core CPU. For each setting, at most 10^4 trials are run on each core, and the loop on each core will be terminated if 50 erroneous frames are detected.

2) Benchmarks: Our benchmarks are the SC and SCAN decoders seeded with $\bar{\sigma}^2$. As shown by sub-Fig. 6(a), compared with the SC decoder, about 0.2dB gain can be achieved by the SCAN decoder with 10 iterations. To show how much gain can be achieved if the channel state is perfectly known at the decoder, we seed the SC and SCAN decoders with $\sigma_{1:N}^2$. As shown by sub-Fig. 6(a), for both SC and SCAN decoders, significant gains can be obtained if x-nodes are seeded with $\sigma_{1:N}^2$ instead of $\bar{\sigma}^2$. For the SC decoder, accurate channel state information can bring about 0.8dB gain, while for the SCAN decoder, accurate channel state information can bring about 1dB gain. Hence, there is still potentially considerable room for improvement with respect to adaptive channel estimation.

3) Gain of Adaptive SCAN Decoders: We compare three adaptive SCAN decoders with the original non-adaptive SCAN decoder in sub-Fig. 6(b), where all decoders are seeded with $\bar{\sigma}^2$. It can be found that adaptive SCAN decoders can achieve about 0.2dB gain. It can also be found that among three adaptive SCAN decoders, LWSCAN is the best, SWSCAN is the worst, and W^{2}SCAN is in between.

It is noticed in [20] that SWBP is robust to the initial estimate of $\sigma_{1:N}^2$. Even though $\hat{\sigma}_{1:N}^2$ is set to $2\bar{\sigma}^2$ or $\bar{\sigma}^2/2$ instead of $\bar{\sigma}^2$, almost the same performance can be achieved [20]. Compared with channel underestimation, *e.g.*, $\hat{\sigma}_{1:N}^2$ is set to $\bar{\sigma}^2/2$, channel overestimation, *e.g.*, $\hat{\sigma}_{1:N}^2$ is set to $2\bar{\sigma}^2$, needs fewer iterations [20]. This finding coincides with the well-known fact that channel overestimation is usually less destructive than channel underestimation. So we also attempt seeding adaptive SCAN decoders with $2\bar{\sigma}^2$. As shown in sub-Fig. 6(c), surprisingly, adaptive SCAN decoders seeded with $2\bar{\sigma}^2$ can achieve about 0.4dB gain, significantly higher than adaptive SCAN decoders seeded with $\bar{\sigma}^2$. This finding not only shows the robustness of adaptive SCAN decoders to initial settings, but also reconfirms the advantage of channel overestimation [20].

Encouraged by the success of channel overestimation for adaptive SCAN decoders, we also attempt seeding non-adaptive decoders with $2\bar{\sigma}^2$. Unfortunately, as shown in sub-Fig. 6(a), the performance of non-adaptive decoders exhibits severe degradation if the channel is overestimated. An unexpected finding from sub-Fig. 6(a) is that for low E_b/N_0, the iterative SCAN decoder is even worse than the non-iterative SC decoder.

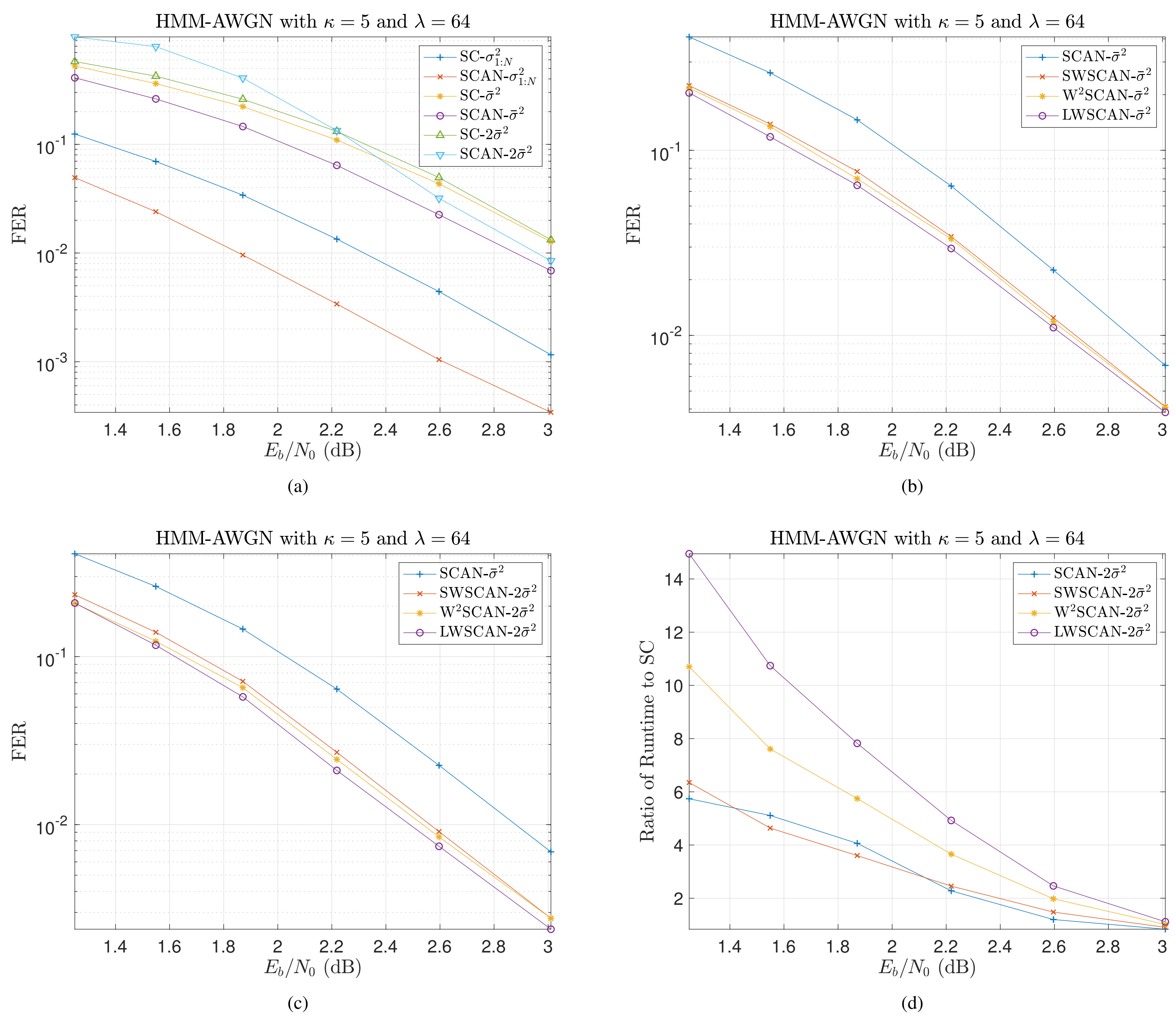


Fig. 6. Comparison of adaptive SCAN decoders with non-adaptive decoders. FER comparison between (a) SC decoder and SCAN decoder, (b) SCAN decoder and adaptive SCAN decoders, and (c) SCAN decoder and adaptive SCAN decoders with overestimated AWGN variances. (d) Runtime comparison between SCAN decoder and adaptive SCAN decoders.

Finally, we compare the complexity of three adaptive SCAN decoders. The ratio of running time between each (adaptive) SCAN decoder and the SC decoder is shown in sub-Fig. 6(d). It can be found that the SWSCAN decoder runs almost as fast as the SCAN decoder. Surprisingly, in some cases, the SWSCAN decoder is even faster than the SCAN decoder. There are two reasons for this phenomenon. On one hand, the metric formulas (25) and (27) for each sliding-window size is very simple, and on the other hand, fewer iterations are needed by the SWSCAN decoder due to refined estimates of channel states. A similar phenomenon regarding the SWBP decoder for LDPC codes is reported in [20]. The W^{2}SCAN decoder runs slower than the SWSCAN decoder, but faster than the LWSCAN decoder. So from the viewpoint of complexity, SWSCAN is better than W^{2}SCAN, and W^{2}SCAN is better than LWSCAN.

VIII. Conclusion

This paper studies the problem of decoding polar codes over a special class of finite-state channels, *i.e.*, generalized Gilbert-Elliott channels, with unknown parameters. It is revealed that, by permuting codewords before transmission and adopting iterative SCAN decoding, it is possible to estimate the time-varying channel state online. Three methods have been proposed in this work. The first is SWSCAN, which is an extension of SWBP from LDPC codes to polar codes. By optimizing the tap weights of SWSCAN with quadratic programming, we get W^{2}SCAN. The third is LWSCAN, which is an extension of the LW algorithm from source estimation to channel estimation. Simulation results show that three adaptive SCAN algorithms significantly improve the performance of polar codes over generalized Gilbert-Elliott channels.

The software package has been released on the first author's GitHub [31].

Acknowledgment

The authors would like to thank I. Tal and B. Shuval for their generosity in sharing the source code for SCTL (which makes the comparison with SWSCAN possible) and for their guidance and suggestions throughout this work. The authors also thank Z.-P. Wang, a professor with Beihang University, China, and F. Jiang, a professor with Xi'an University of Posts and Telecommunications, China, for their supports.

References

[1] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.

[2] *NR Multiplexing and Channel Coding (Release 15)*, document TSG RAN TS 138.212, V15.8.0, 3GPP, Jan. 2020.

[3] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[4] K. Niu and K. Chen, "CRC-aided decoding of polar codes," *IEEE Commun. Lett.*, vol. 16, no. 10, pp. 1668–1671, Oct. 2012.

[5] U. U. Fayyaz and J. R. Barry, "Low-complexity soft-output decoding of polar codes," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 958–966, May 2014.

[6] C. Pillet, C. Condo, and V. Bioglio, "SCAN list decoding of polar codes," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Dublin, Ireland, Jun. 2020, pp. 1–6.

[7] P. Trifonov and V. Miloslavskaya, "Polar codes with dynamic frozen symbols and their decoding by directed search," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Seville, Spain, Sep. 2013, pp. 1–5.

[8] E. Sasoglu and L. Wang, "Universal polarization," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 2937–2946, Jun. 2016.

[9] B. Shuval and I. Tal, "List decoding of universal polar codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angles, CA, USA, Jun. 2020, pp. 389–394.

[10] M. Alsan and E. Telatar, "A simple proof of polarization and polarization for non-stationary memoryless channels," *IEEE Trans. Inf. Theory*, vol. 62, no. 9, pp. 4873–4878, Sep. 2016.

[11] H. Mahdavifar, "Fast polarization for non-stationary channels," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 849–853.

[12] E. Sasoglu and I. Tal, "Polar coding for processes with memory," *IEEE Trans. Inf. Theory*, vol. 65, no. 4, pp. 1994–2003, Apr. 2019.

[13] B. Shuval and I. Tal, "Fast polarization for processes with memory," *IEEE Trans. Inf. Theory*, vol. 65, no. 4, pp. 2004–2020, Apr. 2019.

[14] B. Shuval and I. Tal, "Universal polarization for processes with memory," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, Jul. 2019, pp. 2089–2093.

[15] R. Wang, R. Liu, and Y. Hou, "Joint successive cancellation decoding of polar codes over intersymbol interference channels," Apr. 2014, *arXiv:1404.3001*. [Online]. Available: http://arxiv.org/abs/1404.3001

[16] R. Wang, J. Honda, H. Yamamoto, R. Liu, and Y. Hou, "Construction of polar codes for channels with memory," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Jeju Island, South Korea, Oct. 2015, pp. 187–191.

[17] J. Honda and H. Yamamoto, "Polar coding without alphabet extension for asymmetric models," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7829–7838, Dec. 2013.

[18] L. Li, Z. Xu, and Y. Hu, "Channel estimation with systematic polar codes," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 4880–4889, Jun. 2018.

[19] Y. Fang, "LDPC-based lossless compression of nonstationary binary sources using sliding-window belief propagation," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3161–3166, Nov. 2012.

[20] Y. Fang, "Asymmetric Slepian–Wolf coding of nonstationarily-correlated M-ary sources with sliding-window belief propagation," *IEEE Trans. Commun.*, vol. 61, no. 12, pp. 5114–5124, Dec. 2013.

[21] M. Kamenev, Y. Kameneva, O. Kurmaev, and A. Maevskiy, "Permutation decoding of polar codes," in *Proc. XVI Int. Symp. Problems Redundancy Inf. Control Syst. (REDUNDANCY)*, Moscow, Russia, Oct. 2019, pp. 1–6.

[22] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Belief propagation list decoding of polar codes," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1536–1539, Aug. 2018.

[23] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Belief propagation decoding of polar codes on permuted factor graphs," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Moscow, Russia, Apr. 2018, pp. 1–6.

[24] R. Krichevsky and V. Trofimov, "The performance of universal encoding," *IEEE Trans. Inf. Theory*, vol. IT-27, no. 2, pp. 199–207, Mar. 1981.

[25] M. K. Kozlov, S. P. Tarasov, and L. G. Khachiyan, "The polynomial solvability of convex quadratic programming," *USSR Comput. Math. Math. Phys.*, vol. 20, no. 5, pp. 223–228, Jan. 1980.

[26] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 1996.

[27] F. M. J. Willems, "Coding for a binary independent piecewise-identically-distributed source," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 2210–2217, Nov. 1996.

[28] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory*, vol. IT-19, no. 4, pp. 471–480, Jul. 1973.

[29] J. Garcia-Frias and W. Zhong, "LDPC codes for compression of multi-terminal sources with hidden Markov correlation," *IEEE Commun. Lett.*, vol. 7, no. 3, pp. 115–117, Mar. 2003.

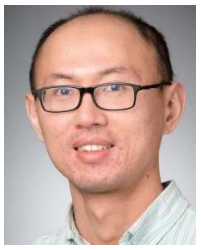
[30] Y. Fang and J. Jeong, "Distributed arithmetic coding for sources with hidden Markov correlation," Dec. 2008, *arXiv:2101.02336*. [Online]. Available: http://arxiv.org/abs/2101.02336

[31] *Polar Codes*. Accessed: Jan. 10, 2021. [Online]. Available: https://github.com/fy79/Polar-Code.git

Yong Fang received the B.Eng., M.Eng., and Ph.D. degrees in communications engineering from the School of Communications Engineering, Xidian University, Xi'an, Shaanxi, China, in 2000, 2003, and 2005, respectively.

In 2007, he was a Lecturer with the School of Electronics and Information Engineering, Northwestern Polytechnic University, Xi'an. From 2007 to 2008, he was a Research Professor with the Department of Electrical and Computer Engineering, Hanyang University, Seoul, South Korea. From 2009 to 2016, he was a Full Professor with the College of Information Engineering, Northwest A&F University, Yangling, Shaanxi. From March to September 2015, he visited the University of Waterloo, Waterloo, ON, Canada, as a Senior Visiting Scholar. From July to August 2016, he visited the University of California-San Diego, San Diego, CA, USA, as a Visiting Professor. He is currently a Full Professor with the School of Information Engineering, Chang'an University, Xi'an. His research interests include information theory, coding techniques, pattern recognition, compressive sensing, image/video coding, processing, and transmission. He had a long experience in hardware development for FPGA-based (Xilinx Vertex series) video codec design and DSP-based (TI C64 series) video surveillance systems.

Dr. Fang was a recipient of the EAI IoTaaS Best Paper Award in 2020 and the AMIA TBI Best Student Paper Award in 2016. He served as an Area Editor for *International Journal of Electronics and Communications* from 2016 to 2017. He is also an Associate Editor of IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE COMMUNICATIONS LETTERS.



Jun Chen (Senior Member, IEEE) received the B.E. degree in communication engineering from Shanghai Jiao Tong University, Shanghai, China, in 2001, and the M.S. and Ph.D. degrees in electrical and computer engineering from Cornell University, Ithaca, NY, USA, in 2004 and 2006, respectively.

From September 2005 to July 2006, he was a Post-Doctoral Research Associate with the Coordinated Science Laboratory, University of Illinois at Urbana–Champaign, Urbana, IL, USA, and a Post-Doctoral Fellow with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA, from July 2006 to August 2007. Since September 2007, he has been with the Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON, Canada, where he is currently a Professor. His research interests include information theory, machine learning, wireless communications, and signal processing.

Dr. Chen was a recipient of the Josef Raviv Memorial Postdoctoral Fellowship in 2006, the Early Researcher Award from the Province of Ontario in 2010, the IBM Faculty Award in 2010, the ICC Best Paper Award in 2020, and the JSPS Invitational Fellowship in 2021. He held the title of the Barber-Gennum Chair of information technology from 2008 to 2013 and the title of the Joseph Ip Distinguished Engineering Fellow from 2016 to 2018. He served as an Editor for IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING from 2020 to 2021. He is also an Associate Editor of IEEE TRANSACTIONS ON INFORMATION THEORY.