# Concatenated Spatially Coupled LDPC Codes With Sliding Window Decoding for Joint Source-Channel Coding

Ahmad Golmohammadi, *Member, IEEE*, and David G. M. Mitchell, *Senior Member, IEEE*

***Abstract*— In this paper, a method for joint source-channel coding (JSCC) based on concatenated spatially coupled low-density parity-check (SC-LDPC) codes is investigated. A construction consisting of two SC-LDPC codes is proposed: one for source coding and the other for channel coding, with a joint belief propagation-based decoder. Also, a novel windowed decoding (WD) scheme is presented with significantly reduced latency and complexity requirements. The asymptotic behavior for various graph node degrees is analyzed using a protograph-based Extrinsic Information Transfer (EXIT) chart analysis for both LDPC block codes with block decoding and for SC-LDPC codes with the WD scheme, showing robust performance for concatenated SC-LDPC codes. Simulation results show a notable performance improvement compared to existing state-of-the-art JSCC schemes based on LDPC codes with comparable latency and complexity constraints.**

***Index Terms*— Low-density parity-check (LDPC) codes, spatially coupled codes, joint source-channel coding, window decoding.**

## I. Introduction

FOR infinite source and channel code block lengths, it is known that arbitrarily high reliability can be attained if the source entropy is less than the channel capacity by the *separation principle*, where source and channel coding are performed separately [1]. On the other hand, in a non-asymptotic regime with delay constraints, a joint source-channel design can be more attractive [2], where the residual redundancy of the source sequence can be used by the channel decoder to improve channel decoding [3], [4]. Block error-correcting codes can be directly applied for source coding where the decoder is used to compress the source data and the encoder is used to reconstruct it [5], [6]. This method was shown to be efficient for memoryless *symmetric sources* under the Hamming distortion measure, where the *average distortion* is measured as the average fraction of source bits that are not correctly reconstructed [6]. However, for many sources, the source sequence is *asymmetric* (*e.g.*, sequences with a small number of ones), for which syndrome source coding [7] can be an efficient method. In syndrome source coding, the source sequence $\mathbf{s}$ is considered as a channel output and the source encoder generates the syndrome $\mathbf{u} = \mathbf{s}\mathbf{H}^{\mathsf{T}}$ as the compressed data, where $\mathbf{H}$ is the parity-check matrix of the linear error-correcting code. At the receiver, the source decoder tries to produce an estimate $\hat{\mathbf{s}}$ consistent with $\mathbf{u}$ [7]. Low-density parity-check (LDPC) codes [8] with a belief propagation (BP) algorithm were proposed for syndrome source coding in [9] and then further investigated with a noisy channel in [10].

Three methods of joint source-channel coding (JSCC) were proposed in [10], specifically 1) two LDPC codes, 2) a single LDPC code, and 3) Lotus codes. This paper is focused on the first method, *i.e.*, two LDPC codes, where LDPC based syndrome source coding is then concatenated with an LDPC channel encoder. Therefore, at the transmitter, there are two concatenated LDPC codes that are applied sequentially. At the receiver, the concatenated codes can be represented as a single bipartite graph and jointly decoded by a BP algorithm. Related JSCC schemes have been successfully employed using turbo codes [11], [12], rate-compatible punctured convolutional codes [13], [14], and two concatenated LDPC block codes [15]. In [15]–[17], a joint protograph extrinsic information transfer (EXIT) analysis was proposed to calculate the decoding threshold concatenated protograph-based for LDPC codes and the source and channel code were jointly optimized to achieve a good bit error rate performance in [18]. Punctured protograph LDPC codes were investigated for a binary Hidden Markov Model (HMM) source in [19].

Spatially coupled LDPC (SC-LDPC) codes can be obtained by coupling together (connecting) a series of $L$ disjoint LDPC block codes to make a larger connected graph, and have been shown to have capacity approaching channel coding performance as a result of the *threshold saturation* phenomenon [20]–[23]. Closely related spatially coupled low-density generator matrix (SC-LDGM) code ensembles, where sparse generator matrices are coupled together to create a connected

Manuscript received February 18, 2021; revised July 5, 2021 and September 24, 2021; accepted October 30, 2021. Date of publication November 8, 2021; date of current version February 17, 2022. This material is based upon work supported by the National Science Foundation under Grant Nos. ECCS-1710920, OIA-1757207, and HRD-1914635. An earlier version of this article was presented in part at the 2018 IEEE International Symposium on Information Theory. The associate editor coordinating the review of this article and approving it for publication was S. Rini. *(Corresponding author: Ahmad Golmohammadi.)*

Ahmad Golmohammadi was with the Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM 88003 USA. He is now with Bose Corporation, Framingham, MA 01701 USA (e-mail: golmoham@nmsu.edu).

David G. M. Mitchell is with the Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM 88003 USA (e-mail: dgmm@nmsu.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2021.3126750.

Digital Object Identifier 10.1109/TCOMM.2021.3126750

graph, were subsequently shown to have rate-distortion bound approaching performance for lossy source compression, displaying an analogous distortion saturation behavior [24], [25]. SC-LDPC codes were also shown to perform well for the problem of transmission of two possibly correlated sources to a common destination with the help of a relay over binary erasure channels (BECs) with a JSCC coding scheme [26].

In this paper, we extend our preliminary results in [27] and present a construction of practically interesting protograph-based concatenated $(J, K)$-regular SC-LDPC codes for JSCC. We show that they can be realized efficiently via sequential source and channel convolutional encoders and can be decoded with a joint BP decoder. Furthermore, we propose a novel low-latency windowed decoding (WD) scheme for the concatenated SC-LDPC-based system with significantly reduced latency and complexity requirements. The main contributions beyond [27] include:

- We present an EXIT chart analysis, extending the approaches of [15], [17], [18] to consider the concatenated convolutional protographs and window structure of the decoder. The results show that while the asymptotic performance (thresholds) of regular, uncoupled LDPC codes for JSCC worsen by increasing the graph density for a fixed coding rate, the thresholds improve with SC-LDPC code ensembles. Furthermore, it is shown that by increasing the decoder window size, the threshold improves up to a certain value, after which negligible gains are observed;
- A comprehensive study of the design parameters is performed, showing that concatenated SC-LDPC codes display good performance for a variety of sources under various latency constraints. Considerations for the protograph design are presented to ensure that the resulting code is amenable to low-complexity partial syndrome former encoding and good window decoding thresholds/performance;
- Simulation results for a binary memoryless source and a binary input additive white Gaussian noise (AWGN) channel confirm the improved BER performance versus comparable concatenated LDPC block codes on an equal latency basis. Moreover, we demonstrate superior threshold and finite-length performance when compared to optimized irregular LDPC codes taken from [16], [18];
- The transmission of binary sources generated by an HMM is considered where it is shown that the behavior of concatenated SC-LDPC codes is similar to a memoryless Bernoulli source. Concatenated SC-LDPC codes are again shown to have competitive performance versus state-of-the-art optimized irregular LDPC codes from [19];
- Finally, rate-compatible concatenated SC-LDPC codes for JSCC are presented and shown to have robust performance over a variety of coding rates by puncturing.

## II. LDPC-Based JSCC

In this section, we provide the necessary background and notation as well as summarizing the LDPC-based JSCC approach proposed in [10].

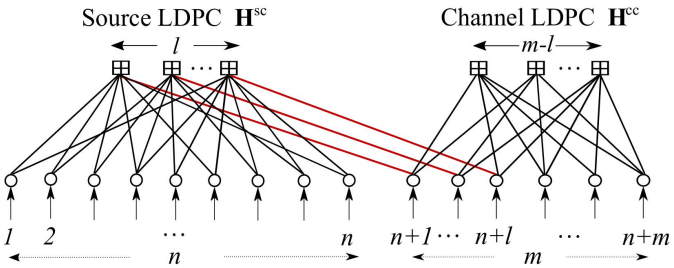


Fig. 1. Concatenated LDPC Tanner graphs for JSCC.

### A. Notation and Background

Throughout the paper, unless otherwise stated, we assume a memoryless Bernoulli source, such that $p_v = \mathbb{P}(s_v = 1)$ for source bit $s_v$, and a binary-input additive white Gaussian noise (AWGN) channel where $r_v$ is the received binary phase shifting keying (BPSK) value for a symbol transmitted on a channel with noise variance $\sigma_n^2$. An LDPC code is described as the null space of a parity-check matrix $\mathbf{H}$ to which we associate a Tanner graph in the usual way [28]. We use the shorthand notation $[a : b]$ to represent the set $\{a, a+1, \ldots, b\}$ for integers $b > a$. Matrices and vectors will be represented by bold fonts and we introduce "sc" or "cc" superscripts to the parameters to indicate their association to the source and channel code, respectively. Table VI in the Appendix displays a table of the commonly used code parameters for reference.

### B. Encoding and Decoding

A first LDPC code with parity-check matrix $\mathbf{H}^{\mathsf{sc}}$ is used to calculate the syndrome $\mathbf{u}$ corresponding to the source input $\mathbf{s}$, and then a second LDPC code with parity-check matrix $\mathbf{H}^{\mathsf{cc}}$ and generator matrix $\mathbf{G}^{\mathsf{cc}}$ is used to encode the compressed sequence for transmission through a noisy channel. For this system, a codeword $\mathbf{v}$ is obtained as

$$\mathbf{v} = \mathbf{u}\mathbf{G}^{\mathsf{cc}} = \left(\mathbf{s}\mathbf{H}^{\mathsf{sc}\mathsf{T}}\right)\mathbf{G}^{\mathsf{cc}}, \tag{1}$$

where $\mathbf{H}^{\mathsf{sc}}$ is an $l \times n$ sparse binary parity-check matrix with *compression rate* $R^{\mathsf{sc}} = l/n < 1$, $\mathbf{s}$ is the length $n$ binary source input, $\mathbf{u}$ is the length $l$ binary compressed source word, and $\mathbf{G}^{\mathsf{cc}}$ is the $l \times m$ binary systematic LDPC channel code generator matrix with *code rate* $R^{\mathsf{cc}} = l/m$. Fig. 1 shows the concatenated Tanner graphs used at the decoder, where the subgraphs associated with $\mathbf{H}^{\mathsf{sc}}$ and $\mathbf{H}^{\mathsf{cc}}$ are referred to as the *source graph* and *channel graph*, respectively, and each check node in the source graph is connected to a single variable node in the channel graph. The *overall code rate* is $R = \frac{R^{\mathsf{cc}}}{R^{\mathsf{sc}}} = \frac{n}{m}$.

Let $L_v^{\mathsf{sc}}$ and $L_v^{\mathsf{cc}}$ denote the incoming log-likelihood ratios (LLRs) for the variable nodes $v \in [1 : n]$ of the source decoder and variable nodes $v \in [n+1 : n+m]$ of the channel decoder, respectively. Then, for a memoryless Bernoulli source such that $p_v = \mathbb{P}(s_v = 1)$, $L_v^{\mathsf{sc}} = \log\left(\frac{1-p_v}{p_v}\right)$, and for binary-input AWGN channel, $L_v^{\mathsf{cc}} = \frac{2r_v}{\sigma_n^2}$, where $r_v$ is the received BPSK value for a symbol transmitted on a channel

with noise variance $\sigma_n^2$. As in [15], we apply BP to the concatenated graph as follows[1]: variable nodes send their message to check nodes at iteration $\ell$ as

$$m_{v\to c}^{\mathsf{sc},(\ell)} = L_v^{\mathsf{sc}} + \sum_{c'\neq c} m_{c'\to v}^{\mathsf{sc},(\ell-1)}, \tag{2}$$

$$m_{v\to c}^{\mathsf{cc},(\ell)} = L_v^{\mathsf{cc}} + m_{c\to v}^{\mathsf{sc}\to\mathsf{cc},(\ell-1)} + \sum_{c'\neq c} m_{c'\to v}^{\mathsf{cc},(\ell-1)}, \tag{3}$$

$$m_{v\to c}^{\mathsf{cc}\to\mathsf{sc},(\ell)} = L_v^{\mathsf{cc}} + \sum_{c'\neq c} m_{c'\to v}^{\mathsf{cc},(\ell-1)}, \tag{4}$$

$$m_{v\to c}^{\mathsf{cc},(\ell)} = L_v^{\mathsf{cc}} + \sum_{c'\neq c} m_{c'\to v}^{\mathsf{cc},(\ell-1)}, \tag{5}$$

where $m_{v\to c}^{\mathsf{sc},(\ell)}$, $m_{v\to c}^{\mathsf{cc},(\ell)}$, and $m_{v\to c}^{\mathsf{cc}\to\mathsf{sc},(\ell)}$ are the messages at iteration $\ell$ passed from the $v^{\text{th}}$ variable node to the $c^{\text{th}}$ check node within the source graph, within the channel graph, and between the channel and the source graphs, respectively. We employ the above BP update rules such that (2) applies for variable node indices $v \in [1:n]$, (3) and (4) for $v \in [n+1:n+l]$, and (5) for $v \in [n+l+1:n+m]$.

For check to variable messages, $m_{c\to v}^{\mathsf{sc},(\ell)}$, $m_{c\to v}^{\mathsf{cc},(\ell)}$, and $m_{c\to v}^{\mathsf{sc}\to\mathsf{cc},(\ell)}$ represent the messages passed from the $c^{\text{th}}$ check node to the $v^{\text{th}}$ variable node within the source graph, within the channel graph, and between the source and channel graphs, respectively, given as

$$m_{c\to v}^{\mathsf{sc},(\ell)} = 2\tanh^{-1}\left(\tanh\left(\frac{m_{v\to c}^{\mathsf{cc}\to\mathsf{sc},(\ell)}}{2}\right)\prod_{v'\neq v}\tanh\left(\frac{m_{v'\to c}^{\mathsf{sc},(\ell)}}{2}\right)\right), \tag{6}$$

$$m_{c\to v}^{\mathsf{sc}\to\mathsf{cc},(\ell)} = 2\tanh^{-1}\left(\prod_{v'\neq v}\tanh\left(\frac{m_{v'\to c}^{\mathsf{sc},(\ell)}}{2}\right)\right), \tag{7}$$

$$m_{c\to v}^{\mathsf{cc},(\ell)} = 2\tanh^{-1}\left(\prod_{v'\neq v}\tanh\left(\frac{m_{v'\to c}^{\mathsf{cc},(\ell)}}{2}\right)\right). \tag{8}$$

Note that $m_{c\to v}^{\mathsf{sc},(0)} = m_{c\to v}^{\mathsf{cc},(0)} = m_{c\to v}^{\mathsf{sc}\to\mathsf{cc},(0)} = 0$. Equations (6) and (7) apply for check node indices $c \in [1:l]$ and (8) for $c \in [l+1:m]$. After $I$ iterations of decoding, BP is terminated by computing the LLR of each source bit $s_v$, *i.e.,* $\text{LLR}(s_v) = L_v^{\mathsf{sc}} + \sum_c m_{c\to v}^{\mathsf{sc},(I)}$, whereby the $v^{\text{th}}$ source bit is estimated as $\hat{s}_v = 0$ if $\text{LLR}(s_v) \geq 0$, and $\hat{s}_v = 1$ otherwise.

## III. Concatenated SC-LDPC Codes for JSCC

In this section, we present our concatenated construction of SC-LDPC codes, design considerations, and the encoding and decoding procedures.

[1]In [29], the two LDPC matrices ($\mathbf{H}^{\mathsf{sc}}$ and $\mathbf{H}^{\mathsf{cc}}$) are combined as one LDPC matrix and standard message passing applied between variable nodes and check nodes; however, we choose to follow separated BP updates as applied in [15] for convenience.

### A. SC-LDPC Protographs

A protograph [30] is a small bipartite graph that connects a set of $n_v$ variable nodes to a set of $n_c$ check nodes by a set of edges, and it can be represented by a parity-check or *base* biadjacency matrix $\mathbf{B} = [B_{x,y}]$, where $B_{x,y}$ is taken to be the number of edges connecting variable node $v_y$ to check node $c_x$. The parity-check matrix $\mathbf{H}$ of a protograph-based LDPC block code can be created by expanding $\mathbf{B}$ using a *lifting factor* $M$, where each non-zero entry in $\mathbf{B}$ is replaced by a sum of $B_{x,y}$ non-overlapping permutation matrices of size $M \times M$ and each zero entry is replaced by the $M \times M$ all-zero matrix. An important property of constructing codes from a protograph is that each lifted code inherits the graph neighborhood structure and degree distribution of the protograph.

*1) Unterminated Convolutional Protographs:* An unterminated SC-LDPC code ensemble code can be represented by means of a *convolutional protograph* [20] with base matrix

$$\mathbf{B}_{[0,\infty]} = \begin{bmatrix} \ddots & & \ddots & & & \\ \mathbf{B}_{m_s} & \mathbf{B}_{m_s-1} & \cdots & \mathbf{B}_0 & & \\ & \ddots & & \ddots & & \\ & & \mathbf{B}_{m_s} & \mathbf{B}_{m_s-1} & \cdots & \mathbf{B}_0 \\ & & & \ddots & & \ddots \end{bmatrix}, \tag{9}$$

where $m_s$ is the *syndrome former memory* of the code and the $b_c \times b_v$ *component base matrices* $\mathbf{B}_i$, $i \in [0:m_s]$, determine the edge connections from the $b_v$ variable nodes at time $t$ to the $b_c$ check nodes at time $t+i$. Starting from a $b_c \times b_v$ block base matrix $\mathbf{B}$, an "edge-spreading" procedure [20] can be applied to obtain the component base matrices $\mathbf{B}_i$, where $\mathbf{B}_0 + \mathbf{B}_1 + \cdots + \mathbf{B}_{m_s} = \mathbf{B}$. An ensemble of time-varying SC-LDPC codes can then be formed from $\mathbf{B}_{[0,\infty]}$ using the protograph construction method described above. For example, a $(3,6)$-regular SC-LDPC code ensemble with $m_s = 2$ can be constructed from the block base matrix $\mathbf{B} = [3\ \ 3]$ by defining the component base matrices $\mathbf{B}_0 = [1\ \ 1] = \mathbf{B}_1 = \mathbf{B}_2$.

*2) Terminated SC-LDPC Code Ensembles:* Suppose that we start the convolutional code with parity-check matrix defined in (9) at time $t = 0$ and terminate it after $t = L$ time instants, the resulting finite-length base matrix is then given by

$$\mathbf{B}_{[0,L-1]} = \begin{bmatrix} \mathbf{B}_0 & & \\ \vdots & \ddots & \\ \mathbf{B}_{m_s} & & \mathbf{B}_0 \\ & \ddots & \vdots \\ & & \mathbf{B}_{m_s} \end{bmatrix}_{(L+m_s)b_c \times Lb_v}. \tag{10}$$

The matrix $\mathbf{B}_{[0,L-1]}$ can be considered as the base matrix of a terminated protograph-based SC-LDPC code ensemble, where $L$ is called the *coupling length*. Termination results in a rate loss: the *design compression rate* for syndrome source coding with $\mathbf{B}_{[0,L-1]}$ is $R_L^{\mathsf{sc}} = \left(\frac{L+m_s}{L}\right)\frac{b_c}{b_v}$ whereas for channel coding the *design rate* $R_L^{\mathsf{cc}}$ of the terminated code ensemble is equal to $R_L^{\mathsf{cc}} = 1 - \left(\frac{L+m_s}{L}\right)\frac{b_c}{b_v}$. As $L$ increases, the rate loss diminishes monotonically so that, as $L \to \infty$,

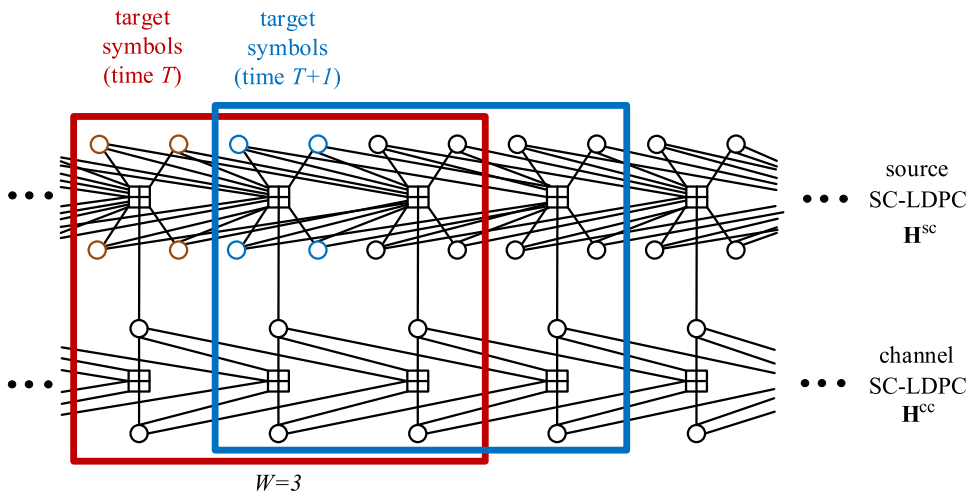


Fig. 2. Concatenated protographs of SC-LDPC codes for JSCC. Also illustrated is the WD procedure with window size $W = 3$.

$R_L^{\mathsf{sc}} \to R^{\mathsf{sc}} = b_c/b_v$ and $R_L^{\mathsf{cc}} \to R^{\mathsf{cc}} = 1 - b_c/b_v$ (the rates of the unterminated convolutional code ensembles).

### B. Concatenating SC-LDPC Graphs

Our proposed concatenated SC-LDPC construction for JSCC involves two SC-LDPC parity-check matrices, one for source compression, $\mathbf{H}^{\mathsf{sc}}$, and another for channel coding, $\mathbf{H}^{\mathsf{cc}}$, with base matrices given in (9) or (10). Notationally, we continue the convention to add "sc" or "cc" superscripts to the parameters to indicate their use in the source and channel codes, respectively, *i.e.*, we add superscripts to $\mathbf{B}_i$, $b_c$, $b_v$, $m_s$, $M$, and $L$. We note that, in order for the parameters to match, we must select $b_c^{\mathsf{sc}} M^{\mathsf{sc}} = (b_v^{\mathsf{cc}} - b_c^{\mathsf{cc}}) M^{\mathsf{cc}}$. We therefore restrict our constructions to have $M^{\mathsf{sc}} = M^{\mathsf{cc}} = M$ in this paper, implying $b_c^{\mathsf{sc}} = b_v^{\mathsf{cc}} - b_c^{\mathsf{cc}}$. We also choose to equate the memories so that $m_s^{\mathsf{sc}} = m_s^{\mathsf{cc}} = m_s$; thus, if the codes are terminated (using base matrices in (10)), we must further have $L^{\mathsf{sc}} + m_s = L^{\mathsf{cc}} - m_s$ and we denote the *overall coupling length* of the concatenated scheme as $L = L^{\mathsf{sc}}$. The *overall coding rate* for the terminated scheme is $R_L = \frac{R_L^{\mathsf{cc}}}{R_L^{\mathsf{sc}}}$ and, as $L \to \infty$, the overall coding rate approaches $R = \frac{R_L^{\mathsf{cc}}}{R^{\mathsf{sc}}} = \frac{b_v^{\mathsf{sc}}}{b_v^{\mathsf{cc}}}$, the rate of the unterminated JSCC scheme.

**Example construction:** We now provide a working example for use in this paper, but it can be easily generalized. In our example, both channel and source encoder have memory $m_s = 2$ and $M^{\mathsf{sc}} = M^{\mathsf{cc}}$. We require $L^{\mathsf{cc}} = L^{\mathsf{sc}} + 4$ and use the notation $L = L^{\mathsf{sc}}$ for the concatenated design as discussed above. We use component matrices $\mathbf{B}_0^{\mathsf{sc}} = \mathbf{B}_1^{\mathsf{sc}} = \mathbf{B}_2^{\mathsf{sc}} = [1\ 1\ 1\ 1]$ to construct $\mathbf{H}_{[0,L-1]}^{\mathsf{sc}}$, with compression rate $R_L^{\mathsf{sc}} = \left(\frac{L+m_s}{L}\right) \frac{b_c^{\mathsf{sc}}}{b_v^{\mathsf{sc}}} \xrightarrow[L\to\infty]{} \frac{b_c^{\mathsf{sc}}}{b_v^{\mathsf{sc}}} = \frac{1}{4}$ and component matrices $\mathbf{B}_0^{\mathsf{cc}} = \mathbf{B}_1^{\mathsf{cc}} = \mathbf{B}_2^{\mathsf{cc}} = [1\ 1]$ with channel code rate $R_L^{\mathsf{cc}} = 1 - \left(\frac{L+4+m_s}{L+4}\right) \frac{b_c^{\mathsf{cc}}}{b_v^{\mathsf{cc}}} \xrightarrow[L\to\infty]{} 1 - b_c^{\mathsf{cc}}/b_v^{\mathsf{cc}} = \frac{1}{2}$. The overall coding rate $R_L = \frac{R_L^{\mathsf{cc}}}{R_L^{\mathsf{sc}}} \xrightarrow[L\to\infty]{} 2$.

The protograph of the proposed construction is shown in Fig. 2. We note that the source and channel graphs are connected periodically block-by-block such that, at each time instant, the left entry of $\mathbf{B}_0^{\mathsf{cc}}$ (shown as the top variable node of the channel graph in Fig. 2) connects to the corresponding check node of $\mathbf{B}^{\mathsf{sc}}$ at that time - this connects the systematic bits of the channel code to the syndrome of the source. This periodic connectivity will allow efficient realization of the window decoder (see Section III-D), but generalizations are possible. The connection between source and channel parts is an important part of the joint code design. Moreover, in order to be able to use $\mathbf{H}^{\mathsf{cc}}$ directly for systematic encoding of the syndrome $\mathbf{u}$, we must also restrict the permutation matrix associated with the right most entry of $\mathbf{B}_0^{\mathsf{cc}}$ (the $M$ edges connected to the nodes that contain the parity bits of the codeword $\mathbf{v}$) to be replaced with the $M \times M$ identity matrix. This is required for syndrome former encoding (see Section III-C). We note that the identity matrix restriction could be achieved by simple column permutations of the $\mathbf{H}^{\mathsf{cc}}$ matrix after an arbitrary lifting.

### C. Encoding Concatenated SC-LDPC Codes

We begin this section by defining the notation required to describe the encoding process. Equation (11) shows the transposed parity check matrix obtained after graph lifting (9), called the *syndrome former matrix*

$$\mathbf{H}_{[0,\infty]}^{\intercal} = \begin{bmatrix} \ddots & & \ddots & & \\ \mathbf{H}_0^{\intercal}(0) & \cdots & \mathbf{H}_{m_s}^{\intercal}(m_s) & & \\ & \ddots & & \ddots & \\ & \mathbf{H}_0^{\intercal}(T) & \cdots & \mathbf{H}_{m_s}^{\intercal}(T+m_s) & \\ & & \ddots & & \ddots \end{bmatrix}, \quad (11)$$

where the submatrices $\mathbf{H}_i$, $i \in [0 : m_s]$, are defined as

$$\mathbf{H}_i(T) = \begin{bmatrix} h_i^{(1,1)}(T) & \cdots & h_i^{(1,b_v M)}(T) \\ \vdots & & \vdots \\ h_i^{(b_c M,1)}(T) & \cdots & h_i^{(b_c M, b_v M)}(T) \end{bmatrix}_{b_c M \times b_v M}. \quad (12)$$

*1) Step 1: Syndrome Source Coding:* Suppose an information sequence $\mathbf{s}_{[0,\infty]}$ is defined as $\mathbf{s}_{[0,\infty]} = [\mathbf{s}_0, \mathbf{s}_1, \ldots]$, where $\mathbf{s}_i$ is a source block of length $b_v^{\mathsf{sc}} M$. We obtain the

compressed syndrome $\mathbf{u}_{[0,\infty]} = [\mathbf{u}_0, \mathbf{u}_1, \ldots] = \mathbf{s}_{[0,\infty]}\mathbf{H}^{\mathsf{sc}\intercal}_{[0,\infty]}$, where $\mathbf{u}_i = \mathbf{s}_i\mathbf{H}^{\mathsf{sc}\intercal}_0(i) + \mathbf{s}_{i-1}\mathbf{H}^{\mathsf{sc}\intercal}_1(i) + \cdots + \mathbf{s}_{i-m_s}\mathbf{H}^{\mathsf{sc}\intercal}_{m_s}(i) = (u_i^{(1)}, u_i^{(2)}, \ldots, u_i^{(b_c^{\mathsf{sc}}M)})$. Note that syndrome source coding can be performed block-by-block in a streaming fashion with memory provided for the previous $m_s$ blocks.

*2) Step 2: Syndrome Former-Based Channel Encoding:* The channel encoder then encodes the compressed binary information sequence $\mathbf{u}_{[0,\infty]}$ into the binary code sequence $\mathbf{v}_{[0,\infty]} = [\mathbf{v}_0, \mathbf{v}_1, \ldots]$, where $\mathbf{v}_i = (v_i^{(1)}, v_i^{(2)}, \ldots, v_i^{(b_v^{\mathsf{cc}}M)})$. The resulting code sequence $\mathbf{v}_{[0,\infty]}$ satisfies $\mathbf{v}_{[0,\infty]}\mathbf{H}^{\mathsf{cc}\intercal}_{[0,\infty]} = 0$. By design, our $\mathbf{H}^{\mathsf{cc}}_{[0,\infty]}$ permits a *systematic* convolutional encoder of rate $R = 1 - b_v^{\mathsf{cc}}/b_c^{\mathsf{cc}}$. We follow the partial syndrome former realization of [31].

Suppose information symbol sequence $\mathbf{u}_{[0,t-1]}$ (the compressed syndrome) is systematically encoded to $\mathbf{v}$ such that for any $t > 0$, this codeword satisfies

$$\mathbf{v}_{[0,t-1]}\mathbf{H}^{\mathsf{cc}}_{[0,t-1]} = \left[\mathbf{0}_{[0,t-1]} | \mathbf{P}_t\right], \tag{13}$$

where $\mathbf{0}_{[0,t-1]}$ is the zero vector of length $b_c^{\mathsf{cc}}Mt$ and

$$\mathbf{P}_t = [\mathbf{P}_{t,1}, \mathbf{P}_{t,2}, \ldots, \mathbf{P}_{t,m_s}], \tag{14}$$

$\mathbf{P}_{t,i} = \left(P_{t,i}^{(1)}, P_{t,i}^{(2)}, \ldots, P_{t,i}^{(Mb_c^{\mathsf{cc}})}\right)$, $i \in [1 : m_{\mathsf{s}}]$. $\mathbf{P}_t$ is called the *partial syndrome*, corresponding to the state of a partial syndrome former encoder at time $t$. The partial syndrome $\mathbf{P}_t$ can be calculated recursively as a function of $\mathbf{P}_{t-1}$ and $\mathbf{v}_{t-1}$ using

$$\mathbf{P}_{t,i} = \begin{cases} \mathbf{P}_{t-1,i+1} + \mathbf{v}_{t-1}\mathbf{H}_i^{\mathsf{cc}\intercal}(t+i-1), & i = 1, 2, \\ & \ldots, m_s - 1, \\ \mathbf{v}_{t-1}\mathbf{H}_{m_s}^{\mathsf{cc}\intercal}(t+m_s-1), & i = m_s. \end{cases} \tag{15}$$

By considering $\mathbf{H}_0^{\mathsf{cc}}(t) = \left[\mathbf{H}_0^{\mathsf{cc}^{(0)}}(t), \mathbf{H}_0^{\mathsf{cc}^{(1)}}(t)\right]$, where $\mathbf{H}_0^{\mathsf{cc}^{(0)}}(t)$ is $b_c^{\mathsf{cc}}M \times (b_v^{\mathsf{cc}} - b_c^{\mathsf{cc}})M$ and $\mathbf{H}_0^{\mathsf{cc}^{(1)}}(t)$ is a $b_c^{\mathsf{cc}}M \times b_c^{\mathsf{cc}}M$ identity matrix (as described in the graph lifting discussion of Section III-B), we have

$$\mathbf{v}_t^{(1)} = \mathbf{v}_t^{(0)}\left[\mathbf{H}_0^{\mathsf{cc}^{(0)}}(t)\right]^{\intercal} + \mathbf{P}_{t,1}, \tag{16}$$

where we assume that encoder is systematic and will therefore use the notation $\mathbf{v}_t = [\mathbf{v}_t^{(0)}, \mathbf{v}_t^{(1)}]$ with $\mathbf{v}_t^{(0)} = \mathbf{u}_t$ of length $(b_v^{\mathsf{cc}} - b_c^{\mathsf{cc}})M$ and $\mathbf{v}_t^{(1)}$ is a parity-check vector of length $b_c^{\mathsf{cc}}M$.

For termination of an SC-LDPC encoder, the information sequences need to be terminated with a sequence of symbols that causes the encoder to reset to the zero state at the end of encoding. While conventional feedforward polynomial convolutional encoders use sequences of zeros as the terminating tail, SC-LDPC encoders use non-zero sequences for the terminating tail; such sequences depend on the encoded information symbols and are needed to solve a system of linear equations. Interested readers can refer to [31] for a full description of the partial syndrome former realization method to terminate the encoder.

*3) Complexity:* The syndrome former matrix of a $(m_s, J, K)$-regular SC-LDPC code, $\mathbf{H}^{\intercal}_{[0,\infty]}$, has exactly $J$ ones in every row and $K$ ones in every column starting from the $(m_s b_v^{\mathsf{cc}}M + 1)$-th column. A syndrome former encoder realization requires $b_v^{\mathsf{cc}}Mm_s + (b_v^{\mathsf{cc}} - b_c^{\mathsf{cc}})M$ memory units. For this scheme, the encoding complexity is independent of the codeword length and the syndrome former memory $m_s$ and it is proportional to $K - 1$ [31]. While the partial syndrome former encoder realization encoding complexity is the same as syndrome former, it needs only $b_c^{\mathsf{cc}}Mm_s$ memory units, which is less than that of syndrome former scheme [31]. On the other hand, a straightforward encoder for a length $N$ LDPC block code that multiplies the information bits by the generator matrix has complexity proportional to $N$ per bit.

### D. Windowed Decoding (WD) Scheme

For practical implementation of concatenated SC-LDPC codes for JSCC with large coupling length $L$, it is essential to reduce the decoding latency. To this end, we propose a joint sliding window decoder, where a window of size $W$ (containing $W$ sections of the concatenated graph) slides over the concatenated graph from left to right. This is a similar concept to the sliding window decoder for channel coding with SC-LDPC codes [32], but here the windowed scheme is applied simultaneously to the source and channel SC-LDPC graphs. At each window position, the BP algorithm described in Section II is applied to the variable and check nodes within the window (also using necessary information from past variable/check nodes) in order to decode one block of source symbols, called *target symbols*. After decoding the set of target symbols (*i.e.*, when they are all assigned 0 or 1), the window slides one section to the right and again executes the BP algorithm to decode the next set of target symbols, using both the nodes in the window and some previously decoded target symbols. Fig. 2 illustrates WD at time $T$ and $T + 1$ with window size $W = 3$ (covering 3 graph sections, or $6M$ channel code symbols and $12M$ source symbols) on the concatenated SC-LDPC codes. Here $2M$ channel code and $4M$ source symbols enter the window at each window position and $4M$ reconstructed source symbols leave (are decoded). In this paper, we refer to the *latency* of the WD scheme as the number of channel code symbols in the window, *i.e.*, how many channel symbols we need to process before we can decode a set of target source symbols.

## IV. EXIT Chart Analysis of Protograph-Based JSCC Schemes

To analyze the asymptotic behavior of the proposed schemes, we employ an EXIT chart analysis. In this section, we present an extension of the EXIT chart approach to codes defined by protographs, following [28]. We begin in Section IV-A by describing the protograph-based EXIT chart analysis for JSCC and establishing notation. We then perform an EXIT chart analysis of the LDPC block code ensembles in Section IV-B and for WD of SC-LDPC code ensembles for JSCC in Section IV-C. Finally, we compare the resulting window decoding thresholds versus those of optimized irregular LDPC codes in Section IV-D.

### A. EXIT Chart Analysis

Concatenated LDPC and SC-LDPC code ensembles for JSCC can be represented by a joint base matrix

(see, *e.g.*, [16], [17]) as follows

$$\mathbf{B}_J = \begin{bmatrix} \mathbf{B}^{\mathsf{sc}} & \mathbf{L}_1 \\ \mathbf{L}_2 & \mathbf{B}^{\mathsf{cc}} \end{bmatrix} = [b_{ij}], \tag{17}$$

where the sizes of source protograph matrix $\mathbf{B}^{\mathsf{sc}}$, channel protograph matrix $\mathbf{B}^{\mathsf{cc}}$, zero matrix $\mathbf{L}_2$, and *linking matrix* $\mathbf{L}_1$, that represents the connectivity from check nodes of the source code to variable nodes of the channel code, are $b_c^{\mathsf{sc}} \times b_v^{\mathsf{sc}}$, $b_c^{\mathsf{cc}} \times b_v^{\mathsf{cc}}$, $b_c^{\mathsf{cc}} \times b_v^{\mathsf{sc}}$, and $b_c^{\mathsf{sc}} \times b_v^{\mathsf{cc}}$, respectively. The following notation is used to denote three types of mutual information (MI) between variable nodes (VNs) and check nodes (CNs) in the rest of the section:

- $I_{\mathsf{E}}^{\mathsf{vc}}(i,j)$ denotes the extrinsic MI from the $j^{\text{th}}$ VN to the $i^{\text{th}}$ CN;
- $I_{\mathsf{E}}^{\mathsf{cv}}(i,j)$ denotes the extrinsic MI from the $i^{\text{th}}$ CN to the $j^{\text{th}}$ VN; and
- $I_{\mathsf{CMI}}^{\mathsf{vc}}(j)$ denotes the cumulative MI for the variable node $V_j$.

Also, the function $J(\sigma_{\mathsf{ch}})$ is defined to represent the MI between a binary bit and its corresponding LLR value $L_{\mathsf{ch}} \sim N(\sigma_{\mathsf{ch}}^2/2, \sigma_{\mathsf{ch}}^2)$, given as [28]

$$J(\sigma_{\mathsf{ch}}) = 1 - \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi\sigma_{\mathsf{ch}}^2}} e^{-\frac{\left(\xi - \sigma_{\mathsf{ch}}^2/2\right)^2}{2\sigma_{\mathsf{ch}}^2}} \log_2(1+e^{-\xi}) d\xi. \tag{18}$$

The proposed EXIT chart algorithm for protograph-based JSCC with a fixed $p_v$ is given as follows:

1) **Select signal-to-noise ratio (SNR)**
Set a vector $\sigma_{\mathsf{ch}} = (\sigma_{\mathsf{ch},0}, \sigma_{\mathsf{ch},1}, \ldots, \sigma_{ch,b_v^{\mathsf{cc}}})$ such that

$$\sigma_{\mathsf{ch},j}^2 = 8R^{\mathsf{cc}} \left(\frac{E_{\mathsf{b}}}{N_0}\right)\Big|_{V_j},$$

where $R^{\mathsf{cc}}$ is the channel code design rate and $E_{\mathsf{b}}/N_0\big|_{V_j}$ represents the SNR associated with the channel input to the $j$th variable node $V_j$.[2]

2) **MI update from VNs to CNs**
We define $\delta_{ab} = 1$ when $a = b$, $\delta_{ab} = 0$ when $a \neq b$, and $I_{\mathsf{E}}^{\mathsf{vc}}(i,j) = 0$ if $b_{ij} = 0$. Then

- For $i \in [1 : b_c^{\mathsf{sc}} + b_c^{\mathsf{cc}}]$ and $j \in [1 : b_v^{\mathsf{sc}}]$,

$$I_{\mathsf{E}}^{\mathsf{vc}}(i,j) = J_{BSC}\left(\sum_{c=1}^{b_c^{\mathsf{sc}}+b_c^{\mathsf{cc}}} (b_{cj} - \delta_{ci}) \left(J^{-1}(I_E^{\mathsf{cv}}(c,j))\right)^2,\ p_v\right). \tag{19}$$

The function $J_{BSC}(\cdot)$ is a manipulation of the $J(\cdot)$ function, defined as [15]

$$J_{BSC}(\mu, p_v) = (1-p_v) I(V;\chi^{(1-p_v)}) + (p_v) I(V;\chi^{(p_v)}),$$

where $I(V;\chi) = 1 - \mathbb{E}[\log_2(1+e^{\chi})]$ is the MI between the VN of the source and LLR value $\chi$, $\chi^{(1-p_v)} \sim N(\mu + L_v^{\mathsf{sc}}, 2\mu)$, $\chi^{(p_v)} \sim N(\mu - L_v^{\mathsf{sc}}, 2\mu)$, and $L_v^{\mathsf{sc}}$ is as defined in Section II.

- For $i \in [1 : b_c^{\mathsf{sc}} + b_c^{\mathsf{cc}}]$ and $j \in [b_v^{\mathsf{sc}} + 1 : b_v^{\mathsf{sc}} + b_v^{\mathsf{cc}}]$,

$$I_{\mathsf{E}}^{\mathsf{vc}}(i,j) = J\left(\sqrt{\sum_{c=1}^{b_c^{\mathsf{sc}}+b_c^{\mathsf{cc}}} (b_{cj} - \delta_{ci})(J^{-1}(I_{\mathsf{E}}^{\mathsf{cv}}(c,j)))^2 + \sigma_{\mathsf{ch}}^2(j)}\right). \tag{20}$$

3) **MI update from CNs to VNs**
For $i \in [1 : b_c^{\mathsf{sc}} + b_c^{\mathsf{cc}}]$ and $j \in [1 : b_v^{\mathsf{sc}} + b_v^{\mathsf{cc}}]$

$$I_{\mathsf{E}}^{\mathsf{cv}}(i,j) = 1 - J\left(\sqrt{\sum_{v=1}^{b_v^{\mathsf{sc}}+b_v^{\mathsf{cc}}} (b_{iv} - \delta_{iv}) \left(J^{-1}(1 - I_{\mathsf{E}}^{\mathsf{vc}}(i,v))\right)^2}\right). \tag{21}$$

4) **Cumulative MI**
For $j \in [1 : b_v^{\mathsf{sc}}]$,

$$I_{\mathsf{CMI}}(j) = J_{BSC}\left(\sqrt{\sum_{i=1}^{b_c^{\mathsf{sc}}} b_{ij} \left(J^{-1}(I_{\mathsf{E}}^{\mathsf{cv}}(i,j))\right)^2}, p_v\right). \tag{22}$$

5) **Stopping criterion**
If $I_{\mathsf{CMI}}(j) = 1$ (up to the desired precision) for all $j$, then stop; otherwise, go to Step 2.

Note that the inverse $J(\cdot)$ function can be approximated as [33]

$$J^{-1}(I) = \begin{cases} a_1 I^2 + b_1 I + c_1\sqrt{I}, & 0 \le I \le 0.3646, \\ -a_2 \log(b_2(1-I)) - c_2 I, & 0.3646 \le I \le 1, \end{cases} \tag{23}$$

where $a_1 = 1.09542$, $b_1 = 0.214217$, $c_1 = 2.33727$, $a_2 = 0.70692$, $b_2 = 0.386013$, and $c_2 = -1.75017$. We define the minimum SNR for which $I_{\mathsf{CMI}}(j) = 1$, $j \in [1 : b_v^{\mathsf{sc}}]$ (or for $j \in$ position of the target source symbols in the case of the WD scheme for SC-LDPC code ensemble) as the *joint threshold* for the concatenated JSCC code ensemble. We remark that the algorithm can also be run for a fixed SNR (dB) to determine the maximum source probability $p_v^{\mathsf{SNR}}$ that allows convergence. We define the maximum probability $p_v^*$ for which (22) is satisfied for the (disconnected) source code protograph as the *source threshold*. This is obtained in the above algorithm by setting the matrix $\mathbf{L}_1$ to zero. As conventionally done, we also define the minimum SNR for which the cumulative MI information converges to 1 in the (disconnected) channel code protograph as the *channel threshold* [28], [34]. This can be obtained from the above algorithm by setting the matrix $\mathbf{L}_1$ to zero and replacing Step 4 with

$$I_{\mathsf{CMI}}(j) = J\left(\sqrt{\sum_{c=b_c^{\mathsf{sc}}+1}^{b_c^{\mathsf{sc}}+b_c^{\mathsf{cc}}} b_{cj} \left(J^{-1}(I_{\mathsf{E}}^{\mathsf{cv}}(c,j))\right)^2 + \sigma_{\mathsf{ch}}^2(j)}\right), \tag{24}$$

for $j \in [b_v^{\mathsf{sc}} + 1 : b_v^{\mathsf{sc}} + b_v^{\mathsf{cc}}]$.

[2]Note that $E_b/N_0\big|_{V_j} = 0$ if $V_j$ is punctured.

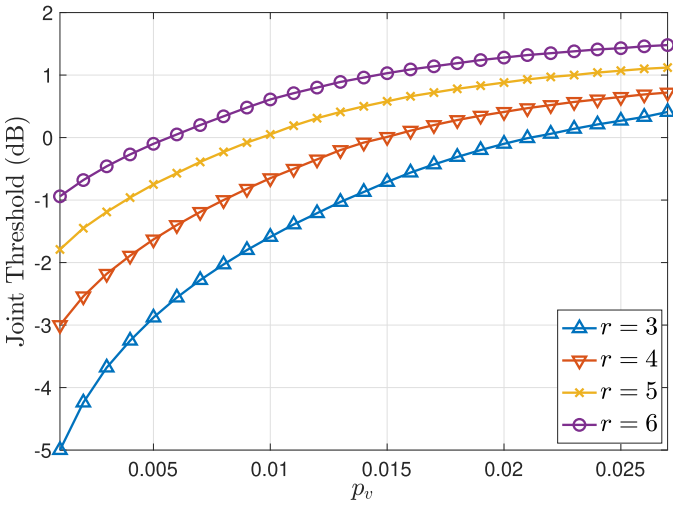


Fig. 3. Thresholds of concatenated $(3, 12)$-regular source and $(r, 2r)$-regular LDPC block code for a given source bit probability $p_v$.

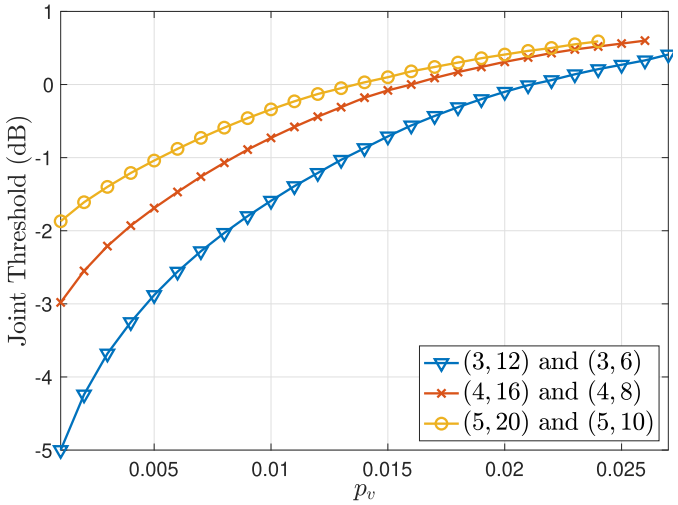


Fig. 4. Joint threshold of concatenated block LDPC code ensembles with respect to source bit probability $p_v$. Labels correspond to source and block degrees, respectively.

### *B. Analysis of Concatenated LDPC Block Codes for JSCC*

In this subsection, we present an EXIT chart analysis for JSCC with the protograph-based concatenated LDPC block code ensemble shown in Fig. 1. Fig. 3 shows the results for several pairs of $(3, 12)$-regular source and $(r, 2r)$-regular channel LDPC block code ensembles, with source and channel design rates of $R^{\text{sc}} = 0.25$ and $R^{\text{cc}} = 0.5$, respectively, giving an overall design code rate of $R = 2$. Here, the joint base matrix $\mathbf{B}_J$, *i.e.*, equation (17), is constructed as

$$\mathbf{B}_J = \begin{bmatrix} 3 & 3 & 3 & 3 & 1 & 0 \\ 0 & 0 & 0 & 0 & r & r \end{bmatrix}. \tag{25}$$

We compute the source threshold for this $R^{\text{sc}} = 0.25$ $(3, 12)$-regular ensemble as $p_v^* = 0.027$, where the source entropy is $0.1791$. It is well-known the threshold value increases (worsens) by increasing the density of regular channel block code ensembles for a fixed code, and this phenomena similarly effects the performance of JSCC ensembles. This is seen in Fig. 3, where the source code graph density is fixed but the channel code graph density increases with $r$. This behavior is due to suboptimality of BP, since increasing $r$ improves the channel code distance properties and has improving ML decoding performance. As we will see in Section IV-C, this drawback for LDPC code ensembles for increasing channel density $r$ is avoided by using SC-LDPC codes, for which threshold saturation effect ensures *improving* threshold for increasing $r$, thereby allowing much more flexibility in the JSCC design.

Finally, Fig. 4 shows the results obtained for the case where the source and channel graph densities are increased simultaneously with joint base matrix

$$\mathbf{B}_J = \begin{bmatrix} r & r & r & r & 1 & 0 \\ 0 & 0 & 0 & 0 & r & r \end{bmatrix}. \tag{26}$$

As expected, similar behavior is observed, with degrading thresholds for increasing $r$, although the increasing source graph density is seen to reduce the gaps between the curves.

### *C. Analysis of Concatenated SC-LDPC Codes With WD for JSCC*

In this section, the protograph EXIT chart analysis is applied to concatenated SC-LDPC codes for JSCC with the WD scheme described in Section III-D for codes with rates $R^{\text{sc}} = 0.25$ and $R^{\text{cc}} = 0.5$ (source and channel parts respectively). The joint base matrix $\mathbf{B}_J$ is constructed from the convolutional source/channel protographs $\mathbf{B}^{\text{sc}}_{[0,L]}$ and $\mathbf{B}^{\text{cc}}_{[0,L]}$ of appropriate lengths given in Section III-B with the linking matrix

$$\mathbf{L}_1 = \begin{bmatrix} \mathbf{C} & & & \\ & \mathbf{C} & & \\ & & \ddots & \\ & & & \mathbf{C} \end{bmatrix}, \tag{27}$$

where $\mathbf{C}_{1\times 2} = [0 \;\; 1]$ and all other elements are zeros.[3] Although it is possible to view a terminated SC-LDPC code as a block protograph and apply conventional JSCC EXIT chart techniques, it is more appropriate to modify the approach to consider the window, both from a numerical implementation point-of-view as well as to obtain practically relevant results. In order to do this, it is sufficient to examine the EXIT chart for a single window. For example, (28), shown at the bottom of the next page, shows the joint base matrix for a window with $W = 3$ for the joint convolutional protograph shown in Fig. 2.

We note that this base graph includes variable nodes with reduced degree corresponding to nodes with edge connections that go outside the window (connecting to future nodes), and as such the protograph must be designed to ensure that a threshold exists for a window. The *stopping criterion* of the EXIT chart algorithm must also be adapted so that it is used only for the *target bits* of the *first window* (e.g., the first four columns of (28)) and the algorithm is terminated when the EXIT values of the target bits reach the value 1. After

[3]This linking matrix corresponds to the periodic connections between source and channel parts in our specific examples where $b_c^{\text{cc}} = 1$ and $b_v^{\text{cc}} = 2$, as described in Section III-B, generalizations are possible.

TABLE I

DECODING THRESHOLD VALUES OF CONCATENATED LDPC BLOCK CODES FOR JSCC

| $p_v$ | Threshold (dB) |
|---|---|
| 0.01 | -1.61 |
| 0.02 | -0.10 |
| 0.03 | 2.49 |
| 0.04 | 3.39 |

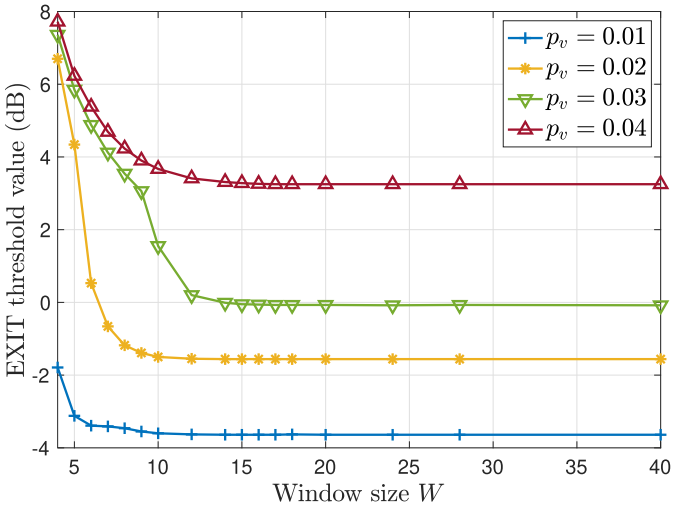


Fig. 5. Decoding threshold values for joint WD of concatenated $(3, 12)$- and $(3, 6)$-regular SC-LDPC Tanner graphs for JSCC.

obtaining a threshold for a given window size, an inductive argument similar to that presented for the channel coding problem in [23] can be applied for future window shifts. This holds since the decoded target bits met the stopping criterion and imply the target bits in the next window position will also converge under the same conditions.

Fig. 5 shows the decoding threshold values for the $(3, 12)$- and $(3, 6)$-regular SC-LDPC code construction discussed above with different source probability $p_v = 0.01, 0.02, 0.03$, and $0.04$ for various window sizes $W$. It is observed that the threshold decreases by increasing $W$ until roughly $W = 15$, where the threshold saturates and does not improve further with $W$. It is also observed that the threshold value is lower for smaller source probability $p_v = \mathbb{P}(s_v = 1)$ and the threshold increases (worsens) by increasing the source probability. We also remark that the thresholds obtained are significantly improved when compared to the underlying block code ensembles from Section IV-B, summarized in Table I. For example, the joint threshold for concatenated SC-LDPC codes with sufficiently large $W$ is $-1.56$ dB, but for block LDPC codes, it is $-0.10$ dB for $p_v = 0.02$.

TABLE II

CHANNEL EXIT THRESHOLDS (DB) FOR DIFFERENT SC-LDPC CODE ENSEMBLES WITH $R^{cc} = 0.5$ USING WD WITH VARIOUS WINDOW SIZES $W$

| Code | $W = 4$ | $W = 5$ | $W = 6$ | $W = 10$ | $W = 15$ | $W = 30$ |
|---|---|---|---|---|---|---|
| $(3, 6)$ | 4.18 | 2.91 | 1.98 | 0.58 | 0.46 | 0.46 |
| $(4, 8)$ | 3.82 | 2.41 | 1.41 | 0.28 | 0.24 | 0.24 |
| $(5, 10)$ | 3.82 | 2.28 | 1.27 | 0.22 | 0.19 | 0.19 |

TABLE III

SOURCE EXIT THRESHOLD ($p_v^*$) FOR DIFFERENT SC-LDPC CODE ENSEMBLES WITH $R^{sc} = 0.25$ USING WD WITH VARIOUS WINDOW SIZES $W$

| Code | $W = 4$ | $W = 5$ | $W = 6$ | $W = 10$ | $W = 15$ | $W = 30$ |
|---|---|---|---|---|---|---|
| $(3, 12)$ | 0.002 | 0.006 | 0.013 | 0.028 | 0.031 | 0.031 |
| $(4, 16)$ | 0.002 | 0.008 | 0.017 | 0.033 | 0.035 | 0.035 |
| $(5, 20)$ | 0.002 | 0.008 | 0.018 | 0.034 | 0.036 | 0.036 |

We also consider higher density graphs with component matrices $\mathbf{B}_0^{cc} = \mathbf{B}_1^{cc} = \mathbf{B}_2^{cc} = \mathbf{B}_3^{cc} = [1\ 1]$ and $\mathbf{B}_0^{cc} = \mathbf{B}_1^{cc} = \mathbf{B}_2^{cc} = \mathbf{B}_3^{cc} = \mathbf{B}_4^{cc} = [1\ 1]$ used to construct $(4, 8)$- and $(5, 10)$-regular SC-LDPC code ensembles, respectively, with rate $R^{cc} = 0.5$ as well as $(4, 16)$- and $(5, 20)$-regular source code ensembles constructed from component matrices $\mathbf{B}_0^{sc} = \mathbf{B}_1^{sc} = \mathbf{B}_2^{sc} = \mathbf{B}_3^{sc} = [1\ 1\ 1\ 1]$ and $\mathbf{B}_0^{sc} = \mathbf{B}_1^{sc} = \mathbf{B}_2^{sc} = \mathbf{B}_3^{sc} = \mathbf{B}_4^{sc} = [1\ 1\ 1\ 1]$, respectively, with rate $R^{sc} = 0.25$ (codes are constructed from these component matrices as described in Section III-A). Tables II and III show the obtained thresholds for the three JSCC code ensembles with varying densities and different window sizes $W$. These tables show that (1) by increasing code density the channel/source threshold improves for a sufficiently large, fixed window size $W$ and (2) the thresholds improve up to a certain value with increasing $W$, after which no further advantage is seen by increasing $W$.

### *D. Comparison With Optimized Irregular LDPC Codes*

Table IV compares the EXIT chart WD thresholds of $(3, 12)$- and $(3, 6)$-regular concatenated SC-LDPC codes with $W = 20$ versus code ensembles given in [16] with the same rates $R^{sc} = 0.25$ and $R^{cc} = 0.5$ for $p_v = 0.01, 0.015$, and $0.02$. We observe a significant threshold improvement for the concatenated SC-LDPC code ensemble, which could be further improved by increasing the graph density for a fixed rate. Simulation results comparing these ensembles are provided in Section V-B, confirming the expected finite-length performance improvement. In Table V, we compare the EXIT chart thresholds of WD with $W = 20$ for some concatenated SC-LDPC code ensembles versus optimized irregular code ensembles with rates $R^{sc} = 0.5$ and $R^{cc} = 0.5$ taken from [18]. Again, SC-LDPC code ensembles are shown to

$$\mathbf{B}_J = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{28}$$

TABLE IV

EXIT THRESHOLDS FOR DIFFERENT CODE ENSEMBLES WITH $R^{sc} = 0.25$ AND $R^{cc} = 0.5$

| Code Ensemble | $p_v = 0.01$ | $p_v = 0.015$ | $p_v = 0.02$ |
|---|---|---|---|
| $\mathbf{B}_{\mathsf{AR4JA}}$ [16] | -2.524 dB | -1.450 dB | -0.632 dB |
| $\mathbf{B}_{\mathsf{M1}}$ [16] | -3.015 dB | -1.797 dB | -0.931 dB |
| $\mathbf{B}_{\mathsf{M2}}(\mathbf{B}_{\mathsf{IARA}-1})$ [16] | -3.145 dB | -1.984 dB | -1.155 dB |
| $\mathbf{B}_{\mathsf{AR3A}}$ [16] | -3.248 dB | -1.910 dB | -0.965 dB |
| $\mathbf{B}_{\mathsf{IARA}-2}$ [16] | -3.438 dB | -2.254 dB | -1.379 dB |
| (3,12)- and (3,6)-regular SC-LDPC | -3.645 dB | -2.463 dB | -1.569 dB |

TABLE V

EXIT THRESHOLDS FOR DIFFERENT CODE ENSEMBLES WITH $R^{sc} = 0.50$ AND $R^{cc} = 0.50$

| Code Ensemble | $p_v = 0.04$ | $p_v = 0.06$ |
|---|---|---|
| (R4JA, AR4JA) [18] | -0.60 dB | -0.05 dB |
| (R4JA, AR3A) [18] | -0.80 dB | -0.15 dB |
| (Regular LDPC, Regular LDPC) [18] | -0.84 dB | 0.02 dB |
| Optimized code pairs and threshold [18] | $(\mathbf{B}_{s1}, \mathbf{B}_{c1})$ -2.10 dB | $(\mathbf{B}_{s2}, \mathbf{B}_{c1})$ -0.80 dB |
| (3,6)- and (3,6)-regular SC-LDPC | -2.54 dB | -1.34 dB |
| (4,8)- and (3,6)-regular SC-LDPC | -2.86 dB | -1.62 dB |
| (5,10)- and (3,6)-regular SC-LDPC | -3.05 dB | -1.75 dB |
| Shannon limit [18] | -4.02 dB | -2.36 dB |

have improved thresholds, indicating superior performance in the large code length regime.

## V. FINITE-LENGTH PERFORMANCE OF SC-LDPC CODES FOR JSCC

In this section, the numerical results of various computer experiments of a C++-based implementation of joint decoding of SC-LDPC codes are reported.

### A. Performance of Concatenated SC-LDPC Codes With WD

In this section, we consider WD of unterminated concatenated $(3, 12)$- and $(3, 6)$-regular SC-LDPC codes for JSCC and demonstrate their behavior for various parameter sets of practical interest. Simulation results were obtained for a binary input AWGN channel and where the source symbols are assumed to be i.i.d. with $\mathbb{P}(s_v = 1) = p_v = 0.02$ and a fixed number of $I = 30$ iterations per window position.[4] The coupling lengths are $L^{sc} = 26$ and $L^{cc} = 30$ unless stated otherwise.

*1) Effect of Increasing the Lifting Factor $M$:* Fig. 6(a) shows the effect of increasing the lifting factor $M$ (improving code strength) with a fixed window size $W = 15$.[5] The resulting latency is $2MW = 30M$. We observe improving performance as $M$ is increased through $M = 80, 120, 160, 200,$ and 240, as expected. The flat error floors occur due to the residual BER of the source, as described in [15]. Indeed, no such floor is observed in BER plots for the channel coding part only.

[4]Stopping rules could be included to reduce the number of iterations performed in many cases. $I = 30$ was chosen to be representative of general WD performance. Slight improvements can be observed by increasing $I$.

[5]A similar figure showing results for $W = 10$ can be found in [27].

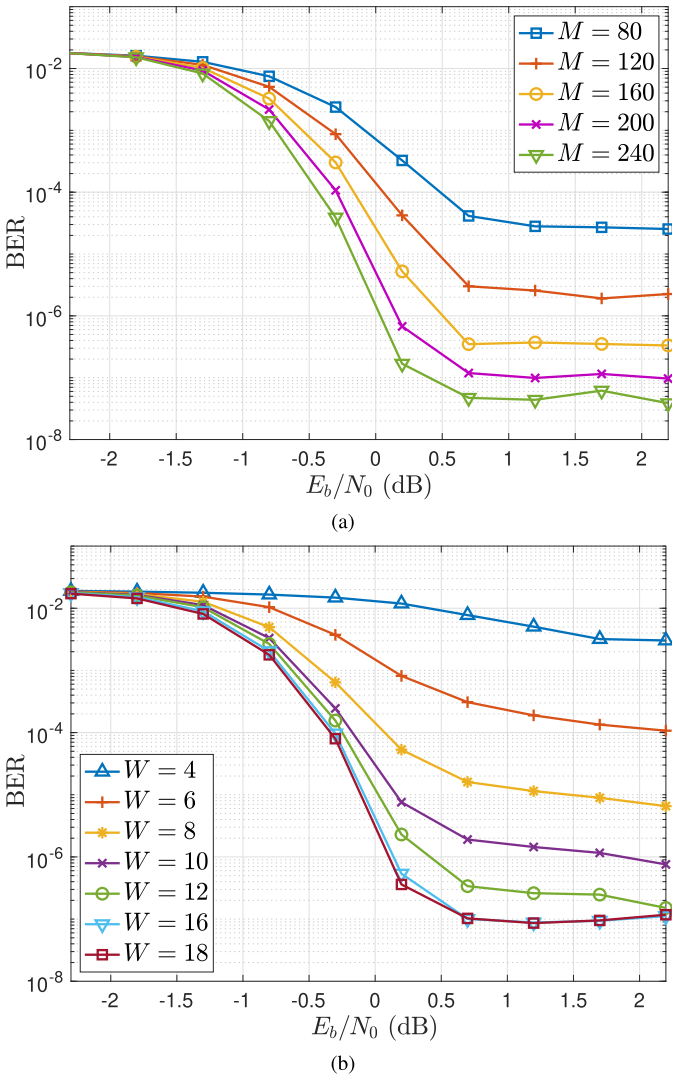


Fig. 6. JSCC performance of SC-LDPC codes with (a) increasing $M$ and fixed $W = 15$, and (b) increasing $W$ and fixed $M = 200$.

*2) Effect of the Window Size $W$:* Fig. 6(b) shows the JSCC performance with increasing window size $W = 4, 6, 8, 10,$ and 12 (improving decoder strength) but fixed $M = 200$; recall that the decoding latency is equal to $2MW = 400W$. For a fixed code strength, we observe again that the BER improves with increasing latency since the decoder performance is improving; however, we see that after a certain point, the improvement diminishes as $W$ is further increased. Our results indicate that, for a fixed latency, one has to carefully consider the trade-off between $M$ and $W$.[6]

*3) Effect of Check Node Degree:* Here, we consider a fixed $(3, 6)$-regular SC-LDPC channel code with $R^{cc} = 0.5$ and $m_c = 2$ with three source codes of rate $R^{sc} = 0.25$ and increasing check node degrees: $(3, 12)$-regular with $m_s = 2$, $(4, 16)$-regular with $m_s = 3$ and $(5, 20)$-regular with $m_s = 4$. The BER of WD for different window lengths is shown in

[6]We note that the performance is shown to continue to improve beyond $W = 10$, up to about $W = 16$, whereas the EXIT chart results from Section IV-C did not indicate such an improvement. We conjecture that this discrepancy is related to the sub-optimality of the finite length code realization and would resolve for larger $M$ (see also Fig.6(b)).

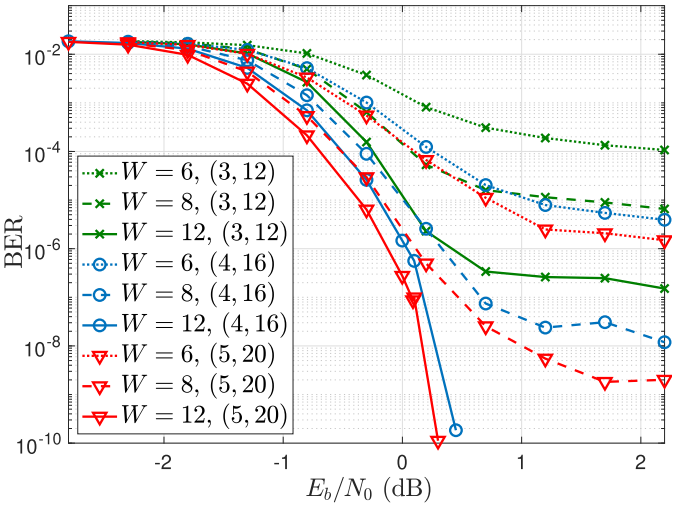


Fig. 7. WD with source code check node degree of 12, 16, and 20 for a concatenated LDPC codes with $I = 30$ and $M = 200$.

Fig. 7. The figure shows that, as expected from the asymptotic analysis in Section IV-C, the performance significantly improves in both the waterfall and error floor regions.

*4) Other Considerations:* There are several features of the scheme that can be improved, such as including stopping rules for BP to reduce complexity and designing good convolutional protographs that permit shorter window size. These features, along with comparisons to other decoding algorithms, are the subject of further work.

### *B. Comparison With Concatenated LDPC Block Codes*

In this section, we present WD results with code and decoder parameters chosen such that we obtain latencies equal to 1600 and 3200 bits.[7] We then compare with the concatenated block LDPC code designs for JSCC given in [15]. To model the codes from [15], we used a regular protograph construction with $\mathbf{B}^{\mathsf{sc}} = \mathbf{1}_{3\times 12}$ and $\mathbf{B}^{\mathsf{cc}} = \mathbf{1}_{3\times 6}$, connected as described in Section II. The overall coding rate is $R = 2$, the same as our unterminated construction, and the regularity (edge complexity) of parity-check matrices is the same. We considered block decoders with both $I = 30$ and $I = 100$ iterations.

Fig. 8(a) shows the results obtained for latency 1600 bits. We observe that for window size $W = 4$ and $M = 200$, the block code scheme outperforms the WD scheme due to the window size limiting the performance. For $W = 6$, the performance is similar to the LDPC block code in the waterfall, but the small window results in a higher error floor. As $W$ increases to 10, the SC-LDPC code outperforms the LDPC block code for all $E_{\mathsf{b}}/N_0$. Fig. 8(b) compares results of the LDPC block code scheme of length 3200 bits with WD results with latency 3200 bits. In this case, each of the parameter sets chosen for the SC-LDPC codes outperform the

[7]We note that, although we equate latency in the sense that both the block and window decoders need to receive the same number of bits to begin decoding (a block and a complete window, respectively), after initialization, the window decoder will produce partial outputs corresponding to each block of $2M$ bits at each time unit whereas the block decoder will produce estimates less often but in larger increments according to the block length.

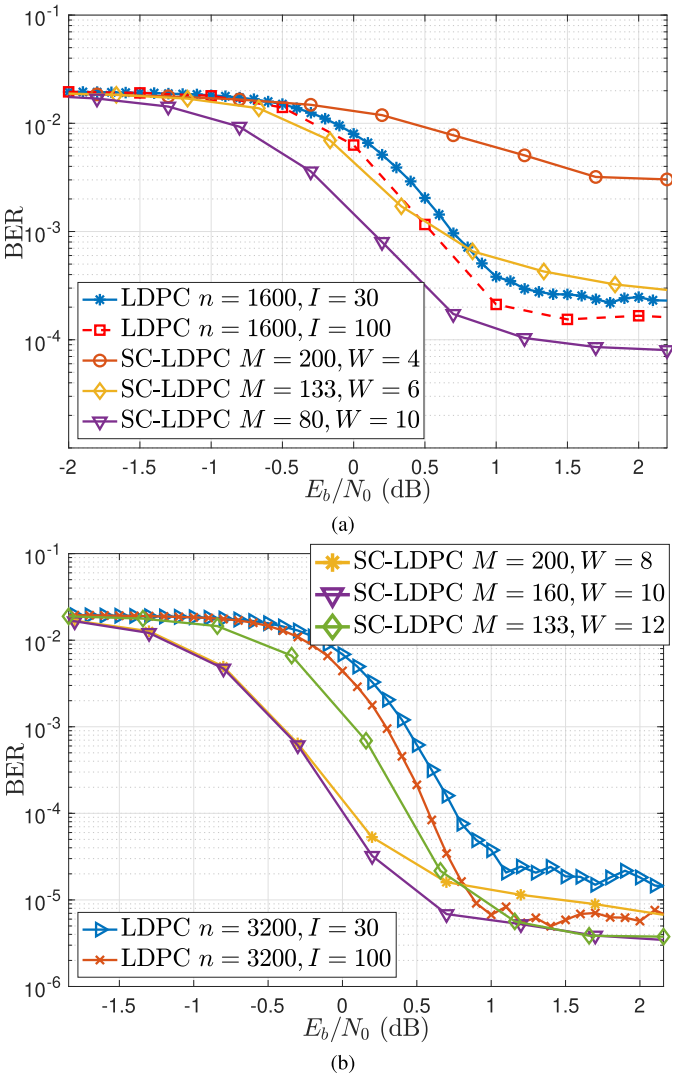


Fig. 8. Equal latency comparison of concatenated SC-LDPC codes vs. LDPC block codes: (a) 1600 bits and (b) 3200 bits.

LDPC block codes in the waterfall, with similar error floor performance.[8] We remark that for larger latencies, where $W$ can be chosen sufficiently large to not limit the performance, SC-LDPC codes hold significant promise for JSCC.

In Fig. 9, we compare the simulated performance of regular SC-LDPC codes versus optimized IARA-1 and IARA-2 codes with block length 3200 bits taken from [16]. Our SC-LDPC code results are obtained using $W = 8$, $I = 30$, and $M = 200$, giving a latency of 3200. For these codes with source rate $R^{\mathsf{sc}} = 0.25$ and channel rate $R^{\mathsf{cc}} = 0.5$, we fix the channel code to be $(3, 6)$-regular and compare three different source graphs that are $(3, 12)$-, $(4, 16)$-, and $(5, 20)$-regular for a source probability $p_v = 0.02$. We observe that, although the waterfall is competitive with the IARA codes in all cases, the

[8]In this section, we have compared on the basis of equal latency. The complexity, measured as number of node updates, is not equal. For the window decoder, the first block will receive $I$ iterations, the next $2I$, and so on up until blocks $W$ and beyond which will receive $WI$ iterations in total. On the other hand, we note that simulation results performed for the LDPC block codes in Fig. 8 where $I$ was increased in order to equate both complexity as well as latency (omitted for figure clarity) do not show significant improvement beyond those shown for $I = 100$.

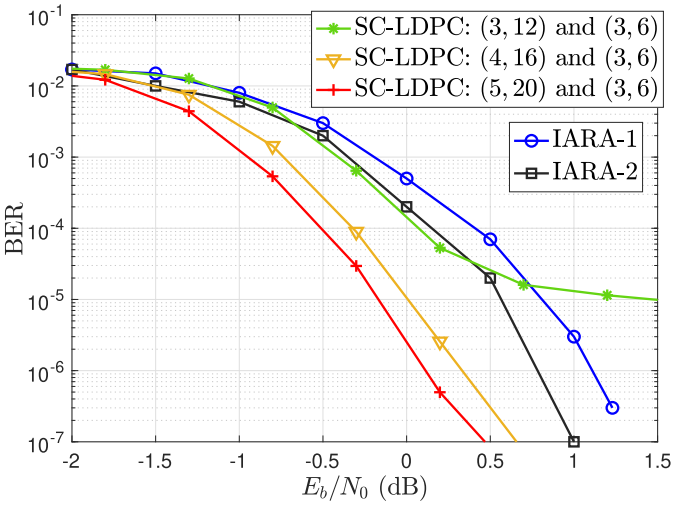


Fig. 9. Simulated decoding performance of concatenated SC-LDPC codes with $W = 8$ and $M = 200$ versus two optimized irregular LDPC codes IARA-1 and IARA-2 from [16].

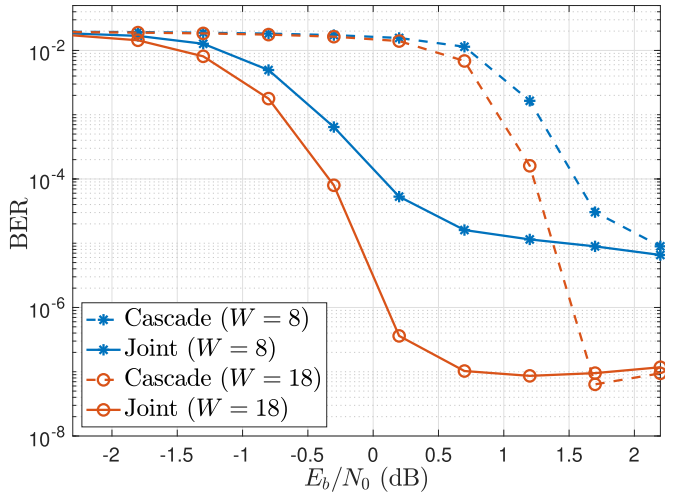


Fig. 10. BER versus $E_b/N_0$ for the joint and cascade WD scheme of the concatenated SC-LDPC codes.

smaller lifting factor in conjunction with the (3,12)-regular source graph results in an error floor for the $(3, 12)$- and $(3, 6)$-regular pair. This is eliminated in the simulated range, for example, by increasing the density of the concatenated SC-LDPC codes.

### C. Comparison With Cascade Decoding

Fig. 10 compares the joint WD scheme versus a separate "cascade" decoder for the source $(3, 12)$-regular and $(3, 6)$-regular channel codes constructed in Section V-A. Assuming that the compressed bits are equally likely and i.i.d., the cascade decoder first decodes the channel code, then decodes the source bits using information obtained by the channel decoder. Results are shown for two different window lengths $W = 8$ and 18. Both the joint decoder (solid curves) and the channel cascade decoder (dashed lines) were allowed $I = 30$ iterations. We observe that the joint decoder offers significant improvement in waterfall region with approximately equal error floors.

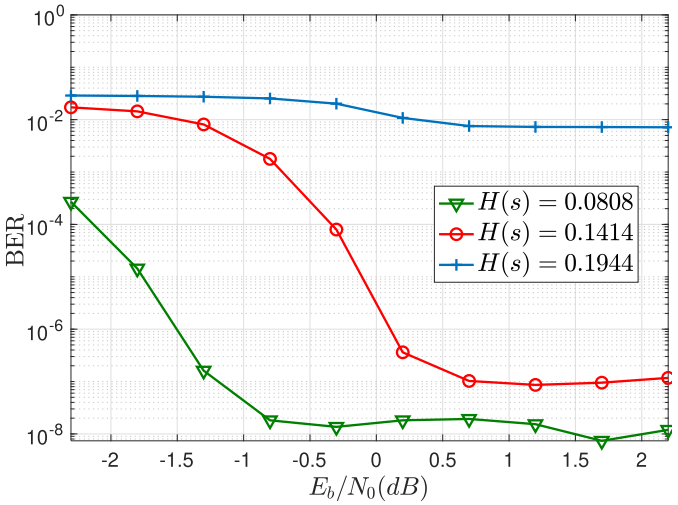


Fig. 11. JSCC performance of concatenated $(3, 12)$- and $(3, 6)$-regular SC-LDPC codes with increasing entropy.

### D. Increasing Source Entropy

Fig. 11 shows the BER of WD of terminated concatenated $(3, 12)$- and $(3, 6)$-regular concatenated SC-LDPC codes with respective, coupling lengths $L^{sc} = 26$ and $L^{cc} = 30$, code rates $R^{sc} = 0.2692$ and $R^{cc} = 0.4665$, window size $W = 18$, lifting factor $M = 200$, and $I = 30$ iterations of the joint BP decoder. Results are obtained for three different i.i.d. sources with $p_v = 0.01$, 0.02, and 0.03, and corresponding source entropy $H(s) = 0.0808$, 0.1414, and 0.1944. All of these source entropies satisfy Shannon's source coding condition $R > H(s)$, but we observe significant BER degradation from $10^{-8}$ to $10^{-2}$ with increasing source entropy. This can be alleviated by either increasing the code length (lifting factor) or by increasing the source coding rate, as seen below.

*1) Increasing Lifting Factor:* Fig. 12 plots the BER for the $(3, 12)$- and $(3, 6)$-regular concatenated SC-LDPC codes for different lifting factor $M = 200$, 1000, and 2000. Source symbols are assumed to be i.i.d. with $p_v = 0.03$ and the code decoded with a WD scheme with size $W = 18$ and $I = 30$ iterations of BP. As expected, by increasing the lifting factor $M$ from 200 to 2000, the performance improves significantly in the waterfall and error floor region at the cost of increased latency.

*2) Increasing Source Coding Rate:* Here, we fixed the lifting factor to be relatively small, $M = 200$, and obtained two new source code with higher rates with respect to previous code, *i.e.*, $(3, 12)$-regular source SC-LDPC code by constructing $(3, 9)$- and $(3, 6)$-regular source SC-LDPC code ensembles with $m_s = 2$ using the protograph construction method described III-A from the block base matrix $\mathbf{B} = [3\ 3\ 3]$ and $\mathbf{B} = [3\ 3]$ by defining the component base matrices $\mathbf{B}_0 = [1\ \ 1\ \ 1] = \mathbf{B}_1 = \mathbf{B}_2$, and $\mathbf{B}_0 = [1\ \ 1] = \mathbf{B}_1 = \mathbf{B}_2$, respectively. Fig. 13 compares the obtained WD results with $W = 18$ and $I = 30$ for i.i.d. source symbols with entropy $H(p_{sv} = 0.03) = 0.1944$. Note that all these three codes use the same channel code with rate $R^{cc} = 0.4705$ and coupling lengths $L^{cc} = 30$. The constructed source codes for $(3, 12)$-, $(3, 9)$-, and $(3, 6)$-regular SC-LDPC code have

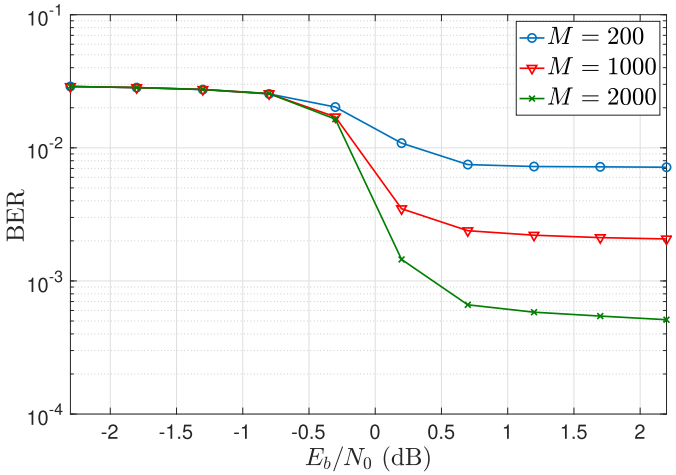


Fig. 12. BER of concatenated $(3, 12)$- and $(3, 6)$-regular SC-LDPC codes with different lifting factors and a fixed source with $p_v = 0.03$.

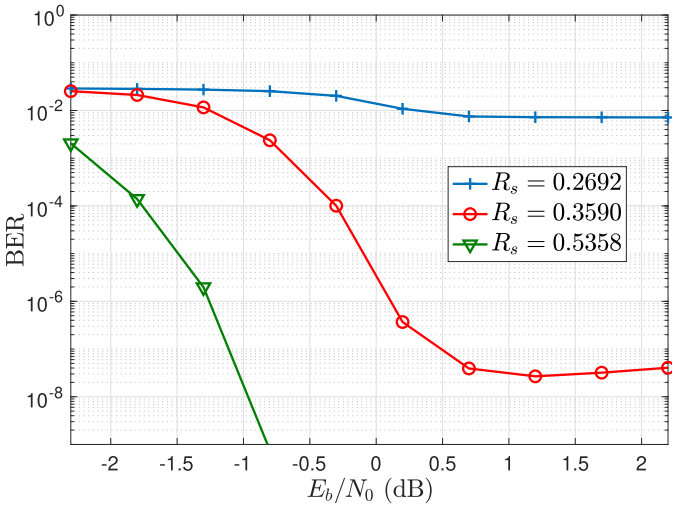


Fig. 13. Performance of concatenated SC-LDPC codes with WD of size $W = 18$, $M = 200$ and $I = 30$. Results shown for $(3, 6)$-regular channel codes and $(3, 12)$-, $(3, 9)$- and $(3, 6)$-regular source codes (top to bottom).

rates $R^{\text{sc}} = 0.2692$, $0.3590$, and $0.5385$, respectively. Again, as expected, the performance improves significantly in the waterfall and error floor regions by fixing the lifting factor and channel code rate at the cost of increasing the source rate.

### *E. Hidden Markov Model (HMM) Source*

Following [19], we also consider concatenated SC-LDPC codes for JSCC with a two-state Markovian source, referred to as the Hidden Markov Model (HMM) source. Fig. 14 shows the source diagram, where states $s_0$ and $s_1$ have transition probabilities $t_{ij}$ from state $i$ to state $j$, *i.e.*, $t_{ij} = P(s_j|s_i)$, where $i \in \{0, 1\}$. Also, we denote the probability of getting binary output $v$ in the state $s_j$ as $p_{jv} = P(v|s_j)$. Fig. 15 shows the results of the HMM source with parameters $[t_{00} = 0.02, t_{11} = 0.02, p_{00} = 0.98, p_{11} = 0.02]$ decoded with WD of concatenated $(3, 12)$- and $(3, 6)$-regular SC-LDPC codes with different window lengths $W = 8, 9, 10, 11$, and 12, but fixed $M = 200$. Similar behavior to the memoryless Bernoulli source is observed where we see that the BER improves with increasing latency.

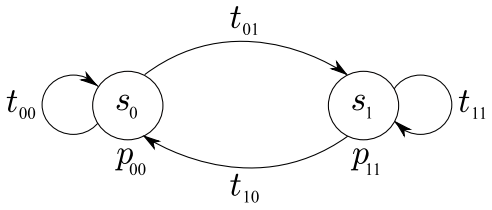


Fig. 14. Two-state Markovian source.

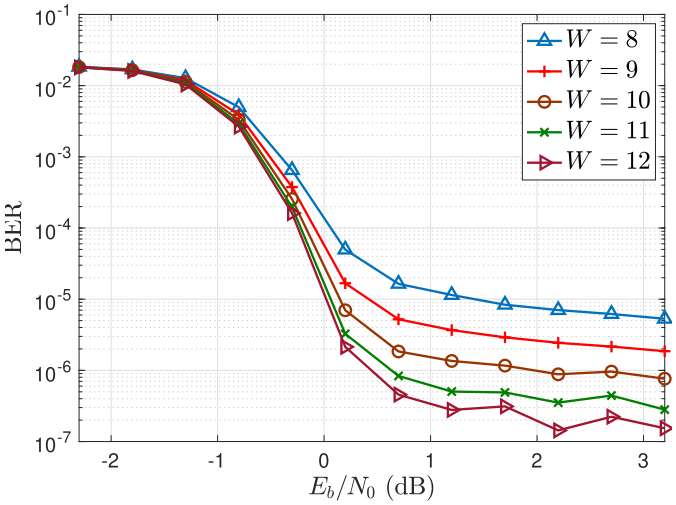


Fig. 15. JSCC performance of concatenated $(3, 6)$- and $(3, 12)$-regular SC-LDPC codes with increasing $W$ for a HMM source.

In Fig. 16, we plot the simulated decoding results of concatenated SC-LDPC codes with $(3, 12)$-, $(4, 16)$-, and $(5, 20)$-regular source graphs with $R^{\text{sc}} = 0.25$ and a fixed $(3, 6)$-regular channel graph with $R^{\text{cc}} = 0.5$, giving an overall rate $R = 2$. We selected $M = 200$ and $W = 8$, giving a latency of 3200 bits and set $I = 30$. We also show the result of the irregular LDPC code with length 3200 bits and source "Src1" from [19], where the accumulate-repeat-by-3-accumulate (AR3A) channel code with $R^{\text{cc}} = 0.5$ is concatenated with a repeat-by-4-jagged-accumulate (R4JA) source code with $R^{\text{sc}} = 0.25$. We observe that the SC-LDPC codes show improved performance in the waterfall and similar or better performance in the error floor for all of the considered parameter sets.

### *F. Punctured Channel Codes*

In a rate-compatible puncturing scheme [35], the transmitter punctures coded symbols by removing a set of $q$ columns from $n$ columns of its generator matrix which reduces the codeword length from $n$ to $n - q$ and, as a result of having fewer transmitted code symbols, the code rate is increased with *puncturing factor* $\alpha = q/n$ [36]. The resulting transmission rate is

$$R^{\text{cc}}(\alpha) = \frac{R^{\text{cc}}}{1 - \alpha}, \quad \alpha \in [0, 1). \tag{29}$$

The rate $R^{\text{cc}}(\alpha)$ is achievable if distinct codewords differ in unpunctured symbols [36], for example, by restricting punctured symbols to the parity-check symbols of a systematic code. Puncturing can be done randomly or according to a

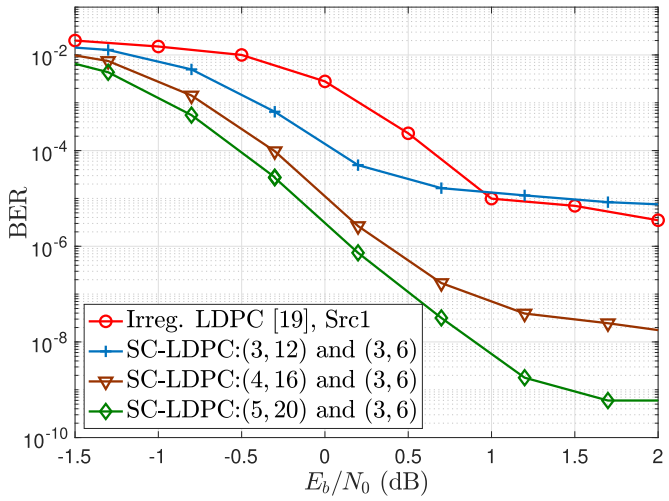


Fig. 16. Simulated decoding performance of concatenated SC-LDPC codes with $W = 8$ and $M = 200$ and a HMM source versus "Src1" from [19].

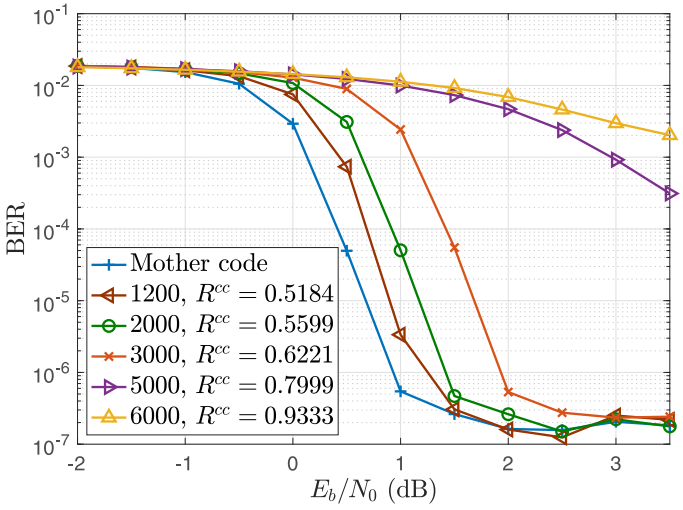


Fig. 17. Random puncturing of rate-compatible concatenated SC-LDPC codes for JSCC with different numbers of punctured bits.

particular pattern and it is assumed that the receiver knows the positions of the punctured symbols, so that both the punctured and transmitted symbols can be estimated during decoding. Since the decoder for the mother code is used to decode the punctured codes, a variety of code rates can be achieved using the same decoding architecture by puncturing different numbers of symbols. For the given $(3, 6)$-regular channel code in Section V-D, we consider punctured concatenated SC-LDPC codes, where we punctured 1200, 2000, 3000, 5000, and 6000 bits of the channel code (which are selected randomly and uniformly over the non-systematic bits). This puncturing causes the rate of the channel code to change from the non-punctured code rate $R^{\mathsf{cc}} = 0.4665$ to $R^{\mathsf{cc}} = 0.5184, 0.5599, 0.6221, 0.7999$, and $0.9333$, respectively.

The BER performance of randomly punctured concatenated SC-LDPC codes transmitted over the BI-AWGNC was also investigated via computer simulation and the results are shown in Fig. 17. As we see, by increasing the number of punctured bits, we see robust performance for moderate puncturing, but the performance degrades rapidly as puncturing increases further (as seen in [36] for non-concatenated SC-LDPC codes).

TABLE VI

COMMONLY USED CODE PARAMETERS. SUPERSCRIPTS "sc" AND "cc" DENOTE THAT THE PARAMETER CORRESPONDS TO THE SOURCE AND CHANNEL CODE, RESPECTIVELY

| Parameter | Definition |
|---|---|
| $\mathbf{s} = [s_0, s_1, \ldots, s_n]$ | Source sequence |
| $\mathbf{u} = [u_0, u_1, \ldots, u_l]$ | Syndrome sequence (compressed source) |
| $\mathbf{v} = [v_0, v_1, \ldots, v_m]$ | Channel codeword |
| $\mathbf{H}$, $\mathbf{H}^{\mathsf{sc}}$, $\mathbf{H}^{\mathsf{cc}}$ | Parity-check matrix |
| $\mathbf{G}$, $\mathbf{G}^{\mathsf{sc}}$, $\mathbf{G}^{\mathsf{cc}}$ | Generator matrix |
| $R^{\mathsf{sc}}$ | Source code rate |
| $R^{\mathsf{cc}}$ | Channel code rate |
| $R$ | Overall code rate |
| $\mathbf{B}$ | Protograph base (biadjacency) matrix |
| $b_c \times b_v$ | Dimensions of $\mathbf{B}$ |
| $\mathbf{B}_i$, $\mathbf{B}_i^{\mathsf{sc}}$, $\mathbf{B}_i^{\mathsf{cc}}$ | Component base matrix, $i \in [0 : m_s]$ |
| $b_c^{\mathsf{sc}} \times b_v^{\mathsf{sc}}$, $b_c^{\mathsf{cc}} \times b_v^{\mathsf{cc}}$ | Dimensions of the component base matrices |
| $\mathbf{B}_{[0,L-1]}$ | SC-LDPC Base matrix for a given $L$ |
| $L$, $L^{\mathsf{sc}}$, $L^{\mathsf{cc}}$ | Coupling length of SC-LDPC protograph |
| $M$, $M^{\mathsf{sc}}$, $M^{\mathsf{cc}}$ | Lifting factor |
| $m_s$, $m_s^{\mathsf{sc}}$, $m_s^{\mathsf{cc}}$ | Syndrome former memory of a SC-LDPC code |
| $t$ | Time |
| $R_L^{\mathsf{sc}}$, $R_L^{\mathsf{cc}}$, $R_L$ | SC-LDPC code rates for coupling length $L$ |
| $\mathbf{s}_{[0,\infty]} = [\mathbf{s}_0, \mathbf{s}_1, \ldots]$ | Source sequence and blocks per unit time |
| $\mathbf{u}_{[0,\infty]} = [\mathbf{u}_0, \mathbf{u}_1, \ldots]$ | Syndrome sequence and blocks per unit time |
| $\mathbf{v}_{[0,\infty]} = [\mathbf{v}_0, \mathbf{v}_1, \ldots]$ | Code sequence and blocks per unit time |
| $\mathbf{H}^{\mathsf{sc}}_{[0,\infty]}$, $\mathbf{H}^{\mathsf{cc}}_{[0,\infty]}$ | Syndrome former matrices lifted from $\mathbf{B}_{[0,\infty]}$ |
| $\mathbf{P}_t$ | The partial syndrome at time $t$ |

## VI. CONCLUSION

In this paper, we introduced a new construction of concatenated SC-LDPC codes based on protographs for JSCC over an AWGN channel and showed that such codes can be efficiently realized with sequential source and channel convolutional encoders and jointly decoded via a sliding window BP decoder. The obtained EXIT chart results show that, unlike LDPC block codes, concatenated SC-LDPC codes have improving thresholds for increasing graph density in *both* the source and channel code parts, with significant improvement over the underlying block code ensembles. Simulation results show the source reconstruction performance to be superior to state-of-the-art LDPC block codes for comparable latency and complexity constraints.

## APPENDIX

See Table VI.

## REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul./Oct. 1948.
[2] J. L. Massey, "Joint source and channel coding," in *Communications Systems and Random Process Theory*, vol. 11. Dordrecht, The Netherlands: Sijthoff and Nordhoff, 1978, pp. 279–293.
[3] G. Buch, F. Burkert, J. Hagenauer, and B. Kukla, "To compress or not to compress?" in *Proc. IEEE Global Telecommun. Conf.*, Nov. 1996, pp. 198–203.
[4] J. Hagenauer, "Source-controlled channel decoding," *IEEE Trans. Commun.*, vol. 43, no. 9, pp. 2449–2457, Sep. 1995.
[5] F. Jelinek, "Tree encoding of memoryless time-discrete sources with a fidelity criterion," *IEEE Trans. Inf. Theory*, vol. IT-15, no. 5, pp. 584–590, Sep. 1969.

[6] T. J. Goblick, "Coding for a discrete information source with a distortion measure," Ph.D. dissertation, Dept. Elect. Eng., Massachusetts Inst. Technol., Cambridge, MA, USA,1963.
[7] T. Ancheta, "Syndrome-source-coding and its universal generalization," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 4, pp. 432–436, Jul. 1976.
[8] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.
[9] G. Caire and S. Verdú, "A new data compression algorithm for sources with memory based on error correcting codes," in *Proc. IEEE Inf. Theory Workshop*, Mar. 2003, pp. 291–295.
[10] G. Caire and S. Verdú, "Almost-noiseless joint source-channel coding-decoding of sources with memory," in *Proc. Int. ITG Conf. Source Channel Coding*, 2004, pp. 295–304.
[11] Z. Peng, Y.-F. Huang, and D. J. Costello, "Turbo codes for image transmission—A joint channel and source decoding approach," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 6, pp. 868–879, Jun. 2000.
[12] A. Ruscitto and E. Biglieri, "Joint source and channel coding using turbo codes over rings," *IEEE Trans. Commun.*, vol. COM-46, no. 8, pp. 981–984, Aug. 1998.
[13] L. P. Kondi, F. Ishtiaq, and A. K. Katsaggelos, "Joint source-channel coding for motion-compensated DCT-based SNR scalable video," *IEEE Trans. Image Process.*, vol. 11, no. 9, pp. 1043–1052, Sep. 2002.
[14] A. J. Goldsmith and M. Effros, "Joint design of fixed-rate source codes and multiresolution channel codes," *IEEE Trans. Commun.*, vol. 46, no. 10, pp. 1301–1312, Oct. 1998.
[15] M. Fresia, F. Perez-Cruz, and H. Poor, "Optimized concatenated LDPC codes for joint source-channel coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2009, pp. 2131–2135.
[16] Q. Chen, L. Wang, S. Hong, and Z. Xiong, "Performance improvement of JSCC scheme through redesigning channel code," *IEEE Commun. Lett.*, vol. 20, no. 6, pp. 1088–1091, Jun. 2016.
[17] Q. Chen, S. Hong, and Y. Chen, "Design of linking matrix in JSCC scheme based on double protograph LDPC codes," *IEEE Access*, vol. 7, pp. 92176–92183, 2019.
[18] C. Chen, L. Wang, and F. C. M. Lau, "Joint optimization of protograph LDPC code pair for joint source and channel coding," *IEEE Trans. Commun.*, vol. 66, no. 8, pp. 3255–3267, Aug. 2018.
[19] L. Wang, H. Wu, and S. Hong, "The sensitivity of joint source-channel coding based on double protograph LDPC codes to source statistics," in *Proc. Int. Symp. Med. Inf. Commun. Technol. (ISMICT)*, Kamakura, Japan, Mar. 2015, pp. 213–217.
[20] D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, "Spatially coupled LDPC codes constructed from protographs," *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 4866–4889, Sep. 2015.
[21] S. Kudekar, C. Méasson, T. Richardson, and R. Urbanke, "Threshold saturation on BMS channels via spatial coupling," in *Proc. 6th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, 2010, pp. 309–313.
[22] S. Kudekar, T. J. 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.
[23] M. Lentmaier, A. Sridharan, D. J. Costello, and K. Sh. Zigangirov, "Iterative decoding threshold analysis for LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 5274–5289, Oct. 2010.
[24] V. Aref, N. Macris, and M. Vuffray, "Approaching the rate-distortion limit with spatial coupling, belief propagation, and decimation," *IEEE Trans. Inf. Theory*, vol. 61, no. 7, pp. 3954–3979, Jul. 2015.
[25] A. Golmohammadi, D. G. M. Mitchell, J. Kliewer, and D. J. Costello, "Windowed encoding of spatially coupled LDGM codes for lossy source compression," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2016, pp. 2084–2088.
[26] S. Schwandter, A. Graell i Amat, and G. Matz, "Spatially-coupled LDPC codes for decode-and-forward relaying of two correlated sources over the BEC," *IEEE Trans. Commun.*, vol. 62, no. 4, pp. 1324–1337, Apr. 2014.
[27] A. Golmohammadi and D. Mitchell, "Concatenated spatially coupled LDPC codes for joint source-channel coding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 631–635.
[28] W. E. Ryan and S. Lin, *Channel Codes: Classical Modern*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[29] H. V. B. Neto and W. Henkel, "Multi-edge optimization of low-density parity-check codes for joint source-channel coding," in *Proc. Int. ITG Conf. Syst., Commun. Coding*, Jan. 2013, pp. 1–6.
[30] JJ. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," Jet Propuls. Lab., Pasadena, CA, USA, Tech. Rep. 42-154, Aug. 2003.
[31] A. E. Pusane, A. J. Felstrom, A. Sridharan, M. Lentmaier, and K. S. Zigangirov, "Implementation aspects of LDPC convolutional codes," *IEEE Trans. Commun.*, vol. 56, no. 7, pp. 1060–1069, Jul. 2008.
[32] 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.
[33] S. Ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. COM-52, no. 4, pp. 670–678, Apr. 2004.
[34] G. Liva and M. Chiani, "Protograph LDPC codes design based on EXIT analysis," in *Proc. IEEE Global Telecommun. Conf.*, Nov. 2007, pp. 3250–3254.
[35] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. Com-36, no. 4, pp. 389–400, Apr. 1988.
[36] D. G. M. Mitchell, M. Lentmaier, A. E. Pusane, and D. J. Costello, "Randomly punctured LDPC codes," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 408–421, Feb. 2016.

**Ahmad Golmohammadi** (Member, IEEE) received the M.S. degree in electrical engineering from the University of Southern California, Los Angeles, USA, in 2012, and the Ph.D. degree in electrical engineering-communication systems from the Klipsch School of Electrical and Computer Engineering, New Mexico State University (NMSU), Las Cruces, NM, USA, in December 2020. Since then, he has been with Bose Corporation, Framingham, MA, USA, where he is currently a Senior Engineer. His research interests include coding theory, information theory, and signal processing.

**David G. M. Mitchell** (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from The University of Edinburgh, U.K., in 2009. From 2009 to 2015, he held the Post-Doctoral Research Associate and Visiting Assistant Professor positions with the Department of Electrical Engineering, University of Notre Dame, USA. Since 2015, he has been an Assistant Professor with the Klipsch School of Electrical and Computer Engineering, New Mexico State University, USA. His research interest is in the area of digital communications, with emphasis on error control coding and information theory. He currently serves as an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY.