# Concatenated Spatially Coupled LDPC Codes for Joint Source-Channel Coding

Ahmad Golmohammadi and David G. M. Mitchell
Klipsch School of Electrical & Computer Engineering, New Mexico State University, Las Cruces, NM 88003, USA.
{golmoham, dgmm}@nmsu.edu

***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. Simulation results show a notable performance improvement compared to existing state-of-the-art JSCC schemes based on LDPC codes.**

## 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 error pattern $\mathbf{e}$ and the source encoder generates the syndrome $\mathbf{u} = \mathbf{s}\mathbf{H}^T$ 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 a possible error pattern $\hat{\mathbf{e}}$ 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], rate-compatible punctured convolutional codes [12], and two concatenated regular LDPC block codes [13].

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 excellent channel coding performance [14]. Closely related spatially coupled low-density generator matrix (SC-LDGM) code ensembles were subsequently shown to have excellent performance for lossy source compression [15], [16]. In this paper, we present a construction of practically interesting protograph-based concatenated $(J, K)$-regular SC-LDPC codes for JSCC that can be encoded sequentially in a convolutional fashion with syndrome source coding then syndrome-former channel coding and 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. Simulation results for a binary memoryless source and a binary input additive white Gaussian noise (AWGN) channel show improved BER performance versus comparable concatenated LDPC block codes.

## II. LDPC-based joint source-channel coding

In this section, we summarize the LDPC-based JSCC proposed in [10]. As described above, the first LDPC code with parity-check matrix $\mathbf{H}^{\text{sc}}$ is used to calculate the syndrome $\mathbf{u}$ corresponding to the source input $\mathbf{s}$, and a second (systematic) LDPC code with parity-check matrix $\mathbf{H}^{\text{cc}}$ and generator matrix $\mathbf{G}^{\text{cc}}$ is used to encode to the compressed sequence for transmission through a noisy channel. For this system, a codeword $\mathbf{v}$ is obtained as

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

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

We follow [13] and apply BP to the concatenated graph as follows[1]: variable nodes send their message to check nodes at

[1]In [17], the two LDPC matrices ($\mathbf{H}^{\text{sc}}$ and $\mathbf{H}^{\text{cc}}$) are combined as one LDPC matrix and standard message passing applied between variable nodes and check nodes; however, we chose to follow separated BP updates as applied in [13].

This material is based upon work by the National Science Foundation under Grant No. CCSS-1710920.

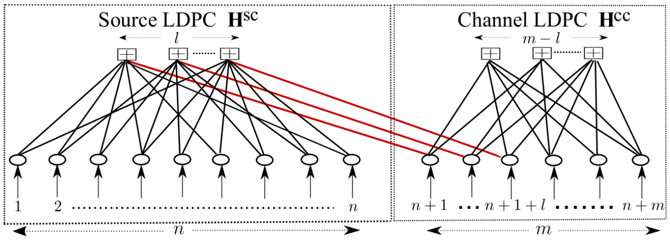


Fig. 1. Concatenated LDPC Tanner graphs for JSCC.

iteration $t$ as

$$m_{v\to c}^{\text{sc},(t)} = L_v^{\text{sc}} + \sum_{c'\neq c} m_{c'\to v}^{\text{sc},(t-1)}, \tag{2}$$

$$m_{v\to c}^{\text{cc},(t)} = L_v^{\text{cc}} + m_v^{\text{sc}\to\text{cc},(t-1)} + \sum_{c'\neq c} m_{c'\to v}^{\text{cc},(t-1)}, \tag{3}$$

$$m_{v\to c}^{\text{cc}\to\text{sc},(t)} = L_v^{\text{cc}} + \sum_{c'\neq c} m_{c'\to v}^{\text{cc},(t-1)}, \tag{4}$$

$$m_{v\to c}^{\text{cc},(t)} = L_v^{\text{cc}} + \sum_{c'\neq c} m_{c'\to v}^{\text{cc},(t-1)}, \tag{5}$$

where $m_{v\to c}^{\text{sc},(t)}$, $m_{v\to c}^{\text{cc},(t)}$, and $m_{v\to c}^{\text{cc}\to\text{sc},(t)}$ are the messages at iteration $t$ passed from the $v^{\text{th}}$ variable node to the $c^{\text{th}}$ check node within the source graph $\mathbf{H}^{\text{sc}}$, within the channel graph $\mathbf{H}^{\text{cc}}$, and between the channel and the source graphs, respectively, and where $L_v^{\text{sc}}$ and $L_v^{\text{cc}}$ denote the log-likelihood ratios (LLRs) for the variable nodes $v = 1,\ldots,n$ of the source decoder and $v = n+1,\ldots,n+m$ of the channel decoder, respectively. We assume a memoryless Bernoulli source, such that $p_v = \mathbb{P}(s_v = 1)$; therefore, for an additive white Gaussian noise channel, $L_v^{\text{sc}} = \log(\frac{1-p_v}{p_v})$ and $L_v^{\text{cc}} = \frac{2r_v}{\sigma_n^2}$, where $r_v$ is the received binary phase shifting keying (BPSK) value for a symbol transmitted on a channel with noise variance $\sigma_n^2$. For check to variable messages, $m_{c\to v}^{\text{sc},(t)}$, $m_{c\to v}^{\text{cc},(t)}$, and $m_{c\to v}^{\text{sc}\to\text{cc},(t)}$ 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

$$m_{c\to v}^{\text{sc},(t)} = 2\tanh^{-1}\left(\tanh\left(\frac{m_v^{\text{cc}\to c,(t)}}{2}\right)\prod_{v'\neq v}\tanh\left(\frac{m_{v'\to c}^{\text{sc},(t)}}{2}\right)\right), \tag{6}$$

$$m_{c\to v}^{\text{sc}\to\text{cc},(t)} = 2\tanh^{-1}\left(\prod_{v'}\tanh\left(\frac{m_{v'\to c}^{\text{sc},(t)}}{2}\right)\right), \tag{7}$$

$$m_{c\to v}^{\text{cc},(t)} = 2\tanh^{-1}\left(\prod_{v'\neq v}\tanh\left(\frac{m_{v'\to c}^{\text{cc},(t)}}{2}\right)\right). \tag{8}$$

Note that $m_{c\to v}^{\text{sc},(0)} = m_{c\to v}^{\text{cc},(0)} = m^{\text{sc}\to\text{cc},(0)} = 0$.

Regarding the separation, note that (2) applies for variable node indices $v = 1,\ldots,n$, (3) and (4) for $v = n+1,\ldots,n+1+l$, and (5) for $v = n+l+1,\ldots,n+m$. Equations (6) and (7) apply for check node indices $c = 1,\ldots,l$ and (8) for $c = l+1,\ldots,m$. After $I$ iterations of decoding, BP is terminated by computing the LLR of each source bit $s_v$, *i.e.*, $\text{LLR}(s_v) = L_v^{\text{sc}} + \sum_c m_{c\to v}^{\text{sc},(T)}$, 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 discuss protograph construction of SC-LDPC code ensembles, present our concatenated construction of SC-LDPC codes with an example, then describe the encoding and decoding procedures.

### A. SC-LDPC protographs

A protograph [18] 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 $\mathbf{B}$, 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 $\mathbf{H}$ 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* [14] with base matrix

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

where $m_s$ is the *syndrome former memory* of the code and the $b_c \times b_v$ *component base matrices* $\mathbf{B}_i$, $i = 0,\ldots,m_s$, 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 [14] 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 $\mathbf{B}_{[-\infty,\infty]}$ 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 $L$ time instants, the resulting finite-length base matrix is then given by

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

The matrix $\mathbf{B}_{[0,L-1]}$ can be considered as the base matrix of a terminated protograph-based SC-LDPC code ensemble. Termination results in a rate loss: without puncturing, the *design compression rate* for syndrome source coding with $\mathbf{B}_{[0,L-1]}$ is $R_L^{\text{sc}} = \left(\frac{L+m_s}{L}\right)\frac{b_c}{b_v}$ whereas for channel coding the *design rate* $R_L^{\text{cc}}$ of the terminated code ensemble is equal to $R_L^{\text{cc}} = 1-\left(\frac{L+m_s}{L}\right)\frac{b_c}{b_v}$. As the termination factor $L$ increases, the rate loss diminishes monotonically so that, as $L \to \infty$, $R^{\text{sc}}_{L^{\text{sc}}} \to R^{\text{sc}} = b_c/b_v$ and $R^{\text{cc}}_{L^{\text{cc}}} \to R^{\text{cc}} = 1-b_c/b_v$ (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 with base matrices given in (9) or (10), one for source compression, $\mathbf{H}^{\text{sc}}$, and another for channel coding, $\mathbf{H}^{\text{cc}}$. Notationally, we introduce superscripts to the parameters to indicate their use in $\mathbf{H}^{\text{sc}}$ or $\mathbf{H}^{\text{cc}}$, *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 scheme to work, we must select $b_c^{\text{sc}}M^{\text{sc}} = (b_v^{\text{cc}} - b_c^{\text{cc}})M^{\text{cc}}$. We restrict our constructions to have $M^{\text{sc}} = M^{\text{cc}} = M$, implying $b_c^{\text{sc}} = b_v^{\text{cc}} - b_c^{\text{cc}}$. We also choose to set $m_s^{\text{sc}} = m_s^{\text{cc}} = m_s$ thus if the codes are terminated (using base matrices in (10)), we must further have $L^{\text{sc}} + m_s = L^{\text{cc}} - m_s$ and we denote the length of the concatenated scheme as $L = L^{\text{sc}}$. The overall coding rate for the terminated scheme is $R_L = \frac{R_L^{\text{cc}}}{R_L^{\text{sc}}}$ and, as $L \to \infty$, the overall coding rate approaches $R = \frac{R^{\text{cc}}}{R^{\text{sc}}} = \frac{b_v^{\text{sc}}}{b_v^{\text{cc}}}$, the rate of the unterminated JSCC scheme.

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 $M^{\text{sc}} = M^{\text{cc}}$. We require $L^{\text{cc}} = L^{\text{sc}} + 4$ and use the notation $L = L^{\text{sc}}$ for the concatenated design. We use component matrices $\mathbf{B}_0^{\text{sc}} = \mathbf{B}_1^{\text{sc}} = \mathbf{B}_2^{\text{sc}} = [1\ \ 1\ \ 1\ \ 1]$ to construct $\mathbf{H}^{\text{sc}}_{[0,L-1]}$, with compression rate $R_L^{\text{sc}} = \left(\frac{L+m_s}{L}\right)\frac{b_c^{\text{sc}}}{b_v^{\text{sc}}} \xrightarrow[L\to\infty]{} \frac{b_c^{\text{sc}}}{b_v^{\text{sc}}} = \frac{1}{4}$ and component matrices $\mathbf{B}_0^{\text{cc}} = \mathbf{B}_1^{\text{cc}} = \mathbf{B}_2^{\text{cc}} = [1\ \ 1]$ with channel code rate $R_L^{\text{cc}} = 1-\left(\frac{L+4+m_s}{L+4}\right)\frac{b_c^{\text{cc}}}{b_v^{\text{cc}}} \xrightarrow[L\to\infty]{} 1-b_c^{\text{cc}}/b_v^{\text{cc}} = \frac{1}{2}$. The overall coding rate $R_L = \frac{R_L^{\text{cc}}}{R_L^{\text{sc}}} \xrightarrow[L\to\infty]{} 2$.

The protograph of the proposed construction is shown in Fig. 2. In order to be able to use $\mathbf{H}^{\text{cc}}$ directly for systematic encoding of the syndrome $\mathbf{u}$, we connect the protographs such that, at each time instant, the left entry of $\mathbf{B}_0^{\text{cc}}$ (top variable node in Fig. 2) connects to the check node of $\mathbf{B}^{\text{sc}}$ at that time - this connects the systematic bits of the channel code to the syndrome of the source. When lifting, we must also restrict the permutation matrix associated with the right most entry of $\mathbf{B}_0^{\text{cc}}$ (the $M$ 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 $\mathbf{H}^{\text{cc}}$ 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*

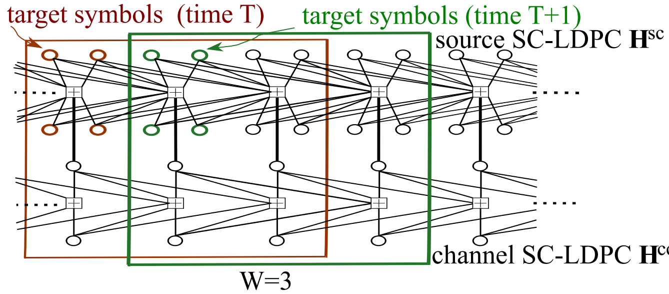


Fig. 2. Concatenated protographs of SC-LDPC codes for JSCC. Also illustrated is the WD procedure with window size $W = 3$.

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

where the submatrices $\mathbf{H}_i$, $i = 0, 1, \ldots, m_s$, are defined as

$$\mathbf{H}_i(T) = \begin{bmatrix} h_i^{(1,1)}(T) & \cdots & h_i^{(1,b_vM)}(T) \\ \vdots & & \vdots \\ h_i^{(b_cM,1)}(T) & \cdots & h_i^{(b_cM,b_vM)}(T) \end{bmatrix}_{b_cM\times b_vM}. \tag{12}$$

*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 $b_v^{\text{sc}}M$. We obtain the compressed syndrome $\mathbf{u}_{[0,\infty]} = [\mathbf{u}_0, \mathbf{u}_1, \ldots] = \mathbf{s}_{[0,\infty]}\mathbf{H}^{\text{sc}^T}_{[0,\infty]}$, where $\mathbf{u}_i = \mathbf{s}_i\mathbf{H}_0^{\text{sc}^T}(i) + \mathbf{s}_{i-1}\mathbf{H}_1^{\text{sc}^T}(i) + \cdots + \mathbf{s}_{i-m_s}\mathbf{H}_{m_s}^{\text{sc}^T}(i) = (u_i^{(1)}, u_i^{(2)}, \ldots, u_i^{(b_c^{\text{sc}}M)})$. 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 $\mathbf{v}_i = (v_i^{(1)}, v_i^{(2)}, \cdots, v_i^{(b_v^{\text{cc}}M)})$. The resulting code sequence $\mathbf{v}_{[0,\infty]}$ satisfies $\mathbf{v}_{[0,\infty]}\mathbf{H}^T_{[0,\infty]} = 0$. By design, our $\mathbf{H}^{\text{cc}}_{[0,\infty]}$ defines a *systematic* convolutional code of rate $R = 1 - b_v^{\text{cc}}/b_c^{\text{cc}}$. In [19], two methods are introduced to encode $\mathbf{u}_{[0,\infty]}$ into $\mathbf{v}_{[0,\infty]}$: 1) syndrome former realization and 2) partial syndrome former realization.

*a) Syndrome Former Realization:* Systematic encoding can be performed as [19]

$$v_i^{(j)} = u_i^{(j)}, \quad j = 1, \ldots, (b_v^{\text{cc}} - b_c^{\text{cc}})M, \tag{13}$$

$$\begin{aligned} v_i^{(j)} = & \sum_{k=1}^{(b_v^{\text{cc}}-b_c^{\text{cc}})M} v_i^{(k)} h_0^{(j-(b_v^{\text{cc}}-b_c^{\text{cc}})M,k)}(i) \\ & + \sum_{c=1}^{m_s}\sum_{k=1}^{b_v^{\text{cc}}M} v_{i-c}^{(k)} h_c^{(j-(b_v^{\text{cc}}-b_c^{\text{cc}})M,k)}(i), \\ & \qquad j = (b_v^{\text{cc}} - b_c^{\text{cc}})M + 1, \ldots, b_v^{\text{cc}}M. \end{aligned} \tag{14}$$

*b) Partial Syndrome Former Realization:* 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 polynomial convolutional encoders use a sequences of zeros as the terminating tail, SC-LDPC encoders use non-zero sequences for the terminating tail, which depend on the encoded information symbols and are needed to solve a system of linear equations. Details are omitted here due to space constraints, but the interested reader can refer to [19] for a full description of the partial syndrome former realization method to terminate the encoder.

In this paper, we applied the partial syndrome former method for encoding the last $m_s$ sections of $\mathbf{v}_{[0,L+4-1]}$ (the termination tail); for all other sections we use the syndrome former realization method directly. The tail length is variable, codes having tail lengths of $\tau = 2(m_s + 1)$ were used and any longer tails were rejected (about $50\%$ of all randomly generated codes).

### 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 [20], 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. Numerical results

In this section, we present numerical results for WD of unterminated concatenated $(3, 6)$- and $(3, 12)$-regular SC-LDPC codes for JSCC. Simulation results were obtained with a binary input AWGN channel and the source symbols are assumed to be i.i.d. with $\mathbb{P}(s_v = 1) = p_v = 0.02$. Results were obtained by averaging over 100000 block samples and

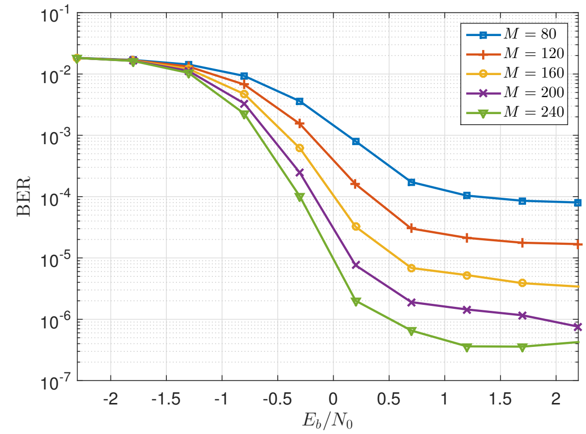


Fig. 3. JSCC performance of SC-LDPC codes with increasing $M$.

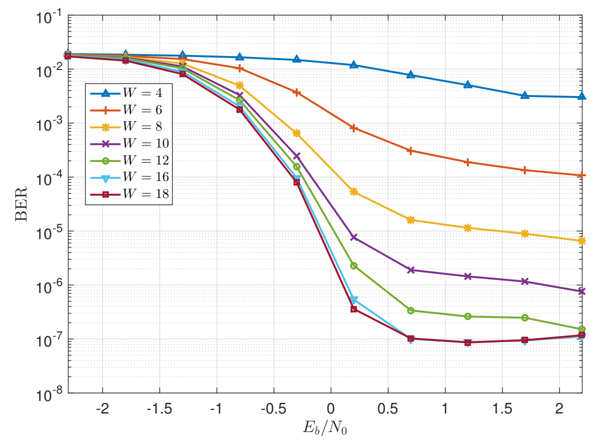


Fig. 4. JSCC performance of SC-LDPC codes with increasing $W$.

a fixed number of $I = 30$ iterations per window position was performed.[2]

### A. Performance of concatenated SC-LDPC codes

*1) Effect of increasing the lifting factor $M$:* Fig. 3 shows the effect of increasing the lifting factor $M$ (improving code strength) with a fixed window size $W = 10$. The resulting latency is $2MW = 20M$. 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 source distortion. Indeed, no such floor is observed in BER plots for the channel coding part only.

*2) Effect of the window size $W$:* Fig. 4 shows the JSCC performance with increasing window size $W = 4, 6, 8, 10, 12$ (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 $W$.

### 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

[2]Stopping rules could be included to reduce the number of iterations performed in many cases. This is the subject of ongoing work.

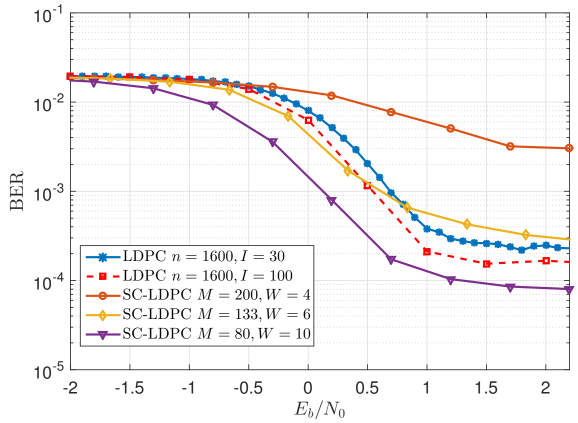


Fig. 5. Equal latency comparison (1600 bits) of concatenated SC-LDPC codes vs. LDPC block codes.

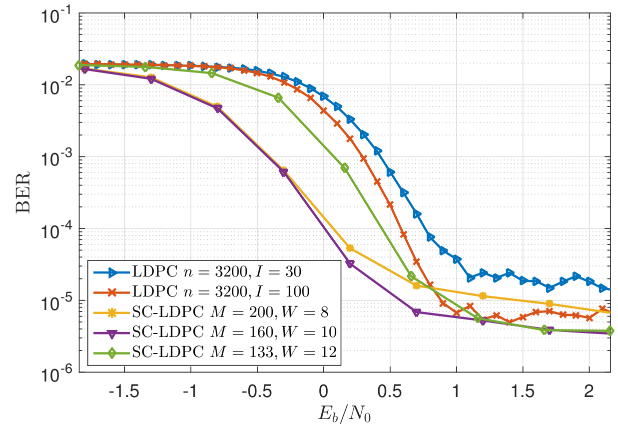


Fig. 6. Equal latency comparison (3200 bits) of concatenated SC-LDPC codes vs. LDPC block codes.

to 1600 and 3200 bits. We then compare with the concatenated block LDPC code designs for JSCC given in [13].[3] To model the codes from [13], we used a regular protograph construction with $\mathbf{B}^{\mathrm{sc}} = \mathbf{1}_{3\times 12}$ and $\mathbf{B}^{\mathrm{cc}} = \mathbf{1}_{3\times 6}$, 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. 5 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 $E_b/N_0$. Fig. 6 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 LDPC block codes in the waterfall, with similar error floor performance. 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.

## V. Conclusion

In this paper we introduced a new construction of concatenated SC-LDPC codes based on protographs for joint source-channel coding and proposed a novel windowed decoding algorithm. Simulation results showed that the proposed decoder has good source reconstruction performance for moderate decoding latency. 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 ongoing work.

[3] Similar performance gains to those found for block decoding in [13] were observed for joint window decoding of concatenated SC-LDPC codes when compared to cascade decoding (where, prior to source decoding, the channel code is decoded independently assuming equally likely i.i.d. compressed bits from the source). The results are omitted due to space constraints.

## References


[1] C. E. Shannon, "A mathematical theory of communication," *Bell Sys. tech. J.*, vol. 27, pp. 379-423, 623-656, Jul.-Oct. 1948.

[2] J. L. Massey, "Joint source and channel coding," *Communication Systems and Random Process Theory*, vol. 11, pp. 279-293, Sijthoff and Nordhoff, 1978.

[3] G. Buch, F. Burkert, J. Hagenauer, and B. Kukla, "To compress or not to compress?" *Proc. IEEE Global Comm. Conference*, pp.198-203, 1996.

[4] J. Hagenauer, "Source-controlled channel decoding," *IEEE Trans. Communications*, pp. 2449-57, 1995.

[5] F. Jelinek, "Tree encoding of memoryless time-discrete sources with a fidelity criterion," *IEEE Trans. Inf. Theory*, vol. 15, no. 5, pp. 584-590, 1969.

[6] T. J. Goblick, "Coding for a discrete information source with a distortion measure," PhD diss., Massachusetts Institute of Technology, 1963.

[7] T. C. Ancheta, "Syndrome-source-coding and its universal generalization," *IEEE Trans. on Inf. Theory*, vol. 22, no.4, pp. 432-436, July 1976.

[8] R. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. theory*, vol. 8, no. 1, pp. 21-28, 1962.

[9] G. Caire, S. Shamai, and S. Verdú, "A new data compression algorithm for sources with memory based on error correcting codes," *Proc. IEEE Inf. Theory Workshop*, pp. 291-295, 2003.

[10] G. Caire, S. Shamai, and Sergio Verdú, "Almost-noiseless joint source-channel coding-decoding of sources with memory," *Proc. Int. ITG Conf. on Source and Channel Coding*, pp. 295-304, 2004.

[11] Z. Peng, Y. Huang, and D. J. Costello, "Turbo codes for image transmission-a joint channel and source decoding approach," *IEEE J. Sel. Areas in Comm.*, vol. 18, no. 6, pp. 868-879, 2000.

[12] L. P. Kondi, F. Ishtiaq, and A. K. Katsaggelos, "Joint source-channel coding for motion-compensated DCT-based SNR scalable video," *IEEE Trans. Image Processing*, vol. 11, no. 9, pp. 1043-1052, 2002.

[13] M. Fresia, F. Perez-Cruz, and H. Poor, "Optimized concatenated LDPC codes for joint source-channel coding," *Proc. IEEE Int. Symp. Inf. Theory*, pp. 2131-2135, Jun. 2009.

[14] D. G. M. Mitchell, M. Lentmaier, D. J. Costello, "Spatially Coupled LDPC Codes Constructed From Protographs," *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 4866-4889, Sept. 2015.

[15] V. Aref, N. Macris, 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, July 2015.

[16] A. Golmohammadi, D. G. M. Mitchell, J. Kliewer, and D. J. Costello, "Windowed encoding of spatially coupled LDGM codes for lossy source compression," *Proc. IEEE Int. Symp. Inf. Theory*, pp. 2084-2088, 2016.

[17] H. V. Beltrão Neto and W. Henkel, "Multi-Edge Optimization of Low-Density Parity-Check Codes for Joint Source-Channel Coding," *Proc. Int. ITG conference on Systems, Communications and Coding,* Jan. 2013.

[18] J. Thorpe,"Low-density parity-check (LDPC) codes constructed from protographs," Jet Propulsion Laboratory, Pasadena, CA, INP Progress Report 42-154, Aug. 2003.

[19] A. E. Pusane, A.J. Feltstrom, A. Sridharan, M. Lentmaier, K. Zigangirov, and D.J. Costello, "Implementation aspects of LDPC convolutional codes," *IEEE Trans. Comm.*, vol. 56, no. 7, pp. 1060-1069, Jul. 2008.

[20] 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, 2012.