Symmetric Product Codes

Henry D. Pfister\textsuperscript{1}, Santosh Emmadi\textsuperscript{2}, and Krishna Narayanan\textsuperscript{2}

\textsuperscript{1}Department of Electrical and Computer Engineering
Duke University

\textsuperscript{2}Department of Electrical and Computer Engineering
Texas A&M University

Coding: From Practice to Theory
Simons Institute
UC Berkeley
Prologue

Let $C$ be an $(n, k, d)$ linear code over $\mathbb{F}$

- generator / parity-check matrix: $G \in \mathbb{F}^{k \times n} / H \in \mathbb{F}^{(n-k) \times n}$
- product code given by $n \times n$ arrays with rows/columns in $C$:
  \[ \mathcal{P} = \{ G^\top U G \mid U \in \mathbb{F}^{k \times k} \} \]

- well-known that $\mathcal{P}$ is an $(n^2, k^2, d^2)$ linear code
Prologue

Let $C$ be an $(n, k, d)$ linear code over $\mathbb{F}$

- generator / parity-check matrix: $G \in \mathbb{F}^{k \times n}$ / $H \in \mathbb{F}^{(n-k) \times n}$
- product code given by $n \times n$ arrays with rows/columns in $C$:

$$\mathcal{P} = \left\{ G^\top U G \mid U \in \mathbb{F}^{k \times k} \right\}$$

- well-known that $\mathcal{P}$ is an $(n^2, k^2, d^2)$ linear code

Let $\mathcal{U}$ be the symmetric subcode of $\mathcal{P}$:

$$\mathcal{U} = \left\{ X \in \mathcal{P} \mid X^\top = X \right\}$$

- if $\text{char}(\mathbb{F}) \neq 2$, then $\mathcal{U} = \left\{ 2^{-1}(X^\top + X) \mid X \in \mathcal{P} \right\}$
- puncturing the lower triangle gives $\binom{n+1}{2}, \binom{k+1}{2}, \binom{d+1}{2}$ code
Product Code

X_{0,0} X_{0,1} X_{0,2} X_{0,3} X_{0,4} X_{0,5} X_{0,6}

X_{1,0} X_{1,1} X_{1,2} X_{1,3} X_{1,4} X_{1,5} X_{1,6}

X_{2,0} X_{2,1} X_{2,2} X_{2,3} X_{2,4} X_{2,5} X_{2,6}

X_{3,0} X_{3,1} X_{3,2} X_{3,3} X_{3,4} X_{3,5} X_{3,6}

X_{4,0} X_{4,1} X_{4,2} X_{4,3} X_{4,4} X_{4,5} X_{4,6}

X_{5,0} X_{5,1} X_{5,2} X_{5,3} X_{5,4} X_{5,5} X_{5,6}

X_{6,0} X_{6,1} X_{6,2} X_{6,3} X_{6,4} X_{6,5} X_{6,6}
Symmetric Subcode

X_{0,0} \quad X_{0,1} \quad X_{0,2} \quad X_{0,3} \quad X_{0,4} \quad X_{0,5} \quad X_{0,6}

X_{0,1} \quad X_{1,1} \quad X_{1,2} \quad X_{1,3} \quad X_{1,4} \quad X_{1,5} \quad X_{1,6}

X_{0,2} \quad X_{1,2} \quad X_{2,2} \quad X_{2,3} \quad X_{2,4} \quad X_{2,5} \quad X_{2,6}

X_{0,3} \quad X_{1,3} \quad X_{2,3} \quad X_{3,3} \quad X_{3,4} \quad X_{3,5} \quad X_{3,6}

X_{0,4} \quad X_{1,4} \quad X_{2,4} \quad X_{3,4} \quad X_{4,4} \quad X_{4,5} \quad X_{4,6}

X_{0,5} \quad X_{1,5} \quad X_{2,5} \quad X_{3,5} \quad X_{4,5} \quad X_{5,5} \quad X_{5,6}

X_{0,6} \quad X_{1,6} \quad X_{2,6} \quad X_{3,6} \quad X_{4,6} \quad X_{5,6} \quad X_{6,6}
Symmetric Product Codes

Prologue (2)

Punctured Symmetric Subcode

$X_{0,0}$ $X_{0,1}$ $X_{0,2}$ $X_{0,3}$ $X_{0,4}$ $X_{0,5}$ $X_{0,6}$

$X_{1,1}$ $X_{1,2}$ $X_{1,3}$ $X_{1,4}$ $X_{1,5}$ $X_{1,6}$

$X_{2,2}$ $X_{2,3}$ $X_{2,4}$ $X_{2,5}$ $X_{2,6}$

$X_{3,3}$ $X_{3,4}$ $X_{3,5}$ $X_{3,6}$

$X_{4,4}$ $X_{4,5}$ $X_{4,6}$

$X_{5,5}$ $X_{5,6}$

$X_{6,6}$
Prologue (3)

- Benefits
  - for moderate $k$ and $n$, length and dimension reduced by $\sim 2$
  - same component code: roughly same rate and half the length
Prologue (3)

- **Benefits**
  - for moderate $k$ and $n$, length and dimension reduced by $\sim 2$
  - same component code: roughly same rate and half the length

- **Drawbacks**
  - minimum distance also drops by $\sim 2$. Can one do better?
Prologue (3)

- **Benefits**
  - for moderate $k$ and $n$, length and dimension reduced by $\sim 2$
  - same component code: roughly same rate and half the length

- **Drawbacks**
  - minimum distance also drops by $\sim 2$. Can one do better?

- Let $\mathcal{V}$ be the anti-symmetric subcode of $\mathcal{P}$:

\[
\mathcal{V} = \left\{ X \in \mathcal{P} \mid X^\top = -X, \, \text{diag}(X) = 0 \right\}
\]

- if $\text{char}(\mathbb{F}) \neq 2$, then $\mathcal{V} = \left\{ 2^{-1}(X^\top - X) \mid X \in \mathcal{P} \right\}$

- Justesen suggested puncturing the lower triangle to get an

\[
\binom{n}{2}, \binom{k}{2}, D \quad \text{Half-Product Code } \mathcal{H}
\]
Prologue (4)

Product Code

\[
\begin{array}{ccccccc}
X_{0,0} & X_{0,1} & X_{0,2} & X_{0,3} & X_{0,4} & X_{0,5} & X_{0,6} \\
X_{1,0} & X_{1,1} & X_{1,2} & X_{1,3} & X_{1,4} & X_{1,5} & X_{1,6} \\
X_{2,0} & X_{2,1} & X_{2,2} & X_{2,3} & X_{2,4} & X_{2,5} & X_{2,6} \\
X_{3,0} & X_{3,1} & X_{3,2} & X_{3,3} & X_{3,4} & X_{3,5} & X_{3,6} \\
X_{4,0} & X_{4,1} & X_{4,2} & X_{4,3} & X_{4,4} & X_{4,5} & X_{4,6} \\
X_{5,0} & X_{5,1} & X_{5,2} & X_{5,3} & X_{5,4} & X_{5,5} & X_{5,6} \\
X_{6,0} & X_{6,1} & X_{6,2} & X_{6,3} & X_{6,4} & X_{6,5} & X_{6,6}
\end{array}
\]
Anti-Symmetric Subcode

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>$X_{0,1}$</th>
<th>$X_{0,2}$</th>
<th>$X_{0,3}$</th>
<th>$X_{0,4}$</th>
<th>$X_{0,5}$</th>
<th>$X_{0,6}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_{0,1}$</td>
<td>0</td>
<td>$X_{1,2}$</td>
<td>$X_{1,3}$</td>
<td>$X_{1,4}$</td>
<td>$X_{1,5}$</td>
<td>$X_{1,6}$</td>
<td></td>
</tr>
<tr>
<td>$X_{0,2}$</td>
<td>$X_{1,2}$</td>
<td>0</td>
<td>$X_{2,3}$</td>
<td>$X_{2,4}$</td>
<td>$X_{2,5}$</td>
<td>$X_{2,6}$</td>
<td></td>
</tr>
<tr>
<td>$X_{0,3}$</td>
<td>$X_{1,3}$</td>
<td>$X_{2,3}$</td>
<td>0</td>
<td>$X_{3,4}$</td>
<td>$X_{3,5}$</td>
<td>$X_{3,6}$</td>
<td></td>
</tr>
<tr>
<td>$X_{0,4}$</td>
<td>$X_{1,4}$</td>
<td>$X_{2,4}$</td>
<td>$X_{3,4}$</td>
<td>0</td>
<td>$X_{4,5}$</td>
<td>$X_{4,6}$</td>
<td></td>
</tr>
<tr>
<td>$X_{0,5}$</td>
<td>$X_{1,5}$</td>
<td>$X_{2,5}$</td>
<td>$X_{3,5}$</td>
<td>$X_{4,5}$</td>
<td>0</td>
<td>$X_{5,6}$</td>
<td></td>
</tr>
<tr>
<td>$X_{0,6}$</td>
<td>$X_{1,6}$</td>
<td>$X_{2,6}$</td>
<td>$X_{3,6}$</td>
<td>$X_{4,6}$</td>
<td>$X_{5,6}$</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Punctured Anti-Symmetric Subcode

\[ \begin{array}{cccccc}
0 & X_{0,1} & X_{0,2} & X_{0,3} & X_{0,4} & X_{0,5} & X_{0,6} \\
0 & X_{1,2} & X_{1,3} & X_{1,4} & X_{1,5} & X_{1,6} \\
0 & X_{2,3} & X_{2,4} & X_{2,5} & X_{2,6} \\
0 & X_{3,4} & X_{3,5} & X_{3,6} \\
0 & X_{4,5} & X_{4,6} \\
0 & X_{5,6} \\
0 &
\end{array} \]
Outline

▶ Background
▶ Applications
▶ Half-Product Codes
▶ Symmetric Product Codes
Background

- Product Codes
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding
Background

- **Product Codes**
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding

Revised block
Background

- **Product Codes**
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding

![Diagram of a product code with row and column decoding](image)
Background

- **Product Codes**
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding

![Diagram of product codes]

Row decoding
Background

- **Product Codes**
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding

![Column decoding diagram]
Background

- **Product Codes**
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding

- Column decoding
Background

- **Product Codes**
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding
Background

- **Product Codes**
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding

Decoding successful
Background

- **Product Codes**
  - introduced by Elias in 1954
  - hard-decision “cascade decoding” by Abramson in 1968
  - “GLDPC” introduced by Tanner in 1981

- Example: 2-error-correcting codes, bounded distance decoding

Or trapped in a stopping set
Applications

- recent interest for high speed optical communication
- focus on 100 Gb/s with 7% redundancy (i.e., $1 - \frac{239}{255} \approx 0.07$)
- high-rate generalized product codes with BCH component codes and iterative algebraic hard-decision
- many designs appeared in ITU 975.1 in 2004
- Justesen recognized the potential in 2010

Decoding

- decoding complexity much lower than comparable LDPC codes
- for hard-decision channels, BER performance is comparable
A Note on Decoding

- Syndrome-Based Iterative Algebraic Decoding
  - Initialization
    - compute and store the syndrome for each row and column
  - Iteration
    - run algebraic decoding on each row using syndromes
    - correct errors by updating the column syndromes
    - run algebraic decoding on each column using syndromes
    - correct errors by updating the row syndromes

- Memory to store syndromes is $2n(n-k) = 2n^2(1-R)$ vs. $n^2$
  - $(1023, 993)$ BCH vs. $n=1023^2$ LDPC: factor 50 less memory
- Well-known trick in industry for many years...
Symmetric Product Codes

What are they?
- subclass of generalized product codes that use symmetry to reduce the block length while using the same component code
- one example, dubbed half-product codes (HPCs) in 2011 by Justesen, based on work by Tanner in 1981
- the minimum distance is also larger than expected

Match the length and rate between product and HPC
- PC is \((n_0^2, k_0^2)\) and HPC is \(\approx (n_1^2/2, k_1^2/2)\)
  - \(n_1 \approx \sqrt{2}n_0, k_1 \approx \sqrt{2}k_0, \text{ and } n_1 - k_1 \approx \sqrt{2}(n_0 - k_0)\)
- HPC component code has \(n\) and \(t\) larger by factor \(\sqrt{2}\)!
Support Sets and Generalized Hamming Weights

- let $\text{supp}(x) \triangleq \{ i \in [n] | [x]_i \neq 0 \}$ denote the support set of $x$

- the 2nd generalized Hamming weight [HKY92] is

$$d_2 = \min_{\substack{x_1,x_2 \in C \setminus \{0\} \atop x_1 \neq x_2}} |\text{supp}(x_1) \cup \text{supp}(x_2)| \geq \lceil 3d_{\min}/2 \rceil$$

- measures minimal total support of two codewords

- Bound: if $d_2$ smaller than $\lceil 3d_{\min}/2 \rceil$, then sum violates $d_{\min}$
Let $\mathcal{V}$ be the anti-symmetric subcode of $\mathcal{P}$
Minimum Distance (2)

- Let $\mathcal{V}$ be the anti-symmetric subcode of $\mathcal{P}$

- For $x_1, x_2 \in \mathcal{C}\setminus\{0\}$, we will show $X = x_1^T x_2 \notin \mathcal{V}$
  - First, note $X \in \mathcal{P}$ because $HX = (Hx_1^T)x_2 = 0$
  - But, $\text{diag}(X) = 0$ for $X \in \mathcal{V}$ and, thus, $[x_1]_i [x_2]_i = 0$ for all $i$
    - implies $\text{supp}(x_1) \cap \text{supp}(x_2) = \emptyset$
    - and $X_{i,j} = [x_1]_i [x_2]_j \neq 0$ implies $X_{j,i} = [x_1]_j [x_2]_i = 0$
  - Thus, $X^T \neq -X$ and $X \notin \mathcal{V}$
Minimum Distance (2)

- Let $\mathcal{V}$ be the anti-symmetric subcode of $\mathcal{P}$

- For $x_1, x_2 \in \mathcal{C}\{0\}$, we will show $X = x_1^T x_2 \notin \mathcal{V}$
  - First, note $X \in \mathcal{P}$ because $HX = (Hx_1^T)x_2 = 0$
  - But, $\text{diag}(X) = 0$ for $X \in \mathcal{V}$ and, thus, $[x_1]_i [x_2]_i = 0$ for all $i$
    - implies $\text{supp}(x_1) \cap \text{supp}(x_2) = \emptyset$
    - and $X_{i,j} = [x_1]_i [x_2]_j \neq 0$ implies $X_{j,i} = [x_1]_j [x_2]_i = 0$
    - Thus, $X^T \neq -X$ and $X \notin \mathcal{V}$

- Thus, no $X \in \mathcal{V}$ where n.z. rows are scalar multiples of a c.w.
No \( X \in \mathcal{V} \) where n.z. rows are scalar multiples of a c.w.
- n.z. codeword in \( \mathcal{V} \) must have \( \geq 2 \) distinct non-zero rows
- Minimum number of n.z. columns is lower bounded by \( d_2 \)
- Likewise, each column must have at least \( d \) non-zero elements
- So, minimum distance of \( \mathcal{V} \) must be \( \geq d_2 d \geq \lceil 3d/2 \rceil d \)
- Puncturing lower triangle gives \( \mathcal{H} \)
  - implies \( D \geq \lceil 3d/2 \rceil d/2 \)
  - Or \( D \geq 3d^2/4 \) if \( d \) even
Minimum Distance (4)

$\mathcal{H}$ is an $(N, K, D)$ code with $N = \binom{n}{2}$, $K = \binom{k}{2}$, and

$$D \geq \begin{cases} \frac{3d^2}{4} & \text{if } d \text{ even} \\ \frac{(3d+1)d}{4} & \text{if } d \mod 4 = 1 \\ \frac{(3d+1)d+2}{4} & \text{if } d \mod 4 = 3 \end{cases}$$

- Also have matching upper bound if $d$ is even and there are minimum distance codewords achieving the minimum for $d^2$.

- **Basic Idea**: Zeros on diagonal prevent standard square pattern codewords. Thus, support in one dimension must contain at least 2 distinct codewords. Thus, there are $d^2$ non-zero rows (or columns) each with weight at least $d$ and $D \geq d^2d$. 
Example: If $C$ is an $(8,4,4)$ extended Hamming code

- then $d = 4$, $d_2 = \lceil 3d/2 \rceil = 6$, and $D \geq 12$
- there exists $x_1, x_2 \in C$ such that $|\text{supp}(x_1) \cup \text{supp}(x_2)| = 6$ and $w(x_1) = w(x_2) = 4$

- Half-product code is a $(28,6,12)$ binary linear code
  - no $(28,6)$ binary linear code with larger $d_{\text{min}}$ exists
Iterative Decoding Analysis (1)

Peeling Decoder for Generalized Product Codes

- received symbols corrected sequentially without mistakes
- for the BEC and, if a genie prevents miscorrection, the BSC
Iterative Decoding Analysis (1)

- Peeling Decoder for Generalized Product Codes
  - received symbols corrected sequentially without mistakes
  - for the BEC and, if a genie prevents miscorrection, the BSC

- Based on “error graph”:
  - vertices are code constraints
  - edges connect code constraints containing same symbol
  - initial observations remove fraction $1 - p$ edges
  - decoder peels any code constraint with $t$ or fewer errors/edges
  - always reaches stopping set after finite number of iterations
Iterative Decoding Analysis (2)

- Asymptotic Results for Half-Product Codes
  - $t$-error-correcting components w/bounded distance decoding
  - complete graph, edges removed i.i.d. prob. $1 - p$

- Assume $n \to \infty$ with fixed $t$ and $p_n = \frac{\lambda}{n}$
  - decoding threshold $\lambda^*$ via $k$-core problem in graph theory
  - observed in 2007 by Justesen and Høholdt
  - thresholds for $t = 2, 3, 4$ are $\lambda^* = 3.35, 5.14, 6.81$
    - information about finite length via $\lambda^* = \lim_{n \to \infty} np_n^*$
Simulation Results (1)

- “Fair comparison” between product and half-product codes
  - can’t match both rate and block length due to numerology
  - we match the rate and let the block lengths differ by < 15%
Simulation Results (1)

- “Fair comparison” between product and half-product codes
  - can’t match both rate and block length due to numerology
  - we match the rate and let the block lengths differ by < 15%

- First Example
  - product code from $(170, 154, 5)$ shortened binary BCH code
    - $(N', K', D') = (28900, 23716, 25)$, rate $\approx 0.82$, $s_{\text{min}} = 9$
  - half-product code from $(255, 231, 7)$ binary BCH code
    - $(N, K, D) = (32385, 26565, 40)$, rate $\approx 0.82$, $s_{\text{min}} = 10$
Simulation Results (1)

- “Fair comparison” between product and half-product codes
  - can’t match both rate and block length due to numerology
  - we match the rate and let the block lengths differ by < 15%

- First Example
  - product code from \((170, 154, 5)\) shortened binary BCH code
    - \((N', K', D') = (28900, 23716, 25), \text{rate } \approx 0.82, \ s_{\text{min}} = 9\)
  - half-product code from \((255, 231, 7)\) binary BCH code
    - \((N, K, D) = (32385, 26565, 40), \text{rate } \approx 0.82, \ s_{\text{min}} = 10\)

- Iterative decoding assuming genie to prevent miscorrection
  - connection to \(k\)-core problem allows “threshold” estimates
  - For the product code, \(p^* \approx 3.35/170 = 0.0197\)
  - For the half-product code, \(p^* \approx 5.14/255 = 0.0201\)
Simulation Results (2)

- DE predicts better HPC threshold because $5.14/3.35 > 3/2$
- Stopping set analysis predicts better HPC error floor
Simulation Results (3)

- Product code from \((383, 356, 7)\) shortened binary BCH code
  - \((146689, 126736, 49)\) code, rate \(\approx 0.86\), \(s_{\text{min}} = 16\)
- Half-product code from \((511, 475, 9)\) binary BCH code
  - \((130305, 112575, 65)\) code, rate \(\approx 0.86\), \(s_{\text{min}} = 15\)
- DE predicts worse HPC threshold because \(6.81/5.14 < 4/3\)
Conclusions

▶ Half-product codes

▶ Length and dimension reduced by half with same component
▶ Normalized minimum distance improved by $3/2$
▶ For same blocklength and rate, one can increase $t$ by $\sqrt{2}$
▶ Changing $t = 2$ to $t = 3$ generally improves performance
▶ More comprehensive simulations are needed

▶ Symmetric product codes (see ITA 2015 paper)

▶ Natural extension to $m$-dimensional product codes
▶ Length and dimension reduced roughly by $m$ factorial
▶ Minimum distance improves
Conclusions

- **Half-product codes**
  - Length and dimension reduced by half with same component
  - Normalized minimum distance improved by $3/2$
  - For same blocklength and rate, one can increase $t$ by $\sqrt{2}$
  - Changing $t = 2$ to $t = 3$ generally improves performance
  - More comprehensive simulations are needed

- **Symmetric product codes (see ITA 2015 paper)**
  - Natural extension to $m$-dimensional product codes
  - Length and dimension reduced roughly by $m$ factorial
  - Minimum distance improves
  - By how much is an open problem...