Received 12 August 2022; revised 18 September 2022; accepted 9 October 2022. Date of publication 14 October 2022; date of current version 26 October 2022.

Digital Object Identifier 10.1109/OJCOMS.2022.3214578

Fast Simulation of Coded QAM Transmission in White Gaussian Noise at Low Packet Error Rates

YOU-ZONG YU (Graduate Student Member, IEEE), DAVID W. LIN (Life Senior Member, IEEE), AND TZU-HSIEN SANG (Member, IEEE)

Institute of Electronics, College of Electrical and Computer Engineering, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan

CORRESPONDING AUTHOR: Y.-Z. YU (e-mail: yozongyu@gmail.com)

This work was supported in part by the Ministry of Science and Technology of R.O.C. under Grant MOST 108-2221-E-009-099 and Grant MOST 111-2221-E-A49-156-MY3.

ABSTRACT Today's communication system design heavily depends on computer simulation for performance evaluation. The burgeoning ultra-reliable communication systems, however, pose a significant simulation challenge as such systems operate at very low packet error rates (PERs) whereas the required simulation time of the conventional Monte Carlo (MC) method is many times the inverse of the PER. Various importance sampling-type techniques have been developed for more efficient simulation of channel-coded transmission, but they typically rely on exploiting code weaknesses (for the generation of error-causing noise samples) and most works on soft-decision decoding only treat binary signaling. In this paper, we propose to use a function, termed the noise gauging function (NGF), that roughly measures the error-causing propensity of noise samples and we present a way to adaptively optimize the noise sampling under such a function for simulation efficiency. Both binary and nonbinary signalings are considered. And the proposed technique does not require detailed knowledge of the code weaknesses, although some high-level understanding of the code properties can benefit the design of efficient NGFs. We investigate the application of the proposed technique to several common channel codes. Numerical results indicate an approximately 10- to 1,000-fold speedup versus MC.

INDEX TERMS BCH codes, channel coding, convolutional codes, fast simulation, importance sampling, Monte Carlo, packet error rate, polar codes, ultra reliable and low latency communication (URLLC).

I. INTRODUCTION

THE FIFTH generation (5G) wireless communication standards spearheaded ultra reliable and low latency communication (URLLC) to support mission-critical communication in application scenarios such as factory automation and intelligent transportation. Later generations of standards and practical systems are expected to further such capability [1]. Low latency aspects notwithstanding, the ultra reliability aspects, as currently conceived, involve packet error rates (PERs) in the range of 10^{-5} to 10^{-9} with packet sizes on the order of 32 bytes [2], [3]. To achieve the required performance, proper channel coded modulation is indispensable. In this regard, current 5G specifications have continued the common recent practice of bit-interleaved coded modulation (BICM) in standard wireless communication systems [4], [5] and employed BICM built on low-density parity check (LDPC) and polar codings [6], [7]. There is no doubt that a similar practice will continue as related research continues to progress, e.g., [8], [9].

Now, today's communication system design makes heavy use of computer simulation for performance evaluation, for which the Monte Carlo (MC) method is a most ready choice due to its simplicity and general applicability. For error-probability evaluation, however, it requires on the order of $1/(\epsilon^2 p_e)$ simulation runs to attain a relative precision of ϵ at an (*a priori* unknown) error probability p_e. Hence it encounters difficulty in dealing with low error probabilities (such as PERs below 10^{-7}), especially for complicated systems. Usually, one is not interested only in obtaining the error performance at one particular operating point, but in tracing out the PER performance over some range of signal-to-noise (SNR) values for various design alternatives. At times the extreme-value theory or some understanding of the tail probability property can be invoked to extrapolate the MC-obtained error performance for lower SNR values to higher SNR values [10], [11]. This can certainly reduce the MC simulation burden significantly. But the prerequisite is that one has reached the tail part of the error performance curve in MC simulation. Otherwise, the extrapolation would be based on an incorrect slope which may result in substantial over- or underestimation of the error probabilities for higher SNRs [12]. But for complicated coded transmission it can be difficult to ascertain whether one has reached the tail part of the performance curve. Hence one may be obliged to conduct simulation over the full range of SNR or PER of concern. The deficiency in MC efficiency thus strongly calls for a robust and efficient alternative for simulation of ultra-reliable communication systems.

One long-standing approach to fast simulation (versus conventional MC) is importance sampling (IS), for which various techniques have been developed [10]. The basic idea of IS is to increase the frequency of error-causing events by generating simulation samples with a biased probability distribution. Lu and Yao [13] consider uncoded transmission over noisy intersymbol-interference (ISI) channels and find that biasing the Gaussian noise by mean translation is increasingly more efficient than biasing by variance scaling as the SNR increases. Making use of the large deviations theory, Sadowsky and Bucklew [14] show that, for Gaussian noise, proper mean-translation biasing is asymptotically efficient at high SNR values. As a result, translation of a Gaussian noise's mean to the decision boundary becomes a standard practice in IS simulation [15].

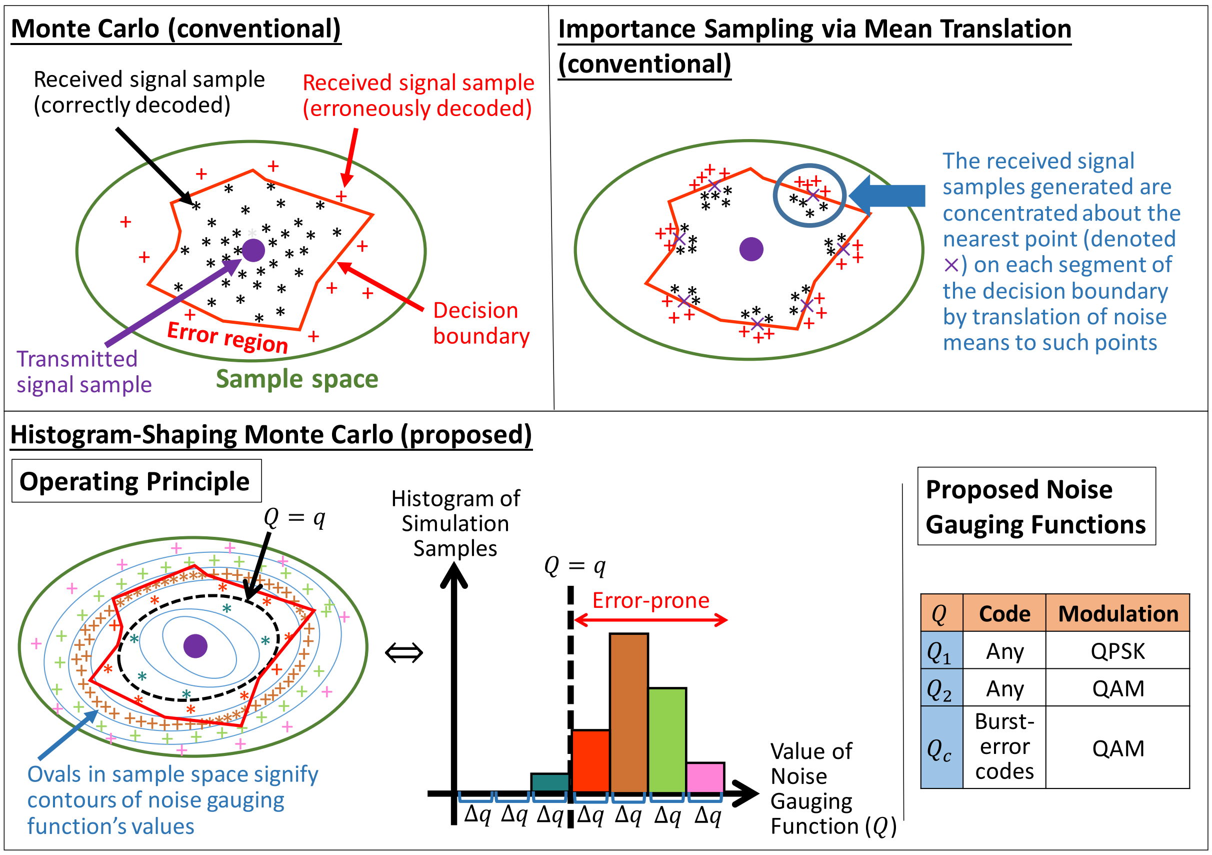
For coded transmission, take LDPC coding as an example. The decoding of such codes is known to be plagued by bit patterns called "trapping sets" which lead to error floors in high SNR. There exist efforts in using mean translation to simulate LDPC decoding in additive white Gaussian noise (AWGN) and Rayleigh fading [11], [16], [17], [18], [19], as mean translation on the trapping sets constitutes an efficient means to estimating the error floors. However, finding the full trapping set is an nondeterministic polynomial-time (NP)-hard problem [20] which is generally much more difficult than estimating the PER itself. Adaptive IS [21], [22] and universal simulation distributions [23], [24] can help the finding of proper mean translations, but they are effective only in low-dimensional code spaces.

Taking a different route, Minja and Šenk [25] propose a quasi-analytic simulation method for so-called "star domain" decoding. The method yields highly remarkable reduction in simulation runs but its application has been limited to several classical decoders for the binary symmetric channel or for binary signaling in AWGN.

The above simulation methods exploit known code weaknesses or the decoder property to concentrate simulation samples in the regions of the noise space that have higher probabilities of causing errors. The issue is that, in a high-dimensional space, the amount of effort required to determine such regions can loom large over the gain in simulation runs. One approach to mitigating this problem is to "map" the multi-dimensional simulation space into a single dimension judiciously and design proper noise sampling over properly organized subsets (or bins) of this one-dimensional (1D) space. In this regard, Holzlöhner et al. [26] define a scalar function of the noise vector whose values are positively correlated with the probability of decoding error and employ the multicanonical Monte Carlo (MMC) method to sample the noise to yield a flat histogram in the function's values. A subsequent study based on a similar flat-histogram approach also shows prominent efficiency advantage over MC [27].

However, a nonflat histogram may be more efficient for a given 1D mapping. For example, in simulating hard-decision decoding performance, Mahadevan and Morris [28] (naturally) use Hamming distance between bit patterns as the 1D mapping and generate more samples in the vicinity of half the minimum Hamming distance of the code. Indeed, Liang et al. [29] derive a general formula for the optimal histogram. Yet, there have been few studies that try to employ such an optimal histogram in simulating coded transmission. The reason is that it requires knowing the *a priori* unknown conditional error probabilities associated with the 1D mapped bins. In addition, for either a flat or nonflat histogram design, an issue is the efficient generation of properly distributed samples for each bin of the employed 1D mapping, as the distribution can be very complicated for a mapping of choice.

In previous works [30], [31], we considered simulating coded transmission performance employing a 1D mapping of the noise space termed the *noise gauging function* (NGF). More particularly, in [30] we developed a prototypical adaptive sampling method that can estimate the optimal (nonflat) histogram on the run, and in [31] we considered specifically the simulation of convolutionally coded systems and enhanced the method for their simulation efficiency. However, one limitation of these works is that they in effect only considered binary signaling, as is the case with all the reported studies on fast simulation techniques. (More exactly, the works considered quadriphase shift keying [QPSK], but that is mathematically equivalent to the direct sum of two biphase shift keying [BPSK] signals.) In the present paper, we revisit the design of the NGF as well as the adaptive sampling method under different situations. In particular, we consider quadrature amplitude modulation (QAM) in addition to QPSK, develop new NGFs along with revamped adaptive sampling mechanisms, and conduct a more in-depth analysis into the associated performance. To reiterate, our proposed technique is essentially code-agnostic in that it does not require detailed knowledge of the code weaknesses. However, some high-level understanding of the code properties can benefit the design of efficient NGFs. We will illustrate this point in several cases, one of which being


Monte Carlo (conventional)
Received signal sample (correctly decoded)
Received signal sample (erroneously decoded)
Error region
Decision boundary
Transmitted signal sample
Sample space
Importance Sampling via Mean Translation (conventional)
The received signal samples generated are concentrated about the nearest point (denoted ×) on each segment of the decision boundary by translation of noise means to such points
Histogram-Shaping Monte Carlo (proposed)
Operating Principle
Q = q
Ovals in sample space signify contours of noise gauging function's values
⇔
Histogram of Simulation Samples
Error-prone
Δq Δq Δq Δq Δq Δq Δq
Value of Noise Gauging Function (Q)
Proposed Noise Gauging Functions

Q	Code	Modulation
Q_1	Any	QPSK
Q_2	Any	QAM
Q_c	Burst-error codes	QAM

FIGURE 1. Key features of the proposed simulation technique (bottom) illustrated in comparison to that of the conventional MC and IS techniques (top). In all plots, green oval signifies the sample space of received signal samples corresponding to a certain transmitted signal sample (purple dot). Jagged red polygon signifies the corresponding decision boundary in decoding (which may or may not be maximum likelihood). Asterisks and crosses of various colors signify the received noisy signal samples generated in simulation, with their distributions illustrating typical sample distributions in the different simulation methods. Details of the proposed technique are explained in later text.

burst-error codes. Fig. 1 illustrates, conceptually, some key features of the conventional MC, conventional IS, and the proposed simulation techniques.

The remainder of this paper is organized as follows. Section II describes the system model and common PER measures. Section III introduces the proposed fast simulation approach. Section IV presents several practical NGFs and develops associated methods for noise sampling conditioned on given NGF values. Section V develops a method for adaptive shaping of the NGF histogram towards the optimal pattern. Section VI presents some numerical results. Section VII extends the proposed technique to simulation of burst-error coding. Finally, Section VIII concludes the paper.

Some notational conventions are as follows. P$(\cdot)$ denotes the probability of an event. Random variables are denoted using uppercase italic letters and random vectors uppercase boldface letters. Corresponding lowercase letters denote their sample values. Depending on whether X (resp. $\mathbf{X}$) is discrete or continuous, $f_X(\cdot)$ (resp. $f_{\mathbf{X}}(\cdot)$) denotes its probability mass or probability density function. E[ ] denotes expectation; any subscript to E denotes the random quantity over which the expectation is taken. $V(\cdot)$ denotes the variance of a random variable. Bernoulli(p) denotes Bernoulli distribution with success probability p, and $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes multivariate

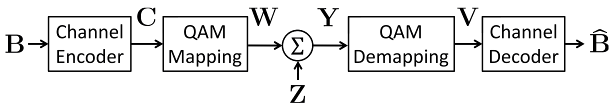


FIGURE 2. System model.

Gaussian probability distribution with mean vector $\boldsymbol{\mu}$ and autocovariance matrix $\boldsymbol{\Sigma}$.

II. SYSTEM MODEL

Fig. 2 shows the considered system model, where $\mathbf{B}$ is a vector of k information bits that observe independent and identical Bernoulli(0.5) distribution. Let n denote the codeword length. The channel encoder, of rate $r = k/n$, encodes $\mathbf{B}$ into a binary codeword vector $\mathbf{C} = [C_1, \ldots, C_n]$ and a modulator operates on $\mathbf{C}$ to yield a channel symbol vector $\mathbf{W} = [W_1, \ldots, W_l]$ where l depends on the modulation method. For convenience, we represent the in-phase and quadrature parts of a QAM (including QPSK) symbol separately in $\mathbf{W}$ so that $\mathbf{W}$ is a real vector and l is equal to two times the number of QAM symbols corresponding to $\mathbf{C}$. Each two successive elements of $\mathbf{W}$ make up a QAM symbol of the form $W_{2i-1} + jW_{2i}$, where $j = \sqrt{-1}$ and $1 \leq i \leq l/2$. We shall often refer to $\mathbf{W}$ as a *packet* in this work.

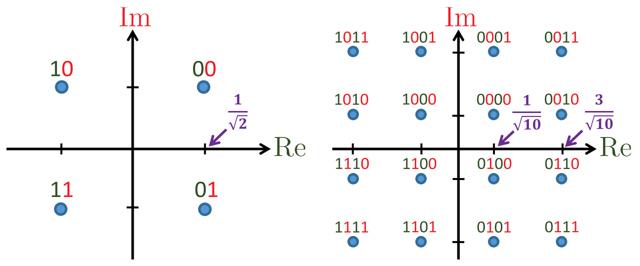


FIGURE 3. Gray-coded QPSK and 16QAM, where odd-indexed bits in a binary word determine the real part value of the modulated signal and even-indexed bits, imaginary part value (with bit indices in a word proceeding from left to right and starting at 1).

We let $\mathrm{E}[W_i^2] = E_s/2$ for $1 \le i \le l$ where recall that E[ ] denotes expectation, and E_s is the average QAM symbol energy. The packet $\mathbf{W}$ is sent over an AWGN channel with two-sided noise power spectral density $N_0/2$, which yields a received symbol vector $\mathbf{Y} = [Y_1, \ldots, Y_l] = \mathbf{W} + \mathbf{Z}$ where $\mathbf{Z} = [Z_1, \ldots, Z_l] \sim \mathcal{N}(\mathbf{0}, \frac{\sigma^2}{2}\mathbf{I})$, i.e., a white Gaussian noise vector with autocovariance $\frac{\sigma^2}{2}\mathbf{I}$ where $\sigma^2 = N_0$ and $\mathbf{I}$ denotes an identity matrix. The receiver demodulates $\mathbf{Y}$ into $\mathbf{V} = [V_1, \ldots, V_n]$ and decodes it into a binary data vector $\hat{\mathbf{B}}$.

A packet error occurs when any bit in $\hat{\mathbf{B}}$ is different from the corresponding bit in $\mathbf{B}$ so that $\hat{\mathbf{B}} \neq \mathbf{B}$. Under our system model, the PER is given by

$$\begin{aligned} p_e &= \mathrm{E}_{\mathbf{B}}\big[p_e(\mathbf{B})\big] = \mathrm{E}_{\mathbf{B}}[\mathrm{P}(\mathbf{Z} \in \mathcal{F}(\mathbf{B}))] \\ &= \mathrm{E}_{\mathbf{B}}\left[\int I_{\mathcal{F}}(\mathbf{B}, \mathbf{z}) f_{\mathbf{Z}}(\mathbf{z}) d\mathbf{z}\right] = \mathrm{E}_{\mathbf{B}}[\mathrm{E}_{\mathbf{Z}}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})]] \quad (1) \end{aligned}$$

where $p_e(\mathbf{b})$ denotes the PER of $\mathbf{b}$, $\mathcal{F}(\mathbf{b}) = \{\mathbf{z} | \hat{\mathbf{b}} \neq \mathbf{b}\}$ is the set of $\mathbf{z}$ that would cause a decoding error for $\mathbf{b}$, and $I_{\mathcal{F}}(\mathbf{b}, \mathbf{z})$ is an indicator function defined as $I_{\mathcal{F}}(\mathbf{b}, \mathbf{z}) = 1$ if $\mathbf{z} \in \mathcal{F}(\mathbf{b})$ and $I_{\mathcal{F}}(\mathbf{b}, \mathbf{z}) = 0$ otherwise. For linear codes with BPSK or Gray-coded QPSK modulation, $p_e(\mathbf{b})$ is equal $\forall \mathbf{b}$ so that p_e can be evaluated by evaluating $p_e(\mathbf{b})$ for any $\mathbf{b}$, such as $\mathbf{b} = \mathbf{0}$. For convenience, define $\mathbf{\Theta} \triangleq (\mathbf{B}, \mathbf{Z})$, and we shall refer to $\mathcal{F}(\mathbf{b})$ either as an error region or an error set.

For reliability reason, URLLC has favored use of low-order modulations. Fig. 3 shows the Gray-coded QPSK and 16QAM constellations [6]. For QPSK, $l = n$ and the bits are mapped to symbol values as

$$W_i = \frac{1}{\sqrt{2}}(1 - 2C_i), \quad i = 1, \ldots, l. \tag{2}$$

Except for the scaling factor of $1/\sqrt{2}$, a Gray-coded QPSK symbol is just the direct sum of two BPSK symbols. Hence, in AWGN, maximum likelihood (ML) decoding under coherent Gray-coded QPSK is mathematically no different from that under coherent BPSK. For 16QAM, $l = n/2$ and

$$W_i = \begin{cases} \frac{1}{\sqrt{10}}(1 - 2C_{2i-1})(1 + 2C_{2i+1}), & \text{if } i \text{ is odd}, \\ \frac{1}{\sqrt{10}}(1 - 2C_{2i-2})(1 + 2C_{2i}), & \text{otherwise}, \end{cases} \tag{3}$$

where $1 \le i \le l$.

For the time being, consider QPSK only. Let a_i be the number of codewords of Hamming weight i, where $0 \le i \le n$. It is well-known that, for coherent BPSK, a union bound on the PER under ML decoding and a high-SNR approximation are given by [32, Ch. 10]

$$p_e \le \sum_{i=1}^{n} a_i \mathcal{Q}\left(\sqrt{iE_s/N_0}\right) \approx a_{\min} \mathcal{Q}\left(\sqrt{d_{\min} E_s/N_0}\right) \tag{4}$$

where $d_{\min}$ is the minimum Hamming weight of the codewords (excluding 0), $a_{\min}$ is the number of minimum-weight codewords, and $\mathcal{Q}(\cdot)$ is the Gaussian Q function.

III. APPROACH TO FAST SIMULATION

While (4) provides a concise characterization of the PER under coherent Gray-coded QPSK (or BPSK), for an arbitrary code $d_{\min}$ and $a_{\min}$ are not always known, not to say the full weight distribution. Indeed, the evaluation of (1) can be highly more intractable for more complicated coding and modulation schemes. As a result, simulation invariably becomes the recourse for determining the PER performance. However, an efficient simulation setup can be obtained often only with substantial design effort. In this section, we introduce the proposed technique for fast simulation of digital communication systems. To put it in perspective, we first review some features of the MC simulation technique and the IS approach. IS reduces the required simulation time by biasing the distribution of the random variables (typically the noise vectors) in the system. Such biasing is conventionally effected by mean translation, that is, translating the mean of the noise to points which best discriminate between error-causing and non-error-causing noise vectors. However, the determination of proper mean translations can be difficult for complicated codes. We develop a method that divides the simulation space into a number of regions ("bins") employing a function that roughly measures the propensity to yield error from a noise vector (a point in the noise space). The method carries out MC simulation in each bin, but seeks to distribute the simulation samples among the bins in a way for best efficiency. The resulting technique is thus termed histogram-shaping Monte Carlo (HSMC).

A. PERFORMANCE OF THE MONTE CARLO TECHNIQUE

Consider (1). MC replaces the last expectation by the sample mean to obtain an estimate of p_e as

$$\hat{p}_{\mathrm{MC}} = \frac{1}{n_{\mathrm{MC}}} \sum_{j=1}^{n_{\mathrm{MC}}} I_{\mathcal{F}}\left(\boldsymbol{\theta}^{(j)}\right) = \frac{N_E}{n_{\mathrm{MC}}} \tag{5}$$

where n_{MC} is the number of simulation runs, N_E is the number of packet errors obtained in simulation, and $\boldsymbol{\theta}^{(j)}$ is the jth simulation sample of $\mathbf{\Theta}$. By the fact that N_E has the binomial distribution arising from n_{MC} Bernoulli(p_e) trials, the variance of $\hat{p}_{\mathrm{MC}}$ is given by

$$V(\hat{p}_{\mathrm{MC}}) = \frac{p_e(1 - p_e)}{n_{\mathrm{MC}}}. \tag{6}$$

Normalizing the standard deviation of the estimate by its mean provides a way to characterize the estimation quality known as the *relative precision*, given by

$$\epsilon_{\mathrm{MC}} = \frac{\sqrt{V(\hat{p}_{\mathrm{MC}})}}{p_{\mathrm{e}}} \approx \sqrt{\frac{1}{n_{\mathrm{MC}} p_{\mathrm{e}}}} \tag{7}$$

where the approximation holds for small p_{e}. To attain a relative precision $\epsilon_{\mathrm{MC}} \leq \epsilon$, therefore, MC requires $n_{\mathrm{MC}} \geq (\epsilon^2 p_{\mathrm{e}})^{-1}$ simulation runs. But since p_{e} is the *a priori* unknown parameter to be estimated, one cannot use it to fix n_{MC}. A common practice is to substitute $\hat{p}_{\mathrm{MC}}$ for p_{e} in (7) and obtain $n_{\mathrm{MC}}\hat{p}_{\mathrm{MC}} \approx \epsilon_{\mathrm{MC}}^{-2}$. Then, for example, for a 10% relative precision, one may stop after collecting 100 packet errors.

B. IMPORTANCE SAMPLING

For small PER values, MC becomes computationally prohibitive because, at given relative precision, n_{MC} is inversely proportional to p_{e}. IS seeks to reduce the number of simulation runs by drawing samples of $\mathbf{\Theta}$ from a biased probability density function (PDF) $f^*_{\mathbf{\Theta}}(\cdot)$, for each $\mathbf{b}$, more concentrated around $\mathcal{F}(\mathbf{b})$. The PER is then given by

$$\begin{aligned} p_{\mathrm{e}} = \mathrm{E}_{\mathbf{\Theta}}[I_{\mathcal{F}}(\mathbf{\Theta})] &= \int I_{\mathcal{F}}(\boldsymbol{\theta}_*)\frac{f_{\mathbf{\Theta}}(\boldsymbol{\theta}_*)}{f^*_{\mathbf{\Theta}}(\boldsymbol{\theta}_*)} f^*_{\mathbf{\Theta}}(\boldsymbol{\theta}_*) d\boldsymbol{\theta}_* \\ &= \mathrm{E}^*_{\mathbf{\Theta}}[I_{\mathcal{F}}(\mathbf{\Theta}_*)\mathcal{W}(\mathbf{\Theta}_*)] \end{aligned} \tag{8}$$

where $\mathrm{E}^*_{\mathbf{\Theta}}[\,]$ denotes the expectation with respect to $f^*_{\mathbf{\Theta}}(\cdot)$ and the ratio $\mathcal{W}(\cdot) \triangleq f_{\mathbf{\Theta}}(\cdot)/f^*_{\mathbf{\Theta}}(\cdot)$ is called the *weight function.*

Similar to MC, IS replaces the last expectation in (8) by the sample mean to yield an estimate of p_{e} as

$$\hat{p}_{\mathrm{IS}} = \frac{1}{n_{\mathrm{IS}}} \sum_{j=1}^{n_{\mathrm{IS}}} I_{\mathcal{F}}\left(\boldsymbol{\theta}_*^{(j)}\right) \mathcal{W}\left(\boldsymbol{\theta}_*^{(j)}\right) \tag{9}$$

where $\boldsymbol{\theta}_*^{(j)}$ is the jth simulation sample of $\mathbf{\Theta}$ generated using the biased PDF. The variance of $\hat{p}_{\mathrm{IS}}$ is given by

$$V\left(\hat{p}_{\mathrm{IS}}\right) = \frac{\mathrm{E}^*_{\mathbf{\Theta}}\left[\left(I_{\mathcal{F}}(\mathbf{\Theta}_*)\mathcal{W}(\mathbf{\Theta}_*) - p_{\mathrm{e}}\right)^2\right]}{n_{\mathrm{IS}}}. \tag{10}$$

The well-known "unconstrained optimal" biased probability distribution is given by [33]

$$f^{*,\mathrm{opt}}_{\mathbf{\Theta}}(\cdot) = I_{\mathcal{F}}(\cdot) f_{\mathbf{\Theta}}(\cdot)/p_{\mathrm{e}} \tag{11}$$

with associated $\mathcal{W}(\boldsymbol{\theta}) = p_{\mathrm{e}} I_{\mathcal{F}}(\boldsymbol{\theta})\ \forall \boldsymbol{\theta}$. The optimal biased PDF requires knowing p_{e} and is hence unworkable. Nevertheless, it has been considered useful in suggesting some IS design heuristics [15]. In particular, it has been used to motivate mean translation as has the large deviations theory [14].

To illustrate the mean translation technique and its issues, consider linear codes with Gray-coded QPSK modulation. Without loss of generality, consider transmitting the all-zero codeword so that $\mathbf{\Theta} = (\mathbf{0}, \mathbf{Z})$. By (2), $w_i = 1/\sqrt{2}\ \forall i$. Consider ML decoding. As indicated in (4), minimum-weight codewords dictate the PER performance in high SNR so that

$$p_{\mathrm{e}} \approx \sum_{a=1}^{a_{\min}} \int_{\mathbf{z}\in\mathcal{F}_a} f_{\mathbf{Z}}(\mathbf{z}) d\mathbf{z} = a_{\min} p_{\min} \tag{12}$$

where a is the index of the minimum-weight codewords, $\mathcal{F}_a$ is the set of noise vectors that would cause a pairwise decision error to codeword a, and $p_{\min} = \mathcal{Q}(\sqrt{d_{\min} E_s/N_0})$ is the corresponding pairwise error probability (PEP). Let $\Pi_a(1), \ldots, \Pi_a(d_{\min})$ be the indices of the 1-bits in codeword a. Ideal mean translation captures all the pairwise error conditions with a Gaussian mixture [14]

$$f^*_{\mathbf{Z}}(\mathbf{z}_*) = \frac{1}{a_{\min}} \sum_{a=1}^{a_{\min}} f_{\mathbf{Z}+\boldsymbol{\mu}_a}(\mathbf{z}_*) \tag{13}$$

where $\mathbf{Z} + \boldsymbol{\mu}_a$ ($\triangleq \mathbf{Z}_*$) denotes a translation of $\mathbf{Z}$ by $\boldsymbol{\mu}_a$, that is, $\mathbf{Z} + \boldsymbol{\mu}_a \sim \mathcal{N}(\boldsymbol{\mu}_a, \frac{\sigma^2}{2}\mathbf{I})$, with $\boldsymbol{\mu}_a$ being a vector whose elements are all zero except for those having indices $\Pi_a(1), \ldots, \Pi_a(d_{\min})$, which have value $-1/\sqrt{2}$. In other words, $\mathbf{Z} + \boldsymbol{\mu}_a$ is centered at the nearest decision boundary associated with pairwise decision error to codeword a. (Incidentally, the simulation for the aforementioned effects of the trapping sets in LDPC codes can be done similarly [11], [16], [17], [18], [19], [20].)

For a first-order understanding of the efficiency of mean translation, consider the hit rate of $\mathbf{Z}_* \in \mathcal{F}_a$ and the expected value of the weight function in $\mathcal{F}_a$. The probability of $\mathbf{Z}_* \in \mathcal{F}_a$ can be obtained as

$$\begin{aligned} \mathrm{P}(\mathbf{Z}_* \in \mathcal{F}_a) &= \frac{1}{a_{\min}} \sum_{a'=1}^{a_{\min}} \mathrm{P}\big(\mathbf{Z} + \boldsymbol{\mu}_{a'} \in \mathcal{F}_a\big) \\ &\approx \frac{1}{a_{\min}} \mathrm{P}\big(\mathbf{Z} + \boldsymbol{\mu}_a \in \mathcal{F}_a\big) = \frac{1}{2a_{\min}} \quad \forall a, \end{aligned} \tag{14}$$

where the approximation is due to that $\mathrm{P}(\mathbf{Z} + \boldsymbol{\mu}_a \in \mathcal{F}_a) = 0.5 \gg \mathrm{P}(\mathbf{Z} + \boldsymbol{\mu}_{a'} \in \mathcal{F}_a)\ \forall a' \neq a$. The expected value of the weight function conditioned on $\mathbf{Z} \in \mathcal{F}_a$ is then given by

$$\mathrm{E}_{\mathbf{Z}_*\in\mathcal{F}_a}[\mathcal{W}(\mathbf{Z}_*)] \approx p_{\min} \Big/ \frac{1}{2a_{\min}} = 2a_{\min} p_{\min} = 2p_{\mathrm{e}} \tag{15}$$

$\forall a$. Thus, with such mean translation the overall hit rate $\mathrm{P}(\mathbf{Z}_* \in \mathcal{F}) = \sum_a \mathrm{P}(\mathbf{Z}_* \in \mathcal{F}_a)$ is raised to 0.5 and the expected value of the weight function is of a similar order as p_{e}. This comes close to the "unconstrained optimal" solution [15] and is optimal in a large deviations theory sense [14].

Its fundamental appeal notwithstanding, an application of the technique to complicated systems requires the appreciation of two issues as follows.

1) UNDERESTIMATION OF PER DUE TO INCOMPLETE KNOWLEDGE OF ERROR STRUCTURES

As already indicated, the "error-prone structures" of a code are not always known, such as the set of minimum-weight codewords of a linear code or the trapping sets of an LDPC code. One could do a computer search for these structures, but for complicated codes this is not necessarily practical, as it can be an NP-complete problem [20], [34]. Hence one may need to settle for a simulation based on known structures only. This could result in underestimation of the PER with mean-translating IS. We explain the mechanism in Appendix A.

2) EXTENSION TO HIGHER-ORDER QAM

Mean translation works only when the nearest points on the error region's boundary are well-characterized [14]. This is the case with Gray-coded QPSK under a channel code with known distance structures. For higher-order QAM, however, the (nonlinear) mapping between code bits and constellation points can drastically complicate the structure of the error region associated with a codeword. For example, the 16QAM plot in Fig. 3 shows that one noise element (say, Z_i) impacts the reliabilities of multiple code bits (two in this case). Moreover, an inner constellation point has more nearest neighbors than an outer point and is thus more susceptible to error than the latter. Further, one cannot only simulate the all-zero codeword because, unlike Gray-coded QPSK, code linearity is not preserved in the QAM domain.

C. HISTOGRAM-SHAPING MONTE CARLO

In view of the above issues of mean translation, we look for a different IS method. Existing theory [14], [15] indicates that an efficient method should seek to identify the error set $\mathcal{F}(\mathbf{B})$ and sample it frequently enough.

To identify the error set, we consider defining a real-valued function (or random variable) $Q(\mathbf{B}, \mathbf{Z})$ whose values are correlated with the PER. In other words, the function measures, in some way, the likelihood that a certain noise vector $\mathbf{z}$ will cause a decoding error to a certain information vector $\mathbf{b}$ and in this way provides a "soft identification" of the error set. (An unrealistic ideal is to have a binary-valued $Q(\mathbf{B}, \mathbf{Z})$ that unambiguously indicates whether any given $\mathbf{z}$ will result in a decoding error for any given $\mathbf{b}$ and thereby provides a "hard identification" of the error set.) From elementary probability theory, the marginal PDF of $Q \triangleq Q(\mathbf{B}, \mathbf{Z})$ is given by

$$f_Q(q) = \sum_{\mathbf{b}} f_{\mathbf{B}}(\mathbf{b}) \left[\int_{\{\mathbf{z}:Q(\mathbf{b},\mathbf{z})=q\}} f_{\mathbf{Z}}(\mathbf{z}) d\mathbf{z} \right] \tag{16}$$

and, by independence between $\mathbf{Z}$ and $\mathbf{B}$, the conditional PDF of $\mathbf{Z}$ on $\mathbf{B}$ is given by

$$f_{\mathbf{Z}}(\mathbf{z}) = f_{\mathbf{Z}|\mathbf{B}}(\mathbf{z}|\mathbf{b}) = \int_{-\infty}^{\infty} f_{Q,\mathbf{Z}|\mathbf{B}}(q, \mathbf{z}|\mathbf{b}) dq. \tag{17}$$

We may therefore rewrite p_e as (cf. (1))

$$\begin{aligned} p_e &= \mathrm{E}_{Q,\mathbf{B},\mathbf{Z}}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})] = \mathrm{E}_Q \left[\underbrace{\mathrm{E}_{\mathbf{B},\mathbf{Z}|Q}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})]}_{\triangleq p_e(Q)} \right] \\ &= \int_q p_e(q) f_Q(q) dq \end{aligned} \tag{18}$$

where $p_e(q)$ is the PER contributed by all $\mathbf{b}$ and $\mathbf{z}$ for which $Q(\mathbf{b}, \mathbf{z}) = q$.

Fig. 4 illustrates typical shapes of $f_Q(q)$, $p_e(q)$, and $f_Q(q)p_e(q)$ that we can normally attain. (Examples will be given later.) Practicality limits us to consider only a finite range of q such as $[q_{\min}, q_{\max}] \triangleq \Omega$ as shown in the figure,

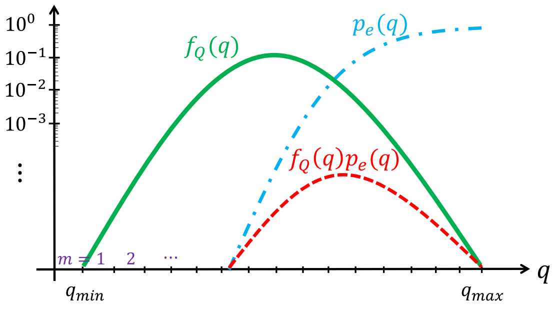


FIGURE 4. Typical shapes of $f_Q(q)$, $p_e(q)$ and $f_Q(q)p_e(q)$.

where Ω should be large enough so that the approximation $p_e \approx \int_{q_{\min}}^{q_{\max}} p_e(q) f_Q(q) dq$ holds. Also for practicality, we divide Ω into a number of bins (intervals) as illustrated.

Having obtained a "soft identifier" of the error set, we now consider the simulation method. The conventional MC is such that the simulation samples would observe the distribution $f_Q(q)$, while only a limited portion of the q values contribute significantly to the PER. Efficient IS should bias the sampling of $\mathbf{Z}$ such that the corresponding q values fall more in the range where $f_Q(q)p_e(q)$ has more substantial values. For this, let Ω be divided into n_b bins denoted Ω_m, $m = 1, 2, \ldots, n_b$, respectively, and denote the number of simulation samples in the mth bin by n_m. We thus have $n_{\mathrm{IS}} = \sum_{m=1}^{n_b} n_m$ and the set $\{n_m, m = 1, \ldots, n_b\}$ constitutes a histogram of the q values. The corresponding biased (sampling) PDF is given by

$$f_Q^*(q) = \sum_{m=1}^{n_b} \frac{n_m}{n_{\mathrm{IS}}} f_{Q|\Omega_m}(q) = \sum_{m=1}^{n_b} \frac{n_m}{n_{\mathrm{IS}}} \frac{I_{\Omega_m}(q) f_Q(q)}{p_m} \tag{19}$$

where $I_{\Omega_m}(q)$ is the indicator function for Ω_m and $p_m \triangleq \mathrm{P}(Q \in \Omega_m)$. Contextualizing (8)–(9) to this setting, the corresponding PER estimator is given by

$$\begin{aligned} \hat{p}_e &= \frac{1}{n_{\mathrm{IS}}} \sum_{j=1}^{n_{\mathrm{IS}}} I_{\mathcal{F}}\left(\mathbf{b}_*^{(j)}, \mathbf{z}_*^{(j)}\right) \frac{f_Q\left(q_*^{(j)}\right)}{f_Q^*\left(q_*^{(j)}\right)} \\ &= \sum_{j=1}^{n_{\mathrm{IS}}} I_{\mathcal{F}}\left(\mathbf{b}_*^{(j)}, \mathbf{z}_*^{(j)}\right) \left[\sum_{m=1}^{n_b} \frac{p_m}{n_m} I_{\Omega_m}\left(q_*^{(j)}\right) \right] \\ &= \sum_{m=1}^{n_b} p_m \frac{E_m}{n_m} \end{aligned} \tag{20}$$

where $q_*^{(j)}$ is the jth sample from $f_Q^*(q)$, $(\mathbf{b}_*^{(j)}, \mathbf{z}_*^{(j)})$ is the associated sample of $(\mathbf{B}, \mathbf{Z})$ conditioned on $Q(\mathbf{B}, \mathbf{Z}) = q_*^{(j)}$, and $E_m \triangleq \sum_{j=1}^{n_{\mathrm{IS}}} I_{\mathcal{F}}(\mathbf{b}_*^{(j)}, \mathbf{z}_*^{(j)}) I_{\Omega_m}(q_*^{(j)})$ $\forall m$ constitute the histogram of decoding errors. Note that the ratio E_m/n_m is precisely the MC estimate of the conditional PER $p_{\mathrm{err}|m}$ in Ω_m from n_m simulation samples, where

$$\begin{aligned} p_{\mathrm{err}|m} &= \mathrm{E}_{Q|\Omega_m}\left[\mathrm{E}_{\mathbf{B},\mathbf{Z}|Q}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})]\right] \\ &= \int_{q \in \Omega_m} p_e(q) f_{Q|\Omega_m}(q) dq. \end{aligned} \tag{21}$$

By the fact that MC estimations are unbiased, $\hat{p}_{\mathrm{err}|m} \triangleq E_m/n_m$ provides an unbiased estimate of $p_{\mathrm{err}|m}$, and hence $\hat{p}_{\mathrm{e}}$ from (20) provides an unbiased estimate of p_{e}. We refer to the simulation technique summarized in (20) as HSMC, which performs MC estimation of the conditional PER in each bin and averages over the bin-wise estimates to obtain the overall PER estimate. The technique is denoted "histogram-shaping" because a key work in its proceeding is to control the histogram $\{n_m\}$, as discussed below.

In HSMC, the set $\{p_m\}$ constitutes the weight function. In contrast to conventional IS, the function is deterministic rather than random and, depending on the design, can be evaluated beforehand so that it does not have to be evaluated per sample. A known weight function also facilitates a simple closed-form characterization of the estimator variance. In particular, since $\hat{p}_{\mathrm{e}}$ is a weighted sum of MC estimates, its variance is given by a square-weighted sum of MC variances (see (6)) as

$$V(\hat{p}_{\mathrm{e}}) = \sum_{m=1}^{n_b} p_m^2 \frac{p_{\mathrm{err}|m}\left(1 - p_{\mathrm{err}|m}\right)}{n_m}. \tag{22}$$

Minimization of the variance therefore depends on optimization of the histogram $\{n_m\}$ subject to a constraint on the total simulation samples n_{IS}. A routine application of the Lagrange multiplier method yields

$$\frac{n_m}{n_{\mathrm{IS}}} = \frac{p_m\sqrt{p_{\mathrm{err}|m}\left(1 - p_{\mathrm{err}|m}\right)}}{\sum_{m=1}^{n_b} p_m\sqrt{p_{\mathrm{err}|m}\left(1 - p_{\mathrm{err}|m}\right)}} \triangleq \rho_m \quad \forall m. \tag{23}$$

Substituting it into (22) gives the minimum variance as

$$V_Q^*(\hat{p}_{\mathrm{e}}) = \frac{1}{n_{\mathrm{IS}}}\left(\sum_{m=1}^{n_b} p_m\sqrt{p_{\mathrm{err}|m}\left(1 - p_{\mathrm{err}|m}\right)}\right)^2 \tag{24}$$

where subscript Q indicates that the minimum depends on the distributions of p_m and $p_{\mathrm{err}|m}$ which in turn depend on how $Q(\mathbf{B}, \mathbf{Z})$ is defined. A useful measure of the efficiency of HSMC versus MC is provided by the ratio $n_{\mathrm{MC}}/n_{\mathrm{IS}}$ at $V(\hat{p}_{\mathrm{MC}}) = V_Q^*(\hat{p}_{\mathrm{e}})$, which may be termed the *ideal speedup factor* and is given by

$$G^* = \frac{p_e(1 - p_e)}{\left(\sum_{m=1}^{n_b} p_m\sqrt{p_{\mathrm{err}|m}\left(1 - p_{\mathrm{err}|m}\right)}\right)^2}. \tag{25}$$

Since $p_{\mathrm{err}|m}$ $\forall m$ can only be estimated after simulation, (23) cannot be used to set the histogram beforehand. But we can use it in an adaptive manner to update the histogram as simulation progresses and some estimates of $p_{\mathrm{err}|m}$ are available. The details are discussed in a subsequent section. In Appendix B, we show (essentially) that $V_Q^*(\hat{p}_{\mathrm{e}})$ is upper bounded by $V(\hat{p}_{\mathrm{MC}})$ and that the upper bound is attained when $p_{\mathrm{err}|m}$ is constant $\forall m$. In other words, any function $Q(\mathbf{B}, \mathbf{Z})$ that yields a nonconstant $p_{\mathrm{err}|m}$ over m would be more efficient than MC under the optimal histogram. Indeed, from (24), zero variance can be achieved by having $p_{\mathrm{err}|m} = 0$ or 1 $\forall m$ and, from (21), this is the case when

$$Q(\mathbf{b}, \mathbf{z}) = I_{\mathcal{F}}(\mathbf{b}, \mathbf{z}) = \begin{cases} 1, & \text{if } \mathbf{z} \in \mathcal{F}(\mathbf{b}), \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

Unsurprisingly, this is in general as unrealizable as the "unconstrained optimal" biased distribution of (11), despite any difference in the underlying mechanism. But it suggests that we find a function that marks well the propensity to cause error by any noise vector $\mathbf{z}$ to any information vector $\mathbf{b}$. We have termed such a function a *noise gauging function* (NGF) in previous work [31].

It can be expected that, for complicated transmission systems, a close-to-ideal NGF may be hard to come by and, if obtainable, the computational complexity may be very high. In particular, the corresponding evaluation of p_m and sampling of Ω_m for any m may be highly arduous work. In this situation, it should be sensible to employ an NGF that may be less ideal but is easy to work with, such as one with a closed-form expression for the PDF of Q and facilitating the sampler design. With a view on the trade-off between efficiency and complexity, we develop several practical NGFs in the next section. The following section develops the proposed method for adaptive shaping of the sample histogram towards its optimal form under a chosen NGF.

IV. PRACTICAL NOISE GAUGING

In this section, we present several practical NGFs for which the PDF of Q can be readily obtained and the corresponding sampling of Ω_m for given m is also relatively straightforward.

To start, we define $\Omega = [q_{\min}, q_{\max}]$ as such that $\mathrm{P}(Q < q_{\min}) = \mathrm{P}(Q > q_{\max}) \approx p_{\mathrm{out}}$ where $p_{\mathrm{out}} = 0.01 p_{\mathrm{e}}$ so as to make Ω cover at least 98% of the total probability of $f_Q(q)p_{\mathrm{e}}(q)$. For example, to simulate ultra-reliable communication we may let $p_{\mathrm{out}} = 10^{-11}$. In absence of knowledge of $p_{\mathrm{e}}(q)$, we divide Ω uniformly into n_b bins with bin width $\Delta q = (q_{\max} - q_{\min})/n_b$, so that $\Omega_1 = [q_{\min}, q_{\min} + \Delta q]$ and $\Omega_m = (q_{\min} + (m-1)\Delta q, q_{\min} + m\Delta q]$ for $2 \le m \le n_b$. For convenience, let $q_m = q_{\min} + (m - 0.5)\Delta q$ $\forall m$. In this manner, p_m can be calculated from the PDF of Q and only $p_{\mathrm{err}|m}$ needs to be evaluated by simulation for any m.

To facilitate sampling of noise vectors conditioned on some value of Q, say $Q = q \in \Omega_m$, we note from [35] that the sampler is particularly easy to implement when $Q(\mathbf{B}, \mathbf{Z})$ is a sufficient statistic for some parameter of the PDF of $\mathbf{Z}$. In particular, the PDF of a Gaussian random vector is characterized by its mean and autocovariance, which have well-known sufficient statistics [36]. In general, the sampling of a random vector conditioned on a certain sufficient statistic having a certain value can be effected by unconditioned sampling of the random vector followed by adjusting of the sample values to satisfy the given condition on the sufficient statistic [35]. Thus in the case of a white Gaussian random $\mathbf{Z}$, its sampling conditioned on a sufficient statistic for mean or variance having a certain value can be effected by unconditioned sampling followed by shifting or scaling of the sample values properly to satisfy the given condition.

In the above regard, in this section we develop two noise gauging methods designated $Q_1(\cdot)$ and $Q_2(\cdot)$ that are closely associated with the sufficient statistics for mean and variance of $\mathbf{Z}$, respectively. Both are code-agnostic in the sense that they assume no knowledge of the error-prone structures of a code. Between them, $Q_1(\cdot)$ is linear in $\mathbf{Z}$. Part of its attractiveness is its amenability to theoretical performance analysis, and it can be shown to be asymptotically (as $n \to \infty$) optimal under random linear coding. But it has some limitations, which prompts our designing the NGF $Q_2(\cdot)$ that takes into account only the noise elements that are conducive to decoding error.

A. CODE-AGNOSTIC LINEAR NOISE GAUGING

Consider QPSK. From (2), we see that a Z_i having a different sign than W_i should be more evocative of decoding error than one having a similar sign. Thus let $\tilde{Z}_i \triangleq -\sqrt{2}W_iZ_i$. For a linear code, it is understood that, in high SNR, the maximum of $\sum_{i=\Pi_a(1)}^{\Pi_a(d_{\min})} \tilde{Z}_i$ over all minimum-weight codewords a well correlates with the ML soft-decision decoding error probability. But using it for noise gauging implies a code-specific design for which some issues have been discussed previously. An appealing *code-agnostic* NGF appears to be

$$Q_1(\mathbf{b}, \mathbf{z}) = -\frac{1}{l}\sum_{i=1}^{l}\sqrt{2}\,w_i z_i = \frac{1}{l}\sum_{i=1}^{l}\tilde{z}_i \tag{27}$$

(where recall that w_i and $\tilde{z}_i$ denote sample values of W_i and $\tilde{Z}_i$, respectively). By the assumption that $\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, \frac{\sigma^2}{2}\mathbf{I})$, we also have $\tilde{\mathbf{Z}} \triangleq [\tilde{Z}_1, \ldots, \tilde{Z}_l] \sim \mathcal{N}(\mathbf{0}, \frac{\sigma^2}{2}\mathbf{I})$, and thus the random variable $Q_1 \triangleq Q_1(\mathbf{B}, \mathbf{Z})$ has PDF $\mathcal{N}(0, \frac{\sigma^2}{2l})$ and p_m can be readily evaluated.

After evaluating p_m, HSMC requires the evaluation of $p_{\mathrm{err}|m}$ $\forall m$. For linear coding with QPSK modulation, by symmetry of the signal structure we only need to simulate the transmission of $\mathbf{b} = \mathbf{0}$ to determine the complete system performance. But for generality, in the below we also consider sampling over $\mathbf{B}$. Then by the law of total expectation, we can rewrite (21) as

$$\begin{aligned} p_{\mathrm{err}|m} &= \mathrm{E}_{\mathbf{B}}\mathrm{E}_{Q_1|\Omega_m}\mathrm{E}_{\mathbf{Z}|Q_1,\Omega_m}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})] \\ &= \mathrm{E}_{\mathbf{B}}\mathrm{E}_{Q_1|\Omega_m}\mathrm{E}_{\mathbf{Z}|Q_1}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})] \end{aligned} \tag{28}$$

where $\mathrm{E}_{Q_1|\Omega_m}$ is expectation with respect to the PDF

$$f_{Q_1|\Omega_m}(q) = \frac{1}{\sqrt{\pi\sigma^2/l}}\exp\left(-\frac{q^2}{\sigma^2/l}\right)\Big/ p_m, \quad q \in \Omega_m, \tag{29}$$

and the last equality in (28) holds because the second expectation has made $Q_1 \in \Omega_m$ and hence the conditioning of the innermost expectation on Ω_m becomes redundant.

According to the sequence of conditioning set forth in (28), the generation of each simulation sample (indexed j) progresses in the order $\mathbf{B} \to Q_1 \to \mathbf{Z}$. To start, we draw an information vector sample $\mathbf{b}^{(j)}$ and encode it into $\mathbf{c}^{(j)}$ and obtain $\mathbf{w}^{(j)} = [w_1^{(j)}, \ldots, w_l^{(j)}]$. Then we employ the acceptance/rejection (A/R) method [10, Sec. 7.2.2.4] to draw $q_1^{(j)}$ according to $f_{Q_1|\Omega_m}(\cdot)$. Finally, to generate the noise vector sample $\mathbf{z}^{(j)} = [z_1^{(j)}, \ldots, z_l^{(j)}]$ conditioned on $Q_1 = q_1^{(j)}$, we first draw a Gaussian vector sample $\bar{\mathbf{z}}^{(j)}$ according to $\mathcal{N}(\mathbf{0}, \frac{\sigma^2}{2}\mathbf{I})$ and add an equal bias to each element of $\bar{\mathbf{z}}^{(j)}$ to make their sum equal to $lq_1^{(j)}$. This, according to [35], yields a proper sample of $\mathbf{Z}|(Q_1 = q_1^{(j)})$, which we denote $\tilde{\mathbf{z}}^{(j)}$. Simple algebra shows that its elements are given by $\tilde{z}_i^{(j)} = \bar{z}_i^{(j)} + \hat{\mu}$ $\forall i$ where $\hat{\mu} = q_1^{(j)} - \frac{1}{l}\sum_i \bar{z}_i^{(j)}$. The desired noise vector sample $\mathbf{z}^{(j)}$ is then obtained by letting $z_i^{(j)} = \tilde{z}_i^{(j)}/(-\sqrt{2}w_i^{(j)})$ $\forall i$. Algorithm shows a summary of the sampling method.

Algorithm 1 Sample Generation Method Under Proposed Linear Noise Gauging

```
1: $\mathbf{b}^{(j)} \leftarrow f_{\mathbf{B}}(\mathbf{b})$
2: Encode $\mathbf{b}^{(j)}$ into $\mathbf{c}^{(j)}$; modulate $\mathbf{c}^{(j)}$ into $\mathbf{w}^{(j)}$
3: $q_1^{(j)} \leftarrow f_{Q_1|\Omega_m}(q)$ (via A/R with (29))
4: $\bar{\mathbf{z}}^{(j)} \leftarrow \mathcal{N}(\mathbf{0}, \frac{\sigma^2}{2}\mathbf{I})$
5: Set $\hat{\mu} = q_1^{(j)} - \frac{1}{l}\sum_i \bar{z}_i^{(j)}$
6: Set $z_i^{(j)} = (\bar{z}_i^{(j)} + \hat{\mu})/(-\sqrt{2}w_i^{(j)})$ $\forall i$
7: Return $\mathbf{b}^{(j)}$, $\mathbf{c}^{(j)}$, $\mathbf{w}^{(j)}$, $\mathbf{z}^{(j)} = [z_1^{(j)}, \ldots, z_l^{(j)}]$
```

It is of interest to understand how $Q_1(\cdot)$ performs statistically over different codes. Thus consider random linear (n, k) coding, for which the probability of any binary n-vector being a codeword is 2^{k-n} and thus the average number of codewords of weight i is equal to $\binom{n}{i}2^{k-n}$. Then for PER at $Q_1 = q$, we have an average union bound as

$$p_{\mathrm{e}}(q) \le \sum_{i=1}^{n'}\binom{n}{i}2^{k-n}p_{\mathrm{e}}(i, q) \tag{30}$$

where $p_{\mathrm{e}}(i, q)$ is the PEP of decoding to a particular weight-i codeword and $n' < n$ may be chosen to yield a suitably tight bound. As $\sum_{i'=1}^{i} \tilde{Z}_{i'}|(Q_1 = q) \sim \mathcal{N}(iq, i\frac{n-i}{n}\frac{\sigma^2}{2})$, we get

$$\begin{aligned} p_{\mathrm{e}}(i, q) &= \mathrm{P}\left(\sum_{i'=1}^{i}\tilde{Z}_{i'} \ge i\sqrt{\frac{E_s}{2}}\,\middle|\, Q_1 = q\right) \\ &= \mathcal{Q}\left(\frac{\sqrt{E_s} - \sqrt{2}\,q}{\sqrt{(i^{-1} - n^{-1})N_0}}\right). \end{aligned} \tag{31}$$

The conditional PER $p_{\mathrm{err}|m}$ can be evaluated by evaluating the mean of $p_{\mathrm{e}}(Q_1)$ over $Q_1 \in \Omega_m$.

Fig. 5(a) shows the ideal speedup factors for some code lengths under different bin counts. It is not surprising that a finer division of the noise space tends to enable a better optimization of the histogram, and $V_Q^*(\hat{p}_{\mathrm{e}})$ should converge to $(\int \sqrt{p_{\mathrm{e}}(q)(1 - p_{\mathrm{e}}(q))}f_{Q_1}(q)dq)^2/n_{\mathrm{IS}}$ when the bin count goes to infinity. The fluctuations in speedup factor at low n_b values, which are particularly prominent for $n \ge 512$, come about because for some values of n_b, a bin edge

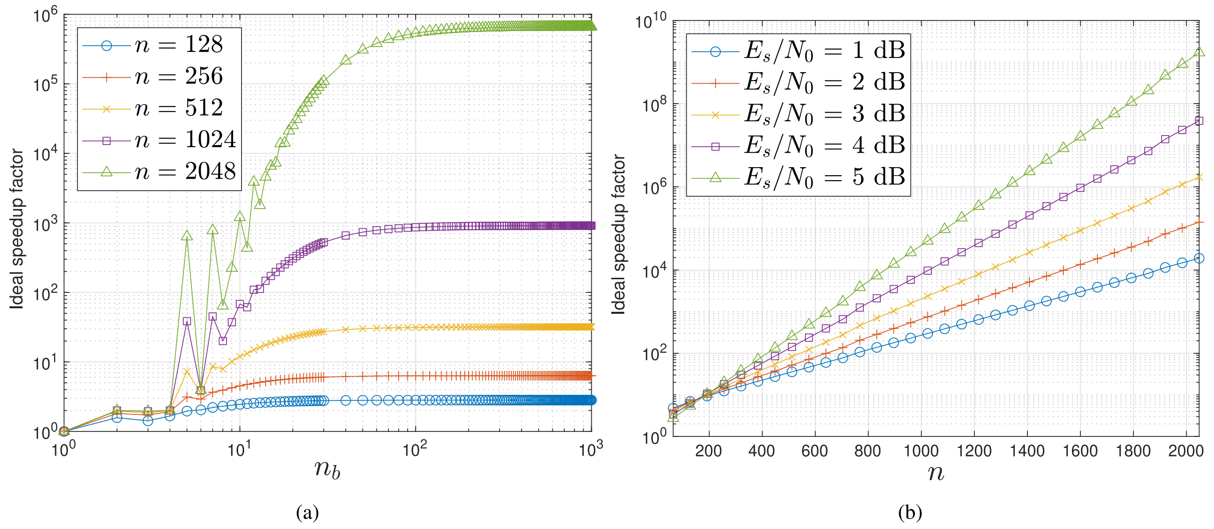


FIGURE 5. Ideal speedup factor of code-agnostic noise gauging method $Q_1(\cdot)$ under random linear coding at rate 1/2. (a) Speedup vs. number of bins (E_s/N_0 = 3 dB). (b) Speedup vs. code length (n_b = 1000).

fortuitously falls on a value of q which dichotomizes Ω into an error-causing subset (where $f_Q(q)p_e(q) > 0$) and an almost non-error-causing subset (where $f_Q(q)p_e(q) \approx 0$). (Recall the illustration in Fig. 4.) Consequently, they result in a higher speedup than their immediate neighboring values which make less clear-cut division between the two subsets. And the effect is especially prominent when n_b is small because the binning of Ω is coarse.

From Fig. 5(a), a reasonable rule of thumb seems to be to let $n_b = 20$ if the ideal speedup factor < 1000 and let $n_b = 100$ otherwise. Fig. 5(b) shows the ideal speedup factors at different SNR values. It indicates that HSMC with Q_1 should be increasingly efficient as the code length and SNR increase. As an example, it can be 10^9 times faster than MC with $n = 2048$ at $E_s/N_0 = 5$ dB.

It is known from information theory that, as $n \to \infty$, typical random linear codes have a minimum distance that grows linearly with n in the sense that $H(d_{\min}/n) = 1 - r$ where $H(p) = p\log_2 p + (1-p)\log_2(1-p)$ is the binary entropy function [37]. Let $\alpha \triangleq n/d_{\min}$. Then with $i = d_{\min}$ in (31) we have

$$p_e(d_{\min}, q) = \mathcal{Q}\left(\frac{\sqrt{E_s/2} - q}{\sqrt{(\alpha-1)\frac{\sigma^2}{2n}}}\right). \tag{32}$$

When $n \to \infty$ or $\sigma^2 \to 0$ (SNR $\to \infty$), we have $\sigma^2/(2n) \to 0$ and $p_e(d_{\min}, q)$ approaches a unit step function in q with the step located at $q = \sqrt{E_s/2}$. Hence the set $\{\mathbf{z}|Q_1(\mathbf{b},\mathbf{z}) \ge \sqrt{E_s/2}\}$ is precisely the error region $\mathcal{F}(\mathbf{b})$ pertaining to some minimum-distance codeword corresponding to a given $\mathbf{b}$, and $Q_1(\cdot)$ becomes an optimal NGF in the sense of (26).

However, the above results for random coding may not hold for common practical codes, because a practical linear code may have $d_{\min} \ll n$ as $n \to \infty$ (e.g., common convolutional codes). From (23), the optimal histogram $\rho_m \propto p_m\sqrt{p_{\mathrm{err}|m}(1-p_{\mathrm{err}|m})}$ where $p_m \to f_{Q_1}(q)$ as $n_b \to \infty$. But since $Q_1 \sim \mathcal{N}(0, \frac{\sigma^2}{2l})$ and $\sqrt{p_{\mathrm{err}|m}(1-p_{\mathrm{err}|m})} \le 0.5$ for $0 \le p_{\mathrm{err}|m} \le 1$, the optimal histogram for q becomes concentrated at 0 as $n \to \infty$, resulting in $p_e(d_{\min}, q) \to \mathcal{Q}(\sqrt{d_{\min}E_s/N_0})$. In other words, the optimal HSMC degenerates to MC in this situation. In addition, $Q_1(\cdot)$ is applicable only to QPSK (as well as binary signaling). Thus we turn to another way of code-agnostic noise gauging, which can alleviate these deficiencies of $Q_1(\cdot)$.

B. CODE-AGNOSTIC QUADRATIC NOISE GAUGING

Based on (26), an efficient NGF should distinguish well between error-causing and non-error-causing noise vectors. Consider QPSK for the moment. Noting that a noise vector with a greater Euclidean norm is more likely to induce a decoding error than one with a smaller Euclidean norm, but a noise component (however large it may be) that increases the distance of a transmitted signal point from other points does not induce error, we define a *quadratic NGF* as

$$Q_2(\mathbf{b}, \mathbf{z}) = \sum_{i \in \Psi(\mathbf{b},\mathbf{z})} z_i^2 \tag{33}$$

where $\Psi(\mathbf{b}, \mathbf{z}) \triangleq \{\psi_1, \ldots, \psi_d\}$ is the set of noise indices such that $i \in \Psi(\mathbf{b}, \mathbf{z})$ if $\tilde{z}_i \ge 0$ and $i \notin \Psi(\mathbf{b}, \mathbf{z})$ otherwise. In this way, z_i contributes to $Q_2(\mathbf{b}, \mathbf{z})$ only if the direction of noise is towards a neighboring constellation point. The squaring of z_i not only accords with the error-inducing propensity of large noise but also facilitates the derivation of the PDF of $Q_2 \triangleq Q_2(\mathbf{B}, \mathbf{Z})$. For this, since $\tilde{z}_i$ has equal probability being positive or negative, the cardinality of $\Psi(\mathbf{B}, \mathbf{Z})$, $D \triangleq |\Psi(\mathbf{B}, \mathbf{Z})|$, is binomial-distributed with probability mass function (PMF)

$$f_D(d) = \binom{l}{d} 2^{-l}, \quad 0 \le d \le l. \tag{34}$$

As a result, Q_2 is a mixture of scaled chi-squared random variables with cumulative distribution function (CDF)

$$F_{Q_2}(q) = \sum_{d=1}^{l} f_D(d) \mathrm{P}(Q_2 < q | D = d) \tag{35}$$

$$= \sum_{d=1}^{l} \frac{\binom{l}{d}}{2^l} F_\chi^{(d)}\left(\frac{q}{\sigma^2/2}\right) \tag{36}$$

where $F_\chi^{(d)}(\cdot)$ represents the CDF of a chi-squared random variable with d degrees of freedom (DoF).

HSMC requires the evaluation of p_m and $p_{\mathrm{err}|m}$ $\forall m$. As p_m can be evaluated from the above CDF, the below concentrates on how to estimate $p_{\mathrm{err}|m}$. Similar to $Q_1(\cdot)$, by the law of total expectation we can rewrite (21) as

$$\begin{aligned} p_{\mathrm{err}|m} &= \mathrm{E}_{\mathbf{B}} \mathrm{E}_{\mathbf{Z},D,Q_2|\Omega_m}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})] \\ &= \mathrm{E}_{\mathbf{B}} \mathrm{E}_{D|\Omega_m} \mathrm{E}_{Q_2|D,\Omega_m} \mathrm{E}_{\mathbf{Z}|D,Q_2}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})] \end{aligned} \tag{37}$$

where the sampling of $Q|\Omega_m$ in (21) has been replaced by sampling of $D|\Omega_m$ and $Q_2|D, \Omega_m$.

According to the sequence of conditioning set forth in (37), the generation of each simulation sample (indexed j) progresses in the order $\mathbf{B} \to D \to Q_2 \to \mathbf{Z}$. To start, we draw an information vector sample $\mathbf{b}^{(j)}$ and encode it into $\mathbf{c}^{(j)}$ to obtain $\mathbf{w}^{(j)}$. Based on the probability distributions of $D|\Omega_m$ and $Q_2|D, \Omega_m$ (detailed expressions given in Appendix C), we employ the A/R method to draw $d^{(j)}$ and $q_2^{(j)}$. Then we draw the noise vector sample $\mathbf{z}^{(j)} = [z_1^{(j)}, \ldots, z_l^{(j)}]$ with the simultaneous requirements that $\Psi^{(j)} \triangleq \Psi(\mathbf{b}^{(j)}, \mathbf{z}^{(j)})$ satisfies $|\Psi^{(j)}| = d^{(j)}$ and that $\sum_{i \in \Psi^{(j)}} (z_i^{(j)})^2 = q_2^{(j)}$. For this, we first draw a standard Gaussian vector sample $\bar{\mathbf{s}}^{(j)} = [\bar{s}_1^{(j)}, \ldots, \bar{s}_l^{(j)}]$ from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ and flip the signs of its elements randomly until we have $|\Psi^{(j)}| = d^{(j)}$. Let the resulting vector be $\mathbf{s}^{(j)} = [s_1^{(j)}, \ldots, s_l^{(j)}]$. Second, we scale the elements in $\mathbf{s}^{(j)}$ whose indices belong to $\Psi^{(j)}$ to make their sum-of-squares equal to $q_2^{(j)}$ (which by [35] yields the proper conditional noise PDF). Specifically, we set $\hat{\sigma}^2 = 2q_2^{(j)} / \sum_{i \in \Psi^{(j)}} (s_i^{(j)})^2$ and obtain the jth sample of $\mathbf{Z}|D, Q_2$ as

$$z_i^{(j)} = \begin{cases} \sqrt{\frac{\hat{\sigma}^2}{2}} s_i^{(j)}, & i \in \Psi^{(j)}, \\ \sqrt{\frac{\sigma^2}{2}} s_i^{(j)}, & \text{otherwise}. \end{cases} \tag{38}$$

Algorithm shows a summary of the sampling method. The most computationally demanding part in it is the numerical evaluation of the chi-squared CDF needed for (60) and (62) (see Appendix C), but the CDF can be precomputed and tabulated so as to make the computational overhead of sampling much lower than the decoding complexity required to determine if the received signal vector $\mathbf{y}^{(j)} \in I_{\mathcal{F}}(\mathbf{b}^{(j)}, \mathbf{z}^{(j)})$.

C. EXTENSION OF QUADRATIC NOISE GAUGING TO HIGHER-ORDER QAM

The above discussion on noise gauging has been restricted to QPSK. In this section, we extend the quadratic NGF to deal with 16QAM, whose idea can be further extended to deal with other higher-order QAMs. The idea is to redefine $\Psi(\mathbf{b}, \mathbf{z})$ (the set of noise indices over which their sum-of-squares is computed) according to the modulation method: an index i is included in $\Psi(\mathbf{b}, \mathbf{z})$ only if z_i shifts w_i towards a neighboring constellation point. Consider the 16QAM constellation in Fig. 3. In the case w_i takes an "outer value," i.e., $w_i = \pm 3/\sqrt{10}$, index i is included in $\Psi(\mathbf{b}, \mathbf{z})$ if z_i has an opposite sign with respect to w_i (a case similar to that of QPSK). But in the case w_i takes an "inner value," i.e., $w_i = \pm 1/\sqrt{10}$, index i is always included in $\Psi(\mathbf{b}, \mathbf{z})$ because w_i is surrounded by other constellation points. In this way, the distribution of $D = |\Psi(\mathbf{B}, \mathbf{Z})|$ depends on the numbers of inner and outer values in $\mathbf{W}$.

Algorithm 2 Sample Generation Method Under Proposed Quadratic Noise Gauging

```
1: $\mathbf{b}^{(j)} \leftarrow f_{\mathbf{B}}(\mathbf{b})$
2: Encode $\mathbf{b}^{(j)}$ into $\mathbf{c}^{(j)}$; modulate $\mathbf{c}^{(j)}$ into $\mathbf{w}^{(j)}$
3: $d^{(j)} \leftarrow f_{D|\Omega_m}(d)$ (via A/R with (60))
4: $q_2^{(j)} \leftarrow f_{Q_2|D,\Omega_m}(q|d^{(j)})$ (via A/R with (62))
5: $\bar{\mathbf{s}}^{(j)} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$; randomly flip the signs of elements until
   $|\Psi^{(j)}| = d^{(j)}$ and let the result be $\mathbf{s}^{(j)}$
6: Set $\hat{\sigma}^2 = 2q_2^{(j)} / \sum_{i \in \Psi^{(j)}} (s_i^{(j)})^2$
7: Set $\mathbf{z}^{(j)} = [z_1^{(j)}, \ldots, z_l^{(j)}]$ according to (38)
8: Return $\mathbf{b}^{(j)}$, $\mathbf{c}^{(j)}$, $\mathbf{w}^{(j)}$, $\mathbf{z}^{(j)}$
```

An exact evaluation of $f_D(d)$ can be very complicated because it depends on the details of the QAM mapping (see Fig. 3) although, in principle, it can be evaluated as $f_D(d) = \sum_{\mathbf{b}} f_{\mathbf{B}}(\mathbf{b}) f_{D|\mathbf{B}}(d|\mathbf{b})$. Nevertheless, we find that $f_D(d)$ can be well approximated in the following way. To begin, let D_i denote the number of terms contributed by Z_i in $\Psi(\mathbf{B}, \mathbf{Z})$, so that $D = \sum_{i=1}^{l} D_i$. We have

$$f_{D_i}(d) = \sum_{w} \mathrm{P}(W_i = w) f_{D_i|W_i=w}(d) \quad \forall i \tag{39}$$

where, for 16QAM, $w \in \{\pm 1/\sqrt{10}, \pm 3/\sqrt{10}\}$ and

$$f_{D_i|W_i \in \left\{\pm 1/\sqrt{10}\right\}}(d) = \begin{cases} 0, & d = 0, \\ 1, & d = 1, \end{cases} \tag{40}$$

$$f_{D_i|W_i \in \left\{\pm 3/\sqrt{10}\right\}}(d) = \begin{cases} 0.5, & d = 0, \\ 0.5, & d = 1. \end{cases} \tag{41}$$

The approximation consists in assuming that $P(W_i = w) = 0.25$ $\forall w$ $\forall i$ and that D_i are independent, so that $f_{D_i}(d)$ are the same $\forall i$ and that

$$f_D(d) \approx f_{D_1}(d) * \ldots * f_{D_l}(d) \triangleq f_{D_1}^{*l}(d), \tag{42}$$

where $*$ denotes convolution and superscript $*l$ denotes l-fold self-convolution of the given function. The approximation is particularly appropriate when the QAM mapping is randomized, such as with an interleaver or scrambler. Substituting $f_D(d)$ into (35) yields an approximation to $F_{Q_2}(\cdot)$ which can be used to evaluate p_m.

To estimate $p_{\mathrm{err}|m}$, we need to sample $(\mathbf{B}, \mathbf{Z})|\Omega_m$. Different from QPSK, the nonlinearity in the bit-to-signal value mapping of 16QAM (or any other higher-order QAM) breaks the symmetry about $\mathbf{b} = \mathbf{0}$ in a linear code, making sampling of $\mathbf{B}$ a necessity rather than an option. Moreover, $\mathbf{B}$ and Ω_m are no longer mutually statistically independent because the modulated signals associated with an information vector may have different counts of inner and outer values than that associated with another information vector, resulting in a distribution of D that depends on $\mathbf{B}$, and hence dependence between Q_2 and $\mathbf{B}$ or between Ω_m and $\mathbf{B}$. Therefore, in contrast to the QPSK case in (37) we have

$$p_{\mathrm{err}|m} = \mathrm{E}_{\mathbf{B}|\Omega_m}\mathrm{E}_{D|\mathbf{B},\Omega_m}\mathrm{E}_{Q_2|D,\Omega_m}\mathrm{E}_{\mathbf{Z}|D,Q_2}[I_{\mathcal{F}}(\mathbf{B},\mathbf{Z})]. \quad (43)$$

Compared to (37), the two outermost expectations are additionally conditioned on $Q_2 \in \Omega_m$ and $\mathbf{B}$, respectively. We derive the corresponding conditional probability distributions of $\mathbf{B}|\Omega_m$ and $D|\mathbf{B}, \Omega_m$ in Appendix D.

To draw samples $\mathbf{b}^{(j)}$ according to $f_{\mathbf{B}|\Omega_m}(\cdot)$ under given Ω_m, the A/R method requires a good grasp of the shape of $f_{\mathbf{B}|\Omega_m}(\cdot)$ over the sample space of $\mathbf{B}$. Otherwise it may risk a very high rejection rate, resulting in a low sampling efficiency. Hence the A/R method is only suitable for very short $\mathbf{B}$ (say $k < 30$) because for long vectors it is difficult to scrutinize the sample spaces sufficiently exhaustively for a detailed understanding of $f_{\mathbf{B}|\Omega_m}(\cdot)$. Therefore, we resort to the Metropolis algorithm [38]. Different from the A/R method which determines the acceptance/rejection of each random sample individually according to $f_{\mathbf{B}|\Omega_m}(\cdot)$, the Metropolis algorithm makes the decision according to the relative probability of two successive samples. Specifically, suppose we have generated $\mathbf{b}^{(j-1)}$. To generate $\mathbf{b}^{(j)}$, we first draw a candidate sample $\mathbf{b}_t$ from $f_{\mathbf{B}}(\cdot)$ and a value u_t from $\mathcal{U}[0, 1]$ (where $\mathcal{U}[0, 1]$ denotes uniform distribution over $[0, 1]$). Then we set $\mathbf{b}^{(j)} = \mathbf{b}_t$ if $f_{\mathbf{B}|\Omega_m}(\mathbf{b}_t)/f_{\mathbf{B}|\Omega_m}(\mathbf{b}^{(j-1)}) > u_t$ and $\mathbf{b}^{(j)} = \mathbf{b}^{(j-1)}$ otherwise. In the end, the set of samples $\{\mathbf{b}^{(j)}, j = 1, 2, \dots\}$ will follow the distribution $f_{\mathbf{B}|\Omega_m}(\cdot)$.

Due to the setting of some $\mathbf{b}^{(j)} = \mathbf{b}^{(j-1)}$, the samples generated by the Metropolis algorithm are not all independent. So the estimator variance is no longer as given in (22) but is changed to

$$V_M(\hat{p}_e) = \sum_{m=1}^{n_b} p_m^2 \frac{p_{\mathrm{err}|m}(1 - p_{\mathrm{err}|m})(1 + \delta_m)}{n_m} \quad (44)$$

where δ_m is the correlation coefficient between samples in the mth bin [39]. This coefficient is system-dependent and generally unknown beforehand, but can be estimated during the simulation along with $p_{\mathrm{err}|m}$ to provide a correction to the optimal histogram in (23).

Given $\mathbf{b}^{(j)}$, the sampling of D according to $f_{D|\mathbf{B},\Omega_m}(\cdot|\mathbf{b}^{(j)})$ does not suffer the dimensionality issue as the sampling of $\mathbf{B}$. Hence we employ the A/R method to draw $d^{(j)}$ based on $f_{D|\mathbf{B},\Omega_m}(\cdot|\mathbf{b}^{(j)})$. Then we follow the steps in Algorithm to draw $q_2^{(j)}$ and $\mathbf{z}^{(j)}$, except that $|\Psi^{(j)}| = d^{(j)}$ is satisfied by only randomly flipping the signs of elements in $\bar{\mathbf{s}}^{(j)}$ corresponding to signal values $\pm 3/\sqrt{10}$ but not those corresponding to signal values $\pm 1/\sqrt{10}$, as the latter are always included in $\Psi^{(j)}$.

Before proceeding further, a remark may be ready: While the above discussion has focused on Gray-coded QPSK and 16QAM, it is not hard to see that the working principle of the proposed method is also applicable to other kinds of signal constellation and other kinds of bit-to-symbol mapping. What needs to be attended to in NGF design for a specific constellation is the error behavior of each signal point in noise. The bit-to-symbol mapping does not affect NGF design but will affect the distance property of the modulated codewords and thus the PER. In the end, each different case may come with a different simulation complexity.

V. ADAPTIVE SHAPING OF HISTOGRAM

As shown in (23), the optimal simulation histogram ρ_m that minimizes the estimator variance under a particular NGF depends on both p_m and $p_{\mathrm{err}|m}$ $\forall m$. However, it is seen in the last section that, even for relatively simple NGFs, $p_{\mathrm{err}|m}$ needs to be estimated by running simulations (not to say that, for more complicated NGFs, even p_m may also need to be estimated using simulations). Hence the determination of ρ_m and estimation of $p_{\mathrm{err}|m}$ become intertwined. In this section, we develop an adaptive method to shape the simulation histogram. We assume that the NGF is suitably designed so that $f_Q(q)$ and $p_e(q)$ exhibit the typical shapes of distribution depicted in Fig. 4 and that $p_{\mathrm{err}|m}$ is an increasing function of m as a higher NGF value q should indicate that the packets are subject to more noise and suffer a higher PER.

The proposed approach to adaptive histogram shaping is illustrated in Fig. 6. Some of the detailed design is geared toward convenience of coarse-grain parallel implementation on multiple processors. Since $p_{\mathrm{err}|m}$ increases with m, the upper part of the bins contribute more to p_e. So we start by concentrating the simulation samples there. As indicated in Fig. 6(a), we initially try to generate a flat histogram over the bins to the right of the peak in p_m (see bars denoted n_m in Fig. 6(a)). Via the A/R formalism, we initialize the sampling with an "acceptance rate vector" over all the bins as

$$\beta_m^{(1)} = \begin{cases} 0, \text{ if } m < \mathrm{argmax}_{m'} p_{m'}, \\ 1, \text{ otherwise}, \end{cases} \quad (45)$$

where the parenthesized superscript is the iteration index. For convenience, each iteration consists of n_w simulation runs (distributed evenly to all parallel processors). For each simulation run, sampling is conducted by first obtaining an accepted bin (say m) and then generating a sample of $(\mathbf{B}, \mathbf{Z})$ in it using Algorithm (for $Q_1(\cdot)$) or Algorithm (for $Q_2(\cdot)$, with the minor tuning stated in the penultimate paragraph of Section IV in the case of 16QAM). Let the sample generated in the jth simulation run be $(\mathbf{b}^{(j)}, \mathbf{z}^{(j)})$. The sample count and error count for the bin (denoted n_m and e_m, respectively) are incremented by 1 and $I_{\mathcal{F}}(\mathbf{b}^{(j)}, \mathbf{z}^{(j)})$, respectively. Iterations continue until $\hat{p}_e = \sum_m p_m \hat{p}_{\mathrm{err}|m} > 0$ where

$$\hat{p}_{\mathrm{err}|m} = \begin{cases} e_m/n_m, & \text{if } n_m > 0, \\ 0, & \text{otherwise}. \end{cases} \quad (46)$$

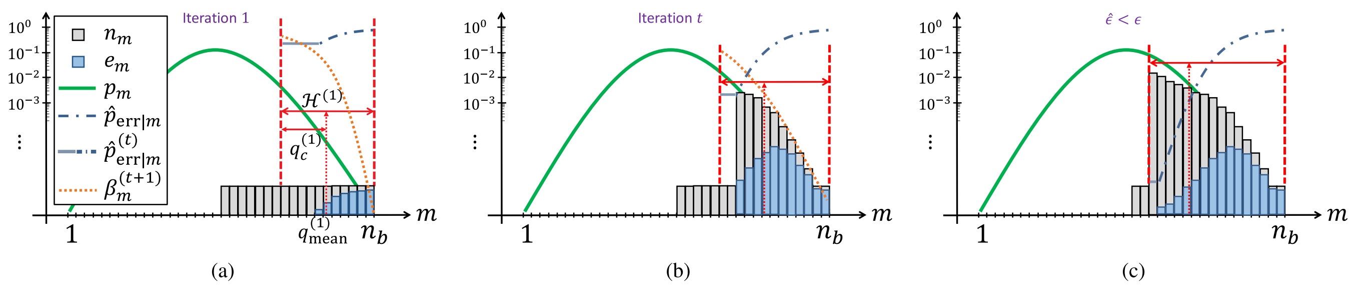


FIGURE 6. The proposed adaptive histogram shaping method. We plot p_m, $\hat{p}_{\mathrm{err}|m}$ and $\beta^{(t)}_m$ as continuous functions for convenience. In each plot, the dotted upward arrow (in red) indicates the location of the value $q^{(t)}_{\mathrm{mean}}$, and $q^{(t)}_c$ is a shorthand for $\max\{2q^{(t)}_{\mathrm{std}}, q_{\mathrm{init}}\}$. The dashdot curve (in grayish blue) illustrates $\hat{p}_{\mathrm{err}|m}$ and the solid horizontal line segment (also in grayish blue) connected to the $\hat{p}_{\mathrm{err}|m}$ curve illustrates the extension of $\hat{p}_{\mathrm{err}|m}$ over the bins where $e_m = 0$; the two, together, make up the whole of $\hat{p}^{(t)}_{\mathrm{err}|m}$. (a) Initialization (iteration 1), with step-like histogram. (b) Typical intermediate condition. (c) Simulation stops when $\hat{\epsilon} \le \epsilon$.

Fig. 6(a) illustrates a case where, after the first iteration, n_m is indeed a step-like function and packet errors are observed so that $e_m \neq 0$ for some m.

To proceed, we need to address two points: first, whether we have attained the desired precision in $\hat{p}_e$ so that no further simulations are needed and, second, how to perform further simulations when they are needed. We address the second point first. The conditional PER estimates in (46) facilitate a rough estimate of the optimal normalized histogram as

$$\hat{\rho}_m = \frac{p_m\sqrt{\hat{p}_{\mathrm{err}|m}}}{\sum_m p_m\sqrt{\hat{p}_{\mathrm{err}|m}}}. \tag{47}$$

We cannot base further simulations solely on it, however, particularly because those bins with $\hat{p}_{\mathrm{err}|m} = 0$ would be excluded from subsequent sampling, yet the reason that we get $\hat{p}_{\mathrm{err}|m} = 0$ in those bins in the early iterations may not be due to a zero $p_{\mathrm{err}|m}$ but due to an insufficient number of total simulation runs to make errors show up therein. To address this issue, we include some bins that have so far yielded no errors in the next iteration of simulations in the following way. First, define the sample mean and standard deviation of $\hat{\rho}_m$ as

$$q^{(t)}_{\mathrm{mean}} = \sum_{m=1}^{n_b} q_m\hat{\rho}_m, \quad q^{(t)}_{\mathrm{std}} = \sqrt{\sum_{m=1}^{n_b}\left(q_m - q^{(t)}_{\mathrm{mean}}\right)^2\hat{\rho}_m}, \tag{48}$$

where t is the iteration index. By Cantelli's inequality, a random variable has at most 20% of probability being smaller than its mean minus two times its standard deviation. Hence we consider the set of bins

$$\mathcal{H}^{(t)} = \left\{m \mid q_m \ge q^{(t)}_{\mathrm{mean}} - \max\left\{2\,q^{(t)}_{\mathrm{std}}, q_{\mathrm{init}}\right\}\right\} \tag{49}$$

where q_{init} is to guard against a too-small $q^{(t)}_{\mathrm{std}}$, which may occur due to insufficient observation of $p_{\mathrm{err}|m}$. Experiments show that a good setting of q_{init} is $q_{\mathrm{init}} = (q_{\max} - q_{\min})/10$, which forces at least 10% of the bins being selected. For each bin in $\mathcal{H}^{(t)}$, we re-parametrize $\hat{p}_{\mathrm{err}|m}$ as $\hat{p}^{(t)}_{\mathrm{err}|m} = \hat{p}_{\mathrm{err}|m_+}$ where m_+ is the index of the nearest bin where packet errors are observed. This makes $\hat{p}^{(t)}_{\mathrm{err}|m} \neq 0\ \forall m \in \mathcal{H}^{(t)}$. If there are two nearest bins (i.e., one on each side), we take the one with the larger index, which tends to yield a larger $\hat{p}^{(t)}_{\mathrm{err}|m}$. Then the estimated normalized optimal histogram is updated as

$$\hat{\rho}^{(t)}_m = \frac{p_m\sqrt{\hat{p}^{(t)}_{\mathrm{err}|m}}}{\sum_m p_m\sqrt{\hat{p}^{(t)}_{\mathrm{err}|m}}}. \tag{50}$$

For the next iteration, we update the acceptance rate as

$$\beta^{(t+1)}_m = \begin{cases} \frac{\hat{\rho}^{(t)}_m - n_m/n^{(t)}_{\mathrm{IS}}}{\max_m\left\{\hat{\rho}^{(t)}_m - n_m/n^{(t)}_{\mathrm{IS}}\right\}}, & \text{if } \hat{\rho}^{(t)}_m > n_m/n^{(t)}_{\mathrm{IS}}, \\ 0, & \text{otherwise}, \end{cases} \tag{51}$$

where $n^{(t)}_{\mathrm{IS}} = \sum_m n_m$, so as to steer n_m towards the optimal distribution. Fig. 6(b) illustrates a typical intermediate condition, where the acceptance rate vector $\beta^{(t+1)}_m$ has been generated at the end of iteration t for iteration $t+1$. The progressively increasing samples over the iterations improve the accuracy of $\hat{p}_{\mathrm{err}|m}$, leading towards a better estimate of ρ_m and a reduced variance in $\hat{p}_e$. The looping decision is made by checking the estimated current precision given by

$$\hat{\epsilon} = \frac{\sqrt{\hat{V}(\hat{p}_e)}}{\hat{p}_e} = \frac{\sqrt{\sum_{m=1}^{n_b} p_m^2\hat{p}^{(t)}_{\mathrm{err}|m}/n_m}}{\sum_{e_m\neq 0} p_m\hat{p}_{\mathrm{err}|m}} \tag{52}$$

where $\hat{p}^{(t)}_{\mathrm{err}|m}$ is used in the numerator to effect a conservative estimate of the current precision.

Fig. 6(c) illustrates a typical condition at the conclusion of the simulation. Not all bins are sampled and not all the sampled bins have yielded packet errors. But the conservative setting of $\mathcal{H}^{(t)}$, $\hat{p}^{(t)}_{\mathrm{err}|m}$, and $\hat{\rho}^{(t)}_m$ should contribute to ensuring having sufficient samples in the bins of nonnegligible $p_{\mathrm{err}|m}$ for a robust PER estimate.

The proposed adaptive HSMC simulation method is summarized in Algorithm .

VI. SOME NUMERICAL RESULTS

In this section, we present some numerical results on several aspects of the proposed adaptive HSMC. To save simulation time, we do coarse-grain parallel computing on 50 processors and set $n_w = 10^4$ and $n_b = 20$ to avoid excessive data exchanges among the processors.

TABLE 1. Speedup factors of HSMC vs. MC for BCH coding with QPSK modulation[†].

	G^* under $Q_1(\cdot)$			$\hat{G}^*$ under $Q_2(\cdot)$			G under $Q_2(\cdot)$		
	$(63,7)$	$(127,8)$	$(255,9)$	$(63,7)$	$(127,8)$	$(255,9)$	$(63,7)$	$(127,8)$	$(255,9)$
$p_e \approx 10^{-5}$	129	176	207	169	68	31	34	15	9
$p_e \approx 10^{-7}$	530	695	990	2632	457	336	359	83	43
$p_e \approx 10^{-9}$	2415	3446	3727	14183	2858	509	1689	363	78

[†]See text for explanation of G^* (ideal speedup factor), $\hat{G}^*$ (estimated ideal speedup factor), and G (actual speedup factor).

Algorithm 3 Adaptive Histogram-Shaping Monte Carlo

```
1:  Initialize $\beta_m^{(1)}$ by (45), $t = 0$, $n_m = e_m = 0\ \forall m$
2:  repeat
3:      $t \leftarrow t + 1$
4:      for $j = 1$ to $n_w$ do
5:          Select $m$ based on $\beta_m^{(t)}$ by A/R method
6:          Draw $(\mathbf{b}^{(j)}, \mathbf{z}^{(j)})$ in bin $m$ by Algorithm  or
7:          $n_m \leftarrow n_m + 1$, $e_m \leftarrow e_m + I_{\mathcal{F}}(\mathbf{b}^{(j)}, \mathbf{z}^{(j)})$
8:      end for
9:      if $\sum_m e_m > 0$ then
10:         Evaluate $\hat{p}_{\mathrm{err}|m}$ by (46) and $\hat{\rho}_m$ by (47) $\forall m$
11:         Determine $\mathcal{H}^{(t)}$ by (48) and (49)
12:         Re-parametrize $\hat{p}_{\mathrm{err}|m}^{(t)}$ and $\hat{\rho}_m^{(t)}$ for $m \in \mathcal{H}^{(t)}$
13:         Determine $\beta_m^{(t+1)}\ \forall m$ by (51)
14:     else
15:         $\beta_m^{(t+1)} \leftarrow \beta_m^{(t)}\ \forall m$
16:     end if
17: until $\hat{\epsilon} \leq \epsilon$
```

We first verify the precision of the HSMC using some BCH codes, where we consider QPSK modulation and employ ML decoding in HSMC. We compare the PER performance obtained under HSMC with the union bound in (4). The (n,k) values of the BCH codes are $(63,7)$, $(127,8)$, and $(255,9)$, where the values of k are small to ease ML decoding. The minimum distances of these codes are 31, 63, and 127, respectively, which are approximately linearly proportional to n (with a decreasing r as n increases). To facilitate QPSK modulation, we add a filler bit to each codeword to make the word length even. Fig. 7 compares the PER estimates of HSMC under $Q_2(\cdot)$ with the union bound. We see that they closely match each other.

We next look at the efficiency of HSMC. For this, Table 1 lists the speedup factors at some PER values under $Q_1(\cdot)$ and $Q_2(\cdot)$, where G^* is the ideal speedup factor (25), $\hat{G}^*$ is an estimate of the ideal speedup factor evaluated using (25) by setting $p_{\mathrm{err}|m}$ therein equal to $\hat{p}_{\mathrm{err}|m}$ and p_e equal to $\sum_m p_m \hat{p}_{\mathrm{err}|m}$, and G is the actual speedup factor given by $n_{\mathrm{MC}}/n_{\mathrm{IS}}$ with $n_{\mathrm{MC}} \triangleq 100/p_{\mathrm{e}}$ and n_{IS} obtained from running HSMC with $\epsilon = 10\%$ in Algorithm . Commensurate with the theory in Section IV-A, the ideal speedup under $Q_1(\cdot)$ increases with n. Interestingly, $Q_2(\cdot)$ outperforms $Q_1(\cdot)$ in ideal speedup for the $(63,7)$ code but is outperformed by $Q_1(\cdot)$ for the longer codes. We deem this relative behavior to have to do with the selective exclusion of some noise

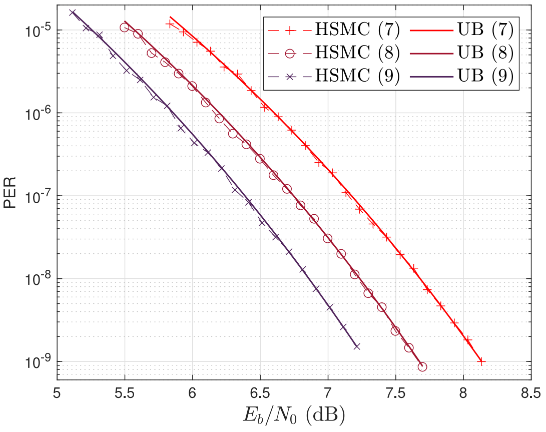


FIGURE 7. PER results for BCH coding at $k = 7$, 8, and 9, where HSMC employs $Q_2(\cdot)$ and "UB" indicates the union bound.

elements in $Q_2(\cdot)$, but a more detailed theoretical analysis of this relative behavior appears difficult and is not pursued in this work. Overall, the actual speedup ranges approximately from 10- to 1000-fold. The suboptimality embodied in the adaptive histogram shaping has led to approximately one order-of-magnitude loss in speedup compared to using the optimal histogram.

We now investigate the performance of HSMC under polar coding, where we consider both the QPSK and 16QAM modulations and hence only examine $Q_2(\cdot)$. Let code rate $r = 1/2$ and $n = 256$, and let the code be of the cyclic redundancy code (CRC)-aided kind with 24 CRC bits attached to the information sequence. We also apply the bit interleaving in [7, Sec. 5.4.1.3], which legitimizes the approximation in (42). At the receiver, we employ successive cancellation list decoding (SCLD) with different list sizes for different decoding complexities. Fig. 8 shows some PER results, where $\mathcal{L}$ denotes the list size. Note that we have run MC only down to a PER of approximately 10^{-7} due to simulation time concern. Similar to the earlier BCH cases, the HSMC and MC results closely match each other.

Concerning the speedup factors, Table 2 shows some results (with "actual speedup factor" defined similarly to Table 1). The proposed HSMC shows better efficiency as $\mathcal{L}$ increases. To better understand this intriguing performance feature of HSMC, we analyze the corresponding codeword error patterns. Specifically, we re-encode the decoded information vector in the case of a packet error and compute the Hamming distance between the re-encoded codeword

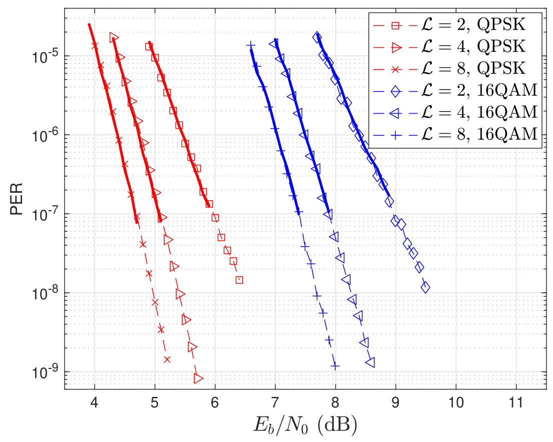


FIGURE 8. PER results for polar coding under different modulations and list sizes, where HSMC employs $Q_2(\cdot)$. The solid lines show MC results down to $p_e \approx 10^{-7}$.

TABLE 2. Actual speedup factors of $Q_2(\cdot)$-based HSMC vs. MC for polar coding.

	QPSK			16QAM		
p_e	$\mathcal{L}=2$	$\mathcal{L}=4$	$\mathcal{L}=8$	$\mathcal{L}=2$	$\mathcal{L}=4$	$\mathcal{L}=8$
10^{-7}	3.00	11.32	55.23	3.45	18.15	89.07
10^{-8}	3.12	25.53	37.45	7.10	15.43	128.52

TABLE 3. Distribution of error patterns in MC simulation of polar decoding at different list sizes, in A total of 100 errors for each list size.

Hamming distance	8	16	20	24	others (≥ 28)
List-2 decoding	87	12	0	0	1
List-4 decoding	16	61	8	7	8
List-8 decoding	5	45	6	17	27

and the transmitted codeword. Table 3 tallies the MC results from 100 packet errors for each of three list sizes. As can be seen, the number of smaller-distance errors (in particular, distance ≤ 16) increases significantly with decreasing list size. This indicates that, when the decoding is further away from being ML (which is the case with a smaller list size), the performance of HSMC with $Q_2(\cdot)$ would suffer more from the presence of lower-weight error patterns. This has some resemblance to the declining of performance of $Q_1(\cdot)$ with decreasing $d_{\min}$. On the negative side, this seems to indicate a deficiency in the proposed HSMC under code-agnostic noise gauging, in that it does not fully surmount the effect of code weaknesses or decoder suboptimality. But on the other hand, this may also indicate that, aside from being a simulation tool, the method can possibly be used to assist system design by detecting code weaknesses or decoder suboptimality via observing how the speedup factor vary with the transmission mechanism or transceiver parameters. Further examination of this perspective is left to potential future work.

Lastly, an aspect of interest is how serious the sample correlation is in the Metropolis algorithm used in $Q_2(\cdot)$-based HSMC 16QAM simulation. For this, Fig. 9 shows p_m, $\hat{p}_{\mathrm{err}|m}$, the estimated δ_m and acceptance rates of codeword samples for the bins where packet errors are observed. As can be

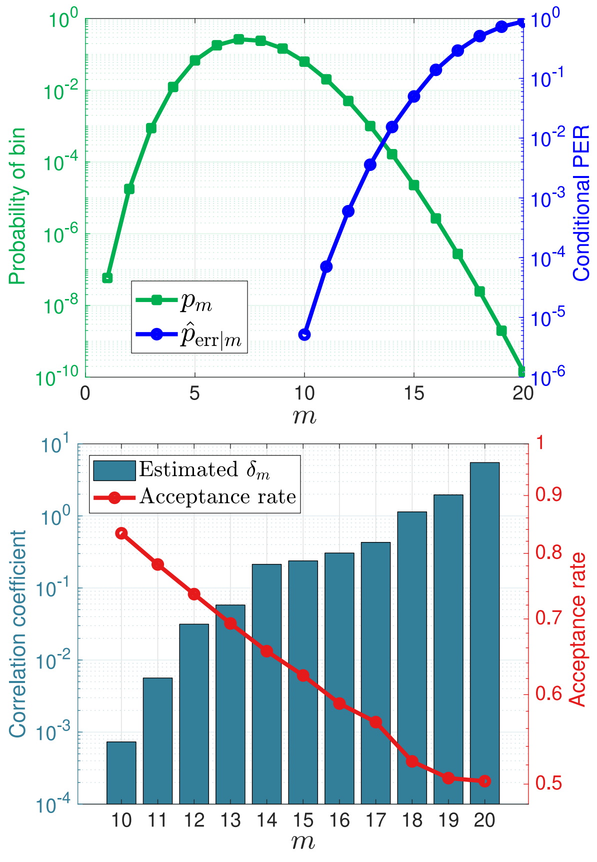


FIGURE 9. Distributions of p_m and $\hat{p}_{\mathrm{err}|m}$ and the effect of the Metropolis algorithm in $Q_2(\cdot)$-based HSMC 16QAM simulation, where $p_e \approx 10^{-5}$.

seen, the acceptance rate decreases with increasing m (i.e., increasing q), because the codeword distribution progressively deviates from the unconditioned $f_{\mathbf{B}}(\cdot)$ as q increases, and the Metropolis algorithm tends to reject a randomly generated codeword samples and keep the previous sample that has sufficient inner signal values to yield the required q value. The correlation factor increases due to increasing rejections. But the high-m bins have low probabilities (p_m). In the end, we find that $V_M(\hat{p}_e)/V(\hat{p}_e) = 1.029$. That is, the effect of the correlation on estimator variance is minor and can be ignored.

VII. EXTENSION TO BURST-ERROR CODES

It is well-known that there is much difficulty in applying IS to complex systems with high dimensions [15]. One way to combat this difficulty is to break the system into several interconnected subsystems of lower dimensions. For PER simulation, insight for how to do this may be gained by considering how packet errors relate to code structures. In the case of burst-error codes, such as a convolutional code, it is known that the Viterbi decoding delay has much to do with decoding performance and hence should serve well to define the subsystem dimension. More generally, any segmentation of the received signal may serve this purpose as long as the segment properties correlate highly with the PER. Thus we propose a *code-aware NGF* as

$$Q_c(\mathbf{b}, \mathbf{z}) = \max_{1 \leq \lambda \leq l/l_m} Q_2^{(\lambda)}(\mathbf{b}, \mathbf{z}) \tag{53}$$

where l_m is a proper segment length which divides the whole received signal into l/l_m segments, λ is the segment index, and $Q_2^{(\lambda)}(\mathbf{b}, \mathbf{z})$ is the segmental NGF given by

$$Q_2^{(\lambda)}(\mathbf{b}, \mathbf{z}) = \sum_{i \in \Psi^{(\lambda)}(\mathbf{b}, \mathbf{z})} z_i^2 \tag{54}$$

where $\Psi^{(\lambda)}(\mathbf{b}, \mathbf{z}) \triangleq \{i | i \in \Psi(\mathbf{b}, \mathbf{z}), (\lambda - 1)l_m < i \leq \lambda l_m\}$. The reason for taking the maximum of the segmental NGF values is based on the intuition that the noisiest segment in the signal may largely determine the error probability (as is the case in Viterbi decoding of convolutional codes).

Recall that HSMC requires the evaluation of p_m and $p_{\text{err}|m}$, where p_m is now determined by the distribution of $Q_c \triangleq Q_c(\mathbf{B}, \mathbf{Z})$. For simplicity, we only treat QPSK herein, although the approach is applicable to higher-order QAM. Define segmental cardinality $D^{(\lambda)} \triangleq |\Psi^{(\lambda)}(\mathbf{b}, \mathbf{z})|$. From the discussion on D in Section IV-B we see that $D^{(\lambda)}$ has PMF $f_D^{(\lambda)}(d) = \binom{l_m}{d}/2^{l_m}$ where $0 \leq d \leq l_m$. Since $Q_2^{(\lambda)}(\mathbf{B}, \mathbf{Z})$, $1 \leq \lambda \leq l/l_m$, are independent and identically distributed random variables, their maximum Q_c has CDF

$$F_{Q_c}(q) = \left[\mathrm{P}\left(Q_2^{(\lambda)} < q\right)\right]^{l/l_m} \triangleq \left(F_{Q_2}^{(\lambda)}(q)\right)^{l/l_m} \quad \forall \lambda \tag{55}$$

where $Q_2^{(\lambda)} \triangleq Q_2^{(\lambda)}(\mathbf{B}, \mathbf{Z})$. Now, $Q_2^{(\lambda)}$ is similarly distributed as Q_2 and hence $F_{Q_2}^{(\lambda)}(q)$ can be evaluated using (36) by letting $l = l_m$. The above suffices for the evaluation of p_m $\forall m$.

Now consider the evaluation of $p_{\text{err}|m}$, for which we need to sample $\mathbf{B}, \mathbf{Z}|\Omega_m$. The symmetry with respect to $\mathbf{b}$ under QPSK in AWGN (as noted in Section IV) makes the sampling of $\mathbf{B}$ independent of $\mathbf{Z}$ and Ω_m (the latter being totally depending on $\mathbf{Z}$). To deal with the maximization in (53), which selects the "noisiest" signal segment based on segmental NGF, let $\Lambda_c \triangleq arg\,max_{1 \leq \lambda \leq l/l_m} Q_2^{(\lambda)}(\mathbf{B}, \mathbf{Z})$ and $D_c \triangleq |\Psi_{\Lambda_c}(\mathbf{B}, \mathbf{Z})|$. By the AWGN assumption, every signal segment is equally probable to be the noisiest, and hence Λ_c is uniformly distributed over $\{1, 2, \ldots, l/l_m\}$ and the distribution of D_c is just that of $D^{(\lambda)}$ for any λ. We thus have (cf. (37))

$$\begin{aligned} p_{\text{err}|m} &= \mathrm{E}_{\mathbf{B}, \Lambda_c, D_c, Q_c, \mathbf{Z}|\Omega_m}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})] \\ &= \mathrm{E}_{\mathbf{B}} \mathrm{E}_{\Lambda_c} \mathrm{E}_{D_c|\Omega_m} \mathrm{E}_{Q_c|D_c,\Omega_m} \mathrm{E}_{\mathbf{Z}|\Lambda_c, D_c, Q_c}[I_{\mathcal{F}}(\mathbf{B}, \mathbf{Z})]. \end{aligned} \tag{56}$$

The generation of simulation samples thus proceeds as $\mathbf{B} \to \Lambda_c \to D_c \to Q_c \to \mathbf{Z}$. The conditional probability distributions of $D_c|\Omega_m$ and $Q_c|D_c, \Omega_m$, which are needed in performing the sampling, are derived in Appendix E, where we show that, under narrow bins (i.e., small Δq), $f_{D_c|\Omega_m}(\cdot) \approx f_{D|\Omega_m}(\cdot)$ and $f_{Q_c|D_c,\Omega_m}(\cdot) \approx f_{Q_2|D,\Omega_m}(\cdot)$, with l replaced by l_m. These approximations work well for $n_b \geq 100$, especially for the bins associated with larger q values for which the CDF $F_{Q_2}^{(\lambda)}(q) \approx 1$. We draw $d_c^{(j)}$ and $q_c^{(j)}$ similarly to $d^{(j)}$ and $q_2^{(j)}$ in Algorithm , except with $f_{D_c|\Omega_m}(\cdot)$ and $f_{Q_c|D_c,\Omega_m}(\cdot)$ replacing $f_{D|\Omega_m}(\cdot)$ and $f_{Q_2|D,\Omega_m}(\cdot)$, respectively. For the $\lambda_c^{(j)}$th segment of the noise vector, the sampling method is similar to lines 5–7 in Algorithm . For every other segment, we merely iterate the same way of sampling until its corresponding segmental NGF value is less than $q_c^{(j)}$.

TABLE 4. Speedup factors for the (256,128) TBCCs†.

TBCC	$Q_1(\cdot)$, G^*	$Q_2(\cdot)$, $\hat{G}^*$	$Q_c(\cdot)$, $\hat{G}^*$
$l_c = 3$	1.08	2.40	17.95
$l_c = 7$	1.20	3.55	12.01
$l_c = 11$	1.37	4.95	11.11

†See discussion on Table 1 for meaning of G^* and $\hat{G}^*$.

To verify the performance of the proposed code-aware NGF, we evaluate the ideal speedup factors for some tail-biting convolutional codes (TBCC). Specifically, we consider the convolutional codes of optimal distance spectra of constraint lengths 3, 7, and 11 given in [40], with $(n, k) = (256, 128)$. The generator polynomials are given by (in octal numbers) $[5, 7]$, $[133, 171]$, and $[3345, 3613]$, respectively, with their free distances equal to 5, 10, and 14, respectively. (Although these codes are originally not optimized for TBCC, we find them well-suited for it in our case. For one thing, the number of minimum-distance codewords of a resulting TBCC is dominated by that associated with the zero initial state. In particular, the $[133, 171]$ code has formed the basis of some standardized TBCC schemes [5], [41].)

It is known that for a rate-1/2 convolutional code, Viterbi decoding yields near ML performance at decoding delays on the order of 5 constraint lengths [32], [42]. Hence typical burst errors under proper convolutional decoding would not exceed this duration. This hints that an l_m on order of 5 times the constraint length or greater may be a good choice. Experiments show that a good choice of l_m for the code-aware NGF is approximately $3l_c/r$ where l_c is the constraint length. Table 4 and Fig. 10 present some numerical results, where we have used the wrap-around Viterbi decoding technique [43] and adjusted the SNR to make $p_e \approx 10^{-5}$ for each code. And we have set $n_b = 100$.

First, Table 4 shows that the speed gain of $Q_1(\cdot)$ is very little and is lower than either $Q_2(\cdot)$ or $Q_c(\cdot)$. This is not surprising in view of the property of $Q_1(\cdot)$ discussed in Section IV-A, for a convolutional code has a fixed free distance (or minimum distance) although one may lengthen the codewords at will. In addition, Table 4 shows that $Q_1(\cdot)$ and $Q_2(\cdot)$ yield greater efficiency as l_c increases, but the advantage of $Q_c(\cdot)$ over them is significant, although its performance decreases with increasing l_c. Fig. 10 shows that, under $Q_2(\cdot)$, the $\hat{p}_{\text{err}|m}$ curve becomes notably steeper as l_c increases whereas under $Q_c(\cdot)$ this is not the case. This should account for the gain in speedup factor with increasing l_c under $Q_2(\cdot)$.

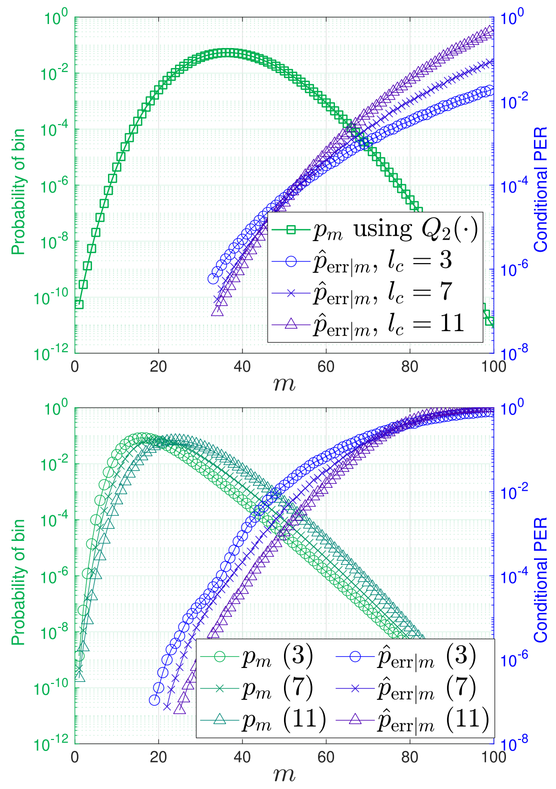


FIGURE 10. Distributions of p_m and $\hat{p}_{\mathrm{err}|m}$ under $Q_2(\cdot)$ and $Q_c(\cdot)$ for the (256, 128) TBCCs. The kink in the conditional PER curve for the $l_c = 3$ TBCC is due to a large percentage of decoding errors to code sequences at Hamming distances greater than the free distance at lower SNR values.

We have also run a simulation using the suboptimal decoding algorithm of [44]. It turns out the speedup behaviors are substantially similar.

Several remarks are in order. First, if the code is particularly prone to noise over a certain codeword segment, then it is effectively a burst-error code that the approach of this section can be applied. Second, if the codewords do not show a burst-error nature but they can be made so with a permutation of the code bits, then the technique of this section can be applied. And third, breaking the code into disjoint segments does not really capture the burst-error nature of a convolutional code, as an error burst could run across a segment boundary. A more pertinent NGF should thus consider arbitrarily located (and thus overlapping) segments [31]. But in this case, p_m does not have a closed-form expression and the sampling mechanism becomes more complicated.

VIII. CONCLUSION

Today's communication system design heavily depends on computer simulation for performance evaluation. In this work, we examined the principles and key issues of the importance sampling approach to efficient simulation and developed a histogram-shaping Monte Carlo (HSMC) technique for coded communication systems simulation. The technique employs a noise gauging function (NGF) to measure the error-causing propensity of channel noise samples and adaptively learns the proper histogram of simulation samples under the NGF.

Concerning the NGF, we proposed two basic kinds that are advantageous in two respects: first, they claim relative ease of sample generation (by being closely related to some sufficient statistics of AWGN parameters); and second, they require no detailed understanding of code properties (and thus code-agnostic in this sense, although some crude knowledge of the code properties can benefit their choice). One of the two is linear and the other is quadratic. The linear one is asymptotically efficient under random coding when the code length and SNR approach infinity but is applicable only under QPSK (or equivalently, BPSK). The quadratic one, with a proper sampling method, can be applied to higher-order modulations. They can therefore be chosen based on code and modulation properties. Under QPSK (or BPSK), the linear one is preferable when the code has a large minimum Hamming distance. Otherwise the quadratic one should be preferable.

We further considered the case of burst-error codes whose minimum distance properties are far inferior to that of random codes, and proposed a quadratic-type NGF to exploit the burst-error property of such codes to improve the simulation efficiency. Use of this NGF requires a little more knowledge of the code properties than the two basic kinds above and it is thus dubbed a code-aware one.

We presented the detailed method of how to adaptively learn the optimal sample histogram under a given NGF.

Simulations on short-length coding typically considered for ultra-reliable communication showed approximately 10- to 1,000-fold speedup of the proposed HSMC versus conventional MC. Some directions of potential future work were pointed out along the way.

APPENDIX A
MECHANISM OF PER UNDERESTIMATION

For a linear code, the Hamming distance d_H between a known and an unknown minimum-weight codeword satisfies $d_{\min} \le d_H \le 2d_{\min}$. This can be understood by considering how the 1-bit positions may be related between them:

$$\begin{array}{lcl} \text{Known (codeword a)} & : & \overbrace{11\cdots1}^{d}\ \overbrace{11\cdots1}^{d_{\min}-d}\ 00\cdots0\ 00\cdots0 \\ \text{Unknown (codeword u)} & : & \underbrace{11\cdots1}_{d}\ 00\cdots0\ \underbrace{11\cdots1}_{d_{\min}-d}\ 00\cdots0 \end{array}$$

In the above, d denotes the number of overlapping 1-bit positions. Since d_H is equal to $2(d_{\min} - d)$, it is always even. In simulating QPSK transmission of the all-zero codeword (term it codeword 0), $\boldsymbol{\mu}_a$ has its first $d_{\min}$ elements equal to $-1/\sqrt{2}$ with all the remaining elements being zero. Let $\mathbf{w}_i$ denote the packets associated codewords i, $i = 0, a, u$, respectively. A pairwise ML decoding error to codeword u results if $\mathbf{w}_0 + \boldsymbol{\mu}_a + \mathbf{Z}$ is closer to $\mathbf{w}_u$ than to $\mathbf{w}_0$. Straightforward algebra shows that this error probability is given by

$$\mathrm{P}\left(\sum_{i=\Pi_u(1)}^{\Pi_u(d_{\min})} Z_i \geq \frac{d_H/2}{\sqrt{2}}\right) = \mathcal{Q}\left(\sqrt{\frac{d_H^2}{4d_{\min}} E_s/N_0}\right). \quad (57)$$

Since $d_{\min} \leq d_H \leq 2d_{\min}$, we have the hit rate $p \triangleq \mathrm{P}(\mathbf{Z} + \boldsymbol{\mu}_a \in \mathcal{F}_u)$ satisfying

$$\underbrace{\mathcal{Q}\left(\sqrt{d_{\min} E_s/N_0}\right)}_{=p_{\min}} \leq p \leq \underbrace{\mathcal{Q}\left(\sqrt{\frac{d_{\min}}{4} E_s/N_0}\right)}_{\triangleq p_{\mathrm{ub}}}, \quad (58)$$

where the upper bound applies when $d_H = d_{\min}$.

By the property of the $\mathcal{Q}$ function, $p_{\mathrm{ub}} < 1/2$. While we cannot characterize p_{ub} more specifically in general, note that p will be small in two conditions: 1) very high SNR and 2) when d_H is close to $2d_{\min}$. Therefore, while mean translation substantially increases the hit rate of known codewords (see (14)), the hit rate of unknown codewords may remain low. When n_{IS} is small (which is the objective of IS), it is highly possible that decoding errors to unknown codewords are insufficiently represented, making $\hat{p}_{\mathrm{IS}}$ underestimate the true PER.

APPENDIX B UPPER BOUND ON THE MINIMUM VARIANCE

For simplicity, assume Ω encompasses the complete support of $f_Q(q)$ so that $\sum_m p_m = 1$ and $\sum_m p_m p_{\mathrm{err}|m} = p_e$. By the Cauchy-Schwarz inequality, the minimum variance given in (24) is upper bounded as

$$\begin{aligned} V_Q^*(\hat{p}_e) &= \frac{1}{n_{\mathrm{IS}}}\left(\sum_{m=1}^{n_b} \sqrt{p_m p_{\mathrm{err}|m}}\sqrt{p_m\left(1 - p_{\mathrm{err}|m}\right)}\right)^2 \\ &\leq \frac{\left(\sum_m p_m p_{\mathrm{err}|m}\right)\left(\sum_m p_m\left[1 - p_{\mathrm{err}|m}\right]\right)}{n_{\mathrm{IS}}} \\ &= \frac{p_e(1-p_e)}{n_{\mathrm{IS}}} = V\left(\hat{p}_{\mathrm{MC}}\right), \end{aligned} \quad (59)$$

where the bound is attained when $1 - p_{\mathrm{err}|m} = \nu p_{\mathrm{err}|m}$ for some constant ν $\forall m$. When this is the case, $p_{\mathrm{err}|m} = 1/(1+\nu)$ $\forall m$, resulting in $p_{\mathrm{err}|m} = p_e$ $\forall m$.

APPENDIX C PROBABILITY DISTRIBUTIONS USED IN (37)

By Bayes' rule, the PMF of $D|\Omega_m$ is given by

$$f_{D|\Omega_m}(d) = \frac{f_D(d)\mathrm{P}(Q_2 \in \Omega_m | D = d)}{\mathrm{P}(Q_2 \in \Omega_m)}, \quad 0 \leq d \leq l \quad (60)$$

where $f_D(d)$ is as in (34), $\mathrm{P}(Q_2 \in \Omega_m) = p_m$ by definition, and

$$\begin{aligned} &\mathrm{P}(Q_2 \in \Omega_m | D = d) \\ &= F_\chi^{(d)}\left(\frac{q_m + \Delta q/2}{\sigma^2/2}\right) - F_\chi^{(d)}\left(\frac{q_m - \Delta q/2}{\sigma^2/2}\right). \end{aligned} \quad (61)$$

For the PDF of $Q_2|D, \Omega_m$, we have

$$f_{Q_2|D,\Omega_m}(q|d) = \frac{f_\chi^{(d)}\left(\frac{q}{\sigma^2/2}\right)}{\mathrm{P}(Q_2 \in \Omega_m | D = d)}, \quad \forall q \in \Omega_m, \quad (62)$$

where $f_\chi^{(d)}(\cdot)$ is the chi-squared PDF with DoF $= d$.

APPENDIX D PROBABILITY DISTRIBUTIONS USED IN (43)

By Bayes' rule, the PMF of $\mathbf{B}|\Omega_m$ is given by

$$\begin{aligned} f_{\mathbf{B}|\Omega_m}(\mathbf{b}) &= \frac{\mathrm{P}(Q_2 \in \Omega_m | \mathbf{B} = \mathbf{b}) f_{\mathbf{B}}(\mathbf{b})}{\mathrm{P}(Q_2 \in \Omega_m)} \\ &= \frac{\sum_{d=0}^{l} \mathrm{P}(Q_2 \in \Omega_m, D = d | \mathbf{B} = \mathbf{b})}{2^k p_m} \\ &= \frac{\sum_{d=0}^{l} \mathrm{P}(Q_2 \in \Omega_m | D = d) f_{D|\mathbf{B}}(d|\mathbf{b})}{2^k p_m} \end{aligned} \quad (63)$$

where $\mathrm{P}(Q_2 \in \Omega_m | D = d)$ is as given in (61) and

$$f_{D|\mathbf{B}}(d|\mathbf{b}) = f^{*n_i}_{D_1|W_1 = \pm 1/\sqrt{10}}(d) * f^{*n_o}_{D_1|W_1 = \pm 3/\sqrt{10}}(d). \quad (64)$$

where n_i and n_o are counts of inner and outer signal values, respectively.

For the PMF of $D|\mathbf{B}, \Omega_m$, we get, again by Bayes' rule,

$$f_{D|\mathbf{B},\Omega_m}(d|\mathbf{b}) = \frac{\mathrm{P}(\mathrm{Q}_2 \in \Omega_\mathrm{m}, \mathrm{D} = \mathrm{d} | \mathbf{B} = \mathbf{b})}{\mathrm{P}(Q_2 \in \Omega_m | \mathbf{B} = \mathbf{b})}, \quad 0 \leq d \leq l, \quad (65)$$

where both the numerator and denominator are already obtained as byproducts in evaluating $f_{\mathbf{B}|\Omega_m}(\mathbf{b})$.

APPENDIX E PROBABILITY DISTRIBUTIONS USED IN (56)

Differentiating (55) yields the PDF of Q_c as

$$f_{Q_c}(q) = \frac{l}{l_m} f_{Q_2}^{(\lambda)}(q)\left(F_{Q_2}^{(\lambda)}(q)\right)^{l/l_m - 1} \quad (66)$$

(with λ arbitrary). To obtain $f_{D_c|\Omega_m}(\cdot)$ and $f_{Q_c|D_c,\Omega_m}(\cdot)$ for sampling use, we first obtain the joint distribution of D_c and Q_c. Since D_c applies only to the maximum of $Q_2^{(\lambda)}$, we have

$$f_{D_c,Q_c}(d,q) = \frac{l}{l_m} f_D^{(\lambda)}(d) f_{Q_2|D}^{(\lambda)}(q|d)\left(F_{Q_2}^{(\lambda)}(q)\right)^{l/l_m - 1} \quad (67)$$

where $f_{Q_2|D}^{(\lambda)}(\cdot)$ is chi-squared PDF with DoF $D^{(\lambda)}$. Due to the term $F_{Q_2}^{(\lambda)}(q)$, $f_{D_c|\Omega_m}(\cdot)$ and $f_{Q_c|D_c,\Omega_m}(\cdot)$ do not have simple closed-form expressions. To facilitate sampling computation, consider the case of narrow bins under a large n_b. We approximate $(l/l_m)(F_{Q_2}^{(\lambda)}(q))^{l/l_m - 1}$ $\forall q \in \Omega_m$ by a constant ξ_m to obtain an approximate PMF of $D_c|\Omega_m$ as

$$f_{D_c|\Omega_m}(d) = \frac{\int_{\Omega_m} f_{D_c,Q_c}(d,q)dq}{\mathrm{P}(Q_c \in \Omega_m)}$$

$$\approx \frac{\xi_m}{\mathrm{P}(Q_c \in \Omega_m)} f_D^{(\lambda)}(d) \int_{\Omega_m} f_{Q_2|D}^{(\lambda)}(q|d)dq$$
$$= \frac{\xi_m}{\mathrm{P}(Q_c \in \Omega_m)} f_D^{(\lambda)}(d) \mathrm{P}\Big(Q_2^{(\lambda)} \in \Omega_m | D^{(\lambda)} = d\Big). \quad (68)$$

And similarly as for (62), it follows that

$$f_{Q_c|D_c,\Omega_m}(q|d) \approx \frac{\xi_m \cdot f_\chi^{(d)}\left(\frac{q}{\sigma^2/2}\right)}{\mathrm{P}\Big(Q_2^{(\lambda)} \in \Omega_m | D^{(\lambda)} = d\Big)} \quad \forall q \in \Omega_m. \quad (69)$$

Comparing (68) and (69) to (60) and (62), we see (by the normalization property of probability) that, under a small bin width, $f_{D_c|\Omega_m}(d) \approx f_{D|\Omega_m}(d)$ and $f_{Q_c|D_c,\Omega_m}(q|d) \approx f_{Q_2|D,\Omega_m}(q|d)$, with l_m in place of l.

REFERENCES

[1] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Netw.*, vol. 34, no. 3, pp. 134–142, May/Jun. 2020.

[2] "Remaining issues on URLLC data channel coding," InterDigital Inc., Wilmington, DE, USA, 3GPP document R1-1804849, Apr. 2018.

[3] P. Schulz et al., "Latency critical IoT applications in 5G: Perspective on the design of radio interface and network architecture," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 70–78, Feb. 2017.

[4] *Supplement to IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High-Speed Physical Layer in the 5 GHZ Band*, IEEE Standard 802.11a-1999, Sep. 1999.

[5] F. Khan, *LTE for 4G Mobile Broadband*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[6] *NR; Physical Channels and Modulation, Version 16.8.0.*, 3GPP Standard TS 38.211, Jan. 2022.

[7] *NR; Multiplexing and Channel Coding, Version 16.8.0*, 3GPP Standard TS 38.212, Jan. 2022.

[8] Y. Liao, M. Qiu, and J. Yuan, "Design and analysis of delayed bit-interleaved coded modulation with LDPC codes," *IEEE Trans. Commun.*, vol. 69, no. 6, pp. 3556–3571, Jun. 2021.

[9] Y. Liao, M. Qiu, and J. Yuan, "Windowed decoding for delayed bit-interleaved coded modulation," *IEEE Commun. Lett.*, vol. 25, no. 11, pp. 3483–3487, Nov. 2021.

[10] M. C. Jeruchim, P. Balaban, and K. S. Shanmugan, *Simulation of Communication Systems*, 2nd ed. New York, NY, USA: Kluwer Acad., 2002.

[11] T. Richardson, "Error floors of LDPC codes," in *Proc. Annu. Allerton Conf. Commun. Control Comput.*, Oct. 2003, pp. 1426–1435.

[12] B. K. Butler and P. H. Siegel, "Error floor approximation for LDPC codes in the AWGN channel," *IEEE Trans. Inf. Theory*, vol. 60, no. 12, pp. 7416–7441, Dec. 2014.

[13] D. Lu and K. Yao, "Improved importance sampling technique for efficient simulation of digital communication systems," *IEEE J. Sel. Areas Commun.*, vol. 6, no. 1, pp. 67–75, Jan. 1988.

[14] J. S. Sadowsky and J. A. Bucklew, "On large deviations theory and asymptotically efficient Monte Carlo estimation," *IEEE Trans. Inf. Theory*, vol. 36, no. 3, pp. 579–588, May 1990.

[15] P. J. Smith, M. Shafi, and H. Gao, "Quick simulation: A review of importance sampling techniques in communications systems," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 4, pp. 597–613, May 1997.

[16] C. A. Cole, S. G. Wilson, E. K. Hall, and T. R. Giallorenzi, "A general method for finding low error rates of LDPC codes," May 2006, *arXiv:cs/0605051*.

[17] E. Cavus, C. L. Haymes, and B. Daneshrad, "Low BER performance estimation of LDPC codes via application of importance sampling to trapping sets," *IEEE Trans. Commun.*, vol. 57, no. 7, pp. 1886–1888, Jul. 2009.

[18] S. Ahn, K. Yang, and D. Har, "Evaluation of the low error-rate performance of LDPC codes over rayleigh fading channels using importance sampling," *IEEE Trans. Commun.*, vol. 61, no. 6, pp. 2166–2177, Jun. 2013.

[19] M. Zhu, M. Jiang, and C. Zhao, "Error floor estimation of QC-LDPC coded modulation with importance sampling," *IEEE Commun. Lett.*, vol. 25, no. 1, pp. 28–32, Jan. 2021.

[20] C.-C. Wang, S. R. Kulkarni, and H. V. Poor, "Finding all small error-prone substructures in LDPC codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 1976–1999, May 2009.

[21] J. S. Stadler and S. Roy, "Adaptive importance sampling," *IEEE J. Sel. Areas Commun.*, vol. 11, no. 3, pp. 309–316, Apr. 1993.

[22] W. A. Al-Qaq, M. Devetsikiotis, and J.-K. Townsend, "Stochastic gradient optimization of importance sampling for the efficient simulation of digital communication systems," *IEEE Trans. Commun.*, vol. 43, no. 12, pp. 2975–2985, Dec. 1995.

[23] J. A. Bucklew and R. Radeke, "On the Monte Carlo simulation of digital communication systems in Gaussian noise," *IEEE Trans. Commun.*, vol. 51, no. 2, pp. 267–274, Feb. 2003.

[24] J. A. Bucklew, S. Nitinawarat, and J. Wierer, "Universal simulation distributions," *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2674–2685, Nov. 2004.

[25] A. Minja and V. Šenk, "Quasi-analytical simulation method for estimating the error probability of star domain decoders," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3101–3113, May 2019.

[26] R. Holzlöhner, A. Mahadevan, C. R. Menyuk, J. M. Morris, and J. Zweck, "Evaluation of the very low BER of FEC codes using dual adaptive importance sampling," *IEEE Commun. Lett.*, vol. 9, no. 2, pp. 163–165, Feb. 2005.

[27] S. Kakakhail, S. Reynal, D. Declercq, and V. Heinrich, "Fast simulation for the performance evaluation of LDPC codes using fast flat histogram method," in *Proc. IEEE Sarnoff Symp.*, Apr. 2008, pp. 1–5.

[28] A. Mahadevan and J. M. Morris, "SNR-invariant importance sampling for hard-decision decoding performance of linear block codes," *IEEE Trans. Commun.*, vol. 55, no. 1, pp. 100–111, Jan. 2007.

[29] D. Liang, W. Liu, M. Peng, and W. Wang, "Monte Carlo simulation with error classification for QAM modulation under Rayleigh fading channel," in *Proc. Int. Conf. Wireless Commun. Netw. Mobile Comput.*, Sep. 2009, pp. 1–4.

[30] Y.-Z. Yu and D. W. Lin, "Fast simulation of ultra-reliable coded communication system via adaptive shaping of noise histogram," in *Proc. IEEE Veh. Technol. Conf.*, May 2020, pp. 1–5.

[31] Y.-Z. Yu, D. W. Lin, and T.-H. Sang, "Fast simulation of convolutionally coded communication system for performance evaluation with a novel noise gauging method," in *Proc. IEEE Veh. Technol. Conf.*, Apr. 2021, pp. 1–5.

[32] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Upper Saddle River, NJ, USA: Pearson, 2004.

[33] J. M. Hammersky and D. C. Handscomb, *Monte Carlo Methods*. New York, NY, USA: Chapman Hall, 1964.

[34] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Trans. Inf. Theory*, vol. 43, no. 6, pp. 1757–1766, Nov. 1997.

[35] B. H. Lindqvist and G. Taraldsen, "Monte Carlo conditioning on a sufficient statistic," *Biometrika*, vol. 92, no. 2, pp. 451–464, Jun. 2005.

[36] S. M. Kay, *Fundamentals of Statistical Signal Processing—Vol. 1: Estimation Theory*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993.

[37] A. Barg and G. D. Forney, "Random codes: Minimum distances and error exponents," *IEEE Trans. Inf. Theory*, vol. 48, no. 9, pp. 2568–2573, Sep. 2002.

[38] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, no. 6, pp. 1087–1092, Jun. 1953.

[39] S.-K. Au and J. L. Beck, "Estimation of small failure probabilities in high dimensions by subset simulation," *Probab. Eng. Mech.*, vol. 16, no. 4, pp. 263–277, Oct. 2001.

[40] P. Frenger, P. Orten, and T. Ottosson, "Convolutional codes with optimum distance spectrum," *IEEE Commun. Lett.*, vol. 3, no. 11, pp. 317–319, Nov. 1999.
[41] *IEEE Standard for Air Interface for Broadband Wireless Access Systems*, IEEE Standard 802.16-2012, Jun. 2012.
[42] B. Moision, "A truncation depth rule of thumb for convolutional codes," in *Proc. Inf. Theory Appl. Workshop*, Jan./Feb. 2008, pp. 555–557.
[43] R. Y. Shao, S. Lin, and M. P. C. Fossorier, "Two decoding algorithms for tailbiting codes," *IEEE Trans. Commun.*, vol. 51, no. 10, pp. 1658–1665, Oct. 2003.
[44] Y.-P. E. Wang and R. Ramésh, "To bite or not to bite—A study of tail bits versus tail-biting," in *Proc. Int. Symp. Pers. Indoor Mobile Radio Commun.*, vol. 2, Oct. 1996, pp. 317–321.

YOU-ZONG YU (Graduate Student Member, IEEE) received the B.S. degree in electronics engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2016. He is currently pursuing the Ph.D. degree with the Graduate Degree Program, College of Electrical and Computer Engineering, National Yang Ming Chiao Tung University, Hsinchu. His research interests include the performance analysis of digital communication systems, with emphasis on importance sampling techniques for error rate evaluation of channel-coded transmission.

DAVID W. LIN (Life Senior Member, IEEE) received the B.S. degree in electronics engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1975, and the M.S. and Ph.D. degrees in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 1979 and 1981, respectively.

He was with Bell Laboratories from 1981 to 1983 and with Bellcore from 1984 to 1990 and from 1993 to 1994. Since 1990, he has been a Professor (currently Emeritus) with the Department of Electronics Engineering (currently Department of Electrical Engineering) and Institute of Electronics, National Chiao Tung University (currently National Yang Ming Chiao Tung University), except for a leave from 1993 to 1994. He has conducted research in digital adaptive filtering and telephone echo cancellation, digital subscriber line and coaxial network transmission, speech and video coding, and wireless communication. His research interests include various topics in signal processing and communication engineering.

TZU-HSIEN SANG (Member, IEEE) received the B.S.E.E. degree from National Taiwan University in 1990 and the Ph.D. degree from the University of Michigan at Ann Arbor in 1999. He is currently with the Department of Electronics Engineering, National Yang Ming Chiao Tung University (NCTU), Taiwan. Prior to joining NCTU in 2003, he had worked in Excess Bandwidth, a start-up company at Sunnyvale, CA, USA, working on physical layer design for broadband technologies. His research interests include signal processing and data-driven machine learning for applications in sensor fusion and wireless communications.