Finite-Length Scaling of SC-LDPC Codes With a Limited Number of Decoding Iterations

Finite-Length Scaling of SC-LDPC Codes With a Limited Number of Decoding Iterations

Roman Sokolovskii, *Graduate Student Member, IEEE*, Alexandre Graell i Amat, *Senior Member, IEEE*, and Fredrik Brännström

***Abstract*—We propose four finite-length scaling laws to predict the frame error rate (FER) performance in the waterfall region of spatially-coupled low-density parity-check code ensembles under full belief propagation (BP) decoding with a limit on the number of decoding iterations and a scaling law for sliding window decoding, also with limited iterations. The laws for full BP decoding provide a choice between accuracy and computational complexity; a good balance between them is achieved by the law that models the number of decoded bits after a certain number of BP iterations by a time-integrated Ornstein-Uhlenbeck process. This framework is developed further to model sliding window decoding as a race between the integrated Ornstein-Uhlenbeck process and an absorbing barrier that corresponds to the left boundary of the sliding window. The proposed scaling laws yield accurate FER predictions for the semi-structured code ensembles proposed by Olmos and Urbanke.**

***Index Terms*—Codes-on-graphs, finite-length code performance, spatially-coupled LDPC codes, window decoding.**

I. Introduction

Spatially-coupled low-density parity-check (SC-LDPC) codes [1], [2] achieve capacity under suboptimal yet computationally feasible belief propagation (BP) decoding, which was first observed numerically [2], then proved for the binary erasure channel (BEC) [3] and later for the broad class of binary-input memoryless symmetric channels [4]. Moreover, it was shown that the minimum distance of regular SC-LDPC code ensembles grows linearly with the block length [5]. Spatial coupling is also a powerful technique more broadly; it has been applied to, e.g., turbo-like codes [6], product-like codes [7], lossy compression [8], and compressed sensing [9].

The Tanner graph of an SC-LDPC code is constructed by arranging the Tanner graphs of several uncoupled LDPC codes into a sequence and interconnecting them according to a predefined pattern. The interconnecting is done in such a way so as to create *structured irregularity* at the boundaries of the resulting chain, such that the bits at the boundaries are better protected than those in the middle of the chain and are more likely to be decoded successfully; during BP decoding, information propagates from those boundaries inward in a wave-like fashion.

The structured irregularity at the boundaries entails a loss in code rate. The rate loss goes to zero as the chain length grows, but so does the probability that the decoding waves successfully propagate through the entire chain. A longer chain also implies higher latency if BP decoding is performed on the whole received sequence simultaneously, a scheme we refer to as *full BP decoding*. To limit decoding latency, so-called *sliding window decoding,* originally proposed in [10], is used in practice. Sliding window decoding limits BP decoding to a window of several spatial positions that slides through the chain from the left boundary rightward, making a decision on the bits that leave the window along the way and thus limiting decoding latency to the size of the window. Further, due to constraints on decoding latency and energy efficiency, both full BP and sliding window decoding must in practice be limited in terms of the maximum number of BP iterations.

To adopt an SC-LDPC code in a practical setting, the system designer would have to specify a range of parameters, including the underlying LDPC code ensemble, the interconnecting pattern, the length of the coupled chain, the size of the sliding window, and the limit on the number of decoding iterations. It is therefore important to understand the influence of these parameters on error-correcting performance, which is the subject of ongoing scientific inquiry.

The asymptotic performance of SC-LDPC codes (i.e., when the block length of the component LDPC codes grows large) is relatively well understood. Much less is known about their finite-length behavior. Olmos and Urbanke proposed a finite-length scaling law to predict the frame error rate (FER) of SC-LDPC code ensembles under full BP decoding with an unlimited number of iterations over the BEC [11]. The law in [11] follows the approach proposed earlier in [12] for uncoupled LDPC code ensembles and focuses on the number of degree-one check nodes (CNs) available for peeling decoding. Peeling decoding is equivalent in error-correcting performance to BP decoding with unlimited iterations but is more tractable analytically. A decoding failure corresponds to the peeling decoder running out of degree-one CNs before recovering the entire codeword. The authors estimate the probability of that event using an exponential approximation to the first-hit time distribution of an appropriately chosen Ornstein-Uhlenbeck process [11]. This framework was later applied to protograph-based [13] and generalized [14] SC-LDPC code ensembles.

The prediction in [11] captures the slope of the FER curve well. However, it exhibits a gap to the simulated FER performance. In [15], we closed this gap by proposing an alternative scaling law that models the number of degree-one CNs available for peeling decoding as a sum of two independent Ornstein-Uhlenbeck processes that correspond to the decoding waves from the left and right boundary of the chain. We provided laws to also predict the bit and block error rate performance. Importantly for practical applications,

This work was funded by the Swedish Research Council (grant 2016-4026).

R. Sokolovskii, A. Graell i Amat, and F. Brännström are with the Communication Systems Group, Department of Electrical Engineering, Chalmers University of Technology, SE-41296 Gothenburg, Sweden (email: {roman.sokolovskii,alexandre.graell,fredrik.brannstrom}@chalmers.se).

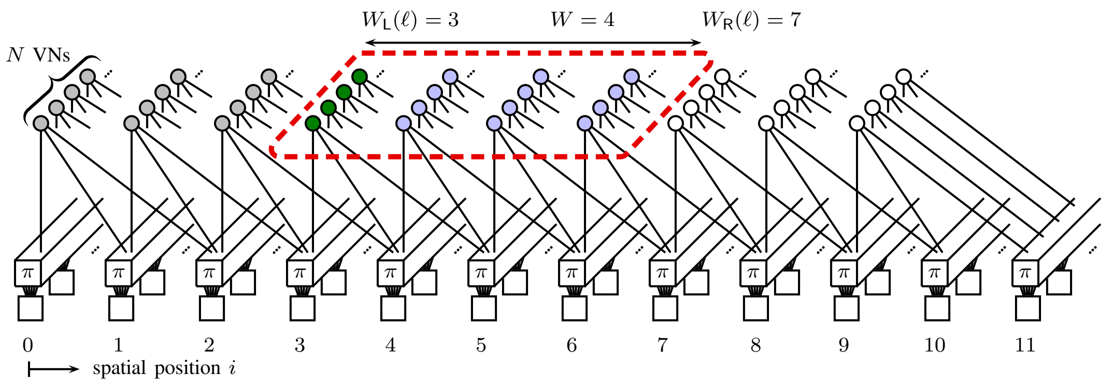

Fig. 1. Tanner graph of the terminated $(d_{\mathsf{v}}, d_{\mathsf{c}}, L, N)$ SC-LDPC ensemble with $d_{\mathsf{v}} = 3, d_{\mathsf{c}} = 6$, $L = 10$, and N VNs per spatial position. The rectangular cuboids marked with π are permutation blocks. The sliding window of length $W = 4$ is shown as the red dashed frame. The decision on the VNs in the first 3 spatial positions (gray) has already been made; the associated values are fixed. The VNs in position $W_{\mathsf{L}}(\ell) = 3$ (green) are next to be decided upon and fixed. The VNs in the next three positions (blue) participate in message passing, while those in positions $W_{\mathsf{R}}(\ell) = 7$ and above (white) do not.

we proposed scaling laws for the frame, bit, and block error rate under sliding window decoding, albeit still in the case of unlimited number of iterations.

The scaling law for the FER under full BP decoding from [15] was used in [16] to elucidate the non-trivial space of trade-offs associated with the choice of code parameters and extended in [17] to predict the error rate performance of periodically-doped SC-LDPC code ensembles for streaming. In [18], the authors optimize protograph-based SC-LDPC code constructions under sliding window decoding using a criterion derived from finite-length scaling models (the so-called window mean parameter); they show that taking into account the finite-length scaling behavior of SC-LDPC code ensembles during code optimization can yield codes that significantly outperform their counterparts designed using asymptotic BP thresholds only.

Our ambition is to make finite-length scaling laws the tool of choice in such code parameter optimization. To that end, this paper tackles the problem of predicting the FER of SC-LDPC code ensembles over the BEC under both full BP and sliding window decoding for the practical case of *a limited number of iterations*. First, we consider full BP decoding and propose four scaling laws that vary in accuracy and computational complexity. One of these laws models the number of bits recovered after a certain number of BP iterations by a time-integrated Ornstein-Uhlenbeck process. We then extend this framework and model sliding window decoding as a race between an absorbing barrier that corresponds to the left boundary of the window and an integrated Ornstein-Uhlenbeck process (with an additional diffusion term) that corresponds to the position of the left decoding wave. We estimate the probability of the wave being absorbed at the barrier (and thus overtaken by the sliding window) by numerically solving the initial value problem of the corresponding Fokker-Planck equation with appropriately chosen boundary conditions. The proposed laws yield accurate predictions of the FER and allow us to quantify performance degradation associated with the introduction of the limit on the number of decoding iterations, making practical code parameter optimization using finite-length scaling laws a step closer to reality.

II. Preliminaries

We consider the semi-structured $(d_{\mathsf{v}}, d_{\mathsf{c}}, L, N)$ SC-LDPC code ensemble introduced in [11]. Its Tanner graph is shown in Fig. 1. To construct the Tanner graph of an element of the $(d_{\mathsf{v}}, d_{\mathsf{c}}, L, N)$ SC-LDPC code ensemble, one must first take L Tanner graphs of length-N $(d_{\mathsf{v}}, d_{\mathsf{c}})$-regular LDPC codes of variable node (VN) degree d_{v} and CN degree d_{c} and arrange them into L spatial positions indexed by $i \in \mathcal{L} = \{0, \ldots, L-1\}$. Each spatial position contains N VNs and $M = \frac{d_{\mathsf{v}}}{d_{\mathsf{c}}} N$ CNs, where we assume M is an integer. We refer to N as the *component code length* and to L as the *chain length*. The set of all LN VNs in the Tanner graph—and thus of all LN code bits—is referred to as the *frame*. The L Tanner graphs of the individual (uncoupled) LDPC codes are then interconnected as follows: each VN at position $i \in \mathcal{L}$ is connected to d_{v} CNs in positions $[i, \ldots, i+d_{\mathsf{v}}-1]$. Specifically, one CN is chosen uniformly at random among M CNs at position i, another one at position $i+1$, and so on until position $i + d_{\mathsf{v}} - 1$. To connect the overhanging edges at the end of the chain, $d_{\mathsf{v}} - 1$ additional positions that contain CNs only are appended, resulting in a *terminated* ensemble. The generation of the elements from this ensemble is described in detail in [11] and can be expressed in terms of choosing the $L + d_{\mathsf{v}} - 1$ permutation blocks in the Tanner graph in Fig. 1, marked by π.

The ensemble is structured from the VN perspective—it is certain that each VN is connected to CNs in d_{v} different spatial positions. The same cannot be said about the CNs. Indeed, a CN at position $i \in \{d_{\mathsf{v}}-1, \ldots, L-1\}$ can be connected to d_{c} VNs from any non-empty subset of positions $[i-d_{\mathsf{v}}+1, \ldots, i]$, depending on how the permutation block at position i reshuffles the edges connected to VNs at positions $[i-d_{\mathsf{v}}+1, \ldots, i]$ (see Fig. 1). This particular "semi-structured" ensemble is proposed in [11] to simplify the analysis. The same approach was later applied to protograph-based ensembles [13].

In addition to the terminated ensemble, we also consider the *truncated* and *unterminated* ensembles. In the truncated ensemble, the additional $d_{\mathsf{v}}-1$ positions with CNs only are not added at the end of the chain of length L, and the overhanging edges emanating from VNs at positions $[L-d_{\mathsf{v}}+1, \ldots, L-1]$ are simply deleted from the Tanner graph. The degree of VNs in these last $d_{\mathsf{v}}-1$ positions is therefore reduced. In the unterminated ensemble, the chain is neither terminated nor truncated, resulting in a “semi-infinite” sequence of coupled codes. We introduce the unterminated ensemble for the analysis of sliding window decoding, and we evaluate the decoding error probability over the first L' positions of this semi-infinite chain. Both the truncated and the unterminated ensemble have higher code rates than the terminated ensemble; however, they are not relevant as candidates for practical code constructions and are instead introduced here for analytical purposes.

Key to impressive error-correcting performance of SC-LDPC codes under BP decoding is the lower degree of CNs at the terminated boundaries of the chain, i.e., at the left boundary of the truncated and unterminated ensembles and at both left and right boundaries of the terminated ensemble. During BP decoding, information propagates from the terminated boundaries of the chain inward in a wave-like fashion. The finite-length scaling laws aim to estimate the probability that such “decoding waves” fail to propagate through the entire chain, which results in decoding error.

Full BP decoding entails a decoding latency of LN bits, which is impractical for long spatially-coupled chains. To limit decoding latency, sliding window decoding is used in practice, where decoding is limited to VNs in a window of W spatial positions (depicted as the red dashed rectangle in Fig. 1). After a certain number of BP iterations, the decoder decides on the values of the bits in the left-most spatial position within the window (colored green in Fig. 1) and the window *slides* by one position to the right over the Tanner graph. We index BP decoding iterations by ℓ and denote the leftmost position of the window by $W_{\mathsf{L}}(\ell)$. The first position just outside the window is denoted by $W_{\mathsf{R}}(\ell)$, as illustrated in Fig. 1. Sliding window decoding has a decoding latency of WN bits [10].

This paper investigates the influence of the limit on the number of BP iterations on the error-correcting performance. In the case of full BP decoding, we denote this limit by I and treat it is as a parameter of the decoder. In the case of sliding window decoding, the system designer should choose two parameters instead: First, the number of BP iterations after which the window slides for the first time, from position 0 to position 1, i.e., $W_{\mathsf{L}}(\ell)=1$, which we denote by I_{in} and whose impact will be clarified later. Second, for $W_{\mathsf{L}}(\ell)\geq 1$, the designer must specify the number of BP iterations before the window slides further, which we denote by I_{s}. The total budget of BP iterations in the case of sliding window decoding can be obtained from I_{in} and I_{s} for a chain of length L as

$$I = I_{\mathsf{in}} + (L-1)I_{\mathsf{s}}\,. \tag{1}$$

We consider transmission over the BEC with erasure probability ϵ.

A. Density Evolution

Let $q^{(\ell)}_{i+j,i}$ denote the probability that a CN at position $i+j$ sends an erasure message to a VN at position i at BP iteration ℓ. Likewise, let $p^{(\ell)}_{i}$ be the probability that a VN at position i is erased, and $p^{(\ell)}_{i,i+j}$ the probability that a VN at position i sends an erasure message to a CN at position $i+j$ at BP iteration ℓ. The CN update for the semi-structured ensemble averages the incoming error probabilities as

$$q^{(\ell)}_{i+j,i} = 1-\left(1-\frac{1}{d_{\mathsf{v}}}\sum_{j'=0}^{d_{\mathsf{v}}-1} p^{(\ell-1)}_{i+j-j',i+j}\right)^{d_{\mathsf{c}}-1}\,. \tag{2}$$

To circumvent the problem of the reduced-degree CNs at the boundaries, the values of $p^{(\ell-1)}_{i+j-j',i+j}$ that correspond to VN indices $i+j-j'$ outside the chain—i.e., when $i+j-j'\notin\mathcal{L}$—are set to zero, implying that the VNs outside the chain are not erased.

Since a VN at position $i\in\mathcal{L}$ is connected to d_{v} consecutive positions $\{i,\dots,i+d_{\mathsf{v}}-1\}$, as we discussed in the beginning of Section II, the VN update for the semi-structured ensemble is

$$p^{(\ell)}_{i,i+j} = \epsilon\prod_{j'\neq j} q^{(\ell)}_{i+j',i}\,, \tag{3}$$

and the *a posteriori* probability that a VN at position i remains erased at iteration ℓ is

$$p^{(\ell)}_{i} = \epsilon\prod_{j=0}^{d_{\mathsf{v}}-1} q^{(\ell)}_{i+j,i}\,. \tag{4}$$

To numerically estimate the BP decoding threshold ϵ^* for a given $(d_{\mathsf{v}},d_{\mathsf{c}},L,N)$ SC-LDPC code ensemble, we initialize $p^{(0)}_{i,i+j}=1$ for all $i\in\mathcal{L}, j\in\{0,\dots,d_{\mathsf{v}}-1\}$ and iterate equations (2)–(3) until the VN erasure probability (4) converges either to zero (for $\epsilon\leq\epsilon^*$) or to another fixed point (for $\epsilon>\epsilon^*$) for all i.

B. Peeling Decoding

The finite-length analysis of BP decoding for the BEC becomes more tractable by considering *peeling decoding* [19], which is equivalent to BP decoding for an infinite number of iterations. The peeling decoder gets stuck in the same stopping sets and therefore yields the same performance as the BP decoder [19]. At the initial stage of peeling decoding, all VNs that correspond to non-erased bits are removed from the Tanner graph. At every subsequent iteration, the decoder selects one degree-one CN uniformly at random among all degree-one CNs in the graph. Since the value of the code bit associated with the VN connected to the chosen degree-one CN can be recovered, the decoder removes both the CN and VN from the Tanner graph along with d_{v} adjacent edges and modifies the parity-check equations associated with the adjacent CNs according to the value of the recovered bit. This may in turn create new degree-one CNs to choose from (or remove some other degree-one CNs collaterally). Each iteration of peeling decoding produces a new *residual* graph, indexed by iteration number ℓ_{PD}. Decoding is successful if eventually the decoder manages to peel off all VNs from the original Tanner graph, resulting in an empty graph. This happens if at every iteration of peeling decoding there is at least one degree-one CN to choose from. On the other hand, if the decoder runs out of degree-one CNs before recovering all VNs, decoding gets trapped in a stopping set and fails.

The goal of scaling laws for LDPC codes in [12] and for SC-LDPC codes in [11], [15] is to estimate the error rate

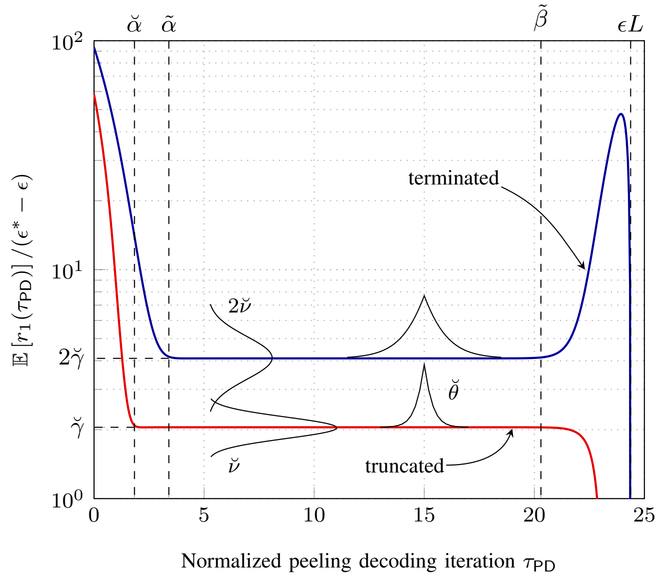

Fig. 2. The evolution of $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$ during peeling decoding, normalized by the distance to the BP threshold, for the $(5, 10, L=50)$ ensemble with $\epsilon^*=0.4994$ at $\epsilon=0.4875$.

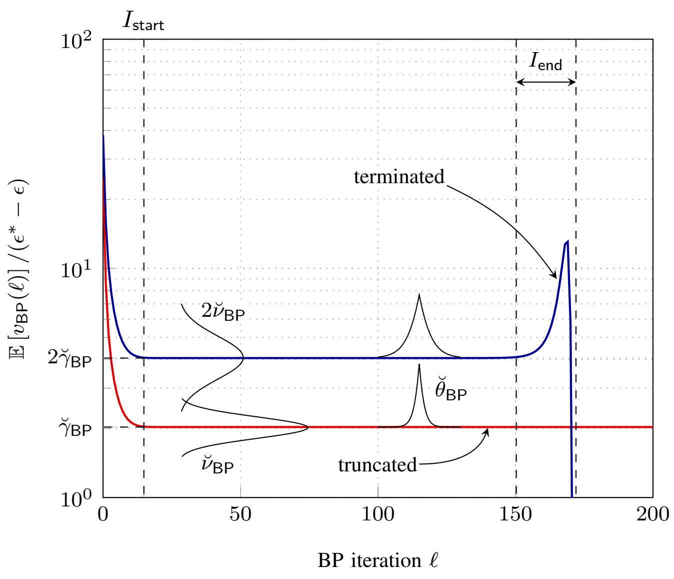

Fig. 3. The evolution of $\mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right]$ during BP decoding, normalized by the distance to the BP threshold, for the $(5, 10, L=50)$ ensemble with $\epsilon^*=0.4994$ at $\epsilon=0.47$.

in the waterfall region, which is dominated by the stopping sets whose size grows linearly with the block length N [12]. A scaling law therefore aims to estimate the probability of the event that some VNs remain in the residual graph when decoding stops, focusing on "large" stopping sets whose size grows linearly with N.

We remark that the finite-length optimization of SC-LDPC code and decoder parameters must consider their effect on both the waterfall and the error-floor region. In the error-floor region—in contrast to the waterfall region—the error probability is dominated by small harmful substructures in the Tanner graph known as stopping and trapping or absorbing sets. These structures have been investigated in the context of SC-LDPC codes in [20]–[22] and used to estimate the error floor for SC-LDPC codes in [23]. Such error-floor analysis complements the waterfall-oriented scaling-law approach developed in [11], [12], [15], and in this paper.

C. The Scaling Laws for Unlimited Number of Iterations

The general approach to finite-length scaling of SC-LDPC code ensembles originally proposed in [11] for full BP decoding and improved and extended to sliding-window decoding in [15] is to focus on the stochastic process associated with the number of degree-one CNs in the residual graphs during peeling decoding normalized by N [24],

$$r_1(\tau_{\mathsf{PD}}) = \frac{1}{N}\sum_{u} R_{1,u}(\tau_{\mathsf{PD}})\,, \tag{5}$$

where $\tau_{\mathsf{PD}} = \ell_{\mathsf{PD}}/N$ is the normalized time of peeling decoding, and $R_{1,u}(\tau_{\mathsf{PD}})$ is the number of degree-one CNs at position u at iteration ℓ_{PD}. Since peeling decoding requires at least one degree-one CN at every iteration, a decoding error occurs if $r_1(\tau_{\mathsf{PD}})$ hits zero before recovering all VNs that are erased by the channel. We refer to a realization of $r_1(\tau_{\mathsf{PD}})$ as a *decoding trajectory*.

For a fixed τ_{PD}, the distribution of $r_1(\tau_{\mathsf{PD}})$ converges to a Gaussian as $N\to\infty$ and concentrates around its mean $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$ with expectation taken over the ensemble, channel, and peeling decoding realizations [11], [12]. The evolution of $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$ over τ_{PD} can be obtained by numerically solving a system of coupled differential equations called *mean evolution*. Mean evolution equations for the semi-structured $(d_{\mathsf{v}}, d_{\mathsf{c}}, L, N)$ SC-LDPC code ensemble are provided in [11]. For illustration, Fig. 2 shows the mean evolution curves $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$ normalized by $\epsilon^*-\epsilon$ for the terminated (blue curve) and truncated (red curve) $(5, 10, L=50, N)$ SC-LDPC code ensemble at $\epsilon=0.4875$.

Notably, $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$ exhibits a steady-state phase where it remains essentially constant [11]. We denote the range of τ_{PD} corresponding to the steady state of the terminated ensemble as $\left[\tilde{\alpha}, \tilde{\beta}\right]$. Here and in the following, we denote the variables associated with the terminated ensemble with a tilde, e.g., $\tilde{\alpha}$, and those associated with the truncated ensemble with a breve, e.g., $\breve{\alpha}$—the two "extrema" in a tilde allude to the presence of two decoding waves in the terminated ensemble, and a single "extremum" in a breve to a single wave in the truncated ensemble.

During the steady state, the two waves propagating in the terminated chain are each equivalent to the single wave present in the truncated chain. In [15], we proposed to isolate a single wave by focusing on the truncated chain and modeling the first two moments of $r_1(\tau_{\mathsf{PD}})$ in the same way as is done for the terminated chain in [11], namely

$$\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right] \approx \breve{\gamma}\left(\epsilon^*-\epsilon\right)\,, \tag{6}$$

$$\mathrm{Var}\left[r_1(\tau_{\mathsf{PD}})\right] \approx \frac{\breve{\nu}}{N}\,, \tag{7}$$

$$\mathrm{Cov}\left[r_1\left(\tau_{\mathsf{PD}}^{(0)}\right), r_1\left(\tau_{\mathsf{PD}}^{(1)}\right)\right] \approx \frac{\breve{\nu}}{N}\exp\left(-\breve{\theta}\left|\tau_{\mathsf{PD}}^{(0)}-\tau_{\mathsf{PD}}^{(1)}\right|\right)\,, \tag{8}$$

for a triple of real positive numbers $(\breve{\gamma}, \breve{\nu}, \breve{\theta})$.

Apart from the BP decoding threshold ϵ^*, the scaling law for unlimited number of BP iterations requires the five parameters $(\tilde{\alpha}, \tilde{\beta}, \breve{\gamma}, \breve{\nu}, \breve{\theta})$. The first three can be estimated by numerically solving mean evolution equations. We estimate them for several ϵ and linearly interpolate the values in between. The last two can be estimated by solving an augmented system of differential equations called *covariance evolution* [11]. Instead, we estimate $(\breve{\nu}, \breve{\theta})$ from a set of realizations of $r_1(\tau_{\mathsf{PD}})$ for a fixed (ϵ, N) and treat them as parameters that depend on $(d_{\mathsf{v}}, d_{\mathsf{c}})$ only. For our running example of the $(5, 10, L, N)$ SC-LDPC code ensemble, we use $\breve{\nu} = 0.424$ and $\breve{\theta} = 1.64$ as in [15]. The meaning of the scaling parameters for peeling decoding is illustrated in Fig. 2.

The key idea behind the scaling laws in [11], [15] is to model $r_1(\tau_{\mathsf{PD}})$ in the steady state by an Ornstein-Uhlenbeck process, parametrized to match the first two moments of $r_1(\tau_{\mathsf{PD}})$ (6)–(8). Olmos and Urbanke model the steady state of the terminated ensemble by a single Ornstein-Uhlenbeck process [11]; we proposed instead a refined model that relies on two independent Ornstein-Uhlenbeck processes and yields more accurate predictions [15]. We summarize the model in [15] here and use it as the starting point for our analysis of decoding with a limited number of iterations.

To estimate the FER, we consider the normalized time of peeling decoding at which the number of degree-one CNs—and hence the value of $r_1(\tau_{\mathsf{PD}})$—drops to zero, referred to as the first-hit time τ_0,

$$\tau_0 = \min\{\tau_{\mathsf{PD}} : r_1(\tau_{\mathsf{PD}}) = 0\}\,. \tag{9}$$

For the terminated ensemble, to exhaust all available degree-one CNs, the peeling decoder must run out of them in both the left and right decoding wave. Accordingly, we model τ_0 as the sum of two independent variables A and B that correspond to the first-hit time of the left and right wave, respectively [15], as

$$\tau_0 = A + B\,. \tag{10}$$

The number of degree-one CNs available to each wave is modeled as an independent Ornstein-Uhlenbeck process. It is known that the first-hit time distribution for an Ornstein-Uhlenbeck process converges to an exponential distribution with mean $\breve{\mu}_0$ as $N \to \infty$ [11],

$$A, B \sim \mathrm{Exp}(\breve{\mu}_0)\,, \tag{11}$$

where $\breve{\mu}_0 = \mu_0(\breve{\gamma}, \breve{\nu}, \breve{\theta})$ and

$$\mu_0(\gamma, \nu, \theta) = \frac{\sqrt{2\pi}}{\theta} \int_0^{\gamma\sqrt{N/\nu}(\epsilon^* - \epsilon)} \Phi(z) \mathrm{e}^{\frac{1}{2}z^2} \mathrm{d}z\,, \tag{12}$$

with $\Phi(z)$ denoting the CDF of the standard Gaussian distribution. The PDF of the number of degree-one CNs recovered by the left and right wave during the steady state is

$$f_A(x) = f_B(x) \approx \breve{\mu}_0^{-1} \exp\left(-\frac{x}{\breve{\mu}_0}\right)\,. \tag{13}$$

Let S denote successful decoding—specifically, the event that a codeword from a randomly sampled element from the code ensemble is fully recovered by the employed decoder after transmission over the BEC with erasure probability ϵ. Since the total number of degree-one CNs that must be recovered during the steady state of the terminated ensemble is $\tilde{\beta} - \tilde{\alpha}$, the probability of successful full BP decoding can be expressed as

$$\begin{aligned} \Pr\{S\} &= \Pr\left\{A + B > \tilde{\beta} - \tilde{\alpha}\right\} \\ &= \int_{\tilde{\beta}-\tilde{\alpha}}^{\infty} \int_0^{x} f_A(z) f_B(x - z) \mathrm{d}z \mathrm{d}x\,. \end{aligned} \tag{14}$$

The approximation for the FER in [15] is obtained from (14) and (13) as

$$P_{\mathsf{f,t}}^{(L)} \approx 1 - \left(1 + \frac{\tilde{\beta} - \tilde{\alpha}}{\breve{\mu}_0}\right) \exp\left(-\frac{\tilde{\beta} - \tilde{\alpha}}{\breve{\mu}_0}\right)\,. \tag{15}$$

In a similar manner, the FER of an unterminated SC-LDPC code ensemble evaluated over L' spatial positions is approximated in [15] as

$$P_{\mathsf{f,u}}^{(L')} \approx 1 - \exp\left(-\frac{\epsilon L' - \breve{\alpha}}{\breve{\mu}_0}\right)\,, \tag{16}$$

which is the same approximation used by Olmos and Urbanke for the *terminated* ensemble, but with the scaling parameters estimated from the truncated ensemble instead of from the terminated ensemble. In other words, the law in (16) estimates the probability that the decoding wave from the left boundary of the unterminated chain successfully propagates through the first L' positions.

The scaling law for the unterminated ensemble is used here and in [15] in the analysis of sliding window decoding, where the sliding window does not allow the right wave to propagate to the left by further than W positions, effectively limiting the first $L - W$ positions to decoding by a single wave only. As we showed in [15], this results in a "two-phase" decoding, where the first phase comprises the first $L - W$ positions and includes a single decoding wave from the left, and the second phase comprises the last W positions and may contain both the wave from the left and the wave from the right. The FER in the case of sliding window decoding with unlimited number of BP iterations can then be approximated as [15]

$$P_{\mathsf{f,t,sw}}^{(L,W)} = 1 - \left(1 - P_{\mathsf{f,u}}^{(L-W)}\right)\left(1 - P_{\mathsf{f,t}}^{(W)}\right)\,, \tag{17}$$

where $P_{\mathsf{f,t}}^{(\cdot)}$ is given in (15), and $P_{\mathsf{f,u}}^{(\cdot)}$ is given in (16).

Lastly, we will use the *speed* of a decoding wave in our analysis. Specifically, we assume that a wave traverses V_{PD} positions in N peeling decoding iterations. We estimate V_{PD} from the average number of erased VNs in the middle of the coupled chain during the steady state as

$$V_{\mathsf{PD}} \approx N \cdot \mathbb{E}\left[V_{\lfloor L/2 \rfloor}\left(\frac{\tilde{\beta} + \tilde{\alpha}}{2}\right)\right]^{-1}\,, \tag{18}$$

where $\mathbb{E}\left[V_u(\tau_{\mathsf{PD}})\right]$, the average number of erased VNs at position u at normalized iteration τ_{PD}, is produced alongside $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$ by numerically solving mean evolution [15].

For future reference, Table I provides the essential notation used throughout the paper. The table is split into three blocks. The first block covers the ensemble, channel, and decoder

TABLE I
ESSENTIAL NOTATION

Alias	Meaning
ϵ	BEC erasure probability
d_{v}	VN degree
d_{c}	CN degree
L	number of VN positions
N	number of bits (VNs) in each position
I	limit on the number of full BP iterations
W	size of the sliding window (in VN positions)
I_{in}	number of iterations before the sliding window slides for the first time
I_{s}	number of iterations in each subsequent sliding window position
V_{W}	$1/I_{\mathsf{s}}$, the speed of the sliding window
ℓ_{PD}	PD iteration number
$r_1(\tau_{\mathsf{PD}})$	the number of degree-one CNs in the Tanner graph, normalized by N, at peeling decoding iteration $\ell_{\mathsf{PD}} = N \cdot \tau_{\mathsf{PD}}$
$(\tilde{\alpha}, \tilde{\beta})$	boundaries of the steady state of $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$ for the terminated ensemble
$\breve{\gamma}$	steady-state level of $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]/(\epsilon^* - \epsilon)$ for the truncated ensemble
$\breve{\nu}$	variance constant for $r_1(\tau_{\mathsf{PD}})$ (truncated ensemble)
$\breve{\theta}$	covariance decay constant for $r_1(\tau_{\mathsf{PD}})$ (truncated ensemble)
V_{PD}	speed of the decoding wave during peeling decoding (truncated ensemble, normalized iterations)
ϵ^*	BP decoding threshold
ℓ	BP iteration number
$v_{\mathsf{BP}}(\ell)$	the number of bits recovered in BP iteration ℓ, normalized by N
I_{start}	the number of BP iterations before the onset of the steady state of $\mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right]$
I_{end}	the number of BP iterations for decoding to finish once the two waves "meet" (terminated ensemble)
$\breve{\gamma}_{\mathsf{BP}}$	steady-state level of $\mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right]/(\epsilon^* - \epsilon)$ for the truncated ensemble
$\breve{\nu}_{\mathsf{BP}}$	variance constant for $v_{\mathsf{BP}}(\ell)$ (truncated ensemble)
$\breve{\theta}$	covariance decay constant for $v_{\mathsf{BP}}(\ell)$ (truncated ensemble)
V_{BP}	speed of the decoding wave during BP decoding

parameters, and thus can be viewed as input to the scaling laws. The second and third blocks cover the notation used to describe the properties of the peeling and BP decoder, respectively.

We now proceed to develop the scaling laws for decoding with a *limited* number of iterations; first, for full BP decoding in Sections III–V, then for sliding window decoding in Section VI.

III. The Speed of the Decoding Waves and the Duration of the Steady State

Besides providing BP thresholds for SC-LDPC code ensembles, density evolution, described in Section II-A, can be used to answer the following questions: How many BP iterations are needed before the steady state begins and the decoding waves establish? How fast do decoding waves propagate under BP decoding? How many iterations does it take for the decoding waves to collapse once they meet at the end of the steady state?

Let $v_{\mathsf{BP},i}(\ell)$ be the fraction of code bits at position i recovered in BP iteration ℓ. The expected value of $v_{\mathsf{BP},i}(\ell)$ can be obtained from the decrease in the erasure probability across iterations of density evolution as

$$\mathbb{E}\left[v_{\mathsf{BP},i}(\ell)\right] = p_i^{(\ell-1)} - p_i^{(\ell)} \tag{19}$$

with $p_i^{(\ell)}$ from (4). Define $v_{\mathsf{BP}}(\ell) \triangleq \sum_{i\in\mathcal{L}} v_{\mathsf{BP},i}(\ell)$. The number of code bits recovered in iteration ℓ is then $N\cdot v_{\mathsf{BP}}(\ell)$, with $\mathbb{E}\left[N\cdot v_{\mathsf{BP}}(\ell)\right] = N \cdot \mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right]$, where

$$\mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right] = \sum_{i\in\mathcal{L}} \mathbb{E}\left[v_{\mathsf{BP},i}(\ell)\right] \ . \tag{20}$$

The evolution of $\mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right]/(\epsilon^* - \epsilon)$ through BP iterations is shown in Fig. 3 for the terminated (blue curve) and truncated (red curve) $(5, 10, L = 50, N)$ SC-LDPC code ensemble at $\epsilon = 0.47$. In contrast to the mean evolution curves (Fig. 2), the onset of the steady state happens at the same time for the truncated and terminated ensemble because BP decoding resolves all code bits corresponding to VNs connected to degree-one CNs in parallel.[1]

Denote the number of BP iterations before the onset of the decoding waves by I_{start} and the number of BP iterations it takes for the waves to collapse once they meet by I_{end}. The parameters I_{start} and I_{end} are illustrated in Fig. 3. We estimate $(I_{\mathsf{start}}, I_{\mathsf{end}})$ by tracking the change in $\mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right]/(\epsilon^* - \epsilon)$ across iterations and comparing it with a numerical threshold (the value of 10^{-2} is used; I_{start} corresponds to the first ℓ when the magnitude of the change becomes smaller than the threshold, and I_{end} corresponds to the number of iterations between the first ℓ when the change subsequently rises above the threshold and $\mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right]$ collapses to zero, see Fig. 3). Both I_{start} and I_{end} are estimated for several values of ϵ, and linear interpolation is used to estimate the intermediate values. A similar procedure was used in [15] to estimate $(\breve{\alpha}, \tilde{\alpha}, \tilde{\beta})$ for peeling decoding.

As introduced in Section II, we denote the limit on the total number of BP iterations by I. For decoding to be successful, the decoding waves must meet before the *effective deadline* of

$$I_{\mathsf{eff}} = I - I_{\mathsf{start}} - I_{\mathsf{end}} \tag{21}$$

iterations of BP decoding after the beginning of the steady state phase.

Denote the speed of the decoding wave under BP decoding by V_{BP}, measured in positions traveled by the wave per BP iteration. Fig. 4 shows V_{BP} as a function of ϵ, $V_{\mathsf{BP}}(\epsilon)$. The black dots represent the values estimated directly from density evolution by tracking the mid-point of the wave fronts of $p_i^{(\ell)}$ over density evolution iterations. Their apparent noisiness stems from the discretization of the estimated positions of the wave fronts. The red solid curve is a quadratic fit to these data; we use V_{BP} from the smoothed (red) curve in the numerical calculations.

There is an alternative way to estimate $V_{\mathsf{BP}}(\epsilon)$. Since a single iteration of BP decoding recovers all VNs connected to degree-one CNs, we assume that a single BP iteration during the steady state is equivalent to as many peeling decoding

[1]This interpretation of BP decoding for the BEC is made rigorous in [25] with the concept of *parallel* peeling decoding, which the authors prove to be equivalent to BP decoding on a per-iteration basis.

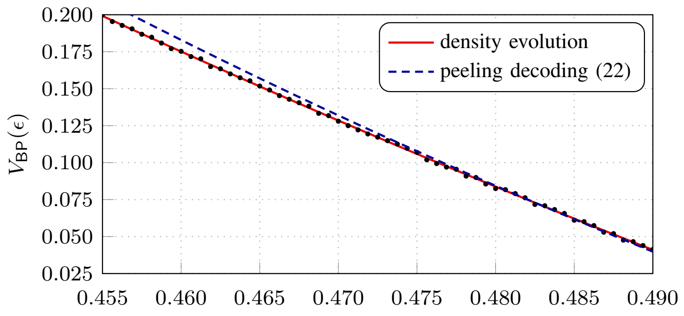

Fig. 4. The speed of the waves for the $(5, 10, L)$ SC-LDPC code ensemble under BP decoding.

iterations as there are degree-one CNs available for each of the waves. This assumption allows us to approximate $V_{\mathsf{BP}}(\epsilon)$ as

$$V_{\mathsf{BP}}(\epsilon) \approx V_{\mathsf{PD}}(\epsilon) \cdot \breve{\gamma}(\epsilon^* - \epsilon)\,, \tag{22}$$

where $V_{\mathsf{PD}}(\epsilon)$ is the speed of the waves under *peeling decoding*, which can be calculated as shown in (18), and $\breve{\gamma}(\epsilon^* - \epsilon)$ is the average number of degree-one CNs available to one decoding wave during the steady state of peeling decoding (see (6)).

The approximation (22) is also shown in Fig. 4 (blue dashed curve). The good match between the direct density evolution-based estimation of $V_{\mathsf{BP}}(\epsilon)$ (red) and the approximation in (22) (blue dashed) indicates that the number of degree-one CNs available to a decoding wave can be used as a crucial bridge between the steady-state behavior of peeling decoding and BP decoding. (This bridge between peeling and BP decoding was used by Olmos and Urbanke to estimate $\tilde{\gamma}$ from density evolution instead of from mean evolution [11, Sec. III.D]; we use it to characterize the effect of imposing a limit on the number of BP iterations.)

IV. Finite-Length Scaling: Constant Propagation Model

For a limited number of BP iterations, the probability of successful decoding (14) should be rewritten as

$$\Pr\{S\} = \Pr\left\{A+B > \tilde{\beta}-\tilde{\alpha} \bigcap \text{ enough iterations}\right\}\,; \tag{23}$$

the decoding waves must jointly recover $\tilde{\beta} - \tilde{\alpha}$ degree-one CNs, and they must do so within the allotted budget of I_{eff} BP iterations. The first condition is the same as in the law for unlimited BP iterations in (14). The second condition is novel and is operationalized below.

Density evolution shows that the decoding waves propagate throughout the chain with a constant speed in the limit of $N \to \infty$. A natural first step toward a finite-length scaling law would be to also assume a constant propagation speed in the context of a finite N—i.e., to assume that with every BP iteration each wave travels by exactly V_{BP} positions.

To understand why a limit on the number of iterations affects the probability of successful decoding, consider the following scenario: imagine one of the waves fails immediately at the beginning of the chain; imagine further that V_{BP} is too small for the other wave to propagate through the whole chain and meet the first wave within I_{eff} iterations. In that case decoding would fail even though it could have been successful without a limit on I.

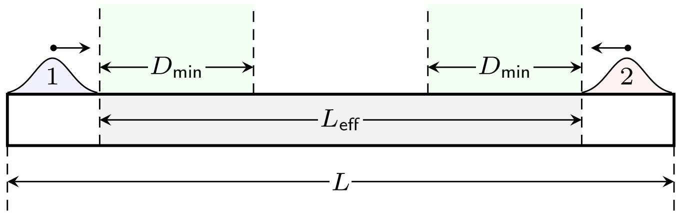

Fig. 5. Schematic illustration of the model.

Our approach is to incorporate a limit on the number of BP iterations into the scaling laws for *peeling decoding*—analyzing BP decoding directly proves to be challenging. First, we need to account for the width of the decoding waves under peeling decoding. It takes $\tilde{\beta} - \tilde{\alpha}$ normalized peeling decoding iterations for the waves to propagate through the chain with a constant speed of V_{PD} positions per normalized peeling decoding iteration (see Fig. 2). This means that the length (in positions) covered during the steady state is

$$L_{\mathsf{eff}} = (\tilde{\beta} - \tilde{\alpha})V_{\mathsf{PD}}\,. \tag{24}$$

We schematically illustrate the meaning of L_{eff}, which can be thought of as the effective length of the chain, in Fig. 5.

How much of that effective length should a wave cover under a limit on the number of BP iterations? Without loss of generality, take the left decoding wave, marked by 1 in Fig. 5. In I_{eff} BP iterations during the steady state, the wave can propagate by up to $V_{\mathsf{BP}}I_{\mathsf{eff}}$ positions. This means that for successful decoding, wave 2 should propagate by at least

$$\begin{aligned} D_{\mathsf{min}} &= \max\{0, L_{\mathsf{eff}} - V_{\mathsf{BP}}I_{\mathsf{eff}}\} \\ &\approx V_{\mathsf{PD}} \cdot \max\left\{0, \tilde{\beta} - \tilde{\alpha} - \breve{\gamma}(\epsilon^* - \epsilon)I_{\mathsf{eff}}\right\} \end{aligned} \tag{25}$$

positions to meet wave 1 before the deadline, as illustrated in Fig. 5. We need to bear in mind that, *mutatis mutandis*, the same logic applies to both the first and second wave, hence both waves need to propagate by at least D_{min} positions for decoding to be successful.

Fig. 6 shows an example of D_{min} calculated using (25) for the terminated $(5, 10, L = 50, N)$ SC-LDPC code ensemble and $I = \{175, 250, 300\}$ (solid curves). We observe that D_{min} is zero up to a certain value of ϵ and then increases with increasing ϵ. $D_{\mathsf{min}} = 0$ means that a single decoding wave has enough time to propagate all the way along the chain (i.e., successful decoding is possible even if the other wave fails immediately after the onset of the steady state), so the limit on the number of iterations does not change the probability of decoding error compared to the case of unlimited number of iterations. For larger ϵ, D_{min} may exceed $L_{\mathsf{eff}}/2$ (black dashed curve), which is when the waves do not get a chance to meet in time at all and decoding is bound to fail.

We can now use D_{min} to translate the limitation on the number of BP iterations into the language of the first-hit times A and B. To require a wave to travel by at least D_{min} positions is to ask it to survive for at least $\tau_{\mathsf{min}} = D_{\mathsf{min}}/V_{\mathsf{PD}}$ peeling

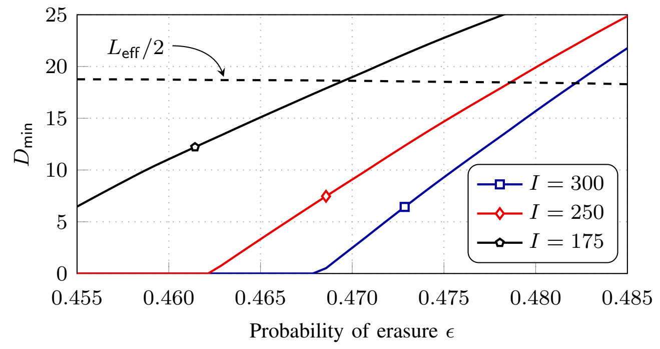

Fig. 6. The minimum propagation distance D_{min} (25) for the $(5, 10, L = 50)$ SC-LDPC code ensemble under BP decoding for $I = \{175, 250, 300\}$.

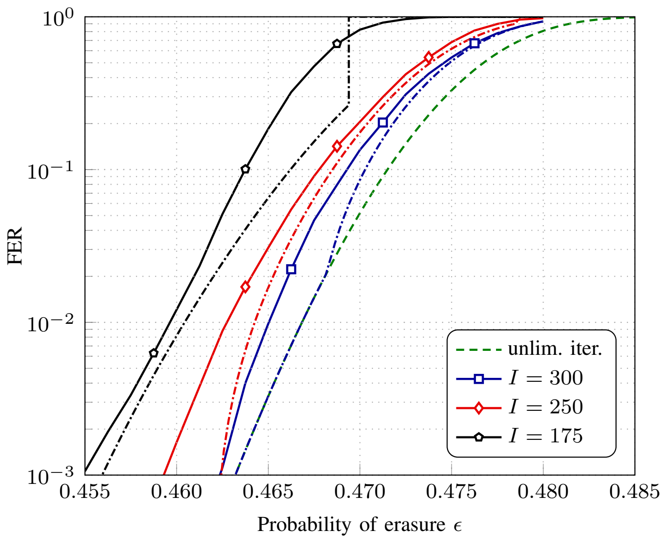

Fig. 7. FER for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under BP decoding for different limits on the number of iterations I (solid curves) and its approximation using (28) (corresponding dash-dotted curves).

decoding iterations. Bearing in mind that *both* waves need to survive that long to meet the deadline, we can rewrite the second condition in (23) as

$$\text{enough iterations} \iff A, B > \tau_{\text{min}} \,. \tag{26}$$

Since the first-hit times A and B are exponentially distributed (11), the two conditions in (23) can be combined as

$$\begin{aligned} \Pr\{S\} &= \Pr\left\{A + B > \tilde{\beta} - \tilde{\alpha} \bigcap A, B > \tau_{\text{min}}\right\} \\ &= \int_{\tilde{\beta}-\tilde{\alpha}}^{\infty} \int_{\tau_{\text{min}}}^{x-\tau_{\text{min}}} f_A(z) f_B(x - z) \mathrm{d}z \mathrm{d}x \\ &= \left(1 + \frac{\tilde{\beta} - \tilde{\alpha} - 2\tau_{\text{min}}}{\breve{\mu}_0}\right) \exp\left(-\frac{\tilde{\beta} - \tilde{\alpha}}{\breve{\mu}_0}\right) , \end{aligned} \tag{27}$$

where we assumed $D_{\text{min}} < L_{\text{eff}}/2$; otherwise, $\Pr\{S\} = 0$.

The FER can therefore be estimated as

$$P_{\text{f}} \approx 1 - \left(1 + \frac{\tilde{\beta} - \tilde{\alpha} - 2\tau_{\text{min}}}{\breve{\mu}_0}\right) \exp\left(-\frac{\tilde{\beta} - \tilde{\alpha}}{\breve{\mu}_0}\right) \tag{28}$$

if $D_{\text{min}} < L_{\text{eff}}/2$ and 1 otherwise.

Fig. 7 compares the simulated FERs (solid curves) for the terminated $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble with the approximation (28) (dash-dotted curves) under different limits on the number of BP iterations $I = \{175, 250, 300\}$. Naturally, as the limit I increases, the solid curves approach the FER curve for unlimited iterations (green dashed) calculated using (15). The scaling law (28) is a reasonably accurate approximation to the simulated FER, especially given that the only change to the scaling law for unlimited iterations (15) is the introduction of the additional term $-2\tau_{\text{min}}$ in (28). However, juxtaposing Figs. 6 and 7 reveals that the analytical approximation does not capture the simulated FER behavior in the regions where D_{min} is close either to zero (where the FER approximation curve joins the green dashed curve, as for $I = 300$ at around $\epsilon = 0.468$) or to $L_{\text{eff}}/2$ (where there may be a discontinuous jump to $P_{\text{f}} = 1$, as for $I = 175$ at around $\epsilon = 0.47$).

V. Finite-Length Scaling: Randomized Propagation Distance Models

In this section, we introduce three scaling laws that drop the assumption that a decoding wave propagates by the same distance in every BP iteration. Indeed, the partial mismatch of the scaling law (28) to the simulated FER curves in Fig. 7 suggests that the assumption of constant wave propagation does not hold in practice for a finite number of iterations—when $V_{\text{BP}}(\epsilon)$ is sufficient for the limitation on the number of iterations not to matter for the scaling law (i.e., when predicted FER curves merge with that for full BP decoding), the waves may still fail to meet before the deadline. Likewise, when $V_{\text{BP}}(\epsilon)$ becomes so low that the law predicts the FER to be equal to one, the waves may still succeed.

Let us denote by $n_{\text{PD}}(K)$ the number of normalized peeling decoding iterations (and hence the number of VNs recovered) that corresponds to K BP iterations for a decoding wave that has not run out of degree-one CNs. The scaling law in Section IV and the conversion between V_{PD} and V_{BP} in (22) effectively assume $n_{\text{PD}}(K)$ to be constant and equal to $\check{\gamma}(\epsilon^* - \epsilon)K$. In this section, we treat $n_{\text{PD}}(K)$ as a random variable instead.

A decoding wave stops either because it runs out of degree-one CNs or because it reaches the limit on the number of BP iterations, whichever event happens first. Correspondingly, let X_1 and X_2 denote the number of VNs recovered by the first and second wave by that time,

$$\begin{aligned} X_1 &= \min\left\{A, n_{\text{PD}}^{(1)}(I_{\text{eff}})\right\} , \\ X_2 &= \min\left\{B, n_{\text{PD}}^{(2)}(I_{\text{eff}})\right\} , \end{aligned} \tag{29}$$

where A and B are the first-hit times of the first and second wave, respectively, and the superscripts to $n_{\text{PD}}(I_{\text{eff}})$ emphasize that these are two independent random variables that correspond to the two decoding waves. (The two waves are numbered left to right as shown in Fig. 5.) The probability of a successful decoding can then be rewritten as

$$\Pr\{S\} = \Pr\left\{X_1 + X_2 > \tilde{\beta} - \tilde{\alpha}\right\} . \tag{30}$$

Essentially, the model (30) incorporates the second requirement in (23)—that there should be enough BP iterations

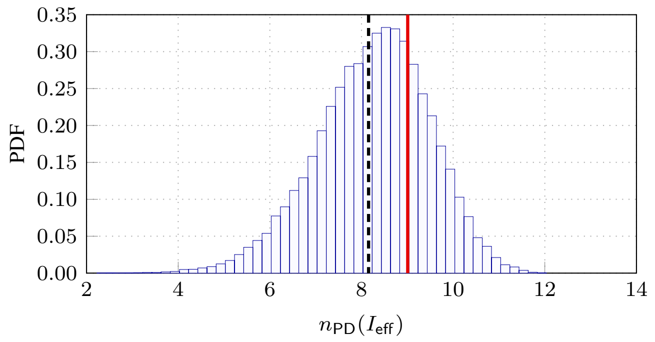

Fig. 8. The PDF of the Ornstein-Uhlenbeck-simulated $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ for the $(5, 10, L, N = 1000)$ SC-LDPC code ensemble under BP decoding for $I = 200$ and $\epsilon = 0.47$.

for the two waves to meet—into X_1 and X_2 via a random variable $n_{\mathsf{PD}}(I_{\mathsf{eff}})$, as opposed to incorporating it via a constant boundary τ_{min} as in (27).

Assuming A and $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ to be independent and A to be exponentially distributed (see (11)), we approximate the PDF of X_1 as

$$\begin{aligned} f_X(x) &= f_A(x)\Big[1 - F_{n_{\mathsf{PD}}}(x)\Big] + f_{n_{\mathsf{PD}}}(x)\Big[1 - F_A(x)\Big] \\ &\approx \frac{1}{\breve{\mu}_0}\exp\left(-\frac{x}{\breve{\mu}_0}\right)\Big[1 - F_{n_{\mathsf{PD}}}(x)\Big] \\ &\quad + f_{n_{\mathsf{PD}}}(x)\exp\left(-\frac{x}{\breve{\mu}_0}\right), \end{aligned} \tag{31}$$

where $f_{n_{\mathsf{PD}}}$ and $F_{n_{\mathsf{PD}}}$ denote the PDF and the CDF of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$, respectively. The same logic applies to the second wave, so the PDF of X_2 is also $f_X(x)$. We can now use (31) to estimate the FER similarly to (14) as

$$P_{\mathsf{f}} = 1 - \Pr\{S\} \approx 1 - \int_{\tilde{\beta}-\tilde{\alpha}}^{\infty}\int_{0}^{x} f_X(z) f_X(x-z)\mathrm{d}z\mathrm{d}x\,. \tag{32}$$

In summary, we need to approximate the distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ and use it in (31)–(32) to predict the FER. We provide three ways to model $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ below. The third way results in a scaling law that offers the best trade-off between accuracy and computational complexity; however, it combines the features of the first two, and we present all three for clarity of exposition.

A. Simulating $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ Using an Ornstein-Uhlenbeck Process

We can approximate the distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ using the following heuristic: We keep the assumption that an iteration of BP corresponds to as many iterations of peeling decoding as there are degree-one CNs available to decode. However, we no longer assume the normalized number of available degree-one CNs across iterations during the steady state, $r_1(\tau_{\mathsf{PD}})$, to be constant. Instead, we assume that each BP iteration is equivalent to advancing along a peeling decoding trajectory—i.e., a realization of $r_1(\tau_{\mathsf{PD}})$—by as much as there are degree-one CNs in the current position on the peeling decoding trajectory. This heuristic model allows us to directly simulate the PDF of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ from Monte-Carlo realizations of the Ornstein-Uhlenbeck process that correspond to realizations

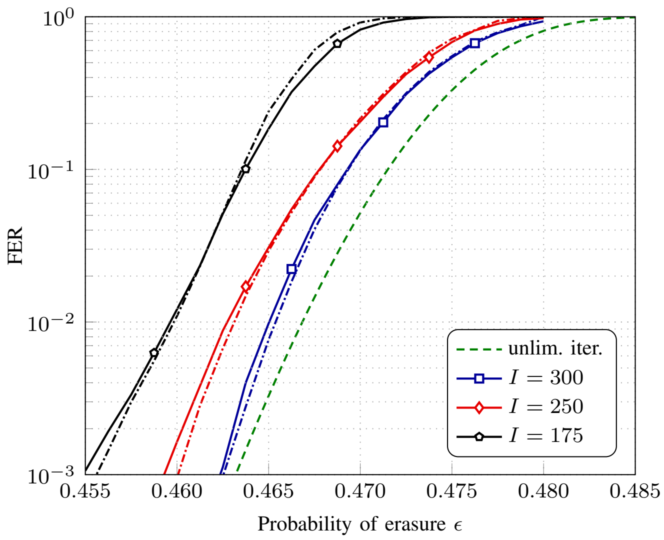

Fig. 9. FER for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under BP decoding for different limits on the number of iterations I (solid curves) and its approximation using the PDF of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ from (33) based on the Ornstein-Uhlenbeck process (corresponding dash-dotted curves).

of the steady-state of $r_1(\tau_{\mathsf{PD}})$. Indeed, from each Ornstein-Uhlenbeck realization of $r_1(\tau_{\mathsf{PD}})$ we can generate a realization of $n_{\mathsf{PD}}(K)$ as

$$n_{\mathsf{PD}}(K) = n_{\mathsf{PD}}(K-1) + r_1\left(n_{\mathsf{PD}}(K-1)\right), \tag{33}$$

starting with $n_{\mathsf{PD}}(1) = r_1(0)$. We remark that when $r_1(\tau_{\mathsf{PD}})$ is set to be constant and equal to $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$ from (6), $n_{\mathsf{PD}}(K)$ boils back down to $\breve{\gamma}(\epsilon^* - \epsilon)K$.

Fig. 8 shows an example PDF of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ simulated via (33) from Monte-Carlo realizations of the Ornstein-Uhlenbeck process with parameters $(\breve{\gamma}, \breve{\nu}, \breve{\theta})$ in (6)–(8) chosen to match the first two moments of $r_1(\tau_{\mathsf{PD}})$ (blue histogram). We observe that the average value of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ (vertical black dashed line) is smaller than $\breve{\gamma}(\epsilon^* - \epsilon)I_{\mathsf{eff}}$ (vertical red solid line). Further, the left tail of the PDF is "heavier" than the right. This can be explained by the asymmetry in the way temporal correlation in $r_1(\tau_{\mathsf{PD}})$ influences $n_{\mathsf{PD}}(K)$ in (33): when $r_1\left(n_{\mathsf{PD}}(K-1)\right)$ is large, $n_{\mathsf{PD}}(K)$ will end up far away from $n_{\mathsf{PD}}(K-1)$, so the next increment in n_{PD} will be statistically close to a sample from an independent Gaussian random variable centered at $\breve{\gamma}(\epsilon^* - \epsilon)$. When $r_1\left(n_{\mathsf{PD}}(K-1)\right)$ is small, on the other hand, the next increment in n_{PD} will also likely be small.

The first scaling law we propose in this section uses the simulated distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ based on the Ornstein-Uhlenbeck model (33) in (31)–(32) to estimate the FER. The corresponding approximations are shown in Fig. 9 for the terminated $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble and $I = \{175, 250, 300\}$ (dash-dotted curves). The match between the predicted and simulated (solid) FER curves is remarkable—evidently, the iterative model based on the Ornstein-Uhlenbeck process (33) is a good approximation of the behavior of BP decoding. The drawback of the model is that it requires Monte-Carlo approximation of the PDF of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ for every combination of (ϵ, N, I); however, simulating $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ via (33) is still far less computationally complex than simulating BP decoding.

B. Gaussian Propagation Distance Model

In this section, we introduce a model that does not require Monte-Carlo simulation of the distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ as in Section V-A. Instead, the model approximates $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ by a Gaussian random variable. To that end, we will rely on the fact that a time integral of an Ornstein-Uhlenbeck process is normally distributed, as we describe below.

1) Ornstein-Uhlenbeck process. Necessary background: A generic Ornstein-Uhlenbeck process Z_t can be described using the following stochastic differential equation:

$$\mathrm{d}Z_t = -b(x_t - m)\mathrm{d}t + \sigma \mathrm{d}B_t\,, \tag{34}$$

where B_t is the standard Wiener process, b and σ are real positive constants, and m is real. The first term on the right-hand side of (34) can be conceptualized as a mean-reverting factor and the second term as a random fluctuation. The constants b and σ determine their relative importance.

For a fixed sufficiently large t, the distribution of Z_t is Gaussian. Specifically,

$$Z_t \sim \mathcal{N}\left(m, \frac{\sigma^2}{2b}\right)\,. \tag{35}$$

Moreover, for sufficiently large $t+s$,

$$\mathrm{Cov}\left[Z_t, Z_s\right] = \frac{\sigma^2}{2b}\exp\left(-b\left|t-s\right|\right)\,. \tag{36}$$

Crucially, Ornstein and Uhlenbeck showed that a time-integrated Ornstein-Uhlenbeck process is a Gaussian random variable [26]. For $t \to \infty$,

$$\int_0^t Z_t \mathrm{d}t \sim \mathcal{N}\left(mt, \frac{\sigma^2}{b^2}t\right)\,. \tag{37}$$

This result is the cornerstone of our approximation of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ as a Gaussian random variable.

2) Ornstein-Uhlenbeck process as a model for BP decoding: Let $v_{\mathsf{BP}}(t)$ denote the number of new VNs (normalized by N) resolved in BP iteration ℓ, where t is

$$t = \ell \cdot (\epsilon^* - \epsilon)\,. \tag{38}$$

The quantity $v_{\mathsf{BP}}(t)$ can be interpreted as the instantaneous decoding speed, measured in VNs (normalized by N) per BP iteration. Here, the time variable t is normalized by the distance to the BP threshold $(\epsilon^* - \epsilon)$ instead of by N as is the case for τ_{PD}, discussed in Section II-C. Up to the normalization of time, $v_{\mathsf{BP}}(t)$ is equivalent to $v_{\mathsf{BP}}(\ell)$ introduced in Section III. We use different time variables in these two cases to avoid confusion.

The process $v_{\mathsf{BP}}(t)$ was introduced in [25] to provide an alternative finite-length scaling law for SC-LDPC code ensembles that yields less accurate predictions of the FER than the law in [15] (and even than the law in [11]) but does not rely on peeling decoding or mean evolution. Indeed, as opposed to $\mathbb{E}\left[r_1(\tau_{\mathsf{PD}})\right]$, $\mathbb{E}\left[v_{\mathsf{BP}}(t)\right]$ can be obtained directly from density evolution, as we discuss in Section III (see (20)).

The authors show that $v_{\mathsf{BP}}(t)$ exhibits a steady-state phase; they approximate the first two moments of this process during the steady-state phase as

$$\mathbb{E}\left[v_{\mathsf{BP}}(t)\right] \approx \breve{\gamma}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)\,, \tag{39}$$

$$\mathrm{Var}\left[v_{\mathsf{BP}}(t)\right] \approx \frac{\breve{\nu}_{\mathsf{BP}}}{N}\,, \tag{40}$$

$$\mathrm{Cov}\left[v_{\mathsf{BP}}(t), v_{\mathsf{BP}}(s)\right] \approx \frac{\breve{\nu}_{\mathsf{BP}}}{N}\exp\left(-\breve{\theta}_{\mathsf{BP}}\left|t-s\right|\right)\,. \tag{41}$$

We apply the same refinement of the law as we did in [15] and model the two-wave process for the terminated ensemble as a combination of two independent Ornstein-Uhlenbeck processes. We therefore estimate the triple $(\breve{\gamma}_{\mathsf{BP}}, \breve{\nu}_{\mathsf{BP}}, \breve{\theta}_{\mathsf{BP}})$ from the truncated ensemble, as we did in [15] for peeling decoding to estimate $(\breve{\gamma}, \breve{\nu}, \breve{\theta})$. (We review the peeling decoding-based laws in Section II-C.) The steady-state level constant $\breve{\gamma}_{\mathsf{BP}}$ is estimated along with I_{start} and I_{end} from density evolution: once I_{start} and I_{end} are estimated as described in Section III, we estimate $\breve{\gamma}_{\mathsf{BP}}$ from the average of $\mathbb{E}\left[v_{\mathsf{BP}}(\ell)\right]$ for $\ell \in [I_{\mathsf{start}}, I_{\mathsf{end}}]$. The covariance parameters $\breve{\nu}_{\mathsf{BP}}$ and $\breve{\theta}_{\mathsf{BP}}$ are estimated from Monte-Carlo simulations of BP decoding for a single fixed (N, ϵ). For our running example of the $(5, 10, N, L)$ SC-LDPC code ensemble, $\breve{\nu}_{\mathsf{BP}}$ and $\breve{\theta}_{\mathsf{BP}}$ are estimated for $(N = 5000, \epsilon = 0.465)$ to be $\breve{\nu}_{\mathsf{BP}} \approx 0.41, \breve{\theta}_{\mathsf{BP}} \approx 2.74$. The scaling parameters $\breve{\gamma}_{\mathsf{BP}}, \breve{\nu}_{\mathsf{BP}}$, and $\breve{\theta}_{\mathsf{BP}}$ are illustrated in Fig. 3 along with I_{start} and I_{end}.

We follow [25] and model $v_{\mathsf{BP}}(t)$ in the steady state by an Ornstein-Uhlenbeck process. Equating the moments (39)–(41) to those in (35)–(36) yields

$$m = \breve{\gamma}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)\,, \qquad b = \breve{\theta}_{\mathsf{BP}}\,, \qquad \sigma^2 = 2\breve{\theta}_{\mathsf{BP}} \cdot \frac{\breve{\nu}_{\mathsf{BP}}}{N}\,. \tag{42}$$

3) Normal approximation based on a time-integrated Ornstein-Uhlenbeck process: The total number of VNs decoded in I_{eff} iterations, $n_{\mathsf{PD}}(I_{\mathsf{eff}})$, can be expressed as a time integral of $v_{\mathsf{BP}}(t)$, divided by $(\epsilon^* - \epsilon)$ to convert the units of speed to VNs per *normalized* iteration,

$$\begin{aligned} n_{\mathsf{PD}}(I_{\mathsf{eff}}) &\approx \frac{1}{\epsilon^* - \epsilon}\int_{t_{\mathsf{start}}}^{t_{\mathsf{start}}+t_{\mathsf{eff}}} v_{\mathsf{BP}}(t)\mathrm{d}t \\ &= \frac{1}{\epsilon^* - \epsilon}\int_0^{t_{\mathsf{eff}}} v_{\mathsf{BP}}(t + t_{\mathsf{start}})\mathrm{d}t\,, \end{aligned} \tag{43}$$

where

$$t_{\mathsf{start}} = I_{\mathsf{start}}\left(\epsilon^* - \epsilon\right)\,, \qquad t_{\mathsf{eff}} = I_{\mathsf{eff}}\left(\epsilon^* - \epsilon\right)\,. \tag{44}$$

We can now use the expression for a time-integrated Ornstein-Uhlenbeck process (37) with appropriately chosen parameters (42) to approximate the distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ from (43) as

$$\begin{aligned} \left(\epsilon^* - \epsilon\right) n_{\mathsf{PD}}(I_{\mathsf{eff}}) &\sim \mathcal{N}\left(m t_{\mathsf{eff}}, \frac{\sigma^2}{b^2} t_{\mathsf{eff}}\right) \\ &\overset{\text{(a)}}{=} \mathcal{N}\left(\breve{\gamma}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right) t_{\mathsf{eff}}, \frac{2\breve{\nu}_{\mathsf{BP}}}{N\breve{\theta}_{\mathsf{BP}}} t_{\mathsf{eff}}\right) \\ &\overset{\text{(b)}}{=} \mathcal{N}\left(\breve{\gamma}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)^2 I_{\mathsf{eff}}, \frac{2\breve{\nu}_{\mathsf{BP}}}{N\breve{\theta}_{\mathsf{BP}}}\left(\epsilon^* - \epsilon\right) I_{\mathsf{eff}}\right)\,, \end{aligned} \tag{45}$$

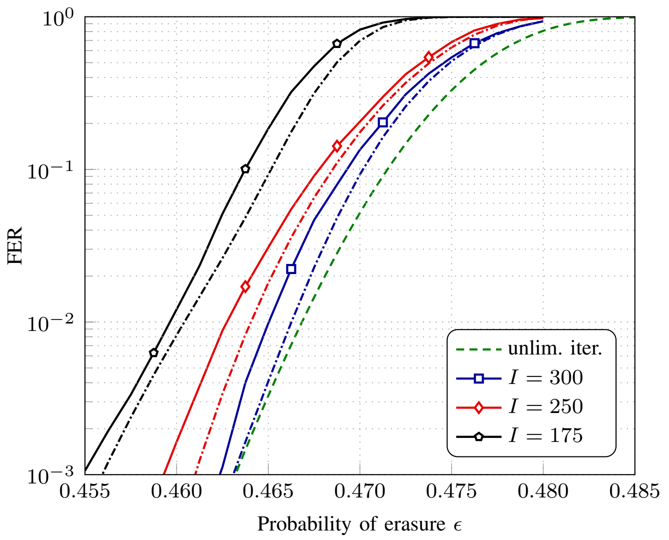

Fig. 10. FER for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under BP decoding for different limits on the number of iterations I (solid curves) and its approximation using (31)–(32) with the distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ from (46) (corresponding dash-dotted curves).

$$n_{\mathsf{PD}}(I_{\mathsf{eff}}) \sim \mathcal{N}\left(\breve{\gamma}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right) I_{\mathsf{eff}}, \frac{2\breve{\nu}_{\mathsf{BP}} I_{\mathsf{eff}}}{N\breve{\theta}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)}\right), \quad (46)$$

where in (a) we used (42) and in (b) we used (44). Notably, the approximation (46) reveals that both mean and variance of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ grow linearly with I_{eff}. We must also remark that the approximation (46) does not account for the possibility that the underlying Ornstein-Uhlenbeck process becomes negative.

The second scaling law we propose in this section uses the normal approximation (46) to the distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ in (31)–(32) to estimate the FER. The corresponding predictions are shown in Fig. 10 for our running example of the terminated $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble and $I = \{175, 250, 300\}$ (dash-dotted curves). We observe that, while being a good approximation to the simulated FER, the predictions are more optimistic than the predictions based on the simulated Ornstein-Uhlenbeck process from Section V-A (cf. Fig. 9).

4) Shifted normal approximation: Part of the reason behind the mismatch between the simulated and predicted FER in Fig. 10 is that (39) overestimates the average number of VNs decoded in a BP iteration for finite N, as shown in Fig. 11 for the truncated $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble and $\epsilon = 0.47$. For the same ensemble and ϵ, Fig. 12 shows that the gap in the steady-state level translates into a shift in the location of the Gaussian approximation (46) to $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ (green curve) relative to the simulated histogram (thin blue curve). Estimating the gap without resorting to Monte-Carlo simulations proves to be difficult; moreover, even if we adjust the location of the Gaussian by shifting its mean to account for the gap, the resulting distribution (red curve) lags behind the simulated histogram (thin blue curve). Overall, the simulated PDF based on the iterative Ornstein-Uhlenbeck model in Section V-A (purple curve) is the most accurate. On the other hand, the figure suggests that the two Gaussian models capture the variance of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ rather well.

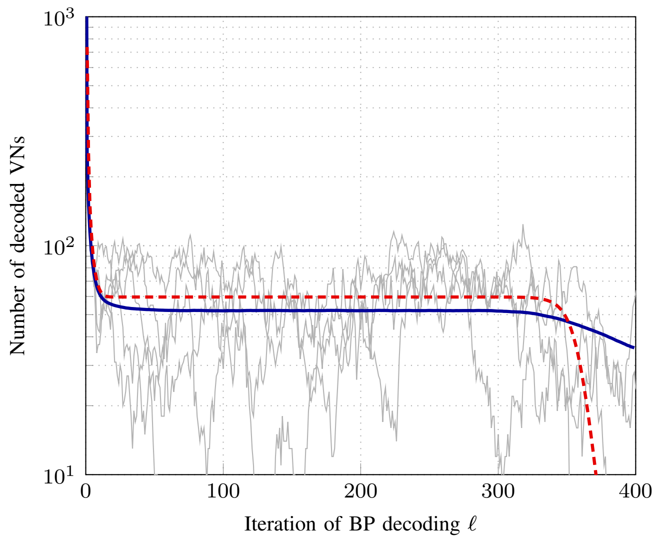

Fig. 11. The number of VNs decoded per BP iteration for the truncated $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble and $\epsilon = 0.47$: the simulated average from 10^5 trajectories (blue solid curve) and the mean from density evolution (red dashed curve). Several simulated trajectories are shown as thin gray lines.

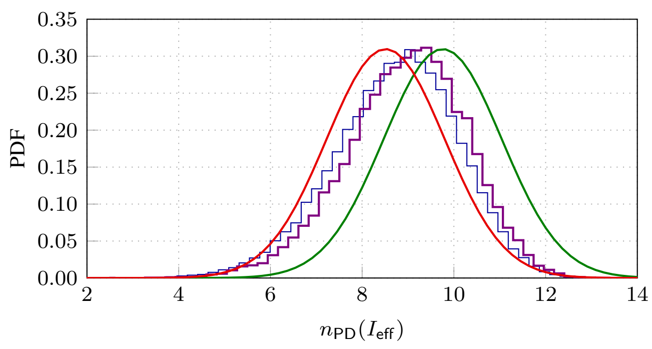

Fig. 12. The PDF of the total number of VNs decoded during the steady state for the truncated $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble, $\epsilon = 0.47$ and $I = 200$: the simulated histogram (thin blue curve), the Gaussian approximation (46) with density evolution- (green curve) or BP simulation-based (red curve) mean, and the simulated PDF based on the iterative Ornstein-Uhlenbeck model from Section V-A (purple curve).

With that in mind, we propose a hybrid model where the Gaussian approximation (46) is shifted to be located at the average of the iterative Ornstein-Uhlenbeck model (33). Specifically, let $f(\epsilon, N, I_{\mathsf{eff}})$ denote that average. Unfortunately, obtaining $f(\epsilon, N, I_{\mathsf{eff}})$ analytically is challenging; we resort to Monte-Carlo simulations of the iterative Ornstein-Uhlenbeck model (33) to estimate it. These simulations indicate that $f(\epsilon, N, I_{\mathsf{eff}})$ depends linearly on I_{eff}—we can thus estimate the slope of $f(\epsilon, N, I_{\mathsf{eff}})$ as $c_{\mathsf{f}} = f(\epsilon, N, I'_{\mathsf{eff}})/I'_{\mathsf{eff}}$ for a specific I'_{eff} and obtain the values of $f(\epsilon, N, I_{\mathsf{eff}})$ for other I_{eff} by scaling that mother curve as $c_{\mathsf{f}} I_{\mathsf{eff}}$, thereby avoiding the need to re-simulate $f(\epsilon, N, I_{\mathsf{eff}})$ for every I_{eff}. This is equivalent to using

$$m = c_{\mathsf{f}} \quad (47)$$

instead of $m = \breve{\gamma}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)$ in (42).

Putting it all together, the third and last scaling law we propose in this section approximates the distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$

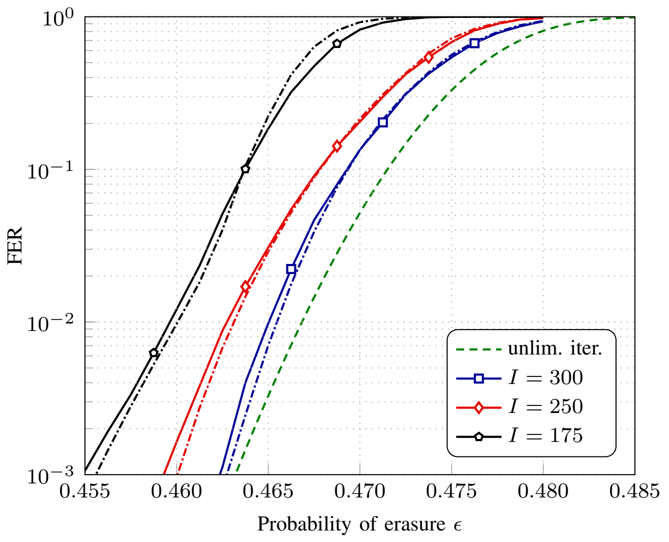

Fig. 13. FER for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under BP decoding for different limits on the number of iterations I (solid curves) and its approximation using (31)–(32) with the distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ from (48) (corresponding dash-dotted curves).

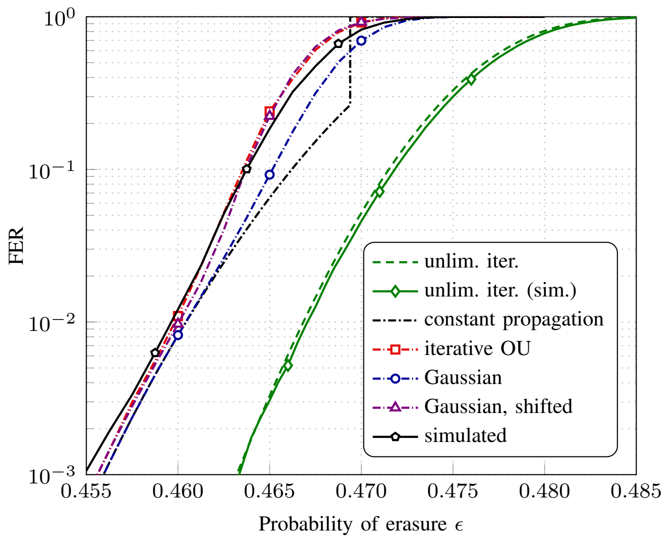

Fig. 14. Simulated FER for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under BP decoding for $I = 175$ (solid curve) and its approximations (dash-dotted curves).

as

$$n_{\mathsf{PD}}(I_{\mathsf{eff}}) \sim \mathcal{N}\left(c_{\mathsf{f}} I_{\mathsf{eff}}, \frac{2\breve{\nu}_{\mathsf{BP}} I_{\mathsf{eff}}}{N \breve{\theta}_{\mathsf{BP}}(\epsilon^* - \epsilon)}\right) \tag{48}$$

and uses it in (31)–(32) to estimate the FER.

The resulting FER predictions for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble are shown in Fig. 13 for $I = \{175, 250, 300\}$ (dash-dotted curves). We chose $I'_{\mathsf{eff}} = 350 - I_{\mathsf{start}} - I_{\mathsf{end}}$, estimated $c_{\mathsf{f}} = f(\epsilon, N = 1000, I'_{\mathsf{eff}})/I'_{\mathsf{eff}}$ for several ϵ, and linearly interpolated it to obtain the intermediate values. (We remark that I_{start} and I_{end} depend on ϵ in their turn; as we discuss in Section III, we also estimate them for several ϵ and use linear interpolation to obtain the intermediate values.) The resulting predictions are very close to those in Fig. 9 by the iterative Ornstein-Uhlenbeck model (i.e., by the first scaling law proposed in this section).

We conclude that the hybrid model provides the best trade-off between accuracy and analytical tractability. Importantly, unlike the first scaling law in this section, it does not require Monte-Carlo simulation for every combination of (ϵ, N, I)—estimating c_{f} from a single value of the number of BP iterations allows us to obtain the whole family of FER curves for different limits on the number of BP iterations.

C. Discussion

Before we tackle the FER prediction for sliding window decoding with a limit on the number of iterations in Section VI, let us summarize the performance-complexity trade-offs associated with the four scaling laws we proposed for full BP decoding. To that end, Fig. 14 groups the simulated (black solid curve with pentagons) and predicted FER curves for the terminated $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under full BP decoding with $I = 175$ using the constant propagation model (28) (black dash-dotted curve) and the randomized propagation model (31)–(32) with the approximated distribution of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ based on the iterative Ornstein-Uhlenbeck model (33) (red dash-dotted curve with squares), on the normal distribution for the integrated Ornstein-Uhlenbeck process (46) (blue dash-dotted curve with circles), and on the shifted normal distribution (48) (purple dash-dotted curve with triangles). For reference, we have also included the simulated FER curve for unlimited number of iterations (green solid curve with diamonds) alongside our prediction from [15] (given in (15)) (green dashed curve).

Besides the scaling parameters also required for the scaling law with unlimited iterations (15), the black dash-dotted prediction curve obtained via the constant propagation model (28) requires calculating τ_{min}, which is a function of quantities we estimate from mean and density evolution, as we describe in Section IV. In that sense, the constant propagation model (28) is no more computationally challenging than the scaling law for unlimited iterations (15). On the other hand, the constant propagation model does not capture well the transition regions of the FER curve, as exemplified by the discontinuous jump of the black dash-dotted curve around $\epsilon = 0.47$ and discussed in Section IV.

The iterative Ornstein-Uhlenbeck model (33) yields very accurate FER predictions (red dash-dotted curve with squares); however, it requires Monte-Carlo simulation of the Ornstein-Uhlenbeck process and numerical approximation of the PDF of $n_{\mathsf{PD}}(I_{\mathsf{eff}})$ from the simulated realizations for every (ϵ, N, I). The normal approximation based on the time-integrated Ornstein-Uhlenbeck process (46), shown in Fig. 14 as the blue dash-dotted curve with circles, is computationally less complex than the iterative Ornstein-Uhlenbeck model once the parameters $(\breve{\gamma}_{\mathsf{BP}}, \breve{\nu}_{\mathsf{BP}}, \breve{\theta}_{\mathsf{BP}})$ are estimated, since it does not require estimating a probability distribution via Monte-Carlo simulation. However, it is also less accurate. The accuracy can be improved by shifting the Gaussian to be located at the mean of the iterative Ornstein-Uhlenbeck model, which we describe in Section V-B4 to obtain the purple dash-dotted curve with triangles in Fig. 14 via (48). The shifted normal approximation is less computationally complex than the iterative Ornstein-Uhlenbeck model—not only does it rely on Gaussian distribution and thus simplify numerical integration in (32), but it also avoids simulating the iterative Ornstein-Uhlenbeck model (33) for every I—while achieving similar accuracy. It offers the best trade-off between complexity and accuracy, and we build upon the same shifted model in Section VI below to provide a scaling law for sliding window decoding with a limit on the number of iterations.

Let us review the steps required to apply the scaling law based on the shifted normal approximation. The user needs to implement density and mean evolution for the ensemble of interest, and the Monte-Carlo simulators of BP and peeling decoding and of the Ornstein-Uhlenbeck process. Density evolution is used to obtain the decoding threshold ϵ^* and, for different ϵ, the values of I_{start} and I_{end} (the unshifted normal approximation requires $\breve{\gamma}_{\mathsf{BP}}$ estimated in the same way). Mean evolution is used to obtain the boundaries and the level of the peeling decoding steady state $(\tilde{\alpha}, \tilde{\beta}, \breve{\gamma})$ for several ϵ. For a single fixed (N, ϵ), Monte-Carlo simulation of the peeling decoding is required to estimate the variance and covariance decay parameters of the steady state, $(\breve{\nu}, \breve{\theta})$; analogously, Monte-Carlo simulation of BP decoding is required to estimate $(\breve{\nu}_{\mathsf{BP}}, \breve{\theta}_{\mathsf{BP}})$. Finally, Monte-Carlo simulations of the Ornstein-Uhlenbeck process for several (ϵ, N) are needed to estimate c_{f}. These parameters are fed to (48) and (31)–(32) to estimate the FER for different (ϵ, N, I).

It would greatly benefit the applicability of the finite-length scaling approach to be able to estimate $(\breve{\nu}, \breve{\theta})$ and $(\breve{\nu}_{\mathsf{BP}}, \breve{\theta}_{\mathsf{BP}})$ without resorting to Monte-Carlo simulation (and without implementing covariance evolution [11]). Similarly, it would be interesting to find a way to estimate c_{f} without simulating the Ornstein-Uhlenbeck process for different (ϵ, N). Providing faster ways to estimate these values would facilitate further adoption of finite-length scaling laws for code and decoder parameter optimization and is an interesting direction of future research. The values that are estimated from density and mean evolution, on the other hand, are relatively easy to obtain.

VI. Finite-Length Scaling: Sliding Window Decoding with a Limited Number of Iterations

The core idea behind the scaling laws in Section V is to model the number of bits recovered by a given number of BP iterations as a time integral of the Ornstein-Uhlenbeck process $v_{\mathsf{BP}}(t)$. In this section, we further develop this approach and use it to estimate the FER under sliding window decoding. First, however, we need to consider the specific ways in which a limit on the number of iterations affects the probability of decoding error, which we discuss in the two following subsections.

A. Competition Between the Left Wave and the Sliding Window

In addition to the potential failure of the decoding waves that we analyzed in [15], a limit on the number of BP iterations in sliding window decoding introduces a kind of race between the left decoding wave and the sliding window. Let $P_{\mathsf{L}}(\ell) \in [0, L-1]$ denote the position of the leftmost VN that remains unrecovered by iteration ℓ. We will refer to $P_{\mathsf{L}}(\ell)$ as the position of the left wave at iteration ℓ. Analogously, let $W_{\mathsf{L}}(\ell)$ denote the leftmost position within the sliding window, and $W_{\mathsf{R}}(\ell) = W_{\mathsf{L}}(\ell) + W$ the next position just outside the right boundary of the sliding window. If, at any iteration ℓ, the left boundary of the window overtakes the wave, decoding fails, even though it could potentially have succeeded had the number of iterations not been limited. Let O denote this overtaking event,

$$O = \left\{ P_{\mathsf{L}}(\ell) < W_{\mathsf{L}}(\ell) \text{ for some } \ell \in [1, I] \right\}. \tag{49}$$

A necessary condition for successful decoding is that O does not happen.

The setup we are considering can be illustrated by means of the following analogy. Suppose a user is watching a video via a streaming service. The video is being downloaded at a rate that fluctuates around a certain average. After an initial buffering period, the device starts playing the video. Then the event O in question is that the buffer is exhausted (and thus the video frozen) at least once during playback.

B. Reduced Maximum Propagation Distance for the Right Wave

Apart from the possibility of the window overtaking the left wave, the limit on the number of iterations also affects decoding by reducing the maximum distance that can be possibly traveled by the right wave (once the sliding window reaches the right boundary of the chain). When the number of decoding iterations is not limited, we assume that the right wave can travel by up to W positions, which is reflected in our scaling law for sliding window decoding (17) proposed in [15]. Here, we account for the presence of a limit on the number of iterations by estimating the maximum number of positions the right wave can travel to the left while the sliding window is moving to the right, which we denote by $W' \leq W$.

We assume the following three-phase process: In the first phase, which begins when the sliding window starts to cover the last VN position (i.e., the right boundary of the window has reached the end of the coupled chain) and lasts I_{start} iterations, the right wave forms. Then, in the second phase, the right wave and the window move toward each other. Finally, in the third phase, which lasts I_{end} iterations, the two decoding waves meet and collapse. During the first and third phase the right wave does not travel, whereas the window moves to the right with speed $V_{\mathsf{W}} = 1/I_{\mathsf{s}}$. During the second phase, the right wave travels leftward with speed V_{BP}, and the window moves rightward with speed V_{W}. The maximum number of positions the right wave can travel, W', corresponds to the distance covered by the right wave in the second phase.

During the first and third phase, the window slides by

$$(I_{\mathsf{start}} + I_{\mathsf{end}})\, V_{\mathsf{W}} \tag{50}$$

positions while the right wave does not move. The remaining

$$W - (I_{\mathsf{start}} + I_{\mathsf{end}}) V_{\mathsf{W}} \tag{51}$$

positions will be covered by the window and the wave jointly with speed $V_{\mathsf{BP}} + V_{\mathsf{W}}$, which will take

$$\Big[W - (I_{\mathsf{start}} + I_{\mathsf{end}})V_{\mathsf{W}}\Big] \cdot \frac{1}{V_{\mathsf{BP}} + V_{\mathsf{W}}} \tag{52}$$

iterations. During that time, the right wave will have traveled by

$$W' = \Big[W - (I_{\mathsf{start}} + I_{\mathsf{end}})V_{\mathsf{W}}\Big] \cdot \frac{V_{\mathsf{BP}}}{V_{\mathsf{BP}} + V_{\mathsf{W}}} \tag{53}$$

positions, which is the adjusted size of the sliding window we use in our model below. Naturally, as the limit on the number of iterations is relaxed, $V_{\mathsf{W}} \to 0$ in (53) and therefore $W' \to W$.

C. General Form of the Scaling Law

We incorporate the limit on the number of iterations during sliding window decoding into our model via the two effects outlined above, namely, the possibility that the window overtakes the left wave (an event denoted by O) and the reduction of the maximum distance the right wave can travel. In essence, successful decoding requires that (*i*) the overtaking O does not happen; (*ii*) the left wave does not run out of VNs to decode while propagating through the first $L - W'$ positions; and (*iii*) the left and right wave jointly cover the last W' positions. Conditions (*ii*) and (*iii*) are equivalent to the two-phase model we proposed in [15] to estimate the FER under unlimited iterations as in (17); the only adjustment is that the size of the second phase is reduced from W to W', defined in (53), to account for the reduction of the maximum reach of the right wave, as discussed in more detail in Section VI-B. Condition (*i*) is introduced in Section VI-A and is novel.

Modeling the three conditions as independent events, we approximate the FER under sliding window decoding with a limit on the number of BP iterations as

$$\begin{aligned} P_{\mathsf{f}} = 1 - \Big(1 - \Pr\{O\}\Big) \\ \cdot \Big(1 - P_{\mathsf{f,u}}^{(L-W')}\Big)\Big(1 - P_{\mathsf{f,t}}^{(W')}\Big)\,, \end{aligned} \tag{54}$$

where $P_{\mathsf{f,u}}^{(L-W')}$ and $P_{\mathsf{f,t}}^{(W')}$ are the estimated FERs for the unterminated and terminated SC-LDPC code ensembles of length $L - W'$ and W', and are defined in (15) and (16), respectively.

The rest of the section is devoted to the estimation of $\Pr\{O\}$, the core of our finite-length scaling law for sliding window decoding with a limited number of iterations.

D. Modeling the Race Between the Left Wave and the Window

The general idea behind our approach is to model the stochastic process associated with the position of the left wave, $P_{\mathsf{L}}(\ell)$, by a scaled time integral of the Ornstein-Uhlenbeck process that corresponds to $v_{\mathsf{BP}}(\ell)$, the normalized number of bits recovered in iteration ℓ defined in Section III, with an additional noise term. The wave cannot overtake the right boundary of the window, $W_{\mathsf{R}}(\ell)$, because no BP iterations are performed there. On the other hand, if the wave is itself overtaken by the left boundary, $W_{\mathsf{L}}(\ell)$, decoding is bound to fail. This motivates us to incorporate the boundaries of the window into our model as an absorbing barrier at $W_{\mathsf{L}}(\ell)$ and a reflecting barrier at $W_{\mathsf{R}}(\ell)$. When a stochastic process hits an absorbing barrier, it remains absorbed indefinitely. When a stochastic process hits a reflecting barrier, it is reflected back inside the domain [27]. The overtaking event O that we introduced in Section VI-A in (49) corresponds to $P_{\mathsf{L}}(\ell)$ having been absorbed by iteration I. The probability of this event can be estimated by tracking the evolution of the PDF of $P_{\mathsf{L}}(\ell)$ across iterations. That evolution can in turn be described by a partial differential equation called the Fokker-Planck equation [27]. We numerically solve the initial value problem for the Fokker-Planck equation for $P_{\mathsf{L}}(\ell)$ with the boundary conditions that correspond to an absorbing barrier at $W_{\mathsf{L}}(\ell)$ and a reflecting barrier at $W_{\mathsf{R}}(\ell)$ and obtain the estimation of the probability of O.

1) Position of the left wave as an integrated Ornstein-Uhlenbeck process: The scaling law for full BP decoding proposed in Section V uses a time integral of the Ornstein-Uhlenbeck process $v_{\mathsf{BP}}(\ell)$ as a model for the total number of VNs (normalized by N) decoded in a given number of BP iterations. In the context of sliding window decoding, we are interested instead in the *position* of the wave after a number of iterations. The basic element of our model is the conversion of the normalized number of VNs decoded in BP iteration ℓ, $v_{\mathsf{BP}}(\ell)$, to the number of positions traveled by the wave in that iteration. We assume that N VNs decoded during the steady state of BP decoding advance the wave by V_{PD} positions—i.e., by the same number of positions as for peeling decoding (18). The number of positions traveled in iteration ℓ is then

$$v_{\mathsf{BP}}(\ell)V_{\mathsf{PD}} \tag{55}$$

and the *total* number of positions traveled in ℓ steady-state BP iterations is

$$P_{\mathsf{L}}(\ell) = n_{\mathsf{PD}}(\ell)V_{\mathsf{PD}}\,, \tag{56}$$

where $n_{\mathsf{PD}}(\ell)$ is the number of VNs decoded in ℓ steady-state BP iterations that we introduced in Section V. We can use the model (43) of $n_{\mathsf{PD}}(\ell)$ in (56) to obtain

$$P_{\mathsf{L}}(\ell) = n_{\mathsf{PD}}(\ell)V_{\mathsf{PD}} = \frac{V_{\mathsf{PD}}}{\epsilon^* - \epsilon} \int_0^{\ell(\epsilon^*-\epsilon)} v_{\mathsf{BP}}(t)\mathrm{d}t\,. \tag{57}$$

For convenience, we have assumed here that iteration $\ell = 0$ corresponds to the beginning of the steady state when the position of the wave is zero.

To declutter notation, we do not use the time t normalized by the distance to the threshold (38) as we do in Section V and in (57). Instead, we use a continuous version of the variable ℓ with a unit of time also measured in BP iterations, denoted by τ. The model (57) can be rewritten in terms of τ as

$$P_{\mathsf{L}}(\ell) = \int_0^{\ell} v_{\mathsf{BP}}(\tau)V_{\mathsf{PD}}\,\mathrm{d}\tau\,. \tag{58}$$

As we do for $v_{\mathsf{BP}}(t)$ in Section V-B2, we model $v_{\mathsf{BP}}(\tau)$ by an Ornstein-Uhlenbeck process of the form (34). The parameters b and σ need to be rescaled relative to those in (42), resulting in

$$\begin{aligned} m &= c_{\mathsf{f}}\,, \quad b = \breve{\theta}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)\,, \\ \sigma^2 &= 2\breve{\theta}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)\frac{\breve{\nu}_{\mathsf{BP}}}{N}\,; \end{aligned} \tag{59}$$

the two models (i.e., the one with rescaled time t and the one with rescaled b and σ) are equivalent and yield identical predictions. We remark that in this section we use the same value for the average steady-state level m of $v_{\mathsf{BP}}(\tau)$ as in the shifted normal approximation (47).

The scaling law for full BP decoding in Section V requires the probability distribution of the time integral of an Ornstein-Uhlenbeck process for a specific t, which is known to be Gaussian (37). The analysis of sliding window decoding requires a more granular approach, since we need to track the evolution of $P_{\mathsf{L}}(\ell)$ over iterations. In other words, we need to treat $P_{\mathsf{L}}(\ell)$ as a stochastic process in its own right. This is complicated by the fact that an integrated Ornstein-Uhlenbeck process of the form (37) or (58) is not Markov. However, the two-dimensional process

$$\left(\eta(\tau) = v_{\mathsf{BP}}(\tau)V_{\mathsf{PD}}\,,\ P_{\mathsf{L}}(\tau) = \int_0^\tau \eta(s)\,\mathrm{d}s\right) \tag{60}$$

is Markov and can be analyzed using standard tools for diffusion processes [27]. (A diffusion process can informally be thought of as a continuous-time Markov processes with continuous sample paths [27].) The corresponding stochastic differential equation is [28, Eq. (35)]

$$\begin{cases} \mathrm{d}\eta(\tau) = -b(\eta(\tau) - m)\mathrm{d}\tau + \sigma_1 \mathrm{d}B_\tau \\ \mathrm{d}P_{\mathsf{L}}(\tau) = \eta(\tau)\mathrm{d}\tau\,, \end{cases} \tag{61}$$

where B_τ is the standard Wiener process. To account for the scaling of the Ornstein-Uhlenbeck process $v_{\mathsf{BP}}(\tau)$ (defined via (34) with parameters (59)) by V_{PD} in (60), m and σ_1 in (61) should be rescaled relative to those in (59), resulting in

$$\begin{aligned} m &= c_{\mathsf{f}}V_{\mathsf{PD}}\,, \quad b = \breve{\theta}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)\,, \\ \sigma_1^2 &= 2\breve{\theta}_{\mathsf{BP}}\left(\epsilon^* - \epsilon\right)V_{\mathsf{PD}}^2\frac{\breve{\nu}_{\mathsf{BP}}}{N}\,. \end{aligned} \tag{62}$$

The need for a subscript in the diffusion coefficient σ_1 will become apparent in the next subsection.

2) *Additional diffusion of the left wave's position:* To answer whether the integrated Ornstein-Uhlenbeck process (61)–(62) is a good model for the propagation distance of the left wave, it can be tested against direct Monte-Carlo simulation of the decoding process. Fig. 15 compares the distribution of $P_{\mathsf{L}}(\ell)$ obtained via direct simulation (blue filled histogram) with the one estimated by simulating $P_{\mathsf{L}}(\tau)$ (red dashed curve) and $\lfloor P_{\mathsf{L}}(\tau)\rfloor$ (green solid curve) using (61)–(62). The integrated Ornstein-Uhlenbeck model captures the average position of the wave relatively well (the means of the blue and green histogram are approximately 78 and 80 positions, respectively, after $\ell = 412$ BP iterations) but underestimates its variance. In other words, the uncertainty in the wave's position does not come solely from the variation in the number of decoded VNs; there must be other factors at play.

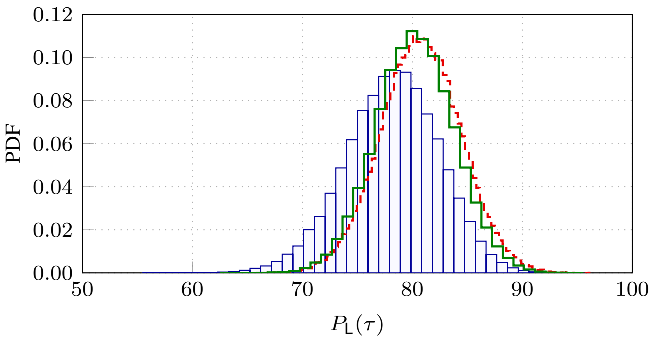

Fig. 15. The histogram of the positions of the left wave $P_{\mathsf{L}}(\tau)$ after $\tau = 412$ BP iterations for the $(5, 10, L, N = 1000)$ SC-LDPC code ensemble with $\epsilon = 0.455$. Direct Monte-Carlo simulation of the decoding process (blue bars), numerical simulation of the integrated Ornstein-Uhlenbeck process (61)–(62): continuous (red dashed curve) and discretized (green solid curve) position. The difference in variance between the blue and green histograms is clear.

We model this additional source of uncertainty via an additional diffusion term that affects $P_{\mathsf{L}}(\tau)$ directly. Throughout iterations, even if the number of decoded VNs is known, the *positions* of these VNs is subject to random fluctuation. As Fig. 15 shows, by time τ that fluctuation accumulates into a difference in variance between simulated $P_{\mathsf{L}}(\ell)$ and $\lfloor P_{\mathsf{L}}(\tau)\rfloor$ from the model (61)–(62). We denote the variance of the simulated $P_{\mathsf{L}}(\ell)$ by $\sigma^2_{\mathsf{sim}}(\ell)$ and that of $\lfloor P_{\mathsf{L}}(\tau)\rfloor$ from (61)–(62) by $\sigma^2_{\mathsf{model}}(\ell)$. We introduce a parameter σ_2^2 that corresponds to additional per-iteration variance in position,

$$\sigma_2^2 = \frac{\sigma^2_{\mathsf{sim}}(\ell) - \sigma^2_{\mathsf{model}}(\ell)}{\ell}\,, \tag{63}$$

and incorporate that additional source of variance into our model by changing (61) to

$$\begin{cases} \mathrm{d}\eta(\tau) = -b(\eta(\tau) - m)\mathrm{d}\tau + \sigma_1 \mathrm{d}B_\tau \\ \mathrm{d}P_{\mathsf{L}}(\tau) = \eta(\tau)\mathrm{d}\tau + \sigma_2 \mathrm{d}B'_\tau\,, \end{cases} \tag{64}$$

where B'_τ is another standard Wiener process independent of B_τ. The rationale behind this model is that the variance of the standard Wiener process grows linearly with time [27]; adding the term $\sigma_2 \mathrm{d}B'_\tau$ in (64) results in an additional variance of $\sigma_2^2\ell$ in $P_{\mathsf{L}}(\ell)$ compared with the model in (61), thereby canceling the mismatch between the simulated and the modeled variance in (63) but "spreading" this variance evenly across iterations.

We treat σ_2 as a scaling parameter that depends on $(d_{\mathsf{v}}, d_{\mathsf{c}})$ only. We rely on a set of simulated realizations of the decoding process for a certain triple (ϵ, N, ℓ) to estimate σ^2_{sim}. For our running example of the $(5, 10, L, N)$ SC-LDPC code ensemble, we use $(\epsilon = 0.455, N = 1000, \ell = 412)$ and obtain $\sigma_2 \approx 0.1179$.

As an aside, we remark that the video streaming analogy we introduced in Section VI-A can be stretched further to include the additional uncertainty we are modeling here. Indeed, the amount of downloaded data is not the only factor that determines how much time of playback is gained. This is also affected by, e.g., how much movement occurs in consecutive frames of the video, assuming the video is compressed before transmission.

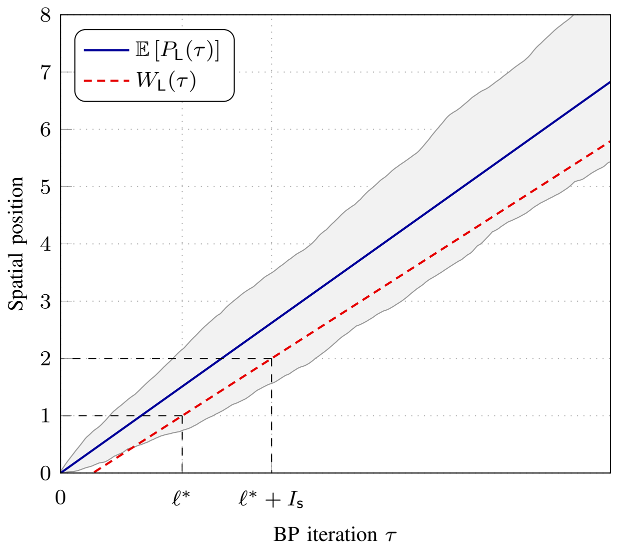

Fig. 16. Schematic illustration of the proposed model for the race between the left wave $P_{\mathsf{L}}(\tau)$ and the left boundary of the window $W_{\mathsf{L}}(\tau)$ from (66). The average position of the wave is shown as the blue solid line. The position of the window is shown as the red dashed line. The gray curves represent the minimum and maximum values observed for the position of the wave for a number of simulated realizations of $P_{\mathsf{L}}(\tau)$. The range of values between them is shadowed. It is apparent that the spread of $P_{\mathsf{L}}(\tau)$ grows with τ. Even though the wave (blue solid line) moves faster than the window (red dashed line) on average, some realizations of $P_{\mathsf{L}}(\tau)$ do get overtaken by the window.

3) Modeling the boundaries of the sliding window: We are now ready to incorporate the sliding window into our model. In essence, we do so by limiting the process $P_{\mathsf{L}}(\ell)$ to the range from $W_{\mathsf{L}}(\ell)$ to $W_{\mathsf{R}}(\ell)$, i.e., to the range of positions covered by the sliding window at iteration ℓ. Indeed, $P_{\mathsf{L}}(\ell)$ cannot be larger than $W_{\mathsf{R}}(\ell)$ because sliding window decoding does not perform any BP iterations there. Similarly, if $P_{\mathsf{L}}(\ell)$ ever becomes smaller than $W_{\mathsf{L}}(\ell)$, the overtaking O happens and decoding fails. We account for these effects in our model by introducing an absorbing barrier at $W_{\mathsf{L}}(\ell)$ and a reflecting barrier at $W_{\mathsf{R}}(\ell)$.

During decoding, the sliding window moves in discrete steps. It is convenient to smoothen this movement and assume it to be continuous and linear instead. We denote the continuously moving boundaries of the window by $W_{\mathsf{L}}(\tau)$ and $W_{\mathsf{R}}(\tau)$. The setup we use is illustrated in Fig. 16. The red dashed line corresponds to $W_{\mathsf{L}}(\tau)$, which will be included in the model as an absorbing barrier. Its slope is known and equal to V_{W}. What remains to be specified is the iteration where $W_{\mathsf{L}}(\tau)$ crosses 1, which we denote by ℓ^* (see Fig. 16).

The value of ℓ^* is what couples the model for $P_{\mathsf{L}}(\tau)$ with $W_{\mathsf{L}}(\tau)$. First, ℓ^* depends not only on I_{in} but also on the time it takes for the wave to form, I_{start}. A natural choice for ℓ^* in that regard would be $I_{\mathsf{in}} - I_{\mathsf{start}}$: it takes I_{start} iterations for the steady state to establish, which eats away from the budget of I_{in} initial iterations, and then the window slides to position 1. However, numerical simulations show that $P_{\mathsf{L}}(I_{\mathsf{start}})$ is already around one, not zero. In our model, we would like instead to set the initial position of the wave to zero. We therefore subtract from I_{start} a term that corresponds to m^{-1} in (62), which is the average time it takes for the wave to propagate by one position, and set

$$\ell^* = I_{\mathsf{in}} - I_{\mathsf{start}} + c_{\mathsf{f}}^{-1} V_{\mathsf{PD}}^{-1} \,. \tag{65}$$

Using our knowledge that $W_{\mathsf{L}}(\ell^*) = 1$ and that the slope of $W_{\mathsf{L}}(\tau)$ is V_{W}, we obtain the continuous version of $W_{\mathsf{L}}(\ell)$ as

$$W_{\mathsf{L}}(\tau) = \tau V_{\mathsf{W}} + 1 - \ell^* V_{\mathsf{W}} \,. \tag{66}$$

For simplicity, we ignore the fact that $W_{\mathsf{R}}(\ell)$ remains flat for the first ℓ^* iterations and let

$$W_{\mathsf{R}}(\tau) = W_{\mathsf{L}}(\tau) + W \,. \tag{67}$$

With (66)–(67) we have ensured that the boundaries of the window, $W_{\mathsf{L}}(\tau)$ and $W_{\mathsf{R}}(\tau)$, grow linearly with τ. They correspond to time-dependent absorbing and reflecting barriers and make the domain of $P_{\mathsf{L}}(\tau)$ change over time. We remove that dependency by subtracting the time-dependent term τV_{W} from $P_{\mathsf{L}}(\tau)$. That corresponds to subtracting V_{W} from $\eta(\tau)$, which can in turn be expressed as a change in m.

Putting it all together, we use the model (64) with

$$\begin{aligned} m &= c_{\mathsf{f}} V_{\mathsf{PD}} - V_{\mathsf{W}} \,, \quad b = \breve{\theta}_{\mathsf{BP}} \left(\epsilon^* - \epsilon\right) \,, \\ \sigma_1^2 &= 2\breve{\theta}_{\mathsf{BP}} \left(\epsilon^* - \epsilon\right) V_{\mathsf{PD}}^2 \frac{\breve{\nu}_{\mathsf{BP}}}{N} \,, \end{aligned} \tag{68}$$

and the boundaries

$$W_{\mathsf{L}} = 1 - \ell^* V_{\mathsf{W}} \,, \quad \text{absorbing} \,, \tag{69}$$

$$W_{\mathsf{R}} = W_{\mathsf{L}} + W \,, \quad \text{reflecting} \,, \tag{70}$$

which no longer depend on τ. The initial conditions are $\eta(0) \sim \mathcal{N}\left(m, \sigma_1^2/(2b)\right)$ as in (35) and $P_{\mathsf{L}}(0) = 0$.

The probability of the overtaking event O corresponds to the probability of $P_{\mathsf{L}}(\tau)$ being absorbed at the barrier W_{L} by the time $\ell^* + (L-1) I_{\mathsf{s}}$. The latter probability can be obtained by solving the Fokker-Planck equation for $(\eta(\tau), P_{\mathsf{L}}(\tau))$ with appropriately chosen initial and boundary conditions. The following subsection provides a summary of the associated results for general multidimensional diffusion processes.

4) Necessary background on the Fokker-Planck equation: Let $\boldsymbol{Y}_\tau$ be a time-homogeneous diffusion process on $\mathbb{R}^d$ defined as the solution to the Itô stochastic differential equation [27, Ch. 3]

$$\mathrm{d}\boldsymbol{Y}(\tau) = \boldsymbol{b}(\boldsymbol{Y}(\tau))\mathrm{d}\tau + \boldsymbol{\sigma}(\boldsymbol{Y}(\tau))\mathrm{d}\boldsymbol{B}_\tau \,, \tag{71}$$

where $\boldsymbol{B}_\tau$ is the standard Wiener process on $\mathbb{R}^n$, $\boldsymbol{b}(\boldsymbol{Y}) \in \mathbb{R}^d$ is a drift vector, and $\boldsymbol{\sigma}(\boldsymbol{Y}) \in \mathbb{R}^{d\times n}$. The matrix $\boldsymbol{\Sigma}(\boldsymbol{Y}) = \boldsymbol{\sigma}(\boldsymbol{Y})\boldsymbol{\sigma}(\boldsymbol{Y})^\mathsf{T} \in \mathbb{R}^{d\times d}$ is known as the diffusion matrix of $\boldsymbol{Y}(\tau)$. Let the initial condition $\boldsymbol{Y}(0)$ be a random vector with probability density $p_0(\boldsymbol{Y})$ independent of $\boldsymbol{B}_\tau$. Then the probability density $p(\boldsymbol{Y},\tau)$ of $\boldsymbol{Y}(\tau)$ is the solution to the initial value problem for the Fokker-Planck (also known as forward Kolmogorov) equation

$$\begin{aligned} \frac{\partial p}{\partial \tau} &= \nabla \cdot \left(-\boldsymbol{b}(\boldsymbol{Y})p + \frac{1}{2}\nabla \cdot (\boldsymbol{\Sigma}(\boldsymbol{Y})p)\right) \\ &= -\sum_{i=1}^{d} \frac{\partial}{\partial y_i}\left(b_i(\boldsymbol{Y})p\right) + \frac{1}{2}\sum_{i,j=1}^{d} \frac{\partial^2}{\partial y_i \partial y_j}\left(\Sigma_{ij}(\boldsymbol{Y})p\right) \,, \end{aligned} \tag{72}$$

$$p(\boldsymbol{Y},0) = p_0(\boldsymbol{Y}) \,,$$

where $y_{\{i,j\}}$, $b_i(\boldsymbol{Y})$, and $\Sigma_{ij}(\boldsymbol{Y})$ denote the components of $\boldsymbol{Y}$, $\boldsymbol{b}(\boldsymbol{Y})$, and $\boldsymbol{\Sigma}(\boldsymbol{Y})$, respectively [27, Proposition 3.3 and Eq. (4.1)]. The Fokker-Planck equation describes the evolution of the probability density of $\boldsymbol{Y}(\tau)$ over time.

Absorbing or reflecting barriers affect $p(\boldsymbol{Y},\tau)$ through additional boundary conditions for the Fokker-Planck equation (72). An absorbing barrier $\mathfrak{B}_{\mathsf{a}}$ imposes a Dirichlet boundary condition on (72), namely

$$p(\boldsymbol{Y},\tau)=0 \quad \forall \boldsymbol{Y}\in\mathfrak{B}_{\mathsf{a}}\,. \tag{73}$$

Likewise, a reflecting barrier $\mathfrak{B}_{\mathsf{r}}$ translates into a Neumann boundary condition for (72). Specifically,

$$\boldsymbol{n}\cdot\left(\boldsymbol{b}(\boldsymbol{Y})p-\frac{1}{2}\nabla\cdot(\boldsymbol{\Sigma}(\boldsymbol{Y})p)\right)=0 \quad \forall \boldsymbol{Y}\in\mathfrak{B}_{\mathsf{r}}\,, \tag{74}$$

where $\boldsymbol{n}$ denotes a vector normal to $\mathfrak{B}_{\mathsf{r}}$. In other words, the probability density must vanish at the absorbing barrier $\mathfrak{B}_{\mathsf{a}}$, and there should be no probability flow at the reflecting barrier $\mathfrak{B}_{\mathsf{r}}$ [27, pp. 90–91]. The probability mass lost by $p(\boldsymbol{Y},\tau)$ is equal to the probability of $\boldsymbol{Y}(\tau)$ having been absorbed at $\mathfrak{B}_{\mathsf{a}}$ by the time τ,

$$\Pr\{\boldsymbol{Y}(\tau)\text{ absorbed at }\mathfrak{B}_{\mathsf{a}}\}=1-\int_{\Omega}p(\boldsymbol{Y},\tau)\mathrm{d}\boldsymbol{Y}\,, \tag{75}$$

assuming $\mathfrak{B}_{\mathsf{a}}$ to be the only absorbing barrier present [27, p. 239]. The integration is performed over Ω that denotes the domain of $\boldsymbol{Y}(\tau)$.

5) The Fokker-Planck equation for the proposed model: The model (64) is of the form (71) with

$$\begin{aligned}\boldsymbol{Y}(\tau)&=\begin{bmatrix}\eta(\tau)\\ P_{\mathsf{L}}(\tau)\end{bmatrix},\quad \boldsymbol{b}(\boldsymbol{Y})=\begin{bmatrix}-b(\eta-m)\\ \eta\end{bmatrix},\\ \boldsymbol{\sigma}(\boldsymbol{Y})&=\begin{bmatrix}\sigma_1 & 0\\ 0 & \sigma_2\end{bmatrix},\quad\text{and } \boldsymbol{B}_\tau=\begin{bmatrix}B_\tau\\ B'_\tau\end{bmatrix}.\end{aligned} \tag{76}$$

We can therefore use (72)–(74) to derive the Fokker-Planck equation for the evolution of the PDF $p(\eta,P_{\mathsf{L}},\tau)$ of the process (64) as

$$\begin{aligned}\frac{\partial p}{\partial\tau}&=\frac{\partial b\,(\eta-m)\,p}{\partial\eta}-\eta\frac{\partial p}{\partial P_{\mathsf{L}}}+\frac{\sigma_1^2}{2}\frac{\partial^2 p}{\partial\eta^2}+\frac{\sigma_2^2}{2}\frac{\partial^2 p}{\partial P_{\mathsf{L}}^2}\,,\\ p(\eta,P_{\mathsf{L}},0)&=p_0(\eta,P_{\mathsf{L}})\,,\end{aligned} \tag{77}$$

with boundary conditions

$$\begin{aligned}&p\,(\eta,W_{\mathsf{L}},\tau)=0\,,\\ &\eta p\,(\eta,W_{\mathsf{R}},\tau)-\frac{\sigma_2^2}{2}\left.\frac{\partial p\,(\eta,P_{\mathsf{L}},\tau)}{\partial P_{\mathsf{L}}}\right|_{P_{\mathsf{L}}=W_{\mathsf{R}}}=0\,,\end{aligned} \tag{78}$$

where W_{L} and W_{R} are defined in (69) and (70), respectively, and the parameters (m,b,σ_1,σ_2) are given in (68) and (63).

As we specify in Section VI-D3, the initial distribution of $\eta(\tau)$ and $P_{\mathsf{L}}(\tau)$ should be $\eta(0)\sim\mathcal{N}\left(m,\sigma_{\mathsf{st}}^2\right)$ and $P_{\mathsf{L}}(0)=0$, where $\sigma_{\mathsf{st}}^2=\sigma_1^2/(2b)$. We should therefore set $p_0(\eta,P_{\mathsf{L}})$ to a PDF whose marginal for η is the PDF of $\mathcal{N}\left(m,\sigma_{\mathsf{st}}^2\right)$ and for P_{L} the Dirac delta function. Handling such a distribution numerically is challenging; instead, we set $p_0(\eta,P_{\mathsf{L}})$ to the PDF of the correlated two-dimensional Gaussian distribution

$$\mathcal{N}\left(\begin{bmatrix}m\\ 0\end{bmatrix},\begin{bmatrix}\sigma_{\mathsf{st}}^2 & \rho\delta\sigma_{\mathsf{st}}\\ \rho\delta\sigma_{\mathsf{st}} & \delta^2\end{bmatrix}\right), \tag{79}$$

where $\rho\to 1$ and $\delta\to 0$. As $\delta\to 0$, the PDF of P_{L} tends to the Dirac delta function as required; the correlation parameter ρ should tend to 1 because as $\tau\to 0$, $\eta(\tau)$ and its integral $P_{\mathsf{L}}(\tau)$ become ever more dependent. The use of the PDF of (79) allows us to avoid the aforementioned numerical issues by backing ρ and δ off from their limits.

6) Numerical solution to the Fokker-Planck equation: To the best of our knowledge, the closed-form solution to the Fokker-Planck equation (77) in the presence of the boundary conditions (78) is not available. We therefore resort to solving (77)–(78) numerically using FiPy, a finite-volume solver of partial differential equations [29]. The range of η is limited to $m\pm 4\sigma_{\mathsf{st}}$ to cover most of the probability mass without overstretching the domain, and that of P_{L} to $[W_{\mathsf{L}},W_{\mathsf{R}}]$. The resulting rectangular domain is discretized into a regular grid with η and P_{L} split into 200 and $20W$ segments, respectively. We set $\rho=0.99$ and $\delta=0.1$. The solver treats (77) as a convection-diffusion equation; we use implicit convection and diffusion terms [29], which allows us to choose a large time step 1 without encountering numerical stability issues.

The numerical solution is propagated forward in time until the window is slid through the entire chain, i.e., until

$$\tau^*=\ell^*+(L-1)I_{\mathsf{s}}\,. \tag{80}$$

The PDF $p(\eta,P_{\mathsf{L}},\tau^*)$ is used to estimate the probability of the overtaking event O according to (75) as

$$\Pr\{O\}=1-\int_{W_{\mathsf{L}}}^{W_{\mathsf{R}}}\int_{m-4\sigma_{\mathsf{st}}}^{m+4\sigma_{\mathsf{st}}}p(\eta,P_{\mathsf{L}},\tau^*)\,\mathrm{d}\eta\,\mathrm{d}P_{\mathsf{L}}\,. \tag{81}$$

The estimated $\Pr\{O\}$ is then used in (54) to estimate the FER.

In addition to the parameters required for the scaling law based on the shifted normal approximation, reviewed in Section V-C, the proposed scaling law for sliding window decoding with a limited number of iterations requires V_{PD}, V_{BP}, and σ_2; V_{PD} can be obtained from mean evolution for several ϵ as shown in (18), V_{BP} can either be obtained directly from density evolution or from V_{PD} using the approximation (22), and σ_2 requires Monte-Carlo simulation of the BP decoder and of the integrated Ornstein-Uhlenbeck process (61)–(62) for a single fixed triple (ϵ,N,ℓ). It is thus relatively easy to obtain V_{PD} and V_{BP}; estimating σ_2 more efficiently, on the other hand, is an open problem of practical interest.

We note that our implementation of the solver takes longer to estimate the FER than direct Euler-Maruyama simulation of 10^4 realizations of (64) to the same end. However, the computational complexity of the estimation based on the Fokker-Planck equation does not depend on the FER to attain a given accuracy, which is not the case for the simulation-based approach. Moreover, no attempt has been made to optimize either implementation.

E. Numerical Results

Fig. 17 compares the FER for the $(5,10,L=50,N=1000)$ SC-LDPC code ensemble under sliding window decoding with $W=20$ and $I_{\mathsf{in}}=60$ for $I_{\mathsf{s}}=\{6,7,10\}$ (solid

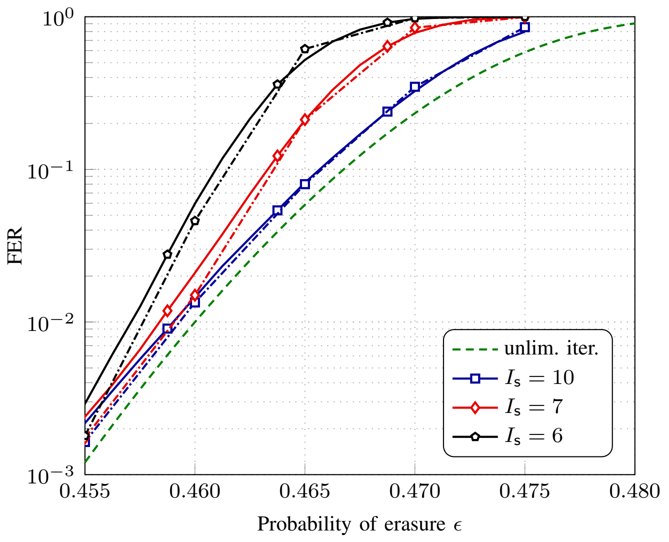

Fig. 17. FER for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under sliding window decoding with $W = 20$ and $I_{in} = 60$ for different I_s (solid curves) and its approximation (54) with Fokker-Planck-based estimation of the overtaking probability O (corresponding dash-dotted curves). The green dashed line corresponds to our approximation for the FER with unlimited number of iterations from [15].

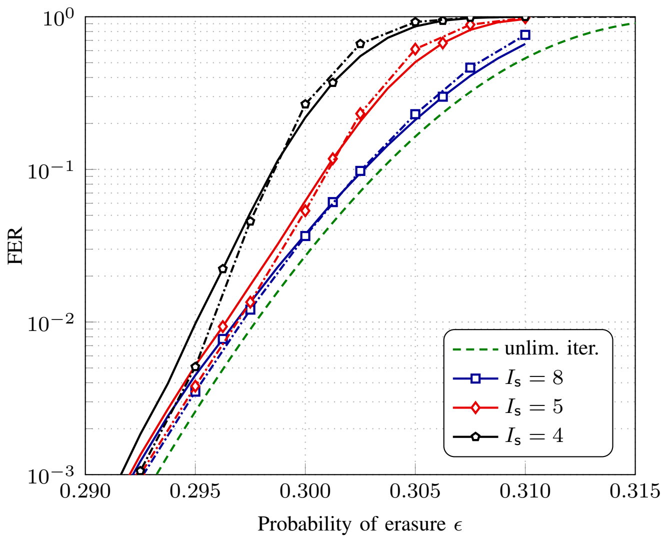

Fig. 19. FER for the $(4, 12, L = 50, N = 996)$ SC-LDPC code ensemble under sliding window decoding with $W = 20$ and $I_{in} = 60$ for different I_s (solid curves) and its approximation (54) with Fokker-Planck-based estimation of the overtaking probability O (corresponding dash-dotted curves). The green dashed line corresponds to our approximation for the FER with unlimited number of iterations from [15].

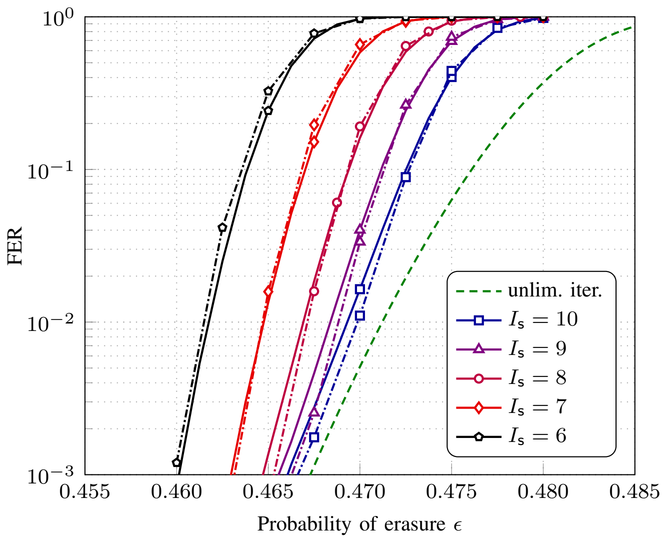

Fig. 18. FER for the $(5, 10, L = 50, N = 2000)$ SC-LDPC code ensemble under sliding window decoding with $W = 20$ and $I_{in} = 60$ for different I_s (solid curves) and its approximation (54) with Fokker-Planck-based estimation of the overtaking probability O (corresponding dash-dotted curves). The green dashed line corresponds to our approximation for the FER with unlimited number of iterations from [15].

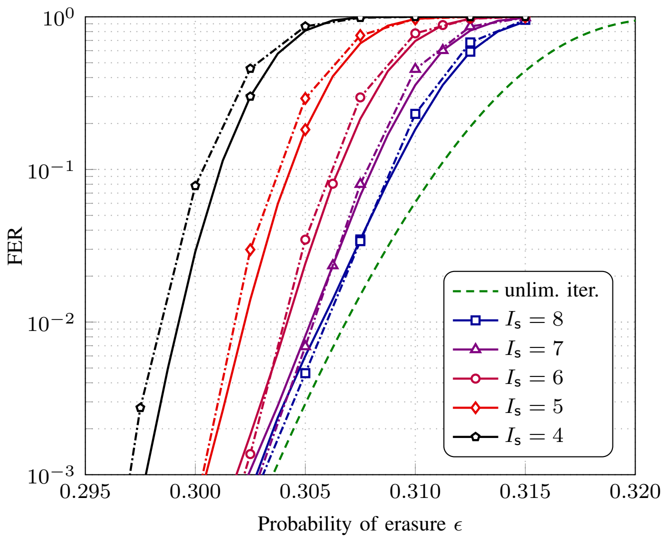

Fig. 20. FER for the $(4, 12, L = 50, N = 1992)$ SC-LDPC code ensemble under sliding window decoding with $W = 20$ and $I_{in} = 60$ for different I_s (solid curves) and its approximation (54) with Fokker-Planck-based estimation of the overtaking probability O (corresponding dash-dotted curves). The green dashed line corresponds to our approximation for the FER with unlimited number of iterations from [15].

curves) with the corresponding approximation using (54), where the overtaking probability O is estimated using (81) (corresponding dash-dotted curves). We observe an impressive match between the simulated and predicted error rates. A similarly accurate prediction is obtained for $N = 2000$ in Fig. 18 and for other values of (d_v, d_c) with $d_v > 3$.

Figs. 19 and 20 compare the simulated (solid curves) and predicted (corresponding dash-dotted curves) FER performance for the $(4, 12, L = 50, N)$ SC-LDPC code ensemble under sliding window decoding with $W = 20, I_{in} = 60$, and different values of I_s for $N = 996$ and $N = 1992$, respectively. For this ensemble, the scaling parameters are estimated as $\breve{\nu} = 0.307, \breve{\theta} = 2.172, \breve{\nu}_{BP} = 0.285, \breve{\theta}_{BP} = 2.921$, and $\sigma_2 = 0.2056$. As for the $(5, 10, L, N)$ SC-LDPC code ensemble, the predictions for the $(4, 12, L, N)$ SC-LDPC code ensemble are fairly accurate.

The quality of the prediction deteriorates for smaller I_{in}, as exemplified in Fig. 21 for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under sliding window decoding with $W = 20, I_s = 9$ and $I_{in} = 25$. We observe that degradation when I_{in} is such that the left wave is often overtaken at the very beginning of the chain. This setup, however, is of limited practical relevance—it is sensible for the system designer to ensure that the wave is firmly established by choosing I_{in}

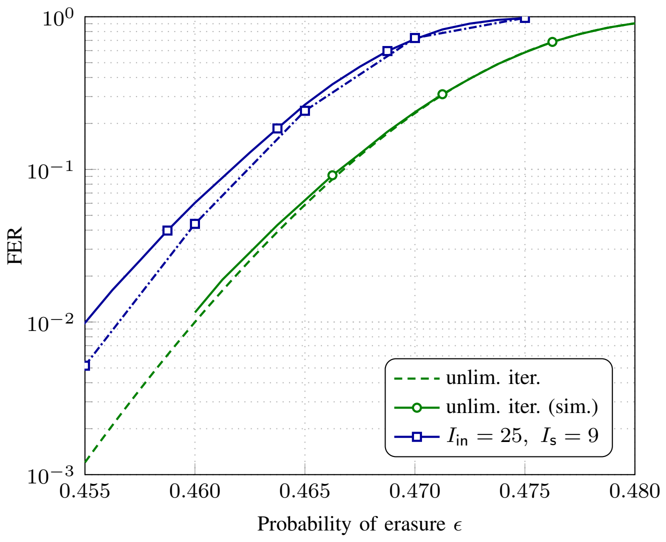

Fig. 21. FER for the $(5, 10, L = 50, N = 1000)$ SC-LDPC code ensemble under sliding window decoding with $W = 20$, $I_{\mathsf{in}} = 25$, and $I_{\mathsf{s}} = 9$ (solid curve with squares) and its approximation (54) with Fokker-Planck-based estimation of the overtaking probability O (dash-dotted curve with squares). The green dashed line corresponds to our approximation for the FER with unlimited number of iterations from [15].

sufficiently large. This also lowers the probability that the wave is overtaken by the window early on during decoding if wave propagation happens to slow down for some time. We have included in Fig. 21 the simulated FER curve for unlimited number of iterations (green solid curve with circles) alongside our prediction from [15] using (17) (green dashed curve) for reference.

We remark that we use $\breve{\theta}_{\mathsf{BP}} \approx 2.34$ estimated at $(\epsilon = 0.455, N = 1000)$ to obtain the predictions in Figs. 17, 18, and 21. This is smaller than $\breve{\theta}_{\mathsf{BP}} \approx 2.74$ used in Section V and estimated at $(\epsilon = 0.465, N = 5000)$. Choosing the latter value makes the prediction curves slightly more optimistic. The covariance decay parameter $\breve{\theta}_{\mathsf{BP}}$ seems to exhibit a stronger dependency on N than $\breve{\theta}$ in the case of peeling decoding; we leave the investigation of this dependency as a subject of future work. As a general rule of thumb, one should estimate $\breve{\theta}_{\mathsf{BP}}$ for (ϵ, N) that lie within the range one is interested in.

VII. Conclusion and Discussion

The proposed scaling laws for full BP decoding with a limited number of iterations and a scaling law for sliding window decoding with a limited number of iterations provide accurate predictions of the FER.[2] Modeling the number of bits decoded in a given number of iterations by a time integral of an Ornstein-Uhlenbeck process—the cornerstone of our scaling laws—proves to be a powerful tool in the analysis of decoding schemes with practically relevant constraints on the maximum number of iterations. More broadly, low-dimensional diffusion processes seem to be able to capture much of the behavior of iterative decoders relevant for error rate prediction in the waterfall region.

Another takeaway is that it is important for sliding window decoding to perform a sufficient number of iterations at the beginning of the chain. It is necessary not only because the decoder must ensure that the decoding wave is established, but also because allowing the decoding wave to propagate further inside the window builds up the decoder's resilience to variation in the wave's propagation speed.

We also remark that the scaling laws we propose in this paper can be extended to predict the bit and block error rate using the same techniques we employed in [15] to the same end. (Block error rate refers to the probability that a spatial position—a *block*—contains unrecovered bits after decoding.) Specifically, the models for both full BP decoding in Sections IV–V and for sliding window decoding in Section VI already keep track of the number of bits decoded across iterations implicitly; what is required to obtain a scaling law for bit and block error rate is to make use of this knowledge and average the bit and block error rate expressions over the probability distributions of when decoding stops, which we did in [15] in the context of unlimited number of iterations.

Further, we do not foresee substantial difficulties in extending the scaling laws proposed here to protograph-based ensembles. What is required to do so is to modify density and mean evolution equations to account for the changed Tanner graph connectivity, as it is done in [13] for unlimited number of decoding iterations.

[2]An implementation of the proposed scaling laws along with density and mean evolution and the necessary Monte-Carlo simulators is available at https://github.com/rsokolovskii/fl_scaling_sc_ldpc

References

[1] A. Jimenéz Feltström and K. S. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 2181–2191, Sep. 1999.

[2] M. Lentmaier, A. Sridharan, D. J. Costello, and K. S. Zigangirov, "Iterative decoding threshold analysis for LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 5274–5289, Oct. 2010.

[3] S. Kudekar, T. J. Richardson, and R. L. Urbanke, "Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 803–834, Feb. 2011.

[4] S. Kudekar, T. Richardson, and R. L. Urbanke, "Spatially coupled ensembles universally achieve capacity under belief propagation," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7761–7813, Dec. 2013.

[5] A. Sridharan, D. Truhachev, M. Lentmaier, D. J. Costello, and K. S. Zigangirov, "Distance bounds for an ensemble of LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4537–4555, Dec. 2007.

[6] S. Moloudi, M. Lentmaier, and A. Graell i Amat, "Spatially coupled turbo-like codes," *IEEE Trans. Inf. Theory*, vol. 63, no. 10, pp. 6199–6215, Oct. 2017.

[7] B. P. Smith, A. Farhood, A. Hunt, F. R. Kschischang, and J. Lodge, "Staircase codes: FEC for 100 Gb/s OTN," *J. Lightw. Technol.*, vol. 30, no. 1, pp. 110–117, Jan. 2012.

[8] V. Aref, N. Macris, R. Urbanke, and M. Vuffray, "Lossy source coding via spatially coupled LDGM ensembles," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Cambridge, MA, USA, Jul. 2012, pp. 373–377.

[9] D. L. Donoho, A. Javanmard, and A. Montanari, "Information-theoretically optimal compressed sensing via spatial coupling and approximate message passing," *IEEE Trans. Inf. Theory*, vol. 59, no. 11, pp. 7434–7464, Nov. 2013.

[10] A. R. Iyengar, M. Papaleo, P. H. Siegel, J. K. Wolf, A. Vanelli-Coralli, and G. E. Corazza, "Windowed decoding of protograph-based LDPC convolutional codes over erasure channels," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2303–2320, Apr. 2012.

[11] P. M. Olmos and R. L. Urbanke, "A scaling law to predict the finite-length performance of spatially-coupled LDPC codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 6, pp. 3164–3184, Jun. 2015.

[12] A. Amraoui, A. Montanari, T. Richardson, and R. Urbanke, "Finite-length scaling for iteratively decoded LDPC ensembles," *IEEE Trans. Inf. Theory*, vol. 55, no. 2, pp. 473–498, Feb. 2009.

[13] M. Stinner and P. M. Olmos, “On the waterfall performance of finite-length SC-LDPC codes constructed from protographs,” *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 345–361, Feb. 2016.

[14] D. J. Costello, D. G. M. Mitchell, P. M. Olmos, and M. Lentmaier, “Spatially coupled generalized LDPC codes: Introduction and overview,” in *Proc. 10th IEEE Int. Symp. Turbo Codes and Iterative Inf. Process. (ISTC)*, Hong Kong, China, Dec. 2018.

[15] R. Sokolovskii, A. Graell i Amat, and F. Brännström, “Finite-length scaling of spatially coupled LDPC codes under window decoding over the BEC,” *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 5988–5998, Oct. 2020.

[16] H.-Y. Kwak, J.-W. Kim, and J.-S. No, “Optimizing code parameters of finite-length SC-LDPC codes using the scaling law,” *IEEE Access*, vol. 9, pp. 118 640–118 650, Aug. 2021.

[17] R. Sokolovskii, A. Graell i Amat, and F. Brännström, “On doped SC-LDPC codes for streaming,” *IEEE Commun. Lett.*, vol. 25, no. 7, pp. 2123–2127, Jul. 2021.

[18] H.-Y. Kwak, J.-W. Kim, H. Park, and J.-S. No, “Optimization of SC-LDPC codes for window decoding with target window sizes,” *IEEE Trans. Commun.*, vol. 70, no. 5, pp. 2924–2938, May 2022.

[19] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, “Practical loss-resilient codes,” in *Proc. Annu. ACM Symp. Theory Comput. (STOC)*, El Paso, TX, USA, 1997, pp. 150–159.

[20] D. G. M. Mitchell, A. E. Pusane, and D. J. Costello, “Minimum distance and trapping set analysis of protograph-based LDPC convolutional codes,” *IEEE Trans. Inf. Theory*, vol. 59, no. 1, pp. 254–281, Jan. 2013.

[21] H. Esfahanizadeh, A. Hareedy, and L. Dolecek, “Finite-length construction of high performance spatially-coupled codes via optimized partitioning and lifting,” *IEEE Trans. Commun.*, vol. 67, no. 1, pp. 3–16, Jan. 2019.

[22] S. Naseri and A. H. Banihashemi, “Construction of time invariant spatially coupled LDPC codes free of small trapping sets,” *IEEE Trans. Commun.*, vol. 69, no. 6, pp. 3485–3501, Jun. 2021.

[23] H. Hatami, D. G. M. Mitchell, D. J. Costello, and T. Fuja, “Performance bounds for quantized spatially coupled LDPC decoders based on absorbing sets,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 826–830.

[24] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, “Efficient erasure correcting codes,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 569–584, Feb. 2001.

[25] M. Stinner, L. Barletta, and P. M. Olmos, “Finite-length scaling based on belief propagation for spatially coupled LDPC codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 2109–2113.

[26] G. E. Uhlenbeck and L. S. Ornstein, “On the theory of the brownian motion,” *Phys. Rev.*, vol. 36, pp. 823–841, Sep. 1930.

[27] G. A. Pavliotis, *Stochastic Processes and Applications*. Springer, New York, NY, USA, 2014.

[28] E. Benedetto, L. Sacerdote, and C. Zucca, “A first passage problem for a bivariate diffusion process: Numerical solution with an application to neuroscience when the process is Gauss-Markov,” *J. Comput. Appl. Math.*, vol. 242, pp. 41–52, 2013.

[29] J. E. Guyer, D. Wheeler, and J. A. Warren, “FiPy: Partial differential equations with Python,” *Comput. Sci. Eng.*, vol. 11, no. 3, pp. 6–15, 2009. [Online]. Available: http://www.ctcms.nist.gov/fipy