

THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Finite-Length Scaling Laws for Spatially-Coupled LDPC Codes

ROMAN SOKOLOVSKII



CHALMERS
UNIVERSITY OF TECHNOLOGY

Communication Systems Group
Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2022

Finite-Length Scaling Laws for Spatially-Coupled LDPC Codes

ROMAN SOKOLOVSKII

Copyright © 2022 ROMAN SOKOLOVSKII, except where otherwise stated. All rights reserved.

ISBN 978-91-7905-660-5

Doktorsavhandlingar vid Chalmers tekniska högskola

Ny serie nr 5126

ISSN 0346-718X

This thesis has been prepared using L^AT_EX, PGF/TikZ, and PSTricks.

Cover photograph (taken by the author): A waterfall in Norway.

Communication Systems Group
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
Phone: +46 (0)31 772 1000
www.chalmers.se

Printed by Chalmers Reproservice
Gothenburg, Sweden, May 2022

To my grandmother

Abstract

This thesis concerns predicting the finite-length error-correcting performance of spatially-coupled low-density parity-check (SC-LDPC) code ensembles over the binary erasure channel. SC-LDPC codes are a very powerful class of codes; their use in practical communication systems, however, requires the system designer to specify a considerable number of code and decoder parameters, all of which affect both the code's error-correcting capability and the system's memory, energy, and latency requirements. Navigating the space of the associated trade-offs is challenging. The aim of the finite-length scaling laws proposed in this thesis is to facilitate code and decoder parameter optimization by providing a way to predict the code's error-rate performance without resorting to Monte-Carlo simulations for each combination of code/decoder and channel parameters.

First, we tackle the problem of predicting the frame, bit, and block error rate of SC-LDPC code ensembles over the binary erasure channel under both belief propagation (BP) decoding and sliding window decoding when the maximum number of decoding iterations is unlimited. The scaling laws we develop provide very accurate predictions of the error rates.

Second, we derive a scaling law to accurately predict the bit and block error rate of SC-LDPC code ensembles with doping, a technique relevant for streaming applications for limiting the inherent rate loss of SC-LDPC codes. We then use the derived scaling law for code parameter optimization and show that doping can offer a way to achieve better transmission rates for the same target bit error rate than is possible without doping.

Last, we address the most challenging (and most practically relevant) case where the maximum number of decoding iterations is limited, both for BP and sliding window decoding. The resulting predictions are again very accurate.

Together, these contributions make finite-length SC-LDPC code and decoder parameter optimization via finite-length scaling laws feasible for the design of practical communication systems.

Keywords: Belief propagation decoding, codes-on-graphs, finite-length code performance, spatially-coupled LDPC codes, window decoding.

List of Publications

This thesis is based on the following publications:

[A] **Roman Sokolovskii**, Alexandre Graell i Amat, and Fredrik Brännström, “Finite-Length Scaling of Spatially Coupled LDPC Codes Under Window Decoding Over the BEC,” *IEEE Transactions on Communications*, vol. 68, no. 10, pp. 5988–5998, October 2020.

[B] **Roman Sokolovskii**, Alexandre Graell i Amat, and Fredrik Brännström, “On Doped SC-LDPC Codes for Streaming,” *IEEE Communications Letters*, vol. 25, no. 7, pp. 2123–2127, July 2021.

[C] **Roman Sokolovskii**, Alexandre Graell i Amat, and Fredrik Brännström, “Finite-Length Scaling of SC-LDPC Codes With a Limited Number of Decoding Iterations,” submitted to *IEEE Transactions on Information Theory*, March 2022.

Other publications by the author, not included in this thesis, are:

[D] **Roman Sokolovskii**, Fredrik Brännström, and Alexandre Graell i Amat, “A Refined Scaling Law for Spatially Coupled LDPC Codes Over the Binary Erasure Channel,” in *Proc. IEEE Information Theory Workshop (ITW)*, Visby, Sweden, Aug. 2019. IEEE Sweden VT/COM/IT Chapter Best Student Conference Paper Award 2020.

Acknowledgments

I acknowledge with pleasure my debts to many people for their help on my Ph.D. journey.

I owe special thanks to my supervisor Professor Alexandre Graell i Amat. Àlex, you showed me by example what it means to be a scientist, a writer, a mentor—all in one. Thank you for your passion, commitment, and brilliance. I am honoured to be able to call you my friend.

I am also very grateful to my supervisor Professor Fredrik Brännström. Thank you for your support, encouragement, and for all the times you found things to try when I thought I was hopelessly stuck.

I would like to thank my friends and colleagues at Chalmers. It has been wonderful to work with you over these years. Special thanks to Chouaib, Laura, Dima (a.k.a. Dyma), Yara, Hao, Jesper, Johan, Liqin, Cristian, Shen, Markus, Rahul, and Sabino.

My wife Guzal, you know better than anyone what a roller-coaster this journey has been—I am happy you have shared its ups and downs with me. Thank you, my love. My family, thank you! I know you are with me, wherever we are.

A handwritten signature in black ink, appearing to read 'Roman Sokolovskii', with a stylized flourish at the end.

Roman Sokolovskii
Gothenburg, May 2022

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

Contents

Abstract	i
List of Papers	iii
Acknowledgments	v
I Introduction	1
1 Background	3
1.1 Thesis Outline	6
2 Linear Block Codes and Iterative Decoding	7
2.1 Linear Block Codes	7
2.2 Tanner Graphs	9
2.3 Belief Propagation Decoding	9
2.4 Binary Erasure Channel	10
2.5 Belief Propagation Decoding for the Binary Erasure Channel .	11
Parallel Peeling Decoding	13
Peeling Decoding	14
2.6 LDPC Codes	16

3	Spatially-Coupled LDPC Codes	21
3.1	SC-LDPC Code Constructions	22
	Termination	23
	Doping	24
3.2	Sliding Window Decoding	26
3.3	Parameter Optimization	28
4	Finite-Length Scaling Laws	31
4.1	Finite-Length Scaling in a Nutshell	32
5	Summary	37
5.1	Contributions	37
	Paper A	37
	Paper B	38
	Paper C	39
5.2	Future Work	40
5.3	Conclusion	41
	References	43
II	Papers	51
A	Finite-Length Scaling of Spatially Coupled LDPC Codes Under Window Decoding Over the BEC	A1
1	Introduction	A3
2	Preliminaries	A5
2.1	Finite-Length Scaling of SC-LDPC Ensembles in [15]	A8
3	Refined Scaling Law	A12
3.1	Decoding Process as Two Independent Ornstein-Uhlenbeck Processes	A13
3.2	Dependence of the Scaling Parameters on the Channel Parameter	A15
3.3	Scaling Law to Predict the Bit Error Rate	A16
3.4	Scaling Law to Predict the Block Error Rate	A16
3.5	Scaling Law for the Unterminated SC-LDPC Code Ensemble	A18

4	Finite-Length Scaling of SC-LDPC Codes Under Window Decoding	A19
4.1	Frame Error Probability	A20
4.2	Bit Error Probability	A21
4.3	Block Error Probability	A22
5	Numerical Results	A24
6	Conclusion	A28
	References	A30
B On Doped SC-LDPC Codes for Streaming		B1
1	Introduction	B3
2	Preliminaries	B5
3	Density and Mean Evolution	B7
3.1	Decoding Thresholds via Density Evolution	B7
3.2	Mean Evolution and Decoding Trajectories	B9
4	Finite-Length Scaling Law	B10
4.1	Error Rates for Regularly Doped Streams	B13
5	Numerical Results	B13
6	Conclusion	B15
	References	B16
C Finite-Length Scaling of SC-LDPC Codes With a Limited Number of Decoding Iterations		C1
1	Introduction	C3
2	Preliminaries	C6
2.1	Density Evolution	C8
2.2	Peeling Decoding	C9
2.3	The Scaling Laws for Unlimited Number of Iterations	C10
3	The Speed of the Decoding Waves and the Duration of the Steady State	C14
4	Finite-Length Scaling: Constant Propagation Model	C17
5	Finite-Length Scaling: Randomized Propagation Distance Models	C21
5.1	Simulating $n_{PD}(I_{\text{eff}})$ Using an Ornstein-Uhlenbeck Process	C22
5.2	Gaussian Propagation Distance Model	C24
5.3	Discussion	C31

6	Finite-Length Scaling: Sliding Window Decoding with a Limited Number of Iterations	C33
6.1	Competition Between the Left Wave and the Sliding Window	C34
6.2	Reduced Maximum Propagation Distance for the Right Wave	C34
6.3	General Form of the Scaling Law	C35
6.4	Modeling the Race Between the Left Wave and the Window	C36
6.5	Numerical Results	C46
7	Conclusion and Discussion	C47
	References	C50

Part I

Introduction

CHAPTER 1

Background

SINCE THE early days of its commercialization in the 1990s, the Internet has taken center stage as the dominant medium of information exchange in the realms of business, governance, and culture. The extent of Internet coverage is starting to be sufficient for the World Wide Web to deserve its name—as of 2021, it is used by 63% of the world’s population, although glaring geographic, economic, and demographic disparities in connectivity levels remain [1]. The amount of data being transmitted is also increasing at a rapid pace, forecast to have tripled in five years to around 400 Exabytes (i.e., 400 million Terabytes) per year by the end of 2022 [2]. Scaling up the global communications infrastructure to meet this growing demand requires innovation at every level, from the design of optical transceivers to network-level optimization.

One prominent area of advance is the development of powerful and energy-efficient means of combating noise and associated errors arising during data transmission. Resilience to noise is achieved by introducing *redundancy* to the transmitted data and leveraging it to reconstruct the parts of the message corrupted by noise. In 1948, in his seminal work that became the cornerstone of *Information Theory*, Shannon showed that there exists a minimum amount

of redundancy necessary to achieve reliable transmission as the size of the messages grows large, and that this minimum depends on the nature of the communication channel and the intensity of noise [3]. However, information theory is not concerned with developing practical ways to achieve the theoretical limits—this is instead the domain of *Coding Theory*, which seeks good trade-offs between error-correcting performance and practical constraints on, e.g., transmission delay and computational feasibility. Specifically, the task of the communication system designer is to propose a way to map a set of information sequences onto a set of sequences to be transmitted—a *code*¹—together with a way to infer the transmitted (and, ultimately, the information) sequence from the received one—a *decoding algorithm*—so as to maximize the probability that the transmitted sequence is reconstructed correctly.

For decades, the northern star of Shannon’s theoretical limits shone from afar, and the question of whether those limits could ever be approached in practice remained open. (Shannon himself stayed hopeful [4].) Today, we can confidently answer this question in the affirmative. The last three decades have witnessed several major breakthroughs in coding theory, starting from Berrou’s invention of turbo codes in 1993 [5]—their performance was so outstanding at the time that the scientific community was initially reluctant to accept the results at face value—and the rediscovery of low-density parity-check (LDPC) codes by MacKay, Luby, and others [6]–[9], apparently independently of their original introduction by Gallager in his Ph.D. thesis in 1963 [10]. Both LDPC and turbo codes leverage pseudo-randomness and iterative decoding algorithms to operate close to theoretical limits with feasible complexity.

Moreover, several code constructions have been subsequently proven to be *capacity-achieving*, such as polar codes [11], Reed-Muller codes [12], irregular LDPC codes [13], as well as spatially-coupled LDPC codes [14] and spatially-coupled turbo-like codes [15], [16]. This means that system designers are now equipped with several concrete ways to construct error-correcting codes that are guaranteed to achieve the fundamental limits of communication when provided with unlimited time, energy, and memory budgets.

Unfortunately, not every system designer can afford to be so prodigal. Instead, the designers have to navigate a complicated space of trade-offs in order

¹ The term ‘code’ is used in this context in the sense of ‘mapping,’ as in ‘Morse code.’ Coding theory, therefore, has nothing to do with ‘coding’ in the sense of ‘computer programming,’ although in practice involves quite a lot of it.

to choose the code and decoder parameters in the non-asymptotic (i.e., finite-length) regime and under practical constraints. To that end, it is desirable to be able to predict the code/decoder performance for a given set of parameters without having to resort to Monte-Carlo simulations for each such combination, which may be prohibitively complex computationally.

This thesis aims to make such finite-length parameter optimization feasible for spatially-coupled LDPC (SC-LDPC) code ensembles, one of the powerful capacity-achieving code constructions, by providing ways to predict a code ensemble's finite-length error-correcting performance as a function of the code and decoder parameters by means of so-called *scaling laws*. An exact formula for decoding error probability suffers from a combinatorial explosion in the number of terms as the length of the transmitted sequence grows; our goal is instead to propose a mathematical model of the decoding process that is simple enough to be analytically tractable but that nevertheless captures the behavior of the decoder sufficiently well to yield accurate predictions of the decoder error probability.

Papers A and C comprise key theoretical contributions of this thesis and can be understood as a progression toward predicting error-correcting performance in ever more practical system setups. Paper A begins by proposing a scaling law that predicts the probability of decoding error more accurately than the previously proposed law by Olmos and Urbanke [17] for full belief propagation (BP) decoding with unlimited iterations. Crucially, Paper A also introduces a new model to predict the performance of sliding window decoding, a decoding algorithm that allows to decouple decoding latency from code length and that is used in practice, albeit still in the case of unlimited decoding iterations. Paper C takes a step further and tackles the analytically challenging case where the number of decoding iterations is limited, as is inevitably the case in practical scenarios. Paper B can be seen as a kind of interlude that considers scaling laws for sliding window decoding of SC-LDPC codes with doping, a technique particularly relevant in streaming applications.

Together, the contributions of Papers A–C take a significant step toward making SC-LDPC code and decoder parameter optimization via finite-length scaling laws feasible and relevant for the design of practical communication systems.

1.1 Thesis Outline

This thesis is organized as a *collection of papers*, appended in Part II, with Part I serving as a brief introduction.

The remainder of Part I is organized as follows: Chapter 2 introduces the concept of linear block codes and their representation in terms of Tanner graphs. It also discusses LDPC codes and iterative BP decoding and introduces the binary erasure channel, which is used as the channel model throughout the thesis. Chapter 3 describes spatial coupling and SC-LDPC codes along with the code and decoder parameters that need to be chosen (and therefore optimized) for SC-LDPC codes in practice. Chapter 4 provides some intuition behind the finite-length scaling laws proposed in the appended papers, and Chapter 5 concludes the introductory part with a brief summary of the contributions of each paper and some discussion of the potential directions of further scientific inquiry.

Linear Block Codes and Iterative Decoding

THE MOST widely used way to introduce redundancy to transmitted data is by employing linear block codes, of which SC-LDPC codes studied in this thesis form a subclass. This chapter briefly introduces the concept of binary linear block codes, LDPC codes and their representation in terms of Tanner graphs, as well as the binary erasure channel and iterative decoding of linear block codes over the binary erasure channel using their Tanner graph representation.

2.1 Linear Block Codes

A binary (n, k) *linear block code* uses n bits to transmit $k < n$ bits of information. It can be defined as the set of binary (column) vectors $\mathbf{x} \in \{0, 1\}^n$ that satisfy

$$\mathbf{H}\mathbf{x} = \mathbf{0} \tag{2.1}$$

with operations performed modulo 2. The binary $m \times n$ matrix \mathbf{H} is referred to as the *parity-check* matrix; \mathbf{H} must be of rank $n - k$. The parameter k is called the *dimension* of the code—the set of *codewords* \mathbf{x} that satisfy (2.1) forms a

k -dimensional linear subspace of $\{0, 1\}^n$ over $\text{GF}(2)$ that can be referred to as the (right) nullspace of \mathbf{H} . The *code rate* $R = k/n$ quantifies the amount of redundancy in the transmitted message in terms of the average number of information (in bits) carried by each code bit.

For example, let us consider the classical $(7, 4)$ Hamming code with parity-check matrix

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}. \quad (2.2)$$

Each codeword $\mathbf{x} = [x_1, x_2, \dots, x_7]^\top$ satisfies

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \quad (2.3)$$

There are 2^4 such codewords; without coding, it would have taken $k = 4$ bits to transmit this amount of information, but it would have been impossible to recover from a single bit flip or erasure because any bit flip would result in a valid sequence, and any erasure would result in an ambiguity over which sequence was transmitted. Instead, $n = 7$ bits are used for transmission of $k = 4$ bits of information, so the code rate of the $(7, 4)$ Hamming code is $R = 4/7$.

Each row of the parity-check matrix \mathbf{H} defines a constraint on the possible values of bits in \mathbf{x} . Indeed, (2.3) can be rewritten as

$$\begin{cases} x_1 + x_2 + x_3 + x_5 = 0 \\ x_2 + x_3 + x_4 + x_6 = 0 \\ x_1 + x_2 + x_4 + x_7 = 0 \end{cases}. \quad (2.4)$$

Each row dictates that there should be an even number of ones among the values of the corresponding bits of the codeword because the summation is performed modulo 2 (hence the name of the *parity*-check matrix). The more (linearly independent) rows in \mathbf{H} , the more constraints are imposed on the bits

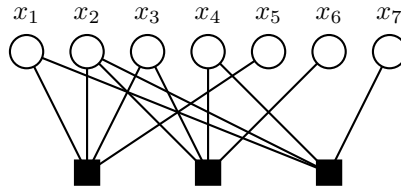


Figure 2.1: Tanner graph of the $(7, 4)$ Hamming code.

of a codeword and the fewer codewords the code will contain. On the other hand, it is the knowledge that the transmitted bits satisfy those constraints that allows the decoder to infer their values and reconstruct the message in the presence of noise.

2.2 Tanner Graphs

An alternative way to specify the parity-check matrix \mathbf{H} and therefore the code is via its *Tanner graph* [18]. A Tanner graph is a bipartite graph where one set of vertices represents the code bits and the other code constraints. The n vertices that represent the code bits are called *variable nodes* (VNs) and are customarily denoted by circles; the m vertices that represent the constraints are called *check nodes* (CNs) and are denoted by squares. An edge connects a VN to a CN if the bit that corresponds to the VN enters the constraint that corresponds to the CN or, in other words, if there is a one at the intersection of the corresponding column (for the VN) and row (for the CN) of the parity-check matrix \mathbf{H} .

The Tanner graph that corresponds to the $(7, 4)$ Hamming code with parity-check matrix \mathbf{H} from (2.2) and the set of constraints (2.4) is shown in Fig. 2.1. The leftmost black square is the CN that corresponds to the first row of \mathbf{H} and the first equation in (2.4); it is thus connected to the VNs that correspond to x_1, x_2, x_3 , and x_5 . The other two CNs are connected analogously.

2.3 Belief Propagation Decoding

The Tanner graph representation leads naturally to a powerful decoding algorithm called *belief propagation* (BP) decoding, which consists of an iterative

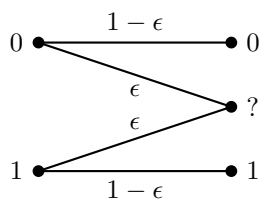


Figure 2.2: Schematic representation of the BEC with erasure probability ϵ .

exchange of messages along the edges of the graph.

Specifically, the Tanner graph representation of \mathbf{H} can be used to implement the BP decoder in the following informally described way: Each node of the graph acts as a local processor that estimates the likelihood of a given bit based on the likelihoods communicated to it from the neighboring processors and, in the case of VNs, the value received from the channel. The local processor then relays its estimations along the edges of the graph to its neighbors, which use the communicated values as input for the next iteration. Such iterative exchange of messages continues until a given stopping criterion is met, e.g., a codeword is found or a limit on the maximum number of iterations is reached.

The specific equations performed by local VN and CN processors during BP decoding are omitted—in the case of the binary erasure channel, BP decoding equations can be greatly simplified, and the general form of BP decoding is not used in this thesis. A detailed introduction to and description of BP decoding for general memoryless channels can be found in [19, Ch. 5.4].

2.4 Binary Erasure Channel

The *binary erasure channel* (BEC) is one of the simplest models for communication channels. It is schematically illustrated in Fig. 2.2. A bit transmitted over the BEC is either erased (i.e., replaced by a symbol $?$) with probability ϵ or received correctly with probability $1 - \epsilon$. In other words, during transmission over the BEC information may be erased completely but is never altered—when a 0 or a 1 is received, the decoder can count on the fact that this was indeed the transmitted value. The task of the decoder becomes therefore to infer the values of the erased bits based on the values on the non-erased bits and on the knowledge of the constraints the bits must satisfy (e.g., the constraints in (2.4) for the running example of the (7, 4) Hamming code).

This thesis considers transmission over the BEC. Despite its simplicity, the BEC proves to be a very useful model in several ways: First, the insights and tools developed for the BEC often carry over to more elaborated channels [20]. Second, the results obtained for the BEC can sometimes serve as bounds for other channels [20]. Third, the BEC can be directly used to optimize the parameters of the code and the decoder in the hope that a code that performs well over the BEC will also perform well for other channels—this hope is particularly justified for SC-LDPC codes, as is discussed in Chapter 3. Finally, the BEC has gained prominence as a “real-world” channel in applications related to packet transmission over the Internet; the underlying TCP/IP protocol ensures that data packets are either transmitted correctly or lost (i.e., “erased”) altogether [20].

2.5 Belief Propagation Decoding for the Binary Erasure Channel

As discussed in Section 2.3, BP decoding consists of an iterative exchange of messages (typically in the form of log-likelihood ratios) that represent beliefs about transmitted code bits between the VNs and the CNs along the edges of the Tanner graph. In the case of the BEC, since the channel does not introduce any errors, the decoder can either be absolutely certain about the value of a bit or remain in complete ignorance. Consequently, it is possible to represent the operation of the BP decoder directly in terms of the inferred values of the code bits instead of in terms of their likelihoods. This alternative representation of BP decoding is described below.

BP decoding over the BEC operates much like one would approach solving a system of linear equations before having been taught about Gaussian elimination or row operations. It first substitutes the values of non-erased code bits in the system of parity-check equations that corresponds to \mathbf{H} , then infers the value of every code bit that happens to remain the only unknown in one of the equations, and then substitutes those newly learned code bit values in all other equations where those code bits participate. This may in turn result in new equations with a single unknown. The decoder proceeds in this fashion until it either resolves the values of all code bits or runs out of trivial single-unknown equations.

Let us illustrate this process using our running example of the (7, 4) Ham-

ming code with the parity-check matrix \mathbf{H} in (2.2). Assume that the codeword $\mathbf{x} = [0, 0, 1, 1, 1, 0, 1]^T$ is transmitted. It can be verified that this codeword satisfies the parity-check constraints (2.3)–(2.4). (Summation is performed modulo 2.) Assume further that after transmission over the BEC the sequence $[?, 0, 1, 1, 1, ?, ?]^T$ is received—i.e., the values of x_1, x_6 , and x_7 were erased and are unknown, whereas the values of x_2, x_3, x_4 , and x_5 are known with certainty. How could one use this knowledge (and the knowledge of the parity-check constraints) to reconstruct the values of the erased bits?

First, the BP decoder substitutes the values of the known bits into the system (2.4), which yields

$$\begin{cases} x_1 + 0 + 1 + 1 = 0 \\ 0 + 1 + 1 + x_6 = 0 \\ x_1 + 0 + 1 + x_7 = 0 \end{cases} \quad (2.5)$$

It then moves all knowns to the right-hand side and adds them modulo 2, resulting in

$$\begin{cases} x_1 = 0 \\ x_6 = 0 \\ x_1 + x_7 = 1 \end{cases} \quad (2.6)$$

The first two equations now contain a single unknown; their values can be filled in whenever they occur. In this case, the value of x_1 can be substituted into the third equation and moved to its right-hand side, which gives

$$\begin{cases} x_1 = 0 \\ x_6 = 0 \\ x_7 = 1 \end{cases} \quad (2.7)$$

and allows the decoder to successfully reconstruct the transmitted codeword.

The BP decoder's reliance on trivial constraints with a single unknown makes tracking the availability of such constraints crucial for the analysis of the decoder's error-correcting performance, as discussed in more detail in Chapter 4.

Reliance on trivial constraints also makes the BP decoder suboptimal. Indeed, in some cases the system of parity-check equations has a unique solution even when there are no trivial constraints available. Optimal decoding could

reconstruct the codeword in those situations, whereas BP decoding would fail. For example, the system

$$\begin{cases} x_1 + x_2 + x_3 = 0 \\ x_2 + x_3 = 1 \\ x_1 + x_2 = 0 \end{cases} \quad (2.8)$$

could be successfully solved by, e.g., substituting the value for $x_2 + x_3$ from the second equation into the first, yielding

$$\begin{aligned} \begin{cases} x_1 + x_2 + x_3 = 0 \\ x_2 + x_3 = 1 \\ x_1 + x_2 = 0 \end{cases} &\longrightarrow \begin{cases} x_1 + 1 = 0 \\ x_2 + x_3 = 1 \\ x_1 + x_2 = 0 \end{cases} \longrightarrow \begin{cases} x_1 = 1 \\ x_2 + x_3 = 1 \\ 1 + x_2 = 0 \end{cases} \longrightarrow \\ &\longrightarrow \begin{cases} x_1 = 1 \\ 1 + x_3 = 1 \\ x_2 = 1 \end{cases} \longrightarrow \begin{cases} x_1 = 1 \\ x_2 = 1 \\ x_3 = 0 \end{cases}, \end{aligned} \quad (2.9)$$

but such manipulations are beyond the capabilities of the BP decoder.

Parallel Peeling Decoding

BP decoding can be alternatively represented in terms of a sequence of graphs, starting from the original Tanner graph. This representation, known as *parallel peeling decoding*, is fully equivalent to BP decoding over the BEC in the sense that parallel peeling decoding recovers the same bits as does BP decoding at every corresponding iteration [21]. The initial step of parallel peeling decoding consists of removing all edges adjacent to VNs that correspond to known code bits from the graph and changing the parity checks accordingly. At every subsequent iteration, the value of every VN connected to a degree-one CN is reconstructed, the edges adjacent to those VNs are removed from the graph, and the affected parity checks are modified if necessary. This may in turn create new degree-one CNs, which correspond to parity-check equations with a single unknown. Parallel peeling decoding proceeds until all edges are thus *peeled off* from the graph, or until there are no more degree-one CNs to be found. Each iteration of parallel peeling decoding produces a new graph, called a *residual* graph. Degree-zero VNs and CNs are also removed from the residual graph; they are kept in the following figures for clarity.

Let us turn back to our running example. The sequence of residual graphs that corresponds to (2.5)–(2.7) is shown in Fig. 2.3. The initial stage of parallel peeling decoding, which corresponds to (2.5), can be represented in terms of the Tanner graph as shown in Fig. 2.3a. The subscripts to the CNs denote the values of the corresponding parity checks. The parallel peeling decoder then removes all edges connected to the VNs associated with the known code bits x_2, \dots, x_5 and accounts for their values in the subscripts to the CNs, with the resulting residual graph shown in Fig. 2.3b (equivalent to (2.6), where all known values are collected at the right-hand side). The resulting residual graph now contains two degree-one CNs (shown in red). The next iteration of parallel peeling decoding results in the graph shown in Fig. 2.3c, where the values of the VNs x_1 and x_6 adjacent to the two degree-one CNs from Fig. 2.3b are deduced (shown in blue) and the connected edges are removed. This creates another degree-one CN (shown in red in Fig. 2.3c), which is used at the last iteration of parallel peeling decoding to reconstruct x_7 (blue) as in (2.7) and produce the empty graph in Fig. 2.3d.

The BP decoder’s reliance on trivial constraints translates into the parallel peeling decoder’s reliance on degree-one CNs. Correspondingly, focusing on the number of degree-one CNs available to parallel peeling decoder over decoding iterations is a crucial step in the analysis of the decoder’s performance, as discussed in Chapter 4.

An example residual graph that corresponds to the system (2.8), unsolvable by the BP decoder, is shown in Fig. 2.4. Such graph structures are known as *stopping sets*. A stopping set is a set of VNs to which every adjacent CN is connected at least twice. Stopping sets play a crucial role in the analysis of BP decoding over the BEC because BP decoding fails whenever a stopping set is present in the corresponding residual graph, so the probability of decoding failure is essentially the probability that a stopping set is present. However, analyzing the error-correcting performance directly by enumerating all possible stopping sets is limited to very short codes since the number of stopping sets explodes combinatorially with increasing n .

Peeling Decoding

At every iteration, parallel peeling decoding removes all degree-one CNs from the residual graph. This renders the direct analysis of parallel peeling decoding and, therefore, of BP decoding complicated. Instead, (sequential) *peeling*

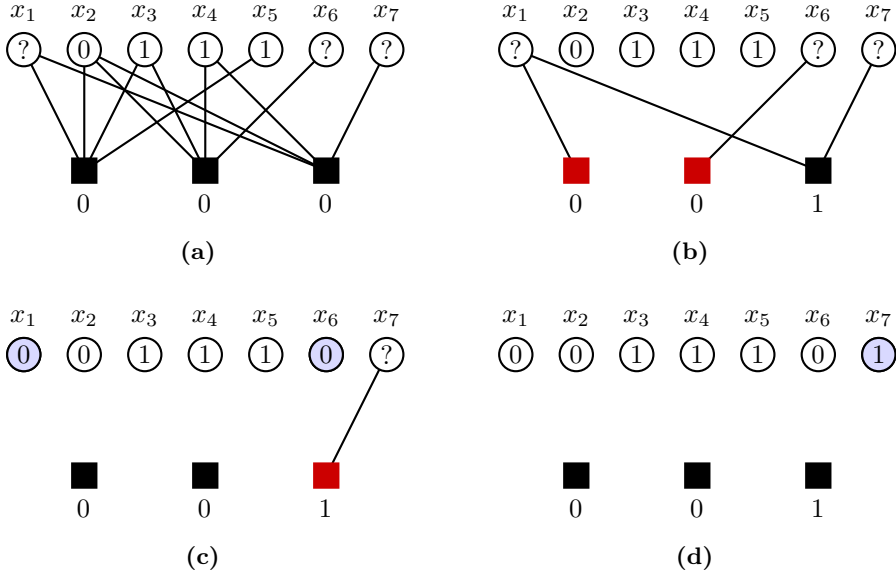


Figure 2.3: Example sequence of residual graphs during parallel peeling decoding.

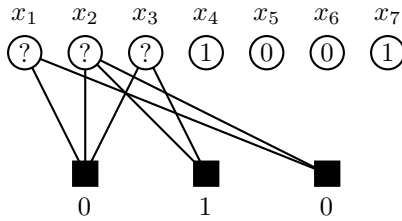


Figure 2.4: Example stopping set.

decoding, first proposed in [22] and later used for the finite-length analysis of LDPC codes in [23] and of SC-LDPC codes in [17], removes a *single* randomly chosen degree-one CN at every iteration, which is equivalent to resolving a single trivial constraint at a time, randomly chosen among all available trivial constraints.

When the number of decoding iterations is not limited, the parallel and sequential versions of peeling decoding are equivalent because they get stuck at the same stopping sets. However, the sequential version is easier to analyze, primarily because each iteration always recovers a single VN.

An example sequence of graphs produced by sequential peeling decoding that corresponds to the running example from Fig. 2.3 is shown in Fig. 2.5. The initialization steps of both parallel and sequential peeling decoding are identical and result in Fig. 2.3b and Fig. 2.5b, respectively. Then, instead of resolving both CNs in Fig. 2.3b to yield Fig. 2.3c, the sequential decoder flips a coin and resolves only the CN connected to x_6 , going from Fig. 2.5b to Fig. 2.5c, and then resolves the other degree-one CN in the next iteration, going from Fig. 2.5c to Fig. 2.5d. The final iteration of sequential peeling decoding reconstructs x_7 and results in the empty graph in Fig. 2.5e.

2.6 LDPC Codes

BP decoding can be applied to any linear block code. However, its performance will be disappointing if the parity-check matrix \mathbf{H} contains a large fraction of ones because this results in a large number of edges in the associated Tanner graph, which, on the one hand, incurs a high computational cost, and, on the other hand, creates a large number of stopping sets. The Tanner graph representation and BP decoding are particularly powerful in the context of LDPC codes, which are specifically designed to perform well under iterative BP decoding.

The “low-density” in LDPC codes stands for a low density of ones in the parity-check matrix \mathbf{H} and therefore a low density of edges in the associated Tanner graph. The term is deliberately vague: in practice, the density of 0.01 can be considered low [19], although the main objective remains to enable effective and computationally efficient iterative decoding [20].

The simplest example of an LDPC code is a code where each bit participates in a fixed number of parity-check equations, d_v , and each parity-check equation comprises d_c code bits. In terms of the parity-check matrix \mathbf{H} , this means that every column of \mathbf{H} contains d_v ones, and each row of \mathbf{H} contains d_c ones; in terms of the associated Tanner graph, this means that every VN has degree d_v and every CN has degree d_c . Such an LDPC code is referred to as a (d_v, d_c) -regular LDPC code.

The values of d_v and d_c define the minimum rate of the code: Suppose the code has n VNs. Then there must be $d_v \cdot n$ edges in the Tanner graph, and

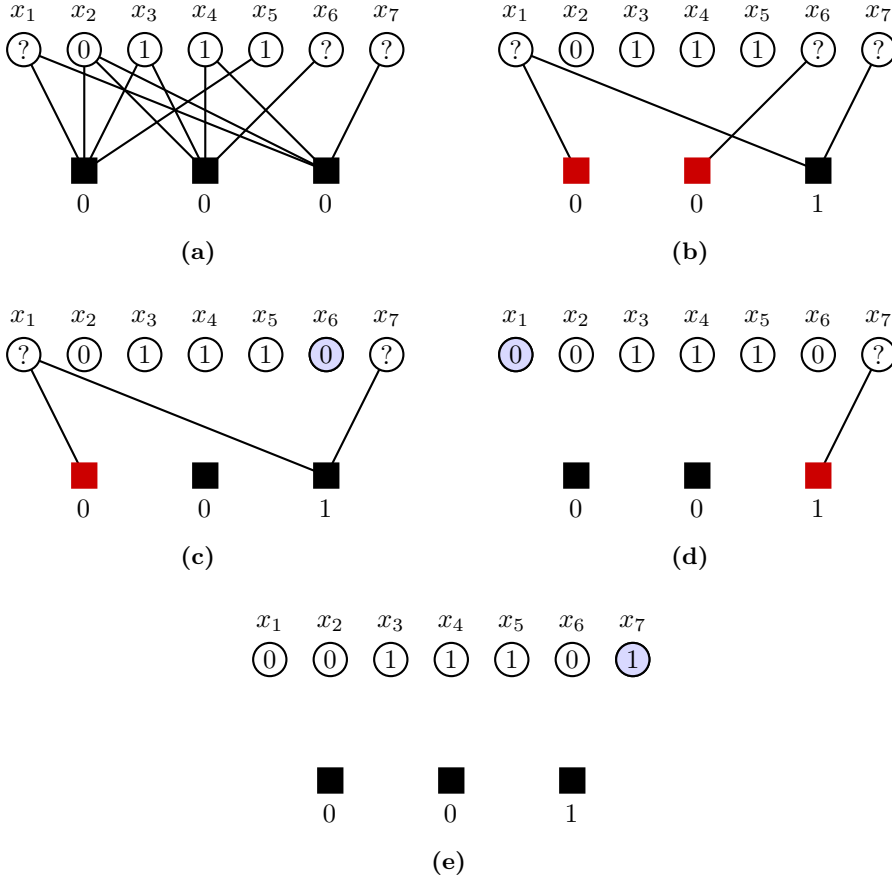


Figure 2.5: Example sequence of residual graphs during peeling decoding.

each CN is connected to d_c of them. This implies that there should be

$$n - k = \frac{d_v}{d_c} \cdot n \quad (2.10)$$

CNs, and the minimum code rate $R = k/n$ must be

$$R = \frac{n - d_v/d_c \cdot n}{n} = 1 - \frac{d_v}{d_c}. \quad (2.11)$$

If the parity-check matrix \mathbf{H} that corresponds to the Tanner graph contains linearly dependent rows, the code rate will be higher.

Instead of analyzing the performance of a specific LDPC code, which is infeasible for long codes, the analysis often involves considering an *ensemble* of LDPC codes that share certain characteristics, such as the block length n and the degrees of VNs and CNs. For example, the (d_v, d_c) -regular LDPC code ensemble of length n is defined as the set of all codes of length n associated with Tanner graphs with VNs of degree d_v and CNs of degree d_c .

The asymptotic error-correcting performance of LDPC code ensembles under BP decoding can be analyzed using *density evolution*. Generally, density evolution tracks the probability density functions of the messages passed along the edges of the Tanner graph across BP iterations. It relies on the fact that asymptotically, i.e., as $n \rightarrow \infty$, the messages incoming to a VN or a CN at a given iteration become independent [20]. (For this to hold, the VN and CN degrees d_v and d_c need to be fixed and finite, which is where the “low-density” assumption plays out.) As with BP decoding itself, in the case of the BEC density evolution can be simplified; instead of tracking the full probability distributions of the log-likelihood ratios, density evolution can simply track the probability that the message sent corresponds to an erasure.

Density evolution shows that the probability of a code bit to remain erased vanishes over BP iterations if the BEC erasure probability ϵ is smaller than a certain value, which is called the *BP threshold* and is denoted in this thesis by ϵ^* . For $\epsilon > \epsilon^*$, on the other hand, the VN erasure probability remains bounded away from zero. This means that as $n \rightarrow \infty$, the decoder error probability as a function of ϵ converges to a step function with a step from zero for $\epsilon \leq \epsilon^*$ to one for $\epsilon > \epsilon^*$. A similar behavior is observed for other channels and other code ensembles.

The parameters of LDPC code ensembles can be optimized in terms of their impact on the asymptotic performance via density evolution thresholds. The validity of this approach rests on the *concentration* of the performance curves for individual LDPC codes around their corresponding ensemble averages [20].

Density evolution analysis reveals that for a fixed code rate the asymptotic performance of (d_v, d_c) -regular LDPC codes under BP decoding degrades with increasing VN degree d_v . At the same time, it is known that the minimum distance (i.e., the minimum number of bits in which any two codewords of a code differ) of such codes *grows* with d_v (and grows linearly with n). In other words, even though the codes themselves become stronger, their performance under suboptimal BP decoding worsens.

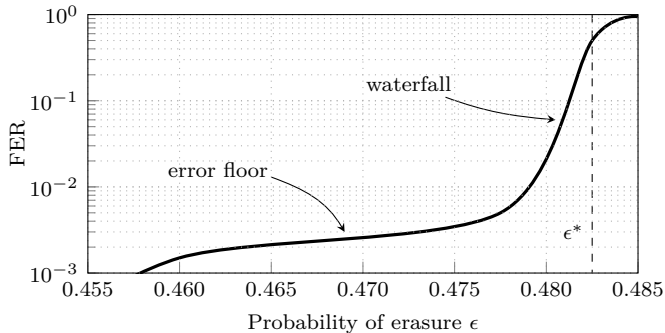


Figure 2.6: Waterfall and error floor regions.

A typical performance curve of an LDPC code (and of other codes-on-graphs under iterative decoding) is sketched in Fig. 2.6. Initially, as ϵ decreases, the decoding error probability falls steeply; this region is informally referred to as the *waterfall*. Then the performance curve flattens out, in the region known as the *error floor*. The error floor region is dominated by small stopping sets (i.e., whose size is sub-linear in n), whereas the waterfall region is dominated by large stopping sets (i.e., linearly-sized with n) [20]. A similar behavior is observed for other channels.

In a series of papers that brought about a revolution in the understanding of BP decoding and LDPC codes, Luby, Mitzenmacher, Shokrollahi, Spielman, and Stemann showed that *irregular* LDPC code ensembles, where the degrees of VNs and CNs are randomized according to a so-called *degree distribution*, can asymptotically perform arbitrarily close to capacity if the degree distribution is appropriately optimized [13], [22], [24], [25]. Irregular LDPC codes can therefore have excellent performance in the waterfall region.

There are a number of disadvantages associated with the use of irregular LDPC codes with degree distributions optimized via density evolution: First, irregular LDPC codes have high error floors, which limits their usability in applications where extremely low error rates are required. Generally, in the case of LDPC codes, better decoding thresholds (and hence better waterfall performance) are associated with higher error probabilities in the error floor region. Second, irregular LDPC codes are not universal, in the sense that a degree distribution optimized for a particular channel will generally not be optimal for another channel, so the optimization must be done anew. Third, their minimum distance does not grow linearly with n . Finally, the presence

of the VNs of different degrees complicates hardware implementation.

For a long time, those downsides—especially the trade-off between the waterfall and error-floor performance—were considered inescapable. Then the discovery of convolutional LDPC codes [26], [27], later re-branded as SC-LDPC codes, shook those assumptions and brought about another breakthrough in modern coding theory.

Spatially-Coupled LDPC Codes

SPATIAL coupling was initially introduced as a coding technique under the name of convolutional LDPC codes [26]. As with classical (non-LDPC) convolutional codes, the encoding process of convolutional LDPC codes may be implemented by introducing memory to the encoder, which in turn translates into a band-like parity-check matrix reminiscent of the parity-check matrices of classical convolutional codes [26]. Later, it became apparent that constructing and analyzing the performance of convolutional LDPC codes is more natural in terms of their Tanner graph representation, where the convolutional aspect manifests itself in the form of coupling between the sub-graphs of component LDPC codes. This resulted in a re-branding of convolutional LDPC codes under the name of *spatially-coupled* LDPC (SC-LDPC) codes.

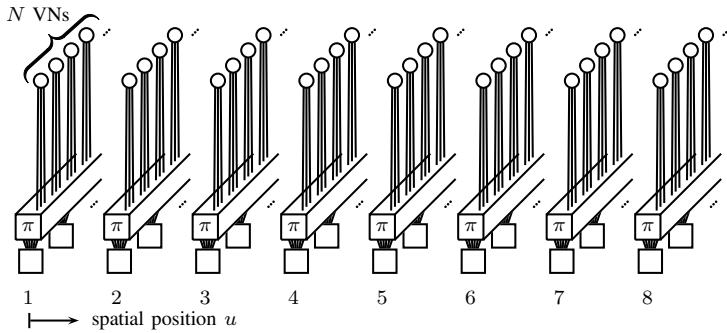
SC-LDPC codes yield outstanding performance under suboptimal BP decoding: First, it was observed numerically that the BP thresholds of SC-LDPC codes ensembles obtained via density evolution (see Section 2.6) are close to the optimal *maximum a posteriori* (MAP) thresholds of the underlying uncoupled LDPC code ensembles [27]. This phenomenon, dubbed *threshold saturation*, unleashes the potential of strong (d_v, d_c) -regular LDPC codes

(Section 2.6) under iterative BP decoding. It was later proved that threshold saturation allows SC-LDPC codes to achieve capacity, first shown for the BEC [28] and later for the broader class of binary-input memoryless symmetric channels [14]. Moreover, unlike irregular LDPC codes, SC-LDPC codes are *universal*—i.e., a sequence of LDPC codes that achieves capacity for one binary-input memoryless symmetric channel also provably achieves capacity for another such channel [14]. This allows the code designer to use the same code construction for different channels without having to re-optimize code parameters, as would be necessary for irregular LDPC codes. Finally, it was shown that spatial coupling preserves the minimum distance growth properties of regular LDPC code ensembles, whose distance grows linearly with the block length [29]; moreover, such linear growth with block length also holds for the smallest stopping set [30] (Section 2.5). Taken together, these remarkable properties mean that SC-LDPC codes can have both good BP thresholds and potentially low error floors, escaping the trade-off between the waterfall and error-floor performance that was hitherto considered unavoidable.

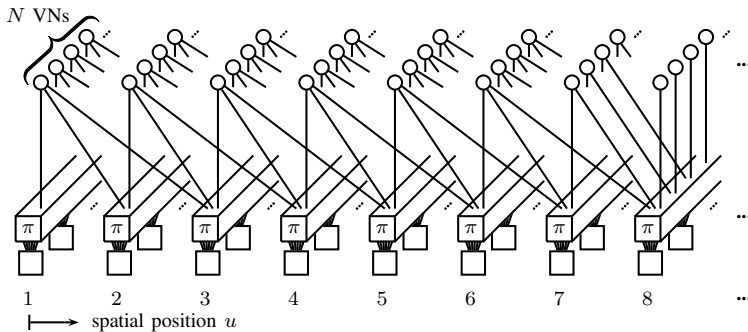
The benefits of spatial coupling proved to apply beyond the realm of LDPC codes in other codes-on-graphs and graph-based systems. Spatial coupling has been used in the context of, e.g., turbo-like codes [15], product-like codes [31], lossy compression [32], and compressed sensing [33].

3.1 SC-LDPC Code Constructions

To construct the Tanner graph of an SC-LDPC code, one must first take a number of Tanner graphs of the underlying uncoupled LDPC codes, each with the same number of VNs, N , and arrange them into a spatial sequence. An example of such a sequence of length $L = 8$ is shown in Fig 3.1a, where the spatial positions are indexed by u . The blocks marked by π denote permutation blocks that shuffle the edges connected to them—a specific permutation results in a specific Tanner graph and thus in a specific code. One could imagine the Tanner graph of the not-so-low-density parity-check (7, 4) Hamming code in Fig. 2.1 in place of the component Tanner graphs in the sequence in Fig. 3.1a. Having arranged the component Tanner graphs in a “spatial” sequence as in Fig. 3.1a, one should interconnect (or “couple”) the component Tanner graphs by rearranging their edges according to a predefined pattern, as shown in Fig. 3.1b. There are many different ways to perform the inter-



(a) Uncoupled



(b) Coupled

Figure 3.1: Spatial coupling.

connecting, which result in different types of SC-LDPC codes; the rule used in Fig. 3.1b as an example is that every VN of degree d_v (with $d_v = 3$ in the figure) is connected to d_v consecutive positions.

Termination

Spatial coupling naturally results in potentially infinite sequences of connected Tanner graphs and thus in potentially infinitely long codes. Such semi-infinite codes are referred to in Papers A–C as *unterminated* SC-LDPC codes. Unterminated codes are a natural fit for streaming applications; they are also used in Paper A in the analysis of sliding window decoding, which is discussed in

Section 3.2.

In practice, the spatially-coupled chain cannot be truly infinite. There are several ways to end the chain after a certain length L . One way is to simply cut it along with the overhanging edges. The resulting code is called *truncated* SC-LDPC code (see Fig. 3.2a). Alternatively, one may connect the overhanging edges to additionally appended positions that contain CNs only; the resulting SC-LDPC code chain is referred to as *terminated* (see Fig. 3.2b). Note that terminating the chain as shown in Fig. 3.2b creates a certain symmetry between the beginning and the end of the chain, in the sense that the first and the last CN positions are connected to the same number of edges.

Termination is key to the outstanding error-correcting performance of SC-LDPC codes. The CNs at the boundaries of the terminated spatially-coupled chain have lower average degrees, which means that during BP decoding (see Section 2.5) those CNs will more likely correspond to trivially-resolvable constraints (i.e., be of degree one), so the adjacent VNs at the boundaries of the chain are more likely to be recovered than the VNs in the middle. Subsequently, reliable information propagates from the boundaries of the chain inward in a wave-like fashion. It is this wave-like decoding effect that gives rise to SC-LDPC codes' improved performance. However, it does not come at no cost: adding CNs at the terminated end of the chain entails a loss in code rate—one may think of adding CNs as subtracting degrees of freedom and thus reducing k in $R = k/n$. The longer the chain, the smaller the loss in code rate; on the other hand, the longer the chain, the more likely it is that wave-like decoding will stall. This trade-off between rate loss and error-correcting performance makes practical SC-LDPC code design nontrivial.

Remarkably, it is sometimes possible to reap the benefits of spatial coupling without termination, e.g., by optimizing bit mappings in the context of the parallel erasure channel [34] and coded modulation [35], or in the form of energy shaping in the context of the binary input additive white Gaussian noise channel [36]. These techniques aim to trigger the same wave-like decoding that is present in terminated chains but by alternative means.

Doping

Some applications cannot afford the loss in code rate associated with frequent termination of spatially-coupled chains to trigger wave-like decoding. Long

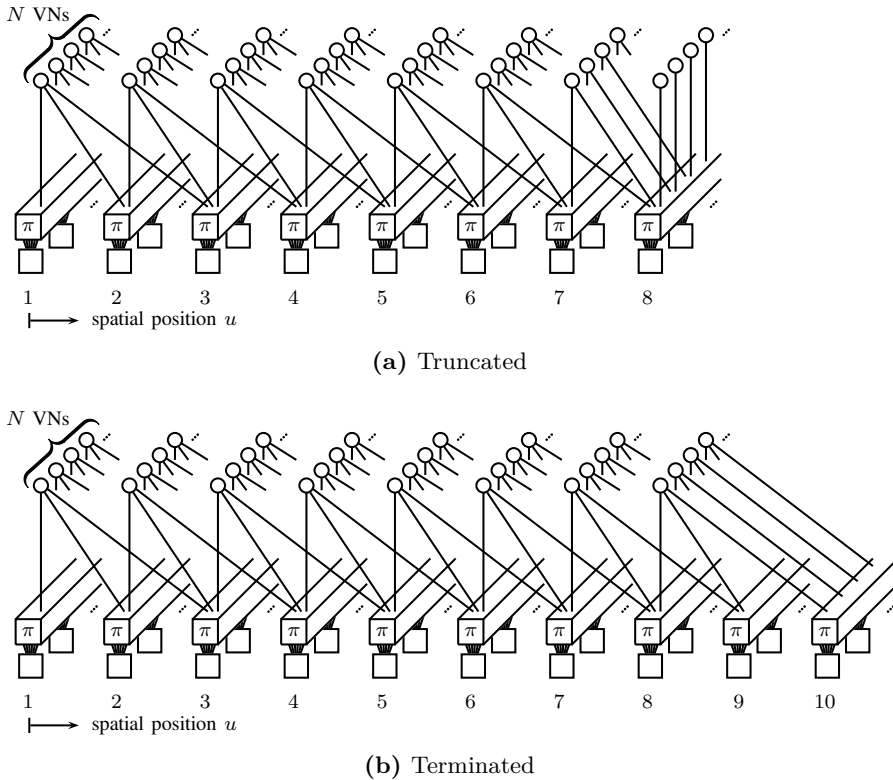


Figure 3.2: Truncation vs termination.

chains, however, suffer from *decoding error propagation*, where the decoding wave gets stuck and a very long stream of errors occurs. To limit the rate loss but at the same time prevent decoding errors from propagating indefinitely, a technique called *doping* involves modifying the chain in such a way so as to trigger the wave without incurring the same loss in rate that is associated with full termination. In a way, doping can be understood as offering a range of solutions in-between full termination and an unobstructed chain.

Doping can be performed in several ways: *CN doping* consists of occasionally inserting spatial positions that contain reduced-degree CNs [37]. Alternatively, *VN doping* involves fixing all VNs at some spatial positions [38]. More

generally, instead of fixing all VNs at a doped position, only a fraction of them may be fixed instead [39], which may be referred to as *soft VN doping*.

Fig. 3.3a illustrates a SC-LDPC code chain with full termination in the middle. Notably, the chain effectively splits in two unconnected sub-chains that can be decoded independently. In contrast, Fig. 3.3b shows the same chain with VN doping, where VNs in a single position (position 5) are fixed. The resulting chain is still fully connected: the VNs at position 4 are still connected to the CNs at position 6, so the sub-chains' performance must be analyzed jointly. Clearly, termination can also be viewed as the result of excessive doping, as the comparison between Fig. 3.3a and Fig. 3.3b reveals—applying VN doping to position 4 as well as to position 5 in Fig. 3.3b would create a chain that is analogous to the chain in Fig. 3.3a with full termination in the middle.

Modeling the effects of doping and quantifying the ensuing trade-offs between rate loss and error-correcting performance are the subject of Paper B. It shows that, in some cases, doping allows to achieve higher code rates for a given target error rate than is possible with fully terminated chains only.

3.2 Sliding Window Decoding

Aside from decoding error propagation, employing long spatially-coupled chains incurs a cost in terms of high decoding latency—when BP decoding is applied to the whole chain (a scheme sometimes referred to as *full BP decoding*), the receiver must wait for the arrival of all bits in the chain before starting decoding. To limit decoding latency, an alternative decoding method called *sliding window decoding* is used in practice, where BP decoding is limited to a window of several spatial positions, as shown in Fig. 3.4a. After a specified number of iterations, the decoder decides on the bits in the leftmost position within the window and slides by one position to the right (Fig. 3.4b). By deciding on the values of the bits and fixing them as the window slides (the fixed bits are shown in Fig. 3.4 in red), sliding window decoding limits decoding latency to the size of the window.

Sliding window decoding was originally proposed in [40]. Later, it was shown that performing BP decoding in a non-uniform fashion within the window can further reduce decoding complexity [41], [42].

Practical constraints on decoding latency and energy efficiency imply that

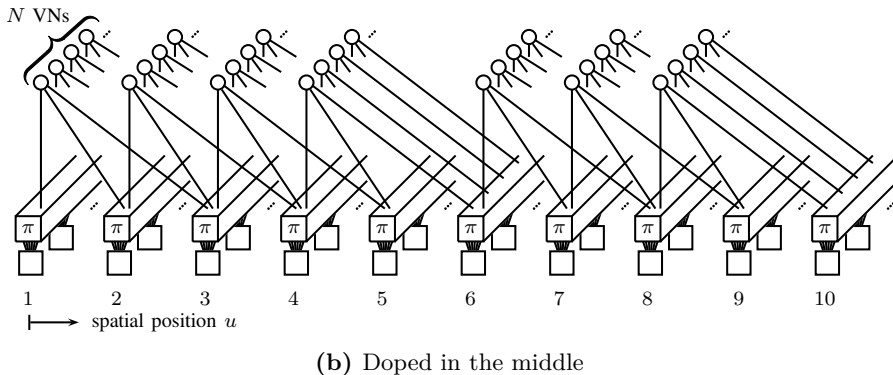
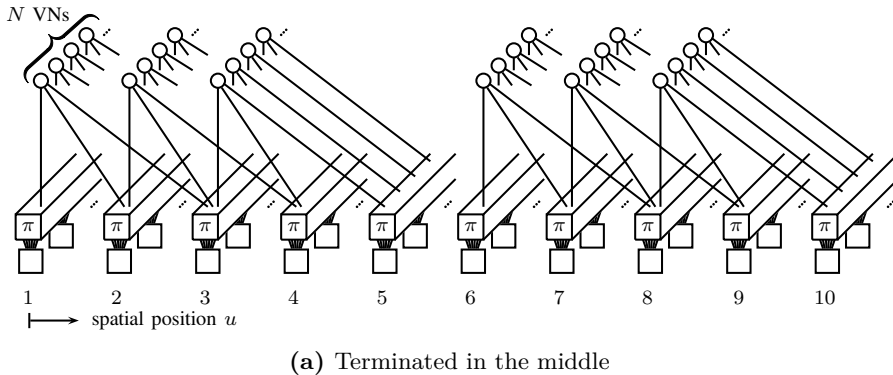


Figure 3.3: Termination vs doping.

both full BP and sliding window decoding must be limited in terms of the maximum number of BP iterations. This can be done with either a hard-set deadline or a conditional rule; devising such heuristic rules (called *stopping criteria*) in the context of the binary-input additive white Gaussian noise channel has attracted considerable research attention [43]–[47]. However, even if a stopping criterion is used, the hard deadline must also be present.

The main downside of using sliding window decoding is that it effectively limits wave-like decoding to a single wave that propagates from the left boundary of the chain since the right boundary is not included in the window. Furthermore, sliding window decoding must be done in a careful way not to lose

sight of the *left* wave as well, which may happen when the maximum number of iterations is limited (the setup we analyze in Paper C).

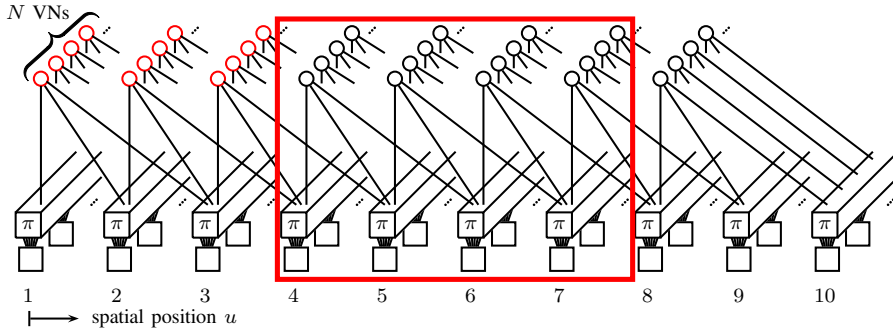
3.3 Parameter Optimization

Error-correcting performance of SC-LDPC codes can be measured in three ways: the *bit error rate* (BER) refers to the probability that a randomly chosen bit remains erased after decoding is finished; the *block error rate* (BLER) refers to the probability that a randomly chosen spatial position contains at least one erased bit; and the *frame error rate* (FER) refers to the probability that the whole chain contains at least one erased bit. These metrics may be specified as targets for code/decoder parameter optimization.

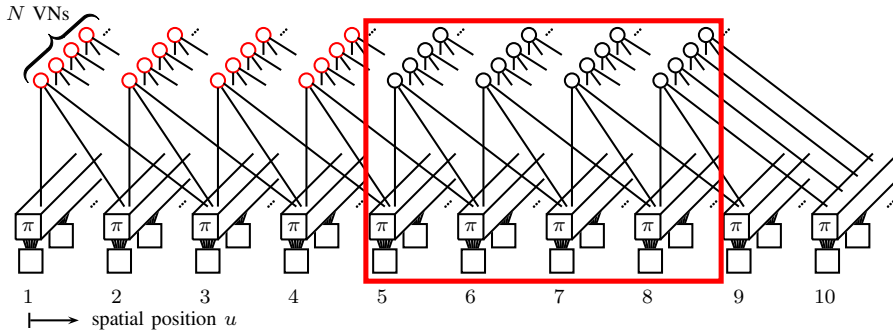
Whereas the asymptotic (i.e., when $N \rightarrow \infty$) performance of SC-LDPC codes is well understood in terms of their iterative decoding thresholds obtained via density evolution analysis, much less is known about their behavior in the finite-length (i.e., finite- N) regime. The code designer is faced with a choice of code and decoder parameters along many dimensions, including:

1. the parameters of the underlying LDPC code, such as
 - a) the code length N ,
 - b) the VN degree d_v and CN degree d_c (in the case of (d_v, d_c) -regular LDPC codes);
2. the specific coupling pattern;
3. the length of the coupled chain L ;
4. the size of the sliding window;
5. possible locations, types, and amounts of doping;
6. the maximum number of decoding iterations;
7. the stopping criterion to use (if any).

All these parameters have an impact on the error-correcting performance, but also on the code rate and on energy, memory, and latency requirements. Navigating this parameter space to choose an optimal combination for a given



(a)



(b)

Figure 3.4: Sliding window decoding.

set of requirements is nontrivial—it would be helpful to have a tool that could predict the code’s error rate as a function of code and decoder parameters.

Finite-length scaling laws aim to do just that.

CHAPTER 4

Finite-Length Scaling Laws

FINITE-LENGTH scaling laws originated in statistical physics, where they arise when studying systems that exhibit phase-transition phenomena [48], including some of the simplest such models studied by percolation theory [49]. Indeed, the BP threshold for LDPC code ensembles (see Section 2.6) can also be regarded as a point of phase transition—from a state where successful decoding is possible to a state where it is not.

The connection between statistical physics and coding theory is in fact even deeper. Sourlas observed in 1989 that error-correcting codes can be expressed in terms of magnetically disordered materials (also known as *spin glasses*) and therefore analyzed using the methods of statistical mechanics [50]. The specific codes he studied had excellent asymptotic performance but very high error floors (Section 2.6); he concluded that such codes are not very useful and abandoned this line of research [20]. However, we can now assert that the connection between coding and spin-glass theory holds beyond the specific class of codes considered by Sourlas.

Montanari applied the tools of spin-glass theory to LDPC codes and introduced the idea of finite-length scaling laws to coding theory [51]. The basic aim of finite-length scaling is to characterize the error-correcting performance

of a code ensemble in the *waterfall region* (Section 2.6) as a function of the finite block length, when operating over channel noise levels close to the iterative BP threshold (i.e., close to the point of phase transition). An example family of waterfall curves is shown in Fig. 4.1 (solid) along with the corresponding predictions by a scaling law (dashed). This line of research led to a finite-length scaling law for LDPC code ensembles that can accurately predict an ensemble's performance over the BEC [23]. Explicit expressions for calculating the scaling parameters for irregular LDPC code ensembles were also developed and used for LDPC code parameter optimization [52]. In [53], the scaling law from [23] was proved to be correct for left-regular right-Poisson LDPC code ensembles and put in a broader perspective of k -core problems for random hypergraphs with applications to error correction and solving large random linear systems. The law was further developed to apply to more general channels in [54], [55]. It has also been used in the realm of uncoordinated multiple access for the analysis of irregular repetition slotted ALOHA [56].

The finite-length scaling framework was used to derive a law to predict the FER of SC-LDPC code ensembles over the BEC in [17], [57]. This law has been applied to protograph-based ensembles [58] in [59] and to generalized SC-LDPC codes in [60], where the component codes are not simple parity-check codes but more elaborated codes such as Reed-Solomon codes (see [61] for an overview).

The scaling law in [17] captures the slope of the FER curve well but shows a significant gap to the simulated curve; closing this gap is the initial challenge that this thesis addresses (Paper A).

4.1 Finite-Length Scaling in a Nutshell

Let us briefly summarize the basic concepts of finite-length scaling of LDPC and SC-LDPC code ensembles over the BEC.

The first crucial step is to realize the importance of degree-one CNs (which represent trivially resolvable parity-check constraints) for the operation of BP decoding and its Tanner graph-based counterparts (i.e., parallel and sequential peeling decoding), as is discussed in Chapter 2. When the number of BP iterations is not limited, the scaling laws focus on sequential peeling decoding because it is easier to analyze—a single iteration of sequential peeling decoding always recovers a single VN (code bit), so the number of iterations performed

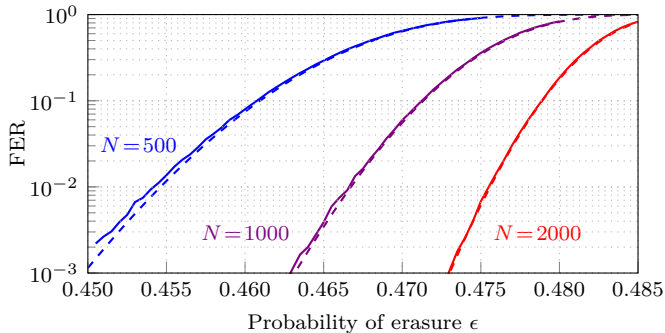


Figure 4.1: Example family of finite-length scaling curves.

by peeling decoding specifies the number of decoded bits.

Peeling decoding halts when it runs out of degree-one CNs. It is therefore natural to focus on the number of degree-one CNs over decoding iterations. Each ensemble, channel, and peeling decoding realization yields a specific sequence of these numbers, referred to as a *decoding trajectory*. Whenever a decoding trajectory hits zero before the codeword is recovered, a decoding error occurs.

The second step is to give up any hope to calculate the probability of that event exactly and to focus instead on statistical properties of decoding trajectories. Specifically, for any given iteration, the distribution of the number of degree-one CNs is shown to converge to a Gaussian whose mean and variance can be calculated by solving numerically a system of coupled differential equations dubbed *mean and covariance evolution* [23]. For uncoupled LDPC codes, mean evolution shows that peeling decoding has a *critical point* where the number of degree-one CNs is at its minimum [23]. For SC-LDPC codes, there is instead a similar *critical phase* where the number of degree-one CNs remains at its minimum steady-state level—this steady state corresponds to the two decoding waves that propagate from the boundaries of the spatially-coupled chain [17] (Chapter 3).

The third step is to let go of the underlying graph-based process that defines decoding trajectories and propose an alternative, simpler stochastic model with matching statistical properties. For both uncoupled LDPC and SC-LDPC codes, the general approach is to make a transition to continuous-time diffusion processes, originally developed for studying particle movements subject to random fluctuations [62].

The final step is to use the simple model to predict the probability that a decoding trajectory hits zero during either the critical point (in the case of uncoupled LDPC codes) or the critical phase (in the case of SC-LDPC codes). For uncoupled LDPC codes, the error probability is estimated as the probability that a Gaussian random variable with positive mean that represents the number of degree-one CNs at the critical point (with mean and variance obtained via mean and covariance evolution) is negative, yielding a Q-function [23].¹

For SC-LDPC codes, the critical steady-state phase of the decoding is modeled in [17] as an Ornstein-Uhlenbeck process [63], and the probability of decoding error is estimated using some known approximations to the first-passage time of that process [64], [65].

An Ornstein-Uhlenbeck process hovers around a certain average (that in the case of SC-LDPC codes represents the average number of degree-one CNs during the critical phase) but experiences fluctuations around it. One could imagine a drunkard doing his best to keep the general direction to the intended destination but experiencing certain difficulties in staying on track (Fig. 4.2). The Ornstein-Uhlenbeck process was first proposed as a model for a particle that experiences a constant drift but also fluctuates as a result of colliding with other particles [63]. It has subsequently been used to model stochastic phenomena in a great variety of applications from finance to neurobiology—and coding theory is yet another one on the list.

Figure 4.2: (right) A mural at Oosterkade, Utrecht, the Netherlands, by the Dutch painters' collective De Strakke Hand (Translation of their name: The Tight Hand.) of Leonard Ornstein writing at a desk. Above Ornstein in the mural, a drunkard is walking down the street, a bottle and a cane in his hand. The stochastic differential equation for the Ornstein-Uhlenbeck process [63] is given. Texts: top left "Prof. Ornstein investigates random movement, 1930", in the middle left the formula " $dx_t = -\theta x_t dt + \sigma dW_t$ ". Source: Wikimedia Commons.²

¹The model that accounts for the possibility of the trajectory to hit zero in a close neighborhood of the critical point—i.e., not only at the critical point itself—results in better predictions. This refined model can also be expressed as a Q-function [23].



²Image credit: Hansmuller / Hans Muller (https://commons.wikimedia.org/wiki/File:Leonard_Ornstein_mural,_Oosterkade,_Utrecht,_2021_-_1.jpg), <https://creativecommons.org/licenses/by-sa/4.0/legalcode>

CHAPTER 5

Summary

THIS CHAPTER briefly summarizes the contributions of the appended papers, highlights some of the conclusions that can be drawn from the obtained results, and outlines the directions of possible future research in the area of finite-length scaling laws for SC-LDPC codes.

5.1 Contributions

Paper A

The first contribution of Paper A is a scaling law to estimate the finite-length FER performance of SC-LDPC code ensembles under full BP decoding with an unlimited number of decoding iterations over the BEC that provides a more accurate estimation than was previously available, closing the gap between the simulated and predicted FER that is observed in [17]. We changed the model used in [17] to predict the FER—instead of treating the steady-state phase of peeling decoding as a single Ornstein-Uhlenbeck process (see Chapter 4), we model it as a combination of two independent Ornstein-Uhlenbeck processes, with each process corresponding to a single decoding wave that propagates

through the spatially-coupled chain (see Chapter 3). This requires estimating the parameters of each Ornstein-Uhlenbeck process from the *truncated* ensemble (instead of from the terminated ensemble) to isolate a single decoding wave. This effectively closes the gap between predicted and simulated FER curves that was present in [17].

Encouraged by the model’s ability to accurately predict the FER, we extended it to also predict the other two figures of merit, namely the BLER and the BER. This was possible because the model for the FER already predicts the distribution in the number of peeling decoding iterations before the failure, from which the fraction of the bits and the distance traveled by the decoding waves can be inferred. The resulting BLER and BER predictions are also accurate.

The model is then powerful enough to capture the behavior of the decoding process relevant for error rate prediction under full BP decoding. Can it be extended to predict the performance of terminated SC-LDPC code ensembles under the more practical *sliding window* decoding? Paper A answers this question in the affirmative. We noted that during sliding window decoding, the window of size W does not allow the wave from the right termination boundary to propagate by further than W spatial positions. We then modeled sliding window decoding as a two-phase process: During the first phase, which corresponds to the first $L - W$ spatial positions (where L is the length of the chain), only a single wave propagating from the left boundary of the wave is present. During the second phase, which corresponds to the last W positions, there may be two waves. With each wave modeled as an independent Ornstein-Uhlenbeck process, we manage to accurately predict the FER, BLER, and BER under sliding window decoding. Since sliding window decoding limits decoding latency and is used in practice (see Chapter 3), the scaling law for sliding window decoding is an important step toward practically relevant code and decoder parameter optimization. The drawback of the model (as well as the models in [23] and [17]) is that it still assumes an unlimited number of decoding iterations—a limitation we overcome in Paper C.

Paper B

Paper B studies SC-LDPC code ensembles with doping (see Section 3.1). Doping is an interesting technique for limiting the rate loss in long spatially-coupled chains relevant for streaming applications. Equipped with the scaling

law for sliding window decoding that we developed in Paper A, we decided to tackle the problem of predicting the BER and BLER of SC-LDPC code ensembles with doping. We derived mean evolution equations for doping points and observed that these too exhibit a critical point, as do uncoupled LDPC codes (Section 4.1). Further, we observed that the doping points either produce two decoding waves or none at all. We followed the approach used in the case of uncoupled LDPC ensembles [23] and estimated the probability of the doping point to survive the critical point with a Q-function. The scaling law that we derived for periodically applied doping yields accurate BER and BLER predictions (predicting the FER is not relevant for infinite chains since the FER in this case is always one).

We used the scaling law for code parameter optimization and showed that SC-LDPC codes with doping can achieve better code rates for a given target error rate than can SC-LDPC codes with full termination only, making doping an interesting extra dimension to be considered.

Paper C

Paper C addresses the challenging case where the number of decoding iterations is limited, both for full BP and for sliding window decoding, which is the most practically relevant setup. All scaling laws proposed for LDPC and SC-LDPC code ensembles, including those we ourselves proposed in Papers A and B, consider decoding with unlimited number of iterations. In that case, the performance of BP decoding (or parallel peeling decoding, its Tanner graph-based counterpart) and sequential peeling decoding (Section 2.5) is the same. When a limit on the maximum number of BP iterations is imposed, that equivalence no longer holds.

For full BP decoding with limited iterations, we developed several scaling laws that offer different trade-offs between prediction accuracy and computational complexity. The main idea is to establish a per-iteration equivalence between BP and (sequential) peeling decoding: How many peeling decoding iterations does a single BP iteration correspond to? We first assumed that a single BP iteration always corresponds to a fixed number of peeling decoding iterations, which we obtained from the average number of degree-one CNs during the steady state of peeling decoding. This resulted in a surprisingly good prediction given the simplicity of that assumption. Our next scaling laws also assume a certain equivalence between BP and peeling decoding iterations

but model that correspondence as a random variable instead. We obtained accurate FER predictions.

Finally, we proposed a scaling law for *sliding window decoding* with a limited number of iterations. Here, there is a certain competition between the sliding window and the position of the left decoding wave; in addition to the potential sources of decoding failure that we considered in Paper A, there is a chance that the window *overtakes* the wave and decoding fails for that reason. We modeled the position of the left decoding wave by an integrated Ornstein-Uhlenbeck process with some additional variance and the left boundary of the window by an absorbing barrier. We then derived the partial differential equation that describes the evolution of the probability density function of the location of the wave in the presence of the absorbing barrier (known as the Fokker-Planck equation) and solved it numerically. This allowed us to obtain an estimate of the probability that the wave is overtaken by the sliding window. We used this estimation to obtain an overall probability of decoding error. The resulting FER predictions are also astonishingly accurate.

5.2 Future Work

Several directions of future research are possible. First, it would be interesting to make sure that the improvement in prediction accuracy of the law we proposed in Paper A compared with the law in [17] carries over to other set-ups where the law in [17] was applied, namely for protograph-based SC-LDPC [59] and generalized [60] SC-LDPC code ensembles. We expect the analysis to carry over and to result in similar improvements of prediction accuracy.

Second, it is of obvious practical interest to extend the scaling laws proposed in Papers A–C to other channels, notably to the binary-input additive white Gaussian noise channel, perhaps by following the steps outlined in [54], [55] for uncoupled LDPC code ensembles.

Third, it would help the case for the use of finite-length scaling laws if it could be shown that *concentration* holds in this context, too. In other words, can we expect the performance of a *specific* code from an SC-LDPC code ensemble to be close to that of the ensemble average, especially in the context of sliding window decoding and/or limited number of iterations?

Fourth, some of the scaling parameters used in the laws in Papers A–C are

obtained via numerical simulation. (The rationale there is that the amount of simulation required to estimate these parameters is significantly smaller than needed to estimate the error rates; furthermore, these parameters need to only be estimated once and do not need to be re-simulated for every combination of code, decoder, and channel setups.) Is it possible to calculate them analytically instead?

Finally, the contributions of this thesis made SC-LDPC codes better studied, in some sense, than are uncoupled LDPC codes, which is quite surprising. In particular, no work has attempted to estimate the finite-length performance of uncoupled LDPC code ensembles in the presence of a limit on the number of BP iterations, whereas Paper C provides a way to do just that for their spatially-coupled counterparts. Could it be possible to derive a similar scaling law for uncoupled LDPC code ensembles as well?

5.3 Conclusion

The contributions of Papers A–C lay the groundwork for finite-length code and decoder parameter optimization of SC-LDPC codes. The tools and insights of these papers have already been used for such optimization [66], [67]; our hope is to see this work continuing in the years to come.

References

- [1] International Telecommunication Union, *Measuring Digital Development, Facts and Figures 2021*. ITU Publications, 2021.
- [2] *Cisco Visual Networking Index, Forecast and Trends, 2017–2022*. Cisco Public, 2019.
- [3] C. E. Shannon, “A mathematical theory of communication,” *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, Jul. 1948.
- [4] M. A. Levinson (Director), *The Bit Player*. [Documentary]. IEEE Information Theory Society, 2019.
- [5] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near shannon limit error-correcting coding and decoding: Turbo-codes (1),” in *Proc. IEEE Int. Conf. Commun. (ICC)*, vol. 2, Geneva, Switzerland, May 1993, pp. 1064–1070.
- [6] D. J. C. MacKay and R. M. Neal, “Good codes based on very sparse matrices,” in *Proc. 5th IMA Conf. Cryptography and Coding*, Cirencester, UK, Dec. 1995, pp. 100–111.
- [7] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.
- [8] N. Alon and M. Luby, “A linear time erasure-resilient code with nearly optimal recovery,” *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 1732–1736, Nov. 1996.

- [9] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, “A digital fountain approach to reliable distribution of bulk data,” *ACM SIGCOMM Computer Communication Rev.*, vol. 28, no. 4, pp. 56–67, Oct. 1998.
- [10] R. G. Gallager, *Low-Density Parity-Check Codes*. MIT Press, Cambridge, MA, USA, 1963.
- [11] E. Arıkan, “Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels,” *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, Jun. 2009.
- [12] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Şaşıođlu, and R. L. Urbanke, “Reed-Muller codes achieve capacity on erasure channels,” *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4298–4316, Feb. 2017.
- [13] 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.
- [14] 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.
- [15] 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.
- [16] M. Qiu, X. Wu, J. Yuan, and A. Graell i Amat, “Generalized spatially-coupled parallel concatenated codes with partial repetition,” *arXiv*, 2022, [Online] Available: <https://arxiv.org/pdf/2201.09414.pdf>.
- [17] 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.
- [18] R. Tanner, “A recursive approach to low complexity codes,” *IEEE Trans. Inf. Theory*, vol. 27, no. 5, pp. 533–547, Sep. 1981.
- [19] W. E. Ryan and S. Lin, *Channel Codes, Classical and Modern*. Cambridge University Press, 2009.
- [20] T. J. Richardson and R. L. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.

-
- [21] 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.
- [22] 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.
- [23] 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.
- [24] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, “Analysis of low density codes and improved designs using irregular graphs,” in *Proc. Annu. ACM Symp. Theory Comput. (STOC)*, 1998, pp. 249–258.
- [25] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, “Improved low-density parity-check codes using irregular graphs,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 585–598, Feb. 2001.
- [26] 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.
- [27] M. Lentmaier, A. Sridharan, D. J. Costello, Jr., 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.
- [28] 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.
- [29] A. Sridharan, D. Truhachev, M. Lentmaier, D. J. Costello, Jr., 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.

- [30] D. G. M. Mitchell, A. E. Pusane, M. Lentmaier, and D. J. Costello, Jr., “Exact free distance and trapping set growth rates for LDPC convolutional codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, St. Petersburg, Russia, Aug. 2011, pp. 1096–1100.
- [31] 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.
- [32] 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, Jul. 2012, pp. 373–377.
- [33] 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.
- [34] C. Häger, A. Graell i Amat, A. Alvarado, F. Brännström, and E. Agrell, “Optimized bit mappings for spatially coupled LDPC codes over parallel binary erasure channels,” in *IEEE Int. Conf. Commun. (ICC)*, Sydney, Australia, 2014, pp. 2064–2069.
- [35] C. Häger, A. Graell i Amat, F. Brännström, A. Alvarado, and E. Agrell, “Terminated and tailbiting spatially coupled codes with optimized bit mappings for spectrally efficient fiber-optical systems,” *J. Lightw. Technol.*, vol. 33, no. 7, pp. 1275–1285, Jan. 2015.
- [36] T. Jerkovits, G. Liva, and A. Graell i Amat, “Improving the decoding threshold of tailbiting spatially coupled LDPC codes by energy shaping,” *IEEE Commun. Lett.*, vol. 22, no. 4, pp. 660–663, Feb. 2018.
- [37] M. Zhu, D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, Jr., “A novel design of spatially coupled LDPC codes for sliding window decoding,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angeles, CA, USA, Jun. 2020.
- [38] —, “Decoder error propagation mitigation for spatially coupled LDPC codes,” in *Proc. IEEE Int. Symp. Inf. Theory Appl.*, Kapolei, USA, 2020.

-
- [39] S. Cammerer, V. Aref, L. Schmalen, and S. ten Brink, “Triggering wave-like convergence of tail-biting spatially coupled LDPC codes,” in *Proc. Annu. Conf. Inf. Sci. Syst. (CISS)*, Princeton, NJ, USA, 2016, pp. 93–98.
- [40] 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.
- [41] N. ul Hassan, A. E. Pusane, M. Lentmaier, G. P. Fettweis, and D. J. Costello, Jr., “Non-uniform windowed decoding schedules for spatially coupled codes,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Atlanta, GA, USA, Dec. 2013, pp. 1862–1867.
- [42] N. Ul Hassan, A. E. Pusane, M. Lentmaier, G. P. Fettweis, and D. J. Costello, Jr., “Non-uniform window decoding schedules for spatially coupled LDPC codes,” *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 501–510, Nov. 2017.
- [43] F. Kienle and N. Wehn, “Low complexity stopping criterion for LDPC code decoders,” in *Proc. 61st IEEE Semiann. Vehicular Technol. Conf. (VTC)*, vol. 1, Stockholm, Sweden, May 2005, pp. 606–609.
- [44] J. Li, X.-h. You, and J. Li, “Early stopping for LDPC decoding: Convergence of mean magnitude (CMM),” *IEEE Commun. Lett.*, vol. 10, no. 9, pp. 667–669, Oct. 2006.
- [45] D. Shin, K. Heo, S. Oh, and J. Ha, “A stopping criterion for low-density parity-check codes,” in *Proc. 65th IEEE Semiann. Vehicular Technol. Conf. (VTC)*, Dublin, Ireland, Apr. 2007, pp. 1529–1533.
- [46] Z. H. Cai, J. Hao, and U. Sethakaset, “Efficient early stopping method for LDPC decoding based on check-node messages,” in *Proc. 42nd Asilomar Conf. Signals, Syst., Comput.*, Pacific Grove, CA, USA, Oct. 2008, pp. 466–469.
- [47] C. Condo, A. Baghdadi, and G. Masera, “Energy-efficient multi-standard early stopping criterion for low-density-parity-check iterative decoding,” *IET Commun.*, vol. 8, no. 12, pp. 2171–2180, Aug. 2014.

- [48] V. Privman, “Finite-size scaling theory,” in *Finite Size Scaling and Numerical Simulation of Statistical Systems*, V. Privman, Ed., Singapore: World Scientific Publ., 1990, pp. 1–98.
- [49] D. Stauffer, “Scaling theory of percolation clusters,” *Physics reports*, vol. 54, no. 1, pp. 1–74, 1979.
- [50] N. Sourlas, “Spin-glass models as error-correcting codes,” *Nature*, vol. 339, no. 6227, pp. 693–695, Jun. 1989.
- [51] A. Montanari, “The glassy phase of Gallager codes,” *Europ. Phys. J. B-Condensed Matter and Complex Systems*, vol. 23, no. 1, pp. 121–136, Apr. 2001.
- [52] A. Amraoui, A. Montanari, and R. Urbanke, “How to find good finite-length codes: From art towards science,” *European Trans. Telecommun.*, vol. 18, no. 5, pp. 491–508, Apr. 2007.
- [53] A. Dembo and A. Montanari, “Finite size scaling for the core of large random hypergraphs,” *Ann. Appl. Probab.*, vol. 18, no. 5, pp. 1993–2040, Oct. 2008.
- [54] J. Ezri, A. Montanari, S. Oh, and R. Urbanke, “The slope scaling parameter for general channels, decoders, and ensembles,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Toronto, Canada, Jul. 2008, pp. 1443–1447.
- [55] J. Ezri, R. Urbanke, A. Montanari, and S. Oh, “Computing the threshold shift for general channels,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Toronto, Canada, Jul. 2008, pp. 1448–1452.
- [56] A. Graell i Amat and G. Liva, “Finite-length analysis of irregular repetition slotted ALOHA in the waterfall region,” *IEEE Commun. Lett.*, vol. 22, no. 5, pp. 886–889, May 2018.
- [57] P. M. Olmos and R. Urbanke, “A closed-form scaling law for convolutional LDPC codes over the BEC,” in *Proc. IEEE Inf. Theory Workshop (ITW)*, Seville, Spain, Sep. 2013.
- [58] D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, Jr., “Spatially coupled LDPC codes constructed from protographs,” *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 4866–4889, Jul. 2015.

-
- [59] 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.
- [60] D. G. M. Mitchell, P. M. Olmos, M. Lentmaier, and D. J. Costello, Jr., “Spatially coupled generalized LDPC codes: Asymptotic analysis and finite length scaling,” *IEEE Trans. Inf. Theory*, vol. 67, no. 6, pp. 3708–3723, Apr. 2021.
- [61] D. J. Costello, Jr., 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.
- [62] G. A. Pavliotis, *Stochastic Processes and Applications, Diffusion Processes, the Fokker-Planck and Langevin Equations*. Springer, New York, NY, USA, 2014.
- [63] G. E. Uhlenbeck and L. S. Ornstein, “On the theory of the brownian motion,” *Phys. Rev.*, vol. 36, pp. 823–841, 5 Sep. 1930.
- [64] M. U. Thomas, “Some mean first-passage time approximations for the Ornstein-Uhlenbeck process,” *J. Appl. Probab.*, vol. 12, no. 3, pp. 600–604, Sep. 1975.
- [65] A. G. Nobile, L. M. Ricciardi, and L. Sacerdote, “Exponential trends of Ornstein-Uhlenbeck first-passage-time densities,” *J. Appl. Probab.*, vol. 22, no. 2, pp. 360–369, Jun. 1985.
- [66] 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.
- [67] 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.*, 2022, Early Access.

