Journal of Algebra and Its Applications
Vol. 23, No. 7 (2024) 2550127 (25 pages)

DOI: 10.1142/S0219498825501270

World Scientific
www.worldscientific.com

Window code parameters of spatially-coupled LDPC codes

Emily McMillon
Department of Mathematics, Rice University
6100 Main Street, Houston, Texas 77005, USA
emily.mcmillon@rice.edu

Christine A. Kelley*
Department of Mathematics, University of Nebraska–Lincoln
1400 R Street, Lincoln, Nebraska 68588, USA
ckelley2@unl.edu

Received 14 December 2022
Accepted 11 June 2023
Published 11 January 2024

Communicated by R. Smarandache

In this paper, we define a window code to be the portion of a Spatially-coupled low-density parity check (SC-LDPC) code seen by a single iteration of a windowed decoder. We consider the design of SC-LDPC codes for windowed decoding via optimization of the window code. In particular, because iterative decoding is optimal on codes with cycle-free graph representations, we ask fundamental questions about the construction and parameters of cycle-free window codes. We show that it is possible to have an SC-LDPC code with cycles and with cycle-free window codes. We consider the relationship between the distance of the window code and the distance of the SC-LDPC code. Further, we show that SC-LDPC codes with MDS window codes exist, and all such codes are asymptotically bad. This work gives insight into the tradeoffs between window code parameters and performance of the SC-LDPC code.

Keywords: Low-density parity-check (LDPC) codes; spatially coupled codes; iterative decoding; minimum distance; Tanner graphs; cycle-free codes; window codes.

Mathematics Subject Classification 2020: 94B05, 94B25, 94B35

1. Introduction

Spatially-coupled low-density parity check (SC-LDPC) codes are a special class of LDPC codes originally introduced by Felström and Zigangirov in [5]. These codes exhibit a phenomenon called threshold saturation [8], meaning that SC-LDPC codes converge quickly under iterative decoding to relatively error-free solutions.

*Corresponding author.

Due in large part to their highly repetitive structures, SC-LDPC codes are amenable to windowed decoding, which is a process by which small portions of a code are decoded sequentially. This sequential decoding is particularly conducive to applications such as wireless communication and audio and video streaming [7, 11]. While there are exceptions, such as [14] that examines decoding thresholds for different window parameters, most existing work has focused on optimizing the parameters of the entire SC-LDPC code. Since at any given time step, only a small window of the code is active, in this paper, we instead optimize the portion of the code seen by the windowed decoder at a given time step, which we call the *window code*.

There are many different aspects one could "optimize" with respect to the window code of an SC-LDPC code, including distance, rate, degree distribution, and cycle structure. In this paper, we focus on cycle structure and distance parameters. In particular, because iterative decoding is optimal on codes with cycle-free graph representations [15], we ask fundamental questions about the construction and parameters of cycle-free window codes. Iterative decoding on a cycle-free graph is optimal in that it avoids combinatorial structures such as stopping sets and absorbing sets that are known to cause iterative decoder failure on certain channels [2, 3]. Cycle-free codes do not see much practical use, as they have poor distance properties [4]. However, because window codes are small pieces of a larger code, they present an opportunity to mix the best of both worlds: optimal iterative decoding and good rate/distance parameters. Many of the results on cycle-free window codes were first presented in [9], though not all with proofs.

Maximum distance separable (MDS) codes are known for their optimal distance/rate tradeoffs. A natural question is whether there exist SC-LDPC window codes that have MDS window codes. We establish necessary conditions on the SC-LDPC code for this property to hold, and show that while such SC-LDPC codes do exist, they have asymptotically poor rates. We also provide initial bounds on the relationship between the window code minimum distance and the distance of the SC-LDPC code given certain conditions.

While this work exclusively considers time-invariant SC-LDPC codes, a potential future avenue of research would be to study window code parameters of time-varying SC-LDPC codes.

This paper is organized as follows. In Sec. 2, we provide notation and background on SC-LDPC codes. In Sec. 3, we provide the definition and motivation for window codes. Section 4 presents results on parameters of cycle-free window codes. Section 5 presents results on distance parameters of window codes, including MDS window codes. Section 6 concludes the paper.

2. Preliminaries

In this section, we introduce relevant notation and terminology. Let $\mathcal{C}$ be a binary $[n, k]$ linear code with block length n, dimension k, and minimum Hamming distance

denoted by $d_{\min}(\mathcal{C})$. If $\mathcal{C}$ is represented by parity check matrix H, its associated Tanner graph is defined as the bipartite graph $G = (V, W; E)$, where the vertices in part V correspond to the columns of H, and the vertices in W correspond to the rows of H. These are referred to as variable and check nodes, respectively. There is an edge between $v_i \in V$ and $w_j \in W$ if and only if $H_{j,i} \neq 0$. When H is sparse, $\mathcal{C}$ is a low-density parity check (LDPC) code.

LDPC codes can be defined over any field. In the case of a binary LDPC code, the associated Tanner graph has unlabeled edges. In the case of an LDPC code over $\mathbb{F}_q$ where $q \neq 2$, the entry $H_{j,i}$ corresponds to an edge label on the edge between $v_i \in V$ and $w_j \in W$. Unless stated otherwise, all results in this paper are valid over all $\mathbb{F}_q$.

SC-LDPC codes may be constructed by coupling L copies of a base Tanner graph, where L is called the coupling length [11]. Viewing these copies as being in positions $0, 1, \ldots, L-1$, this coupling is done by a process termed "edge spreading," whereby edges connected to variable nodes at position i are spread to check nodes at positions $i, i+1, \ldots, i+m$ such that the variable node degrees of the base graph are maintained and any edge is connected to some copy of the same check node to which it was incident in the base graph. Here, m is called the coupling width. This edge spreading is done so that the way in which edges are spread at position 0 is replicated at positions $1, 2, \ldots, L-1$. If additional check nodes are introduced at the end of the coupled graph to complete the connections, the code is said to be terminated. Alternatively, allowing the edges at the end of the coupled graph to wrap around and connect to the beginning of the graph results in a tailbiting code. The coupled graph is often taken as the protograph of an SC-LDPC code.

We introduce new terminology, that of a *forward edge*, to aid in forthcoming arguments regarding the edge spreading process.

Definition 2.1. Let $\mathcal{C}$ be an SC-LDPC code with memory m. We define a *forward edge* to be any edge connected to a variable node in position i of an SC-LDPC code that is spread to a check node at position $i + k$, where $1 \leq k \leq m$.

Remark 2.1. Alternatively, SC-LDPC codes may be constructed from array-based codes and making use of a cutting vector [10]. It has been shown that many common SC-LDPC code constructions can be obtained from an algebraic graph lift with appropriate constraints on the base graph [1].

Example 2.1. Figure 1 shows a Tanner graph and one possible coupled version of that Tanner graph. Variable nodes are represented by circles, and check nodes by squares. The coupling length is three and coupling width is one. Note that the resulting code is terminated, as the check nodes at position 3 have not been wrapped around to coincide with those in position 0.

The edge-spreading process can also be viewed as splitting a base parity check matrix H into $m+1$ matrices of the same dimension such that $H = H_0 + H_1 + \cdots + H_m$, and then arranging them in a matrix with L block columns. When viewing

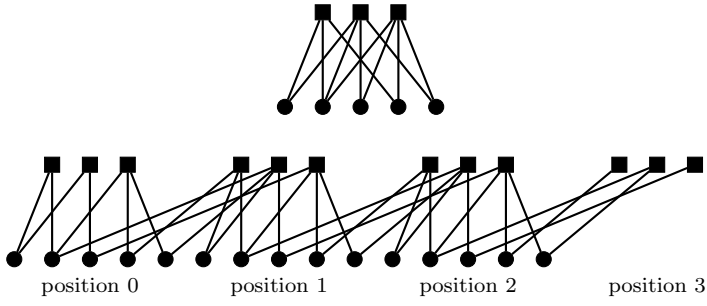

Fig. 1. A terminated SC-LDPC Tanner graph (bottom) is made using the base Tanner graph (above) with $L = 3$ and $m = 1$.

the process as the splitting of a parity check matrix, it is possible for H to have multi-edges. Edges corresponding to ones in $H_1, \ldots, H_m$ are forward edges. If $\mathcal{C}$ is terminated, then its parity check matrix $\mathcal{H}$ has dimensions $(L+m)n_c \times Ln_v$ and the form:

$$\mathcal{H} = \begin{bmatrix} H_0 & & & & & & & \\ H_1 & H_0 & & & & & & \\ \vdots & & \ddots & & & & & \\ H_m & H_{m-1} & \ldots & H_0 & & & & \\ & H_m & \ldots & H_1 & H_0 & & & \\ & & \ddots & & & \ddots & & \\ & & & H_m & \ldots & H_1 & H_0 & \\ & & & & \ddots & & \vdots & \\ & & & & & H_m & H_{m-1} & \\ & & & & & & H_m & \end{bmatrix}.$$

If $\mathcal{C}$ is a tailbiting SC-LDPC code, then its parity check matrix $\mathcal{H}$ has dimensions $Ln_c \times Ln_v$ and the form:

$$\mathcal{H} = \begin{bmatrix} H_0 & & & & H_m & \ldots & H_1 \\ H_1 & H_0 & & & & \ddots & \vdots \\ \vdots & & \ddots & & & & H_0 \\ H_m & H_{m-1} & \ldots & H_0 & & & \\ & H_m & \ldots & H_1 & H_0 & & \\ & & \ddots & & & \ddots & \\ & & & H_m & \ldots & H_1 & H_0 \end{bmatrix}.$$

Unless otherwise noted, we will assume that the dimensions of the base matrix of the resulting SC-LDPC code are $n_c \times n_v$. Consequently, the dimensions of each submatrix H_i in an SC-LDPC code are $n_c \times n_v$.

A terminated SC-LDPC code has design rate

$$R_L = 1 - \frac{(L+m)n_c}{Ln_v} = 1 - \left(\frac{L+m}{L}\right)(1-R),$$

where $R = 1 - \frac{n_c}{n_v}$ is the design rate of the base matrix. Note that $R_L < R$ and $\lim_{L\to\infty} R_L = R$. In practice, L is chosen to be large, so the design rate of the base matrix is a good approximation of the design rate of the overall code.

3. Window Codes

In this section we review windowed decoding and introduce the notion of window code.

Definition 3.1. A *windowed decoder* runs on a small *window* of nodes and slides left to right along the received bits as it decodes. The *window length*, denoted W, is defined to be W consecutive positions of constraint nodes and all of their adjacent variable nodes [6]. For i between 0 and $L-1$, the *window at position i*, denoted W_i, consists of the check nodes in positions i through $i+W-1$, their adjacent variable nodes, and the edges between them. The set of edges in a given window is called a *window configuration* of the windowed decoder.

In the case of a terminated SC-LDPC code, the end window configurations will differ from the typical window configuration. Recall that the variable nodes in a single position may be adjacent to check nodes in at most $m+1$ positions. Thus, requiring a minimum window length of $m+1$ ensures that at least one position of variable nodes has all its adjacent check nodes within a window. Hence, we assume that the window length satisfies $m+1 \leq W \leq m+L$, as in [6, 13].

Definition 3.2. For a window configuration starting at position i, the variable nodes in the ith position are called the *targeted symbols*.

Decoding in a window stops once the window's targeted symbols reach a target error probability or after a fixed number of iterations have been completed [6]. The window then moves one position to the right and the process repeats.

Because the decoder operates on each window, it is natural to consider the code defined by the graph seen on that window and ask what the optimal parameters are for such a code. To that end, we introduce the following definition.

Definition 3.3. Let $\mathcal{C}$ be a SC-LDPC code of length L. We define the *window code at position i*, denoted $\mathcal{C}_{W_i}$, to be the code consisting of the check nodes in positions i through $i+W-1$ and all their adjacent variable nodes. In other words, $\mathcal{C}_{W_i}$ is the code with Tanner graph W_i. Note that, necessarily, $i \in \{0, 1, \ldots, L-1\}$. Further,

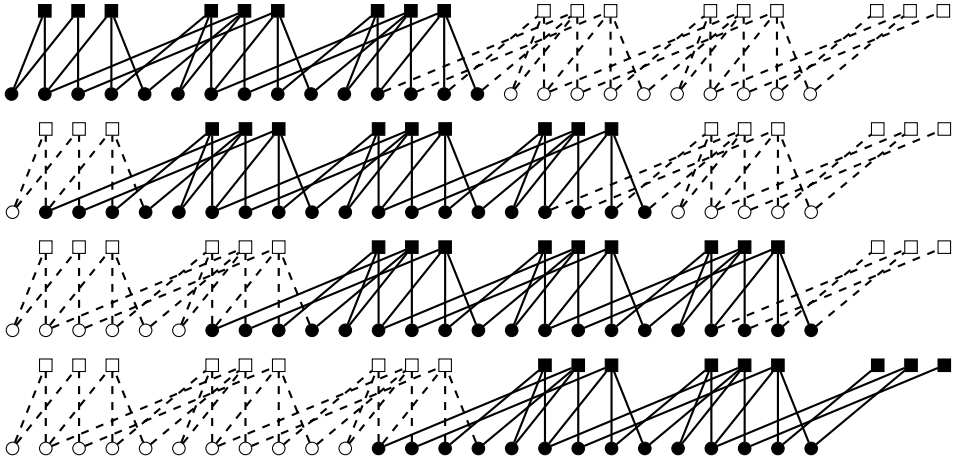

Fig. 2. A terminated SC-LDPC Tanner graph using the base graph in Fig. 1 with $L = 5$, $m = 1$, and $W = 3$. The windows at position i for $i \in \{0, 1, 2, 3\}$ are highlighted by dark vertices and solid edges.

if $d = d_{\min}(\mathcal{C})$, then we define $d_{W_i} = d_{\min}(\mathcal{C}_{W_i})$ to be the *minimum distance of the window code* $\mathcal{C}_{W_i}$.

Example 3.1. Recall that in Fig. 1, a smaller base Tanner graph was used to form an SC-LDPC code. In Fig. 2, we use the same edge spreading choices as in Fig. 1 but with a coupling length $L = 5$ to highlight the windowed decoding process when the window length is $W = 3$. As an example, the third configuration in the figure corresponds to the second decoding window, W_2, and the targeted symbols are the variable nodes at position 2. Notice that W_1 and W_2 are identical, but W_0 and W_3 are each unique. This implies that $\mathcal{C}_{W_1} = \mathcal{C}_{W_2}$. For L larger than 5, additional window configurations identical to W_1 and W_2 appear. In general, as L gets larger, the number of identical window configurations will increase.

Due to the repetitive structure of SC-LDPC codes, in practice, almost all window codes of an SC-LDPC code will be identical. Hence, in optimizing the window code structure of an SC-LDPC code, the majority of our attention can be focused on a single window code, which we call the typical window code, defined next.

Definition 3.4. Let $\mathcal{C}$ be a terminated SC-LDPC code of length L, window length W, and memory m. Then the window codes at positions m through $L - W$ are identical. We call this window code the *typical window code of* $\mathcal{C}$ and denote it $\mathcal{W}(\mathcal{C})$, or, when $\mathcal{C}$ is understood, we simply denote it $\mathcal{W}$. Further, we define the *window code distance* to be the minimum distance of the typical window code, and denote it by $d_{\mathcal{W}}$, i.e. $d_{\mathcal{W}} = d_{\min}(\mathcal{W}(\mathcal{C}))$, and we define the *typical window code parity check matrix* to be the parity check matrix of $\mathcal{W}(\mathcal{C})$ and denote it by $H_{\mathcal{W}}$. In general, if a window code is not a typical window code, we will refer to it as an

atypical window code. In the case of a tailbiting SC-LDPC code, all window codes are typical window codes.

Example 3.2. In Example 3.1, the codes corresponding to W_1 and W_2 are typical window codes of $\mathcal{C}$, i.e. $\mathcal{C}_{W_1} = \mathcal{C}_{W_2} = \mathcal{W}(\mathcal{C})$. On the other hand, $\mathcal{C}_{W_0}$ and $\mathcal{C}_{W_3}$ are atypical window codes of $\mathcal{C}$.

It is also helpful to keep in mind the form of a parity check matrix of a typical window code. Such a parity check matrix has dimensions $Wn_c \times (W+m)n_v$, and has form given below.

$$H_{\mathcal{W}} = \begin{bmatrix} H_m & H_{m-1} & \dots & H_0 & & & \\ & H_m & \dots & H_1 & H_0 & & \\ & & \ddots & & & \ddots & \\ & & & H_m & \dots & H_1 & H_0 \end{bmatrix}. \tag{3.1}$$

Unless otherwise noted, we will assume that the dimensions of the base matrix H of the resulting SC-LDPC code are $n_c \times n_v$. Consequently, the dimensions of each submatrix H_i in an SC-LDPC code are $n_c \times n_v$.

4. Cycle-Free Window Codes

Iterative decoding has been shown to be optimal on codes with cycle-free graph representations [15]. However, codes with cycle-free graph representations were shown by Etzion *et al.* [4] to have poor distance properties, as given in the bound restated next.

Theorem 4.1 ([4]). *Let $\mathcal{C}$ be an $[n,k,d]$ cycle-free linear code. Then*

$$d \leq \left\lfloor \frac{n}{k+1} \right\rfloor + \left\lfloor \frac{n+1}{k+1} \right\rfloor.$$

In particular, notice that when $\frac{k}{n} \geq \frac{1}{2}$, Theorem 4.1 reduces to $d \leq 2$.

Because spatially-coupled codes are amenable to windowed decoding, it is possible for the code to have cycles while having cycle-free window codes, and hence, are optimal with respect to decoding. This means spatially-coupled codes are not limited by the distance bounds of cycle-free codes.

As in [4], we begin by recalling the following well-known result.

Lemma 4.1. *A graph $G = (V, E)$ is cycle-free if and only if $\#E = \#V - \omega(G)$, where $\omega(G)$ denotes the number of connected components in G.*

Any graph contains at least one connected component, and so this bound reduces to $\#E \leq \#V - 1$. If H is an $m \times n$ matrix, then the number of vertices in the Tanner graph corresponding to H is $m+n$, and the number of edges in this Tanner graph is $\mathrm{wt}(H)$, the number of nonzero entries in H. Hence, if H is cycle-free, then $\mathrm{wt}(H) \leq m + n - 1$.

Lemma 4.2. *Given an SC-LDPC code $\mathcal{C}$ with cycle-free window codes, memory m, and window size W,*

$$\mathrm{wt}(H_{\mathcal{W}(\mathcal{C})}) \leq (W+m)n_v + Wn_c - 1.$$

Proof. Because $\mathcal{W}(\mathcal{C})$ has a $Wn_c \times (W+m)n_v$ cycle-free parity check matrix, this result follows immediately from the argument in the preceding paragraph. □

Remark 4.1. Because the Tanner graphs corresponding to atypical window codes are subgraphs of the Tanner graph of a typical window code, assuming that the typical window code of an SC-LDPC code is cycle-free is sufficient to guarantee that all window codes of an SC-LDPC code are cycle-free. In other words, if $\mathcal{W}(\mathcal{C})$ is cycle-free, then all atypical window codes of $\mathcal{C}$ are also cycle-free.

Throughout the remainder of this paper, we will assume that $d_{\min}(\mathcal{W}(\mathcal{C})) \geq 2$. This restriction is sensible, as it guarantees that $H_{\mathcal{W}(\mathcal{C})}$ has no zero columns. Because the bulk of the arguments in this section depend upon Lemma 4.2, it is important to have a lower bound on $\mathrm{wt}(H_{\mathcal{W}(\mathcal{C})})$, and the absence of zero columns allows us to do so.

4.1. *Column weight constraint only*

Lemma 4.3. *If $\mathcal{C}$ is an SC-LDPC code with cycle-free window codes, memory m, window size W, and $d_{\mathcal{W}} \geq 2$, then $\mathrm{wt}(H_0) \geq n_v$ and $\mathrm{wt}(H_m) \geq n_v$.*

Proof. Since $d_{\mathcal{W}} \geq 2$, $H_{\mathcal{W}}$ has no zero columns. Recall that $H_{\mathcal{W}}$ has W block rows and $W+m$ block columns. The first block column's only nonzero block is H_m, which implies that H_m has no zero columns. Similarly, the last block column's only nonzero block is H_0, which implies that H_0 has no zero columns. Hence, $\mathrm{wt}(H_0) \geq n_v$ and $\mathrm{wt}(H_m) \geq n_v$. □

Lemma 4.4. *Given an SC-LDPC code $\mathcal{C}$ with cycle-free window codes, memory m, window size W, and $d_{\mathcal{W}} \geq 2$,*

$$n_v \leq \frac{Wn_c - 1}{W - m}.$$

In addition, assuming that $W \geq m+1$,

$$m \geq \frac{n_v - n_c + 1}{n_c}.$$

Proof. By Lemma 4.2, $\mathrm{wt}(H_{\mathcal{W}}) \leq (W+m)n_v + Wn_c - 1$. By Lemma 4.3, $\mathrm{wt}(H_0) \geq n_v$ and $\mathrm{wt}(H_m) \geq n_v$. Because each block row of $H_{\mathcal{W}}$ has exactly one copy each of H_0 and H_m, and $H_{\mathcal{W}}$ has W block rows, this implies that $2Wn_v \leq \mathrm{wt}(H_{\mathcal{W}})$. Putting these arguments together yields the bound

$$2Wn_v \leq (W+m)n_v + Wn_c - 1$$

which simplifies to $n_v \leq \frac{Wn_c-1}{W-m}$. Solving instead for W and assuming $W \geq m+1$, we obtain $\frac{mn_v-1}{n_v-n_c} \geq W \geq m+1$. Solving for m yields $m \geq \frac{n_v-n_v+1}{n_c}$. □

Corollary 4.1. *Given an SC-LDPC code $\mathcal{C}$ with cycle-free window codes, memory m, window size $W \geq m+1$, and $d_{\mathcal{W}} \geq 2$, the maximum design rate, $R_{\max}$, of the base matrix is*

$$R_{\max} = 1 - \frac{(W-m)n_v + 1}{Wn_v}.$$

In particular, when $m = 1$, $R \leq 1 - \frac{n_v+1}{2n_v} < \frac{1}{2}$.

Proof. By Lemma 4.4, $n_c \geq \frac{(W-m)n_v+1}{W}$. The design rate, R, is $R = 1 - \frac{n_c}{n_v} \leq 1 - \frac{(W-m)n_v+1}{Wn_v}$. Notice we can rewrite this upper bound as $R \leq \frac{mn_v-1}{Wn_v}$. Because $W \geq m+1$, it is clear that the rate is maximized when $W = m+1$. In other words, $R \leq \frac{mn_v-1}{(m+1)n_v}$. Equivalently,

$$R \leq \frac{(W-1)n_v - 1}{Wn_v}.$$

In particular, when $m = 1$, $R \leq \frac{n_v-1}{2n_v} < \frac{1}{2}$. □

We now strengthen this result to show that $R = 1 - \frac{n_v+1}{2n_v}$ is the maximum rate of any SC-LDPC code that has cycle-free window codes with $d_{\mathcal{W}} \geq 2$ and base matrix of size $n_c \times n_v$. In particular, this bound is independent of both the memory and the window length of the code.

Theorem 4.2. *Let $\mathcal{C}$ be an SC-LDPC code with cycle-free window codes, $d_{\mathcal{W}} \geq 2$, memory m, and window length $W \geq m+1$. Then*

(1) *$n_v \leq 2n_c - 1$ and*
(2) *the maximum design rate R of the base matrix is at most $R \leq 1 - \frac{n_v+1}{2n_v} < \frac{1}{2}$.*

Proof. In order for a matrix to correspond to a cycle-free Tanner graph, any submatrix (subgraph) must also be cycle-free. Because we $W \geq m+1$, $H_{\mathcal{W}}$ has at least one block column with both H_0 and H_m as sub-blocks. Hence, the matrix $M = \begin{bmatrix} H_0 \\ H_m \end{bmatrix}$ is a submatrix of $H_{\mathcal{W}}$, and so must be cycle-free. The submatrix M has n_v columns and $2n_c$ rows. Because it is cycle-free, $\mathrm{wt}(M) \leq n_v + 2n_c - 1$. By Lemma 4.3, $\mathrm{wt}(H_0) \geq n_v$ and $\mathrm{wt}(H_m) \geq n_v$. Putting these together this yields the bound $2n_v \leq n_v + 2n_c - 1$, which simplifies to $n_v \leq 2n_c - 1$ and proves (1). Alternatively, we see that $n_c \geq \frac{n_v+1}{2}$. Using this, we conclude that the design rate of the base matrix can be at most

$$R = 1 - \frac{n_c}{n_v} \leq 1 - \frac{n_v+1}{2n_v} = \frac{n_v-1}{2n_v} < \frac{1}{2},$$

proving (2). □

The next Lemma defines cycle-free matrices that can be used to construct a class of window codes that meet the general bound given in Theorem 4.2. We note that the purpose of this construction is purely to prove existence, and the resulting codes are not intended to be examples of good codes (for some values of n_c and n_v, these codes are not even connected).

Lemma 4.5. *For $n_c \geq 2$, let $n_c + 1 \leq n_v \leq 2n_c - 1$. Define H_m and H_0 to be $n_c \times n_v$ matrices such that*:

$$(H_0)_{i,j} = \begin{cases} 1, & \text{when } j = 2i \text{ or } j = 2i-1, \\ 0, & \text{otherwise.} \end{cases}$$

$$(H_m)_{i,j} = \begin{cases} 1, & \text{when } j = 2i-1 \text{ or } j = 2i-2, \\ 0, & \text{otherwise.} \end{cases}$$

Then the matrix $M = \begin{bmatrix} H_m & H_0 \\ H_m & H_0 \end{bmatrix}$ is cycle-free.

Proof. Note that if we assume n_c is fixed, then it suffices to show the statement for $n_v = 2n_c - 1$ with matrices H_0 and H_m, as for all smaller n_v with matrices H'_m and H'_0, H'_m is a submatrix of H_m, and H'_0 is a submatrix of H_0, and hence $\begin{bmatrix} H'_m & H'_0 \\ H'_m & H'_0 \end{bmatrix}$ is a submatrix of $M = \begin{bmatrix} H_m & H_0 \\ H_m & H_0 \end{bmatrix}$. Hence, we assume that $n_v = 2n_c - 1$.

We will begin by showing that $\begin{bmatrix} H_0 \\ H_m \end{bmatrix}$ is cycle-free. Consider the $2n_c \times 2n_c$ matrix with 1's along both the diagonal and subdiagonal and zeroes elsewhere. This matrix corresponds to a graph that is a path on $2n_c$ vertices, hence is cycle-free. Remove column $2n_c$ to give a $2n_c \times (2n_c - 1)$ matrix (that is also necessarily cycle-free), which we will call A. More succinctly, we define A to be the $2n_c \times (2n_c - 1)$ matrix such that

$$(A)_{ij} = \begin{cases} 1, & \text{when } j = i \text{ or } j = i-1, \\ 0, & \text{otherwise.} \end{cases}$$

We will show that H_0 is exactly the even index rows of A and H_m is exactly the odd index rows of A. If 2ℓ is an even index row of A, then this row is zeroes everywhere except columns 2ℓ and $2\ell - 1$. Referring to how H_0 is defined, it is clear that row 2ℓ of A is the same as row ℓ of H_0. If $2k - 1$ is an odd index row of A, then this row is zeroes everywhere except columns $2k - 1$ and $2k - 1 - 1 = 2k - 2$. Referring to how H_m is defined, it is clear that row $2k - 1$ of A is the same as row k of H_0. This shows that, up to row ordering, A is equivalent to $\begin{bmatrix} H_0 \\ H_m \end{bmatrix}$. Hence, $\begin{bmatrix} H_0 \\ H_m \end{bmatrix}$ is cycle-free.

Because both H_m and H_0 have column weights of 1, adding the two block matrices to form the matrix M only adds degree one variable nodes to the graph corresponding to $\begin{bmatrix} H_0 \\ H_m \end{bmatrix}$. Hence, M is cycle-free. □

Example 4.1. For illustrative purposes, we will give examples of matrices constructed as indicated by Lemma 4.5. Let $n_c = 5$ and $n_v = 9$. Then the matrices are

as follows.

$$H_0 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad H_m = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

To construct matrices with $n_c = 5$ and $6 \leq n_v \leq 8$ instead, we remove the appropriate number of columns from the right side of the matrices.

Lemma 4.6. *Let $\mathcal{C}$ be an SC-LDPC code with cycle-free window codes, $d_{\mathcal{W}} \geq 2$ and memory m. Then the maximum design rate $R = 1 - \frac{n_v+1}{2n_v}$ of H, where $R \neq 0$, is theoretically achievable only for window lengths W such that $m + 1 \leq W \leq 2m$.*

Proof. Suppose H has design rate $R = 1 - \frac{n_v+1}{2n_v} > 0$. Then, necessarily, $n_c = \frac{n_v+1}{2}$, or, equivalently, $n_v = 2n_c - 1$. By Lemma 4.3, $\mathrm{wt}(H_0) \geq n_v$ and $\mathrm{wt}(H_m) \geq n_v$. We will show that the induced subgraph corresponding to the submatrix

$$M = \begin{bmatrix} H_m & H_0 & & \\ & H_m & H_0 & \\ & & H_m & H_0 \end{bmatrix}$$

necessarily has cycles. Because $\mathrm{wt}(M) \geq 6n_v$ by the argument above, using Lemma 4.2 with $W = 3$ and $m = 1$, we can conclude that $6n_v \leq 4n_v + 3n_c - 1$. Rearranging, this requires that $n_v \leq \frac{3n_c-1}{2}$, which is strictly less than $2n_c - 1$ for values n_c greater than 1. (Note that when $n_c = 1$, $n_v = 1$, and $R = 0$, so we need not consider this case.) Hence, M is not cycle-free, and so it cannot appear as a submatrix of $H_{\mathcal{W}}$.

In order for M to not appear as a submatrix of $H_{\mathcal{W}}$, W must be bounded above. The first column in $H_{\mathcal{W}}$ in which H_0 shares a column with H_m is column $m + 1$. This column is in $H_{\mathcal{W}}$ as long as $W \geq m + 1$ (the trivial lower bound on W). This copy of H_m shares a row with a second copy of H_0, which is in column $2m + 1$. Column $2m + 1$ has a copy of H_m provided that $W \geq 2m + 1$. Such a copy of H_0 would guarantee that the matrix M was a submatrix of $H_{\mathcal{W}}$. Hence, this yields the upper bound $W \leq 2m$. □

Corollary 4.2. *Let $\mathcal{C}$ be an SC-LDPC code with cycle-free window codes, $d_{\mathcal{W}} \geq 2$, memory m, and window length W such that $m + 1 \leq W \leq 2m$. Then if n_v is odd, the maximum design rate of the base matrix, $R = 1 - \frac{n_v+1}{2n_v}$, is achievable.*

Proof. Let $n_c \geq 2$ and $n_v = 2n_c - 1$. Define H_0 and H_m as in Lemma 4.5. Define $H_1, \ldots, H_{m-1}$ to be all zero matrices. The resulting SC-LDPC code has base matrix with design rate $R = 1 - \frac{n_c}{n_v} = 1 - \frac{n_v+1}{2n_v}$ and has window code parity check matrix as in Eq. (3.1), except where only H_0 and H_m are nonzero. By inspection, we see that

each H_0 shares a row/column with at most one copy of H_m, and each H_m shares a row/column with at most one copy of H_0. Further, because $m+1 \leq W \leq 2m$, the number of rows of $H_{\mathcal{W}}$ is upper bounded by $2m$. The first column in $H_{\mathcal{W}}$ in which H_m shares a column with H_0 is column $m+1$. This copy of H_0 shares a row with a second copy of H_m, which is in column $2m+1$. Because $W \leq 2m$, the second copy of H_m does not share a column with a second copy of H_0; in other words, the matrix

$$\begin{bmatrix} H_m & H_0 & & \\ & H_m & H_0 & \\ & & H_m & H_0 \end{bmatrix}$$

is not a submatrix of $H_{\mathcal{W}}$, and so the largest induced subgraph of $H_{\mathcal{W}}$ that could possibly have cycles is $\left[\begin{smallmatrix} H_m & H_0 \\ H_m & H_0 \end{smallmatrix}\right]$, which is cycle-free by Lemma 4.5. So $H_{\mathcal{W}}$ has no cycles, and we have constructed an SC-LDPC code with cycle-free window codes with maximum base matrix design rate. □

For sufficiently large window length, any SC-LDPC code with $d_{\mathcal{W}} \geq 2$ has window codes with cycles. The following theorem gives an upper bound on the necessary window length to give cycles in a typical window code.

Theorem 4.3. *Let $\mathcal{C}$ be an SC-LDPC code with memory m, base matrix size $n_c \times n_v$ with $n_c + 1 \leq n_v$, window length W, cycle-free window codes, and $d_{\mathcal{W}} \geq 2$. Then, if we consider a new SC-LDPC code $\mathcal{C}'$, identical to $\mathcal{C}$ except with window length $W' > W$, for sufficiently large W', $\mathcal{W}(\mathcal{C}')$ has a cycle. In particular, for any n_c, n_v, and m, for $W' > m(n_c+1)-1$, $\mathcal{W}(\mathcal{C}')$ has cycles.*

Proof. By Lemma 4.2, $\mathrm{wt}(H_{\mathcal{W}}) \leq (W+m)n_v + Wn_c - 1$. By Lemma 4.3, $\mathrm{wt}(H_0) \geq n_v$ and $\mathrm{wt}(H_m) \geq n_v$. Because there are at least W copies of these matrices in $H_{\mathcal{W}}$, $\mathrm{wt}(H_{\mathcal{W}}) \geq 2Wn_v$. Putting these together yields the inequality $2Wn_v \leq Wn_c + (W+m)n_v - 1$. Rearranging, we see that $n_v \leq \frac{Wn_c-1}{W-m}$. The expression $\frac{Wn_c-1}{W-m}$ has a decreasing horizontal asymptote at n_c as $W \to \infty$. Hence, there exists a W' such that $\frac{W'n_c-1}{W'-m} < n_c+1$. Rewriting this inequality, we obtain $W' > m(n_c+1)-1$. For this W', $n_v < n_c+1$, a contradiction. So, for sufficiently large W', the standard window code cannot be cycle-free (and hence must have a cycle). □

An immediate corollary of Theorem 4.3 is that, for sufficiently large coupling length, any SC-LDPC code with cycle-free window codes has cycles.

Corollary 4.3. *Let $\mathcal{C}$ be an SC-LDPC code with memory m, base matrix size $n_c \times n_v$ with $n_c+1 \leq n_v$, cycle-free window codes, and $d_{\mathcal{W}} \geq 2$. Then for sufficiently large coupling length, the SC-LDPC code has a cycle.*

Proof. By the theorem, there exists a window length W such that a standard window code of that length would have a cycle. Take the coupling length to be at least W. Then the SC-LDPC code has a cycle. □

In other words, this result demonstrates one can have cycle-free window codes within an SC-LDPC code with cycles.

We conclude this section by providing a small but nontrivial example of a cycle-free window code for illustrative purposes. Before the example, however, we will discuss a few additional considerations when constructing a cycle-free window code. The graph-theoretic Lemma 4.1 tells us that a cycle-free graph $G = (V, E)$ is connected if and only if $\#V = \#E - 1$. The result given by Lemma 4.2 is a bound; if we want the Tanner graph of our window code to be connected, we need equality, i.e. we need that

$$\mathrm{wt}(H_{\mathcal{W}}) = (W + m)n_v + Wn_c - 1.$$

Having the smallest possible number of ones in H_0 and H_m gives the largest rate and the most flexibility with construction. Hence, it is reasonable to let $\mathrm{wt}(H_0) = \mathrm{wt}(H_m) = n_v$. Thus, when designing a connected, cycle-free window code, we are seeking a set of parameters n_c, n_v, m, and W and a set of matrix weights $\{\mathrm{wt}(H_i)\}_{i=1}^{m-1}$ that satisfy:

$$\begin{aligned}(W + m)n_v + Wn_c - 1 &= \mathrm{wt}(H_{\mathcal{W}}) \\ &= W(\mathrm{wt}(H_0) + \mathrm{wt}(H_1) + \cdots + \mathrm{wt}(H_{m-1}) + \mathrm{wt}(H_m)) \\ &= W(n_v + \mathrm{wt}(H_1) + \cdots + \mathrm{wt}(H_{m-1}) + n_v) \\ &= 2Wn_v + W(\mathrm{wt}(H_1) + \cdots + \mathrm{wt}(H_{m-1}))\end{aligned}$$

in addition to the constraints given by Theorem 4.2. An example follows.

Example 4.2. Let $m = 2$, $W = 3$, $n_v = 8$, and $n_c = 6$. Define matrices H_0, H_1, and H_2 as below.

$$H_0 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad H_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$H_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

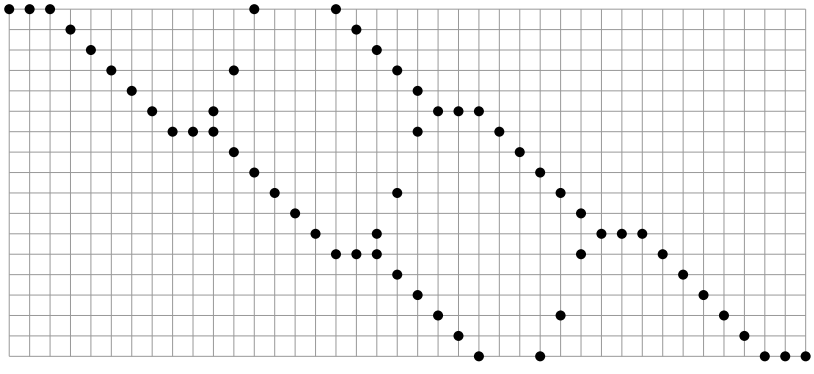

Fig. 3. A representation of the matrix structure of $H_{\mathcal{W}}$ from Example 4.2. Dots represent ones in the parity check matrix; all other entries are zeroes.

The resulting window code has parity check matrix $H_{\mathcal{W}}$ of dimensions 18×40, is connected, and is cycle-free. See Fig. 3 for a visual representation of $H_{\mathcal{W}}$. The maximum weight of a cycle-free matrix of this size is $18 + 40 - 1 = 57$, and we can calculate that $\mathrm{wt}(H_{\mathcal{W}}) = 19 \cdot 3 = 57$, hence, this matrix has the maximum number of nonzero entries for a cycle-free window code and its Tanner graph is connected. Further, $d_{\mathcal{W}} = 2$. By Theorem 4.1, using $n = 40$ and $k = 40 - 18 = 22$, $d_{\mathcal{W}} \leq \lfloor \frac{40}{22+1} \rfloor + \lfloor \frac{40+1}{22+1} \rfloor = 2$, so this code has maximum minimum distance for a cycle-free code of its dimensions.

We note that while the examples given in this section correspond to base matrices with double edges, it is possible to find examples of SC-LDPC codes with cycle-free window codes whose base matrices have no double edges.

4.2. *Row weight constraint*

For implementation purposes, the degree of all check nodes in an SC-LDPC code should be at least 2 [12]. In a tailbiting SC-LDPC code, matrices H_0 and H_m occur together in each block row, and so as long as each of H_0 and H_m has row weight at least 1, the corresponding tailbiting SC-LDPC code minimum has check node degree of at least 2. However, in a terminated SC-LDPC code, the first m block rows have no copy of H_m and the last m block rows have no copy of H_0. In particular, the first block row contains only H_0 and the last block row contains only H_m. Hence, to guarantee that the minimum check node degree of a terminated SC-LDPC code is at least 2, we must have row weights of at least 2 in both H_0 and H_m. In this section, we will show that this constraint is not realizable for terminated SC-LDPC codes with cycle-free window codes.

Lemma 4.7. *If $\mathcal{C}$ is a terminated SC-LDPC code with cycle-free window codes, memory m, window size W, $d_{\mathcal{W}} \geq 2$, and $\min \deg(c) \geq 2$, then $\mathrm{wt}(H_0) \geq 2n_c$ and $\mathrm{wt}(H_m) \geq 2n_c$.*

Proof. By Lemma 4.3, $\mathrm{wt}(H_0) \geq n_v$ and $\mathrm{wt}(H_m) \geq n_v$. Because both H_0 and H_m appear in block rows alone, for $\deg(c) \geq 2$ for all check nodes, each row of H_0 and H_m must have row weight at least 2. Hence, $\mathrm{wt}(H_0) \geq 2n_c$ and $\mathrm{wt}(H_m) \geq 2n_c$. By Theorem 4.2, $n_v \leq 2n_c - 1$. It follows that

$$\mathrm{wt}(H_0) \geq 2n_c > 2n_c - 1 \geq n_v$$

and similarly for $\mathrm{wt}(H_m)$. □

Theorem 4.4. *Let $\mathcal{C}$ be a terminated SC-LDPC code with memory m, window size $W \geq m+1$, $d_{\mathcal{W}} \geq 2$, and $\min \deg(c) \geq 2$. Then any typical window code of $\mathcal{C}$ has at least one cycle.*

Proof. Let $\mathcal{W}$ be a typical window code of $\mathcal{C}$ and $H_{\mathcal{W}}$ its parity check matrix. By Lemma 4.7, $\mathrm{wt}(H_0) \geq 2n_c$ and $\mathrm{wt}(H_m) \geq 2n_c$. In order for $H_{\mathcal{W}}$ to be cycle-free, any submatrix of $H_{\mathcal{W}}$ must also be cycle-free. In particular, because $W \geq m+1$, there is a block column with both H_0 and H_m, and so $M = \left[\begin{smallmatrix} H_0 \\ H_m \end{smallmatrix}\right]$ must be cycle-free. By Lemma 4.1, this requires $\mathrm{wt}(M) \leq n_v + 2n_c - 1$. But

$$\mathrm{wt}(M) = \mathrm{wt}(H_0) + \mathrm{wt}(H_m) \geq 2n_c + 2n_c \geq 2n_c + n_v + 1 > n_v + 2n_c - 1.$$

Hence, any typical window code of $\mathcal{C}$ is not cycle-free. □

This result means that terminated SC-LDPC codes with cycle-free (typical) window codes are likely not usable in practice.

5. Window Code Distance Parameters

In this section, we consider the fundamental question of the possible minimum distances of the window code of a spatially coupled code. Without making additional assumptions about the structure of an SC-LDPC code, we can make some observations about the parameters of the window code.

Lemma 5.1. *Let $\mathcal{C}$ be an $[n,k,d]$ SC-LDPC code and let $\mathcal{W}(\mathcal{C})$ be a $[n_{\mathcal{W}}, k_{\mathcal{W}}, d_{\mathcal{W}}]$ typical window code of $\mathcal{C}$. Then $d \geq d_{\mathcal{W}}$.*

Proof. Let $\mathcal{C}$ and $\mathcal{W}(\mathcal{C})$ be as given and let $\mathcal{H}$ be the parity-check matrix of $\mathcal{C}$ and $H_{\mathcal{W}}$ the parity-check matrix of $\mathcal{W}(\mathcal{C})$. Let A be a set of linearly dependent columns in $\mathcal{H}$. Necessarily, $\#A \geq d$. Because $H_{\mathcal{W}}$ is a submatrix of $\mathcal{H}$, any set of linearly dependent columns in $\mathcal{H}$ is also linearly dependent in $H_{\mathcal{W}}$. So $\#A \geq d \geq d_{\mathcal{W}}$. □

Recall that a code is MDS if it meets the Singleton Bound. In other words, an $[n,k,d]$ code is MDS if $d = n-k+1$. We will both consider the question of creating MDS window codes and more general distance questions. We begin by considering a general distance bound on the minimum distance $d_{\mathcal{W}}$ of a window code of an SC-LDPC code, then consider the existence and properties of SC-LDPC codes with MDS window codes.

Theorem 5.1. *Suppose $\mathcal{C}$ is an SC-LDPC code and let $\mathcal{W}$ be a typical window code of $\mathcal{C}$. If* $\operatorname{rank} H_i < n_v$ *for either* $i \in \{0, m\}$, *then*

$$d_{\mathcal{W}} \leq \min_{i\in\{0,m\}} \operatorname{rank} H_i + 1 \leq n_c + 1.$$

Proof. Recall that the minimum distance of a code is the size of the smallest set of linearly dependent columns in its parity check matrix. Because H_m (respectively, H_0) is the only nonzero block matrix in the first (respectively, last) block column of $H_{\mathcal{W}}$, any set of linearly dependent columns of H_m (respectively, H_0) is also a set of linearly dependent columns in $H_{\mathcal{W}}$.

If $\operatorname{rank} H_m < n_v$, the nullspace of H_m has positive dimension, so H_m has some set of linearly dependent columns. Hence, $d_{\mathcal{W}} \leq d_{\min}(\mathcal{C}(H_m))$. Similarly, if $\operatorname{rank} H_0 < n_v$, the nullspace of H_0 has positive dimension, so H_0 has some set of linearly dependent columns. Hence, $d_{\mathcal{W}} \leq d_{\min}(\mathcal{C}(H_0))$.

By the Singleton Bound, for $i \in \{0, m\}$,

$$d_{\min}(\mathcal{C}(H_i)) \leq n_v - (n_v - \operatorname{rank} H_i) + 1 = \operatorname{rank} H_i + 1 \leq n_c + 1.$$

If $\operatorname{rank} H_i < n_v$ for both $i \in \{0, m\}$, the result is clear, as both $d_{\min}(\mathcal{C}(H_i))$ are upper bounds for $d_{\mathcal{W}}$. Assume without loss of generality that only $\operatorname{rank} H_m < n_v$. Then necessarily $\operatorname{rank} H_0 = n_v$, so $\operatorname{rank} H_m + 1 < \operatorname{rank} H_0 + 1$, and hence the result holds. □

Remark 5.1. Note that when H_i has no redundancy, it has full row rank, so $\operatorname{rank} H_i = n_c$. In this case, the statement in Theorem 5.1 reduces to if $n_c < n_v$, $d_{\mathcal{W}} \leq n_c + 1$. It is worth noting that if $n_c < n_v$, $\operatorname{rank} H_i \leq n_c < n_v$, which also satisfies the assumptions.

We can bound the minimum distance of atypical window codes similarly as in Theorem 5.1.

Theorem 5.2. *Suppose $\mathcal{C}$ is an SC-LDPC code made up of $n_c \times n_v$ base matrices.*

(1) *If* $\operatorname{rank} H_0 < n_v$, *for atypical window codes at positions* $i \in \{1, \ldots, m-1\}$, $d_{\min}(\mathcal{C}_{W_i}) \leq \operatorname{rank} H_0 + 1 \leq n_c + 1$.
(2) *If* $\operatorname{rank} H_m < n_v$, *for atypical window codes at positions* $i \in \{L-W+1, \ldots, L-1\}$, $d_{\min}(\mathcal{C}_{W_i}) \leq \operatorname{rank} H_m + 1 \leq n_c + 1$.

Corollary 5.1. *Suppose $\mathcal{C}$ is an SC-LDPC code with design rate* $1 - \frac{n_c}{n_v} > 0$. *Then for any window code $\mathcal{W}$ of $\mathcal{C}$,* $d_{\min}(\mathcal{W}) \leq n_c + 1$.

Proof. By definition, a code with design rate $1 - \frac{n_c}{n_v}$ is made from $n_c \times n_v$ base matrices. If $1 - \frac{n_c}{n_v} > 0$, then $n_c < n_v$. The conclusion follows from Theorem 5.1. □

The preceding results give bounds on $d_{\mathcal{W}}$ when we assume that $\operatorname{rank} H_i < n_v$ for either $i = 0$ or $i = m$. We now consider the case that $\operatorname{rank} H_i = n_v$ for $i \in \{0, m\}$.

We will see that this condition is necessary for the existence of MDS window codes. Note that if an $n_c \times n_v$ matrix H_i has $\operatorname{rank} H_i = n_v$, necessarily $n_c \geq n_v$.

Lemma 5.2. *Let $\mathcal{C}$ be an SC-LDPC code with memory m, coupling length L, and window width W. If $\mathcal{C}$ has MDS typical window codes, then*

$$\operatorname{rank} H_0 = \operatorname{rank} H_m = n_v.$$

Proof. The parity check matrix $H_{\mathcal{W}}$ of a typical window code of $\mathcal{C}$ has dimensions $Wn_c \times (W+m)n_v$, so the dimension of this code is at least $k \geq (W+m)n_v - Wn_c$. If this code is MDS, then, by the Singleton Bound,

$$d_{\mathcal{W}} = Wn_c - k \leq (W+m)n_v - ((W+m)n_v - Wn_c) + 1 = Wn_c + 1.$$

By way of contradiction, suppose $\operatorname{rank} H_i < n_v$ for either $i \in \{0, m\}$. Then $\operatorname{rank} H_i \leq n_c < n_v$, and so by Theorem 5.1, $d_{\mathcal{W}} \leq n_c + 1$. Then, if $\mathcal{W}(\mathcal{C})$ is MDS, this bound must be met, i.e. $d_{\mathcal{W}} = n_c + 1$. But we also know that $d_{\mathcal{W}} \leq Wn_c + 1$, which implies that $W = 1$. Because SC codes require $W \geq m + 1$ and $m \geq 1$, this is not possible. □

We now use Lemma 5.2 to give an upper bound on the rate of an SC-LDPC code with MDS window codes.

Theorem 5.3. *Let $\mathcal{C}$ be an $[n, k]$ SC-LDPC code with memory m, coupling length L, window width W, and sub-block matrices of size $n_c \times n_v$ that has MDS window codes. Then $\mathcal{C}$ is an $[Ln_v, k]$ code, and*

(1) *if $\mathcal{C}$ is terminated, $k = 0$, and*
(2) *if $\mathcal{C}$ is tailbiting, $k \leq mn_c$.*

Proof. By Lemma 5.2, $\operatorname{rank} H_0 = \operatorname{rank} H_m = n_v \leq n_c$. Importantly, this implies that $H_0\mathbf{x} = \mathbf{0}$ and $H_m\mathbf{x} = \mathbf{0}$ both imply $\mathbf{x} = \mathbf{0}$.

If $\mathcal{C}$ is terminated, its parity check matrix $\mathcal{H}$ has the form:

$$\mathcal{H} = \begin{bmatrix}
H_0 & & & & & & & \\
H_1 & H_0 & & & & & & \\
\vdots & & \ddots & & & & & \\
H_m & H_{m-1} & \dots & H_0 & & & & \\
 & H_m & \dots & H_1 & H_0 & & & \\
 & & \ddots & & & \ddots & & \\
 & & & H_m & \dots & H_1 & H_0 & \\
 & & & & \ddots & & \vdots & \\
 & & & & & H_m & H_{m-1} & \\
 & & & & & & H_m &
\end{bmatrix}.$$

We compute the dimension of $\mathcal{C}$ by finding the dimension of the nullspace of $\mathcal{H}$. Let $\mathbf{x}_i$ for $i \in [L]$ be a length n_c vector and let $\mathbf{x} = \mathbf{x}_1\mathbf{x}_2, \ldots, \mathbf{x}_L$ be the concatenation of these vectors. Then the solutions to the equation $\mathcal{H}\mathbf{x}^T = \mathbf{0}$ are the solutions to the following system of equations.

$$\begin{array}{cccccccccc}
H_0\mathbf{x}_1^T & & & & & & & & & = \mathbf{0} \\
H_1\mathbf{x}_1^T & +H_0\mathbf{x}_2^T & & & & & & & & = \mathbf{0} \\
\vdots & & \ddots & & & & & & \vdots & \vdots \\
H_m\mathbf{x}_1^T & +H_{m-1}\mathbf{x}_2^T & \ldots & +H_0\mathbf{x}_m^T & & & & & & = \mathbf{0} \\
 & H_m\mathbf{x}_2^T & \ldots & +H_1\mathbf{x}_m^T & +H_0\mathbf{x}_{m+1}^T & & & & & = \mathbf{0} \\
 & & \ddots & & & & \ddots & & \vdots & \vdots \\
 & & & H_m\mathbf{x}_{L-m}^T & \ldots & & +H_1\mathbf{x}_{L-1}^T & +H_0\mathbf{x}_L^T & = \mathbf{0} \\
 & & & & \ddots & & & \vdots & \vdots & \vdots \\
 & & & & & & H_m\mathbf{x}_{L-1}^T & +H_{m-1}\mathbf{x}_L^T & = \mathbf{0} \\
 & & & & & & & H_m\mathbf{x}_L^T & = \mathbf{0}.
\end{array}$$

From the first equation, $H_0\mathbf{x}_1^T = \mathbf{0}$ implies $\mathbf{x}_1 = \mathbf{0}$. From the second equation, $H_1\mathbf{x}_1^T + H_0\mathbf{x}_2^T = \mathbf{0}$ reduces to $H_0\mathbf{x}_2^T = \mathbf{0}$, implying that $\mathbf{x}_2 = \mathbf{0}$. This argument repeats for the first L equations, implying that $\mathbf{x} = \mathbf{0}$. Hence, the dimension of $\mathcal{C}$ when it is terminated is 0. This proves (1).

If $\mathcal{C}$ is tailbiting, its parity check matrix $\mathcal{H}$ has dimensions $Ln_c \times Ln_v$ and the form:

$$\mathcal{H} = \begin{bmatrix}
H_0 & & & & H_m & \ldots & H_1 \\
H_1 & H_0 & & & & \ddots & \vdots \\
\vdots & & \ddots & & & & H_m \\
H_m & H_{m-1} & \ldots & H_0 & & & \\
 & H_m & \ldots & H_1 & H_0 & & \\
 & & \ddots & & & \ddots & \\
 & & & H_m & \ldots & H_1 & H_0
\end{bmatrix}.$$

Removing the first mn_c rows of $\mathcal{H}$ results in the $(L-m)n_c \times Ln_v$ matrix

$$H = \begin{bmatrix}
H_m & H_{m-1} & \ldots & H_0 & & & \\
 & H_m & \ldots & H_1 & H_0 & & \\
 & & \ddots & & & \ddots & \\
 & & & H_m & \ldots & H_1 & H_0
\end{bmatrix}.$$

Each block row has submatrix $[H_m\, H_{m-1} \dots H_0\,]$ of dimensions $n_c \times (m+1)n_v$. Because H_m has rank n_v, this submatrix has rank r where $n_v \le r \le \min\{n_c, (m+1)n_v\}$. So there exists a series of row operations such that $[H_m\, H_{m-1} \dots H_0]$ is equivalent to

$$\begin{bmatrix} I_r & A \\ 0_{(n_c-r)\times r} & 0_{(n_c-r)\times[(m+1)n_v-r]} \end{bmatrix},$$

where $0_{j\times\ell}$ is the $j \times \ell$ all zeroes matrix. Perform the same set of row operations to all $(L-m)$ block rows in H. The result is $(L-m)$ pivot rows and $(L-m)(n_c-r)$ zero rows. Because the only difference between $\mathcal{H}$ and H is additional rows, the rank of this matrix is a lower bound for the rank of $\mathcal{H}$, i.e. $(L-m)r \le \operatorname{rank}\mathcal{H}$. By rank-nullity,

$$\begin{aligned} k = \operatorname{nullity}\mathcal{H} &= Ln_v - \operatorname{rank}\mathcal{H} \\ &\le Ln_v - (L-m)r = Ln_v - Lr + mr \\ &\le Ln_v - Ln_v + mr = mr \\ &\le mn_c. \end{aligned}$$

So the dimension of $\mathcal{C}$ when it is tailbiting is at most mn_c. This proves (2). □

The result of Theorem 5.3 is that the maximum dimension of an SC-LDPC code $\mathcal{C}$ with MDS window codes is mn_c. In particular, this dimension does not depend on L. Because the benefit of SC codes is their repetitive structure, one usually chooses a large L. Hence, the increasing length of the code depends on L, but the dimension does not. This means that SC-LDPC codes with MDS window codes have asymptotically bad rates. This is captured in the following corollary.

Corollary 5.2. *If $\mathcal{C}$ is an $[Ln_v, k]$ SC-LDPC code with MDS window codes, then* $\lim_{L\to\infty} \frac{k}{Ln_v} = 0$.

We conclude this section by presenting a construction of an infinite family of MDS window codes, to show that such a family exists. Note that the family constructed here is not usable in coding theory, as it will result in rate 0 codes. From computer search results, we strongly suspect that such families are relatively common.

Theorem 5.4. *Let $\alpha \in \mathbb{F}_q$ be an element of multiplicative order $q-1$. Define* $A = \left[\begin{smallmatrix} 1 & 1 \\ 1 & \alpha \end{smallmatrix}\right]$ *and* $B = \left[\begin{smallmatrix} 1 & 1 \\ \alpha^2 & \alpha^3 \end{smallmatrix}\right]$. *Let H be the $(n-2)\times n$ block parity check matrix over*

$\mathbb{F}_q$ *as defined below.*

$$H = \begin{bmatrix} A & B & 0 & \dots & 0 & 0 \\ 0 & A & B & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & A & B \end{bmatrix}_{(n-2)\times n}.$$

Then H is MDS if and only if $q > n$.

Proof. The proof proceeds by induction on n, using the fact that H is MDS if and only if it has full rank, which is true if and only if all $n \times n$ submatrices of H have nonzero determinant. For details, see Appendix A. □

6. Conclusion

In this paper, we introduced the notion of a window code of an SC-LDPC code under windowed decoding and provided bounds on the rates and dimensions of SC-LDPC codes with cycle-free window codes. We also showed that, given sufficiently large coupling length, SC-LDPC codes with cycle-free window codes have cycles. Moreover, we showed that while SC-LDPC codes with MDS window codes exist, such SC-LDPC codes have provably bad rates.

We gave an initial lower bound on the minimum distance of the SC-LDPC code in terms of the minimum distance of the typical window code. We believe we can strengthen this result and hope to include that in the final version of this paper. It remains open to determine the optimal rate/distance tradeoff for a window code that yields good SC-LDPC codes.

Appendix A. Proof of Theorem 5.4

Proof. Let $\alpha \in \mathbb{F}_q$ be an element of multiplicative order $q-1$. Define $A = \left[\begin{smallmatrix} 1 & 1 \\ 1 & \alpha \end{smallmatrix}\right]$ and $B = \left[\begin{smallmatrix} 1 & 1 \\ \alpha^2 & \alpha^3 \end{smallmatrix}\right]$. Throughout, let H_k be the matrix

$$H_k = \begin{bmatrix} A & B & 0 & \dots & 0 & 0 \\ 0 & A & B & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & A & B \end{bmatrix}$$

with k copies of $[A\ B]$. Note that H_k is necessarily a $(2k+2) \times 2k$ matrix. Further, for a matrix M, define $M_{I\times J}$ to be the submatrix of M with rows in index set I and columns in index set J.

For $k = 1$, we compute the determinants of all 2×2 submatrices of H_1.

$$(1)\ \begin{vmatrix} 1 & 1 \\ 1 & \alpha \end{vmatrix} = \alpha - 1, \qquad (4)\ \begin{vmatrix} 1 & 1 \\ \alpha & \alpha^2 \end{vmatrix} = \alpha(\alpha - 1),$$
$$(2)\ \begin{vmatrix} 1 & 1 \\ 1 & \alpha^2 \end{vmatrix} = \alpha^2 - 1, \quad (5)\ \begin{vmatrix} 1 & 1 \\ \alpha & \alpha^3 \end{vmatrix} = \alpha(\alpha^2 - 1),$$
$$(3)\ \begin{vmatrix} 1 & 1 \\ 1 & \alpha^3 \end{vmatrix} = \alpha^3 - 1, \quad (6)\ \begin{vmatrix} 1 & 1 \\ \alpha^2 & \alpha^3 \end{vmatrix} = \alpha^2(\alpha - 1).$$

The matrix H_1 has 4 columns, so we assume $q > 4$. Thus, $\alpha \in \mathbb{F}_q$ has multiplicative order at least 4. Hence, all of the above determinants are nonzero. Because all 2×2 submatrices of H_1 are nonzero, H_1 has full rank, and its associated code is MDS. Note that if $q \leq n$, $\alpha^{q-1} - 1 = 0$, so at least one of the above determinants is zero.

We now use an inductive argument to compute the determinants of all $2k \times 2k$ submatrices of H_k when $k > 1$. Let S be a $2k \times 2k$ submatrix of H_k, $C \subset [2k+2]$ the set of column indices of S, and $R = [2k]$ the set of row indices of S. Each possible submatrix S has all rows of H_k but two columns removed. We consider cases based upon which columns are in C.

If $\{1, 2\} \subset C$, then

$$\det(S) = (\alpha - 1) \det(S_{R \setminus \{1,2\} \times C \setminus \{1,2\}}).$$

Note that $S_{R \setminus \{1,2\} \times C \setminus \{1,2\}}$ is a submatrix of H_{k-1}, and so, by induction, has nonzero determinant for all $q > 2k$. So this also holds for $q > 2k + 2$. Hence, $\det(S) \neq 0$.

If $\{n - 1, n\} \subset C$, then

$$\det(S) = (\alpha^3 - 1) \det(S_{R \setminus \{n-3,n-2\} \times C \setminus \{n-1,n\}}).$$

Note that $S_{R \setminus \{n-3,n-2\} \times C \setminus \{n-1,n\}}$ is a submatrix of H_{k-1}, and so, by induction, has nonzero determinant for all $q > 2k$. So this also holds for $q > 2k + 2$. Hence, $\det(S) \neq 0$.

If neither of the two above apply, then the two columns missing from C are one of $\{1, 2\}$ and one of $\{2k+1, 2k+2\}$. We will show that the following hold. Note that here and for the rest of the argument, we use the fact that $(\alpha^{m-1} + \alpha^{m-2} + \ldots + \alpha + 1) = \frac{\alpha^m - 1}{\alpha - 1}$, as $\alpha \neq 1$. Further, for each of the below, we present two equivalent expressions to make later calculations more clear.

- If $C = [2k + 2] \setminus \{2, 2k + 2\}$,

$$\begin{aligned} \det(S) &= (-1)^{k-1}(\alpha - 1)^{k-1}(\alpha^{2k} - 1) \\ &= (-1)^{k-1}(\alpha - 1)^k(\alpha^{2k-1} + \alpha^{2k-2} + \cdots + \alpha + 1). \end{aligned}$$

- If $C = [2k + 2] \setminus \{2, 2k + 1\}$,

$$\begin{aligned} \det(S) &= (-1)^{k-1}(\alpha - 1)^{k-1}(\alpha^{2k+1} - 1) \\ &= (-1)^{k-1}(\alpha - 1)^k(\alpha^{2k} + \alpha^{2k-1} + \cdots + \alpha + 1). \end{aligned}$$

- If $C = [2k+2]\backslash\{1, 2k+2\}$,

$$\begin{aligned}\det(S) &= (-1)^{k-1}\alpha(\alpha-1)^{k-1}(\alpha^{2k-1}-1)\\ &= (-1)^{k-1}\alpha(\alpha-1)^{k}(\alpha^{2k-2}+\alpha^{2k-3}+\cdots+\alpha+1).\end{aligned}$$

- If $C = [2k+2]\backslash\{1, 2k+1\}$,

$$\begin{aligned}\det(S) &= (-1)^{k-1}\alpha(\alpha-1)^{k-1}(\alpha^{2k}-1)\\ &= (-1)^{k-1}\alpha(\alpha-1)^{k}(\alpha^{2k-1}+\alpha^{2k-2}+\cdots+\alpha+1).\end{aligned}$$

Note that we have already shown the above hold for $k = 1$. These are (2) through (5) in the $k = 1$ determinant calculations, respectively. For $k > 1$,

- If $C = [2k+2]\backslash\{2, 2k+2\}$,

$$\begin{aligned}\det(S) &= \det(S_{R\backslash\{1\}\times C\backslash\{1\}}) - \det(S_{R\backslash\{2\}\times C\backslash\{1\}})\\ &= (\alpha^2\det(S_{R\backslash\{1,2\}\times C\backslash\{1,3\}} - \alpha^3\det(S_{R\backslash\{1,3\}\times C\backslash\{1,4\}}))\\ &\quad - (\det(S_{R\backslash\{1,2\}\times C\backslash\{1,3\}}) + \det(S_{R\backslash\{1,2\}\times C\backslash\{1,4\}}))\\ &= (\alpha^2-1)\det(S_{R\backslash\{1,2\}\times C\backslash\{1,3\}}) - (\alpha^3-1)\det(S_{R\backslash\{1,2\}\times C\backslash\{1,4\}})\\ &= (\alpha-1)[(\alpha+1)\det(S_{R\backslash\{1,2\}\times C\backslash\{1,3\}})\\ &\quad - (\alpha^2+\alpha+1)\det(S_{R\backslash\{1,2\}\times C\backslash\{1,4\}})].\end{aligned}$$

Notice that $S_{R\backslash\{1,2\}\times C\backslash\{1,3\}}$ is the $2(k-1)\times 2(k-1)$ submatrix of H_{k-1} without columns 1 and $2k$, and $S_{R\backslash\{1,2\}\times C\backslash\{1,4\}}$ is the $2(k-1)\times 2(k-1)$ submatrix of H_{k-1} without columns 2 and $2k$. By inductive hypothesis, we then have

$$\begin{aligned}\det(S) &= (\alpha-1)[(\alpha+1)(-1)^{k-2}\alpha(\alpha-1)^{k-1}(\alpha^{2k-4}+\alpha^{2k-5}+\cdots+\alpha+1)\\ &\quad - (\alpha^2+\alpha+1)(-1)^{k-2}(\alpha-1)^{k-1}(\alpha^{2k-3}+\alpha^{2k-4}+\cdots+\alpha+1)]\\ &= (-1)^{k-1}(\alpha-1)^{k}[(\alpha^2+\alpha+1)(\alpha^{2k-3}+\alpha^{2k-4}+\cdots+\alpha+1)\\ &\quad - (\alpha^2+\alpha)(\alpha^{2k-4}+\alpha^{2k-5}+\cdots+\alpha+1)]\\ &= (-1)^{k-1}(\alpha-1)^{k}(\alpha^{2k-1}+\alpha^{2k-2}+\cdots+\alpha+1)\\ &= (-1)^{k-1}(\alpha-1)^{k-1}(\alpha^{2k}-1).\end{aligned}$$

- If $C = [2k+2]\backslash\{2, 2k+1\}$,

$$\begin{aligned}\det(S) &= \det(S_{R\backslash\{1\}\times C\backslash\{1\}}) - \det(S_{R\backslash\{2\}\times C\backslash\{1\}})\\ &= (\alpha^2\det(S_{R\backslash\{1,2\}\times C\backslash\{1,3\}}) - \alpha^3\det(S_{R\backslash\{1,2\}\times C\backslash\{1,4\}}))\end{aligned}$$

$$
\begin{aligned}
&\quad - (\det(S_{R\setminus\{1,2\}\times C\setminus\{1,3\}}) - \det(S_{R\setminus\{1,2\}\times C\setminus\{1,4\}})) \\
&= (\alpha^2-1)\det(S_{R\setminus\{1,2\}\times C\setminus\{1,3\}}) - (\alpha^3-1)\det(S_{R\setminus\{1,2\}\times C\setminus\{1,4\}}) \\
&= (\alpha-1)[(\alpha+1)\det(S_{R\setminus\{1,2\}\times C\setminus\{1,3\}}) \\
&\quad - (\alpha^2+\alpha+1)\det(S_{R\setminus\{1,2\}\times C\setminus\{1,4\}})].
\end{aligned}
$$

Notice that $S_{R\setminus\{1,2\}\times C\setminus\{1,3\}}$ is the $2(k-1)\times 2(k-1)$ submatrix of H_{k-1} without columns 1 and $2k-1$ and $S_{R\setminus\{1,2\}\times C\setminus\{1,4\}}$ is the $2(k-1)\times 2(k-1)$ submatrix of H_{k-1} without columns 2 and $2k-1$. By inductive hypothesis, we then have

$$
\begin{aligned}
\det(S) &= (\alpha-1)[(\alpha+1)(-1)^{k-2}\alpha(\alpha-1)^{k-1}(\alpha^{2k-3}+\alpha^{2k-4}+\cdots+\alpha+1) \\
&\quad - (\alpha^2+\alpha+1)(-1)^{k-2}(\alpha-1)^{k-1}(\alpha^{2k-2}+\alpha^{2k-3}+\cdots+\alpha+1)] \\
&= (-1)^{k-1}(\alpha-1)^k\big[(\alpha^2+\alpha+1)(\alpha^{2k-2}+\alpha^{2k-3}+\cdots+\alpha+1) \\
&\quad - (\alpha^2+\alpha)(\alpha^{2k-3}+\alpha^{2k-4}+\cdots+\alpha+1)] \\
&= (-1)^{k-1}(\alpha-1)^k(\alpha^{2k}+\alpha^{2k-1}+\cdots+\alpha+1) \\
&= (-1)^{k-1}(\alpha-1)^{k-1}(\alpha^{2k+1}-1).
\end{aligned}
$$

- If $C=[2k+2]\setminus\{1,2k+2\}$,

$$
\begin{aligned}
\det(S) &= \det(S_{R\setminus\{1\}\times C\setminus\{1\}}) - \alpha\det(S_{R\setminus\{2\}\times C\setminus\{1\}}) \\
&= (\alpha^2\det(S_{R\setminus\{1,2\}\times C\setminus\{2,3\}}) - \alpha^3\det(S_{R\setminus\{1,2\}\times C\setminus\{2,4\}})) \\
&\quad - \alpha(\det(S_{R\setminus\{1,2\}\times C\setminus\{2,3\}}) - \det(S_{R\setminus\{1,2\}\times C\setminus\{2,4\}})) \\
&= (\alpha^2-\alpha)\det(S_{R\setminus\{1,2\}\times C\setminus\{2,3\}}) - (\alpha^3-\alpha)\det(S_{R\setminus\{1,2\}\times C\setminus\{2,4\}}) \\
&= \alpha(\alpha-1)[\det(S_{R\setminus\{1,2\}\times C\setminus\{2,3\}}) - (\alpha+1)\det(S_{R\setminus\{1,2\}\times C\setminus\{2,4\}})].
\end{aligned}
$$

Notice that $S_{R\setminus\{1,2\}\times C\setminus\{2,3\}}$ is the $2(k-1)\times 2(k-1)$ submatrix of H_{k-1} without columns 1 and $S_{R\setminus\{1,2\}\times C\setminus\{2,4\}}$ is the $2(k-1)\times 2(k-1)$ submatrix of H_{k-1} without columns 2 and $2k$. By inductive hypothesis, we then have

$$
\begin{aligned}
\det(S) &= \alpha(\alpha-1)\big[(-1)^{k-2}\alpha(\alpha-1)^{k-1}(\alpha^{2k-4}+\alpha^{2k-5}+\cdots+\alpha+1) \\
&\quad \times(\alpha+1)(-1)^{k-2}(\alpha-1)^{k-1}(\alpha^{2k-3}+\alpha^{2k-4}+\cdots+\alpha+1)] \\
&= (-1)^{k-1}\alpha(\alpha-1)^k[(\alpha+1)(\alpha^{2k-3}+\alpha^{2k-4}+\cdots+\alpha+1) \\
&\quad - \alpha(\alpha^{2k-4}+\alpha^{2k-5}+\cdots+\alpha+1)] \\
&= (-1)^{k-1}\alpha(\alpha-1)^k(\alpha^{2k-2}+\alpha^{2k-3}+\cdots+\alpha+1) \\
&= (-1)^{k-1}\alpha(\alpha-1)^{k-1}(\alpha^{2k-1}-1).
\end{aligned}
$$

- If $C = [2k+2]\backslash\{1, 2k+1\}$,

$$\begin{aligned}
\det(S) &= \det(S_{R\backslash\{1\}\times C\backslash\{2\}}) - \alpha\det(S_{R\backslash\{2\}\times C\backslash\{2\}}) \\
&= (\alpha^2\det(S_{R\backslash\{1,2\}\times C\backslash\{2,3\}}) - \alpha^3\det(S_{R\backslash\{1,2\}\times C\backslash\{2,4\}})) \\
&\quad - \alpha(\det(S_{R\backslash\{1,2\}\times C\backslash\{2,3\}}) - \det(S_{R\backslash\{1,2\}\times C\backslash\{2,4\}})) \\
&= (\alpha^2-\alpha)\det(S_{R\backslash\{1,2\}\times C\backslash\{2,3\}}) - (\alpha^3-\alpha)\det(S_{R\backslash\{1,2\}\times C\backslash\{2,4\}}) \\
&= \alpha(\alpha-1)\big[\det(S_{R\backslash\{1,2\}\times C\backslash\{2,3\}}) - (\alpha+1)\det(S_{R\backslash\{1,2\}\times C\backslash\{2,4\}})\big].
\end{aligned}$$

Notice that $S_{R\backslash\{1,2\}\times C\backslash\{2,3\}}$ is the $2(k-1)\times 2(k-1)$ submatrix of H_{k-1} without columns 1 and $2k-1$ and $S_{R\backslash\{1,2\}\times C\backslash\{2,4\}}$ is the $2(k-1)\times 2(k-1)$ submatrix of H_{k-1} without columns 2 and $2k-1$. By inductive hypothesis, we then have

$$\begin{aligned}
\det(S) &= \alpha(\alpha-1)\big[(-1)^{k-2}\alpha(\alpha-1)^{k-1}(\alpha^{2k-3}+\alpha^{2k-4}+\cdots+\alpha+1) \\
&\quad - (\alpha+1)(-1)^{k-2}(\alpha-1)^{k-1}(\alpha^{2k-2}+\alpha^{2k-3}+\cdots+\alpha+1)\big] \\
&= (-1)^{k-1}\alpha(\alpha-1)^k\big[(\alpha+1)(\alpha^{2k-2}+\alpha^{2k-3}+\cdots+\alpha+1) \\
&\quad - \alpha(\alpha^{2k-3}+\alpha^{2k-4}+\cdots+\alpha+1)\big] \\
&= (-1)^{k-1}\alpha(\alpha-1)^k(\alpha^{2k-1}+\alpha^{2k-2}+\cdots+\alpha+1) \\
&= (-1)^{k-1}\alpha(\alpha-1)^{k-1}(\alpha^{2k}-1).
\end{aligned}$$

If we assume $q > 2k+2$, then α has multiplicative order at least $2k+2$. Hence, none of the determinants have a multiplicative factor equal to zero for such an α. Because all $2k\times 2k$ submatrices of H_k have full rank, the code associated with H_k is MDS. However, if $q \leq 2k+2$, α has multiplicative order less than $2k+2$, and so at least one factor of these determinants is zero. □

Acknowledgments

The authors would like to thank Joachim Rosenthal for his mentoring and collaboration over the years. In particular, the second author is grateful to have had Joachim Rosenthal as a PhD advisor. The authors would additionally like to thank the anonymous reviewer for their helpful comments that improved the paper. The first author was partially supported by the National Science Foundation grant DMS-1745670. The second author was partially supported by the Simons Foundation, Award #965274.

ORCID

Emily McMillon https://orcid.org/0000-0002-1239-3564

References

[1] A. Beemer, S. Habib, C. A. Kelley and J. Kliewer, A generalized algebraic approach to optimizing SC-LDPC codes, in *55th Annual Allerton Conf. Communication, Control, and Computing* (Allerton, 2017), pp. 672–679.

[2] C. Di, D. Proietti, I. E. Telatar, T. J. Richardson and R. L. Urbanke, Finite-length analysis of low-density parity-check codes on the binary erasure channel, *IEEE Trans. Inf. Theory* **48** (2002) 1570–1579.

[3] L. Dolecek, Z. Zhang, V. Anantharam, M. Wainwright and B. Nikolic, Analysis of absorbing sets for array-based LDPC codes, in *IEEE Int. Conf. Communications* (Glasgow, Scotland, 2007), pp. 6261–6268.

[4] T. Etzion, A. Trachtenberg and A. Vardy, Which codes have cycle-free Tanner graphs? *IEEE Trans. Inf. Theory* **45** (1999) 2181–2191.

[5] A. J. Felstrom and K. S. Zigangirov, Time-varying periodic convolutional codes with low-density parity-check matrix, *IEEE Trans. Inf. Theory* **45** (1999) 2181–2191.

[6] 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* **58** (2011) 2303–2320.

[7] A. R. Iyengar, P. H. Siegel, R. L. Urbanke and J. K. Wolf, Windowed decoding of spatially coupled codes, *IEEE Trans. Inf. Theory* **59** (2012) 2277–2292.

[8] S. Kudekar, T. J. Richardson and R. L. Urbanke, Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC, *IEEE Trans. Inf. Theory* **57** (2011) 803–834.

[9] E. McMillon and C. A. Kelley, Cycle-free windows of SC-LDPC codes, in *Proc. IEEE Int. Symp. Information Theory* (Espoo, Finland, 2022), pp. 390–395.

[10] D. G. Mitchell, L. Dolecek and D. J. Costello, Absorbing set characterization of array-based spatially coupled LDPC codes, in *Proc. IEEE Int. Symp. Information Theory* (Honolulu, HI, USA, 2014), pp. 886–890.

[11] D. G. Mitchell, M. Lentmaier and D. J. Costello, Spatially coupled LDPC codes constructed from protographs, *IEEE Trans. Inf. Theory* **61** (2015) 4866–4889.

[12] S. Mo, L. Chen, D. J. Costello, D. G. Mitchell, R. Smarandache and J. Qiu, Designing protograph-based quasi-cyclic spatially coupled LDPC codes with large girth, *IEEE Trans. Commun.* **68** (2020) 5326–5337.

[13] M. Papaleo, A. R. Iyengar, P. H. Siegel, J. K. Wolf and G. E. Corazza, Windowed erasure decoding of LDPC convolutional codes, in *Proc. IEEE Information Theory Workshop* (Cairo, Egypt, 2010), pp. 1–5.

[14] L. Wei, D. G. Mitchell, T. E. Fuja and D. J. Costello, Design of spatially coupled LDPC codes over GF(q) for windowed decoding, *IEEE Trans. Inf. Theory* **62** (2016) 4781–4800.

[15] N. Wiberg, Codes and decoding on general graphs, PhD thesis, Linkoping University (1996).