Multilevel LDPC Lattices with Efficient Encoding and Decoding and a Generalization of Construction D′

Danilo Silva   Paulo R. B. da Silva

Department of Electrical and Electronic Engineering
Federal University of Santa Catarina (UFSC), Brazil
danilo.silva@ufsc.br

Lattice Coding & Crypto Meeting
Imperial College London
London, January 15, 2018

Outline

1. Introduction (background, motivation)
2. Constructions of low-complexity lattices
3. New results
   - Efficient encoding and decoding for Construction D′
   - A generalization of Construction D′
   - Design examples and simulation results
4. Conclusions and open problems

Introduction

Motivation

1. Lattice codes provide a structured solution to achieve the capacity of the point-to-point AWGN channel [Erez-Zamir'04]
    - Goal: achieve capacity with efficient encoding and decoding

Motivation

1. Lattice codes provide a structured solution to achieve the capacity of the point-to-point AWGN channel [Erez-Zamir'04]
   - Goal: achieve capacity with efficient encoding and decoding
   - Solved by polar lattices [Yan-Liu-Ling-Wu'14]

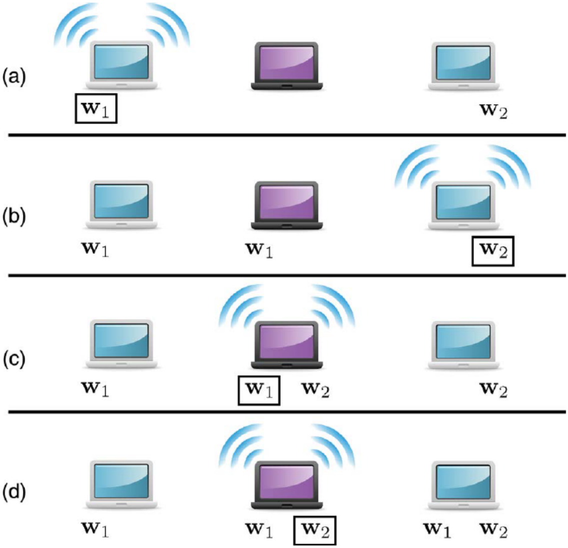
Motivation

1. Lattice codes provide a structured solution to achieve the capacity of the point-to-point AWGN channel [Erez-Zamir'04]
   - Goal: achieve capacity with efficient encoding and decoding
   - Solved by polar lattices [Yan-Liu-Ling-Wu'14]
2. For many network information theory problems, lattice codes can achieve strictly better performance than existing non-structured codes
   - Compute-and-forward for relay networks [Nazer-Gastpar'11]
   - Integer forcing for MIMO systems [Zhan-Nazer-Erez-Gastpar'14]
   - Distributed source coding [Krithivasan-Pradhan'09]
   - Physical-layer security [Ling-Luzzi-Belfiore-Stehlé'14]
   - And more (see Zamir's book)

Example: The Two-Way Relay Channel

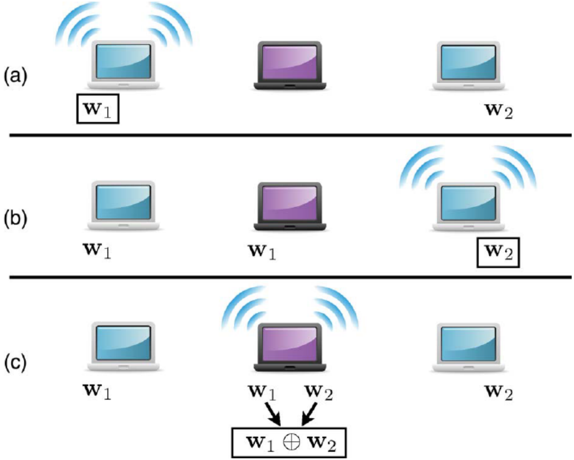
[1] Source: [Nazer-Gastpar'13]

Routing



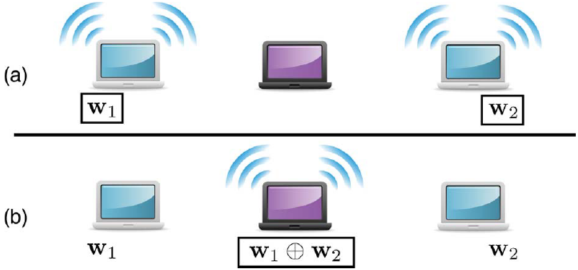
[2] Source: [Nazer-Gastpar'13]

Network Coding



[3] Source: [Nazer-Gastpar'13]

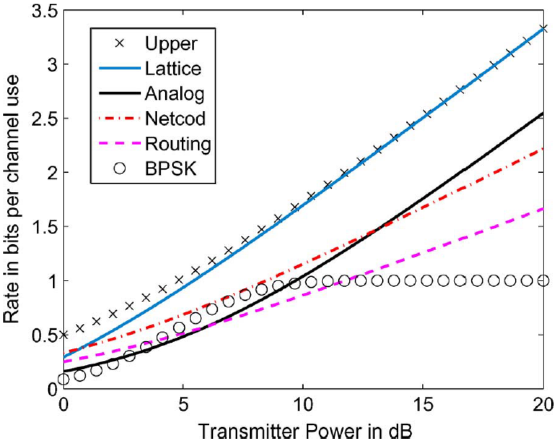
Physical-Layer Network Coding



[4]Source: [Nazer-Gastpar'13]

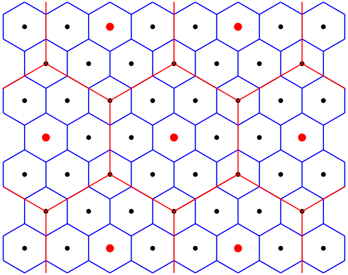
Compute-and-Forward

Physical-Layer Network Coding $+$ Lattices $=$ Compute-and-Forward



[5] Source: [Nazer-Gastpar'13]

Nested Lattice Codes



- If $\Lambda' \subseteq \Lambda$ is a sublattice of Λ with a fundamental region $\mathcal{R}_{\Lambda'}$, then

$$\mathcal{C} = \Lambda \cap \mathcal{R}_{\Lambda'} = \Lambda \bmod \Lambda'$$

  is said to be a nested lattice code

- A decoder that finds the nearest lattice point (ignoring the shaping region) is called a lattice decoder

- Nested lattice codes with lattice decoding are capacity-achieving for the AWGN channel if Λ is AWGN-good and Λ' is quantization-good [EZ'04]

Compute-and-Forward (special case)

- The users transmit $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C} = \Lambda \cap \mathcal{R}_{\Lambda'}$

Compute-and-Forward (special case)

- The users transmit $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C} = \Lambda \cap \mathcal{R}_{\Lambda'}$
- The relay receives

$$\mathbf{y} = \mathbf{c}_1 + \mathbf{c}_2 + \mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$$

Compute-and-Forward (special case)

- The users transmit $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C} = \Lambda \cap \mathcal{R}_{\Lambda'}$
- The relay receives

$$\mathbf{y} = \mathbf{c}_1 + \mathbf{c}_2 + \mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$$

  and wishes to compute

$$\mathbf{c}_3 \triangleq \mathbf{c}_1 + \mathbf{c}_2 \bmod \Lambda' \in \mathcal{C}$$

Compute-and-Forward (special case)

- The users transmit $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C} = \Lambda \cap \mathcal{R}_{\Lambda'}$
- The relay receives

$$\mathbf{y} = \mathbf{c}_1 + \mathbf{c}_2 + \mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$$

  and wishes to compute

$$\mathbf{c}_3 \triangleq \mathbf{c}_1 + \mathbf{c}_2 \bmod \Lambda' \in \mathcal{C}$$

- To do so, it computes

$$\mathbf{y} \bmod \Lambda' = \mathbf{c}_3 + \mathbf{z} \bmod \Lambda'$$

  from which it can then decode $\mathbf{c}_3 \in \mathcal{C}$.

Constructions of Low-Complexity Lattices

Main Problem

How to construct capacity-approaching lattice codes
that admit efficient encoding and decoding?

efficient $\triangleq$ linear or quasi-linear complexity in number of information bits

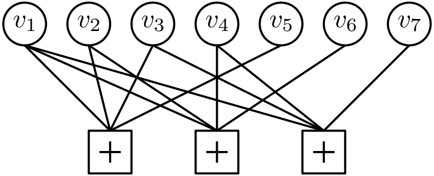
Background on Low-Density Parity-Check Codes

- An LDPC code is a linear code with a sparse parity-check matrix

$$\mathcal{C} = \{\mathbf{x} \in \mathbb{F}_2^n : \mathbf{H}\mathbf{x}^{\mathrm{T}} = \mathbf{0}\}, \qquad \mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$$

- Equivalently represented by a Tanner graph (a bipartite graph, with n variable nodes and m check nodes, whose incidence matrix is $\mathbf{H}$)

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$



- Can be decoded in $O(n)$ by the belief propagation algorithm
- Performance depends largely (but not only) on the degree distribution
- Approaches the BI-AWGN capacity (achieves it if spatially coupled)

Main Approaches

- Low-Density Construction A (LDA) Lattices [di Pietro *et al.*'12]
  - Requires an LDPC code over $\mathbb{Z}_p$ with large p
  - High-complexity decoding: $O(p^2 n)$ with belief propagation

Main Approaches

- Low-Density Construction A (LDA) Lattices [di Pietro *et al.*'12]
  - Requires an LDPC code over $\mathbb{Z}_p$ with large p
  - High-complexity decoding: $O(p^2 n)$ with belief propagation
- Low-Density Lattice Codes (LDLC) [Sommer-Feder-Shalvi'08]
  - Designed directly in $\mathbb{R}^n$ with a sparse parity-check matrix
  - BP decoder must process probability density functions

Main Approaches

- Low-Density Construction A (LDA) Lattices [di Pietro *et al.*'12]
  - Requires an LDPC code over $\mathbb{Z}_p$ with large p
  - High-complexity decoding: $O(p^2 n)$ with belief propagation
- Low-Density Lattice Codes (LDLC) [Sommer-Feder-Shalvi'08]
  - Designed directly in $\mathbb{R}^n$ with a sparse parity-check matrix
  - BP decoder must process probability density functions
- Multilevel Lattices [Forney-Trott-Chung'00]
  - Uses multiple nested binary linear codes
  - Efficient decoding is possible (in principle) using multistage decoding
  - AWGN-good if each component code is capacity-achieving

Multilevel Lattices: Construction D

- Let $\mathcal{C}_0 \subseteq \mathcal{C}_1 \subseteq \cdots \subseteq \mathcal{C}_{L-1} \subseteq \mathbb{Z}_2^n$ be a family of nested linear codes, where each $\mathcal{C}_\ell$ has dimension k_ℓ and generator matrix

$$\mathbf{G}_\ell = \begin{bmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k_\ell} \end{bmatrix} \in \{0,1\}^{k_\ell \times n}$$

- Construction D:

$$\Lambda = \left\{ \sum_{\ell=0}^{L-1} 2^\ell \mathbf{u}_\ell \mathbf{G}_\ell : \mathbf{u}_\ell \in \{0,1\}^{k_\ell},\ 0 \leq \ell < L \right\} + 2^L \mathbb{Z}^n$$

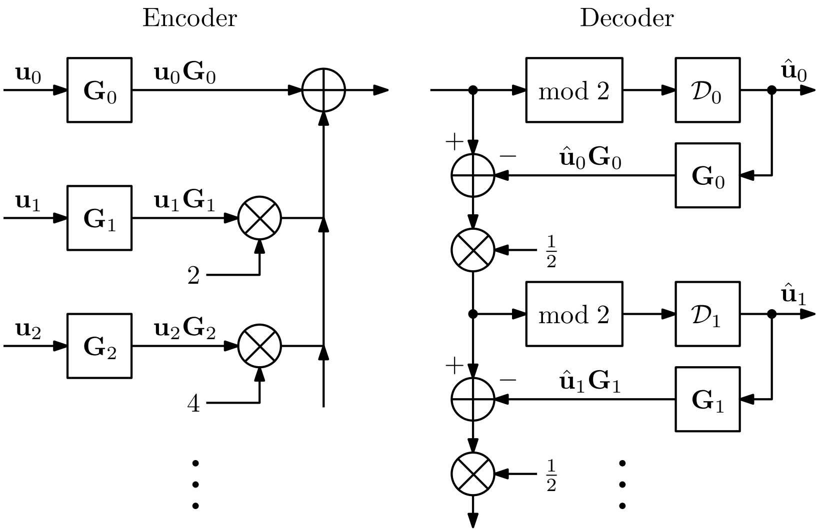
(note that $\mathbf{u}_\ell \mathbf{G}_\ell$ is computed over $\mathbb{Z}$)

- **Remark:** Should not be confused with the "Code Formula"

$$\Gamma = \mathcal{C}_0 + 2\mathcal{C}_1 + \cdots + 2^{L-1}\mathcal{C}_{L-1} + 2^L \mathbb{Z}^n$$

which does not generally produce lattices

Encoding and Multistage Decoding of Construction D



Multilevel Lattices: Construction D′

- Let $\mathcal{C}_0 \subseteq \mathcal{C}_1 \subseteq \cdots \subseteq \mathcal{C}_{L-1} \subseteq \mathbb{Z}_2^n$ be a family of nested linear codes, where each $\mathcal{C}_\ell$ has dimension $n - m_\ell$ and parity-check matrix

$$\mathbf{H}_\ell = \begin{bmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{m_\ell} \end{bmatrix} \in \{0,1\}^{m_\ell \times n}$$

- Construction D′:

$$\Lambda = \{\mathbf{x} \in \mathbb{Z}^n : \mathbf{h}_j \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} \pmod{2^{\ell+1}},\ m_{\ell+1} < j \le m_\ell,\ 0 \le \ell < L\}$$

- Matrix description:

$$\Lambda = \left\{\mathbf{x} \in \mathbb{Z}^n : \mathbf{H}_\ell \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} \pmod{2^{\ell+1}},\ 0 \le \ell < L\right\}$$

Example of Construction D′

For nested codes $\mathcal{C}_0 \subseteq \mathcal{C}_1 \subseteq \mathcal{C}_2 \subseteq \mathbb{Z}_2^4$, let

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad \mathbf{H}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$$

Then

$$\Lambda = \left\{ \mathbf{x} \in \mathbb{Z}^4 : \begin{array}{rl} \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} & \pmod 8 \\ \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} & \pmod 4 \\ \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix} \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} & \pmod 2 \end{array} \right\}$$

or equivalently

$$\Lambda = \left\{ \mathbf{x} \in \mathbb{Z}^4 : \begin{array}{rl} \mathbf{H}_2 \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} & \pmod 8 \\ \mathbf{H}_1 \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} & \pmod 4 \\ \mathbf{H}_0 \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} & \pmod 2 \end{array} \right\}$$

Multilevel Lattices: Previous Work

- Polar Lattices [Yan-Liu-Ling-Wu'14]
  - Based on Construction D
  - Capacity-achieving under MSD
  - Encoding and decoding complexity $O(Ln \log n)$

Multilevel Lattices: Previous Work

- Polar Lattices [Yan-Liu-Ling-Wu'14]
  - Based on Construction D
  - Capacity-achieving under MSD
  - Encoding and decoding complexity $O(Ln \log n)$
- LDPC Lattices [Sadeghi-Banihashemi-Panario'06] [Baik-Chung'08]
  - Based on Construction D′
  - Only joint decoding considered—complexity $O(2^L n)$
  - Encoding complexity not addressed

Multilevel Lattices: Previous Work

- Polar Lattices [Yan-Liu-Ling-Wu'14]
  - Based on Construction D
  - Capacity-achieving under MSD
  - Encoding and decoding complexity $O(Ln \log n)$
- LDPC Lattices [Sadeghi-Banihashemi-Panario'06] [Baik-Chung'08]
  - Based on Construction D′
  - Only joint decoding considered—complexity $O(2^L n)$
  - Encoding complexity not addressed
- Spatially-Coupled LDPC Lattices [Vem-Huang-Narayanan-Pfister'14]
  - AWGN-good under BP MSD
  - Based on Construction D $\implies$ generally dense generator matrices
  - High-complexity encoding and MSD cancellation step

Challenges with Construction D′

- How to encode (efficiently)?
- How to cancel past levels (efficiently) in MSD?
- Nested parity-check matrices:
  - are difficult to design (for non-SC LDPC codes)
  - do not perform well under BP MSD (for non-SC LDPC codes)

New Results

(Submitted to ISIT 2018)

1. A new description of Construction D$'$ that enables sequential encoding
   - Encoding done entirely over the binary field
   - Avoids the need for explicit re-encoding in MSD
   - Existing algorithms for LDPC codes can be easily adapted
     $\Longrightarrow$ encoding and decoding complexity $O(Ln)$
2. A generalization of Construction D$'$ that relaxes the constraints on $\mathbf{H}_\ell$
   - Enlarged design space $\Longrightarrow$ better performance under BP
   - Easier to design (needs only $\mathbf{H}_{L-1}$ and $m_0, \ldots, m_{L-2}$ as inputs)
3. Examples with performance comparable to polar lattices in the power-unconstrained AWGN channel

Efficient Encoding and Decoding for Construction D′

Sequential Encoding

Theorem

Let Λ be a lattice given by Construction D$'$ with matrices $\mathbf{H}_0, \ldots, \mathbf{H}_{L-1}$ and let $\mathcal{C} = \Lambda \cap [0, 2^L)^n$ be a lattice code. Then $\mathcal{C}$ is the set of all possible vectors $\mathbf{c} \in \mathbb{Z}^n$ produced by the following (well-defined) procedure:

1. For $\ell = 0, 1, \ldots, L-1$, choose some vector

$$\mathbf{c}_\ell \in \mathcal{C}_\ell(\mathbf{s}_\ell)$$

where

$$\mathcal{C}_\ell(\mathbf{s}_\ell) \triangleq \left\{\mathbf{x} \in \{0,1\}^n : \mathbf{H}_\ell \mathbf{x}^{\mathrm{T}} \equiv \mathbf{s}_\ell \pmod 2\right\}$$

$$\mathbf{s}_\ell = \frac{-\mathbf{H}_\ell \sum_{i=0}^{\ell-1} 2^i \mathbf{c}_i^{\mathrm{T}}}{2^\ell} \bmod 2 \in \{0,1\}^{m_\ell}$$

2. Compute $\mathbf{c} = \mathbf{c}_0 + 2\mathbf{c}_1 + \cdots + 2^{L-1}\mathbf{c}_{L-1}$

Note: $\mathcal{C}_\ell(\mathbf{s}_\ell)$ is a coset code (linear iff $\mathbf{s}_\ell = 0$)

Example of Sequential Encoding

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad \mathbf{H}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$$

Example of Sequential Encoding

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad \mathbf{H}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$$

1. Choose $\mathbf{c}_0$ satisfying $\mathbf{H}_0\mathbf{c}_0^{\mathrm{T}} \equiv \mathbf{0} \pmod 2$, e.g., $\mathbf{c}_0 = (1, 1, 1, 1)$.

Example of Sequential Encoding

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad \mathbf{H}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$$

1. Choose $\mathbf{c}_0$ satisfying $\mathbf{H}_0\mathbf{c}_0^{\mathrm{T}} \equiv \mathbf{0} \pmod 2$, e.g., $\mathbf{c}_0 = (1, 1, 1, 1)$.
2. Compute
$$\mathbf{s}_1 = -\frac{1}{2}\mathbf{H}_1\mathbf{c}_0^{\mathrm{T}} \bmod 2 = \frac{1}{2}\begin{bmatrix} 4 \\ 2 \end{bmatrix} \bmod 2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
and choose $\mathbf{c}_1$ satisfying $\mathbf{H}_1\mathbf{c}_1^{\mathrm{T}} \equiv \mathbf{s}_1 \pmod 2$, e.g., $\mathbf{c}_1 = (0, 1, 1, 0)$.

Example of Sequential Encoding

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad \mathbf{H}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$$

1. Choose $\mathbf{c}_0$ satisfying $\mathbf{H}_0\mathbf{c}_0^{\mathrm{T}} \equiv \mathbf{0} \pmod 2$, e.g., $\mathbf{c}_0 = (1, 1, 1, 1)$.
2. Compute
$$\mathbf{s}_1 = -\frac{1}{2}\mathbf{H}_1\mathbf{c}_0^{\mathrm{T}} \bmod 2 = \frac{1}{2}\begin{bmatrix} 4 \\ 2 \end{bmatrix} \bmod 2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
and choose $\mathbf{c}_1$ satisfying $\mathbf{H}_1\mathbf{c}_1^{\mathrm{T}} \equiv \mathbf{s}_1 \pmod 2$, e.g., $\mathbf{c}_1 = (0, 1, 1, 0)$.
3. Compute
$$\mathbf{s}_2 = -\frac{1}{4}\mathbf{H}_2(2\mathbf{c}_1^{\mathrm{T}} + \mathbf{c}_0^{\mathrm{T}}) \bmod 2 = 0$$
and choose $\mathbf{c}_2$ satisfying $\mathbf{H}_2\mathbf{c}_2^{\mathrm{T}} \equiv \mathbf{s}_2 \pmod 2$, e.g., $\mathbf{c}_2 = (0, 0, 1, 1)$.

Example of Sequential Encoding

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad \mathbf{H}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$$

1. Choose $\mathbf{c}_0$ satisfying $\mathbf{H}_0\mathbf{c}_0^{\mathrm{T}} \equiv \mathbf{0} \pmod 2$, e.g., $\mathbf{c}_0 = (1, 1, 1, 1)$.
2. Compute
$$\mathbf{s}_1 = -\frac{1}{2}\mathbf{H}_1\mathbf{c}_0^{\mathrm{T}} \bmod 2 = \frac{1}{2}\begin{bmatrix} 4 \\ 2 \end{bmatrix} \bmod 2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
and choose $\mathbf{c}_1$ satisfying $\mathbf{H}_1\mathbf{c}_1^{\mathrm{T}} \equiv \mathbf{s}_1 \pmod 2$, e.g., $\mathbf{c}_1 = (0, 1, 1, 0)$.
3. Compute
$$\mathbf{s}_2 = -\frac{1}{4}\mathbf{H}_2(2\mathbf{c}_1^{\mathrm{T}} + \mathbf{c}_0^{\mathrm{T}}) \bmod 2 = 0$$
and choose $\mathbf{c}_2$ satisfying $\mathbf{H}_2\mathbf{c}_2^{\mathrm{T}} \equiv \mathbf{s}_2 \pmod 2$, e.g., $\mathbf{c}_2 = (0, 0, 1, 1)$.
4. Finally, $\mathbf{c} = \mathbf{c}_0 + 2\mathbf{c}_1 + 4\mathbf{c}_2$
$$= (1, 1, 1, 1) + (0, 2, 2, 0) + (0, 0, 4, 4) = (1, 3, 7, 5).$$

Efficient Systematic Encoding

- Computing each $\mathbf{s}_\ell$ is efficient since $\mathbf{H}_\ell$ is sparse. Thus, the overall complexity will be $O(Ln)$ if encoding each coset code $\mathcal{C}_\ell(\mathbf{s}_\ell)$ is $O(n)$

Efficient Systematic Encoding

- Computing each $\mathbf{s}_\ell$ is efficient since $\mathbf{H}_\ell$ is sparse. Thus, the overall complexity will be $O(Ln)$ if encoding each coset code $\mathcal{C}_\ell(\mathbf{s}_\ell)$ is $O(n)$
- Any coset code can be converted to a linear code:

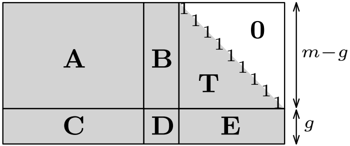
$$\mathbf{H}_\ell \mathbf{c}_\ell^T \equiv \mathbf{s}_\ell \pmod 2 \iff \begin{bmatrix} -\mathbf{s}_\ell & \mathbf{H}_\ell \end{bmatrix} \begin{bmatrix} 1 & \mathbf{c}_\ell \end{bmatrix}^{\mathrm{T}} \equiv \mathbf{0} \pmod 2$$

Efficient Systematic Encoding

- Computing each $\mathbf{s}_\ell$ is efficient since $\mathbf{H}_\ell$ is sparse. Thus, the overall complexity will be $O(Ln)$ if encoding each coset code $\mathcal{C}_\ell(\mathbf{s}_\ell)$ is $O(n)$
- Any coset code can be converted to a linear code:

$$\mathbf{H}_\ell \mathbf{c}_\ell^T \equiv \mathbf{s}_\ell \pmod 2 \iff \begin{bmatrix} -\mathbf{s}_\ell & \mathbf{H}_\ell \end{bmatrix} \begin{bmatrix} 1 & \mathbf{c}_\ell \end{bmatrix}^{\mathrm{T}} \equiv \mathbf{0} \pmod 2$$

- Assume each $\mathbf{H}_\ell$ is of the form required by Richardson-Urbanke's linear-time encoding algorithm:

$\mathbf{H}_\ell =$ 

Since $\mathbf{H}'_\ell = \begin{bmatrix} -\mathbf{s}_\ell & \mathbf{H}_\ell \end{bmatrix}$ has the same structure, the encoding complexity is still $O(n)$ and the overall encoding complexity is $O(Ln)$

Efficient Multistage (Lattice) Decoding

- If $\mathbf{r} = \mathbf{c} + \mathbf{z} \bmod 2^L$:

$$\mathbf{r}_0 \triangleq \mathbf{r} \bmod 2 = \mathbf{c}_0 + \mathbf{z} \bmod 2, \quad \mathbf{c}_0 \in \mathcal{C}_0$$

$$\mathbf{r}_1 \triangleq \frac{\mathbf{r} - \mathbf{c}_0}{2} \bmod 2 = \mathbf{c}_1 + \frac{\mathbf{z}}{2} \bmod 2, \quad \mathbf{c}_1 \in \mathcal{C}_1(\mathbf{s}_1)$$

$$\mathbf{r}_\ell \triangleq \frac{\mathbf{r} - \sum_{i=0}^{\ell-1} 2^i \mathbf{c}_i}{2^\ell} \bmod 2 = \mathbf{c}_\ell + \frac{\mathbf{z}}{2^\ell} \bmod 2, \quad \mathbf{c}_\ell \in \mathcal{C}_\ell(\mathbf{s}_\ell)$$

Efficient Multistage (Lattice) Decoding

- If $\mathbf{r} = \mathbf{c} + \mathbf{z} \bmod 2^L$:

$$\mathbf{r}_0 \triangleq \mathbf{r} \bmod 2 = \mathbf{c}_0 + \mathbf{z} \bmod 2, \quad \mathbf{c}_0 \in \mathcal{C}_0$$
$$\mathbf{r}_1 \triangleq \frac{\mathbf{r} - \mathbf{c}_0}{2} \bmod 2 = \mathbf{c}_1 + \frac{\mathbf{z}}{2} \bmod 2, \quad \mathbf{c}_1 \in \mathcal{C}_1(\mathbf{s}_1)$$
$$\mathbf{r}_\ell \triangleq \frac{\mathbf{r} - \sum_{i=0}^{\ell-1} 2^i \mathbf{c}_i}{2^\ell} \bmod 2 = \mathbf{c}_\ell + \frac{\mathbf{z}}{2^\ell} \bmod 2, \quad \mathbf{c}_\ell \in \mathcal{C}_\ell(\mathbf{s}_\ell)$$

- If each $\mathcal{C}_\ell(\mathbf{s}_\ell)$ admits efficient decoding, then re-encoding is not needed
  - This can be easily accomplished by running BP on

$$\mathbf{H}'_\ell = \begin{bmatrix} -\mathbf{s}_\ell & \mathbf{H}_\ell \end{bmatrix}$$

  with input $\text{LLR}' = \begin{bmatrix} \infty & \text{LLR} \end{bmatrix}$ (corresponding to $\mathbf{c}'_\ell = \begin{bmatrix} 1 & \mathbf{c}_\ell \end{bmatrix}$)

- Overall complexity $O(Ln)$

Consequences of Sequential Encoding

Corollary

Let Λ be a Construction D′ lattice with component codes $\mathcal{C}_0, \ldots, \mathcal{C}_{L-1}$, where each $\mathcal{C}_\ell$ has dimension $n - m_\ell$, and let $\mathcal{C} = \Lambda \cap [0, 2^L)^n$. Then

$$|\mathcal{C}| = |\mathcal{C}_0| \cdot \cdots \cdot |\mathcal{C}_{L-1}|$$

and therefore

$$V(\Lambda) = \frac{V(2^L \mathbb{Z}^n)}{|\mathcal{C}|} = 2^{m_0 + \cdots + m_{L-1}}.$$

- Note: The result in Conway & Sloane's book (Chapter 8, Theorem 14) assumes that *"some rearrangement of $\mathbf{h}_1, \ldots, \mathbf{h}_{m_0}$ forms the rows of an upper triangular matrix"*, which is not required here

A Generalization of Construction D$'$

Revisiting Construction D′

- Construction D′:

$$\Lambda = \left\{ \mathbf{x} \in \mathbb{Z}^n : \mathbf{H}_\ell \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} \pmod{2^{\ell+1}},\ 0 \leq \ell < L \right\}$$

  where $\mathbf{H}_{L-1} \subseteq \cdots \subseteq \mathbf{H}_1 \subseteq \mathbf{H}_0 \subseteq \{0,1\}^{n \times n}$ ($\subseteq$ denotes "submatrix of")

- Can we get rid of this nesting constraint? No, because we would lose:
  - sequential encoding; and thus
  - multistage decoding and
  - the cardinality/volume guarantee

Revisiting Construction D′

- Construction D′:

$$\Lambda = \left\{ \mathbf{x} \in \mathbb{Z}^n : \mathbf{H}_\ell \mathbf{x}^{\mathrm{T}} \equiv \mathbf{0} \pmod{2^{\ell+1}},\ 0 \le \ell < L \right\}$$

  where $\mathbf{H}_{L-1} \subseteq \cdots \subseteq \mathbf{H}_1 \subseteq \mathbf{H}_0 \subseteq \{0,1\}^{n \times n}$ ($\subseteq$ denotes "submatrix of")

- Can we get rid of this nesting constraint? No, because we would lose:
  - sequential encoding; and thus
  - multistage decoding and
  - the cardinality/volume guarantee

- However, sequential encoding requires only the following condition

$$\mathbf{H}_\ell \equiv \mathbf{F}_\ell \mathbf{H}_{\ell-1} \pmod{2^\ell}$$

  - This is needed so that $\mathbf{s}_\ell$ is well-defined
  - The nesting constraint $\mathbf{H}_\ell \subseteq \mathbf{H}_{\ell-1}$ is clearly a special case

Generalized Construction D′

Definition

Let the matrices $\mathbf{H}_\ell \in \mathbb{Z}^{m_\ell \times n}$, $\ell = 0, \ldots, L-1$, be such that

1. $\mathbf{H}_\ell \bmod 2$ is full-rank
2. $\mathbf{H}_\ell \equiv \mathbf{F}_\ell \, \mathbf{H}_{\ell-1} \pmod{2^\ell}$, for some $\mathbf{F}_\ell \in \mathbb{Z}^{m_\ell \times m_{\ell-1}}$

Then the Generalized Construction D′ produces the lattice

$$\Lambda = \left\{ \mathbf{x} \in \mathbb{Z}^n : \mathbf{H}_\ell \mathbf{x}^{\mathrm{T}} \equiv 0 \pmod{2^{\ell+1}}, \ 0 \le \ell \le L-1 \right\}$$

Remarks:

- Clearly a lattice, admits sequential encoding, same cardinality
- Binary codes $\mathcal{C}_\ell$ defined by $\mathbf{H}_\ell \bmod 2$ are still nested ($\mathcal{C}_{\ell-1} \subseteq \mathcal{C}_\ell$)
- $\mathbf{H}_\ell$ need not be binary

Example of Generalized Construction D′

- Let $L = 3$, $n = 4$, let

$$\mathbf{F}_1 = \begin{bmatrix} 2 & 7 & 4 \\ 11 & 9 & 6 \end{bmatrix} \qquad \mathbf{F}_2 = \begin{bmatrix} 3 & 5 \end{bmatrix}$$

be arbitraly chosen integer matrices, and let

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

$$\mathbf{H}_1 = \mathbf{F}_1 \mathbf{H}_0 \bmod 2 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$$\mathbf{H}_2 = \mathbf{F}_2 \mathbf{H}_1 \bmod 4 = \begin{bmatrix} 3 & 1 & 3 & 1 \end{bmatrix}$$

- Generalized Construction D′ produces a lattice Λ and associated lattice code $\mathcal{C} = \Lambda \cap [0, 2^L)^n$ for which $|\mathcal{C}| = 2^{1+2+3}$.

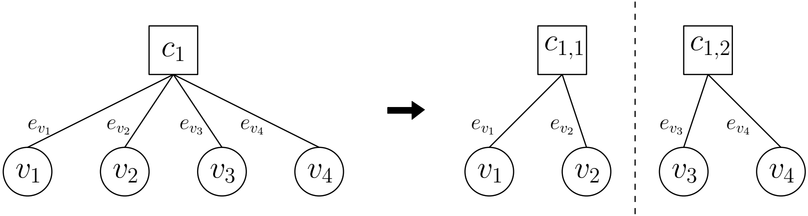
Check Splitting

- One way to produce binary matrices that satisfy

$$\mathbf{H}_\ell = \mathbf{F}_\ell \, \mathbf{H}_{\ell-1} \qquad \text{(exactly, without mod)}$$

  is by splitting rows of $\mathbf{H}_\ell$ (shorter) to produce $\mathbf{H}_{\ell-1}$ (taller)

- This is useful since when designing regular LDPC codes it is best not to increase the column weights (variable-node degrees)



Example of Check Splitting

- Starting with

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

we partition it into

$$\mathbf{H}_1 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

and, in turn, into

$$\mathbf{H}_0 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

- Note that the column weights are preserved and

$$\mathbf{H}_1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \mathbf{H}_0 \quad \text{and} \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 1 \end{bmatrix} \mathbf{H}_1$$

PEG-Based Check Splitting

- We propose two check splitting algorithms based on Progressive Edge Growth (PEG) techniques [Hu *et al.*, 2005]:
  1. PEG-based check splitting: greedily attempts to maximize girth
  2. Triangular PEG-based check splitting: returns a matrix in approximate triangular form, allowing linear-time encoding
- All our design examples are based on the triangular construction

Design Examples and Simulation Results

Power-Unconstrained AWGN Channel

- Channel model:

$$\mathbf{x} \in \Lambda \quad \longrightarrow \quad \mathbf{y} = \mathbf{x} + \mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(0, \sigma^2)$$

- Multilevel partition with multistage decoding [Forney *et al.*, 2000]:

$$\mathbf{x} = \mathbf{c} + \boldsymbol{\lambda}', \quad \mathbf{c} \in \mathcal{C} = \Lambda \cap \mathcal{R}_{\Lambda'}, \quad \boldsymbol{\lambda}' \in \Lambda' = 2^L \mathbb{Z}^n$$

- First, compute

$$\mathbf{r} = \mathbf{y} \bmod \Lambda' = \mathbf{c} + \mathbf{z} \bmod 2^L$$

- Then, decode $\mathbf{c} \in \mathcal{C}$ on the modulo-2^L channel

- Finally, subtract $\mathbf{c}$ from $\mathbf{y}$ and then decode $\boldsymbol{\lambda}' \in \Lambda'$

$$P_e(\Lambda, \sigma^2) \leq P_e(\mathcal{C}, \sigma^2) + P_e(\Lambda', \sigma^2)$$

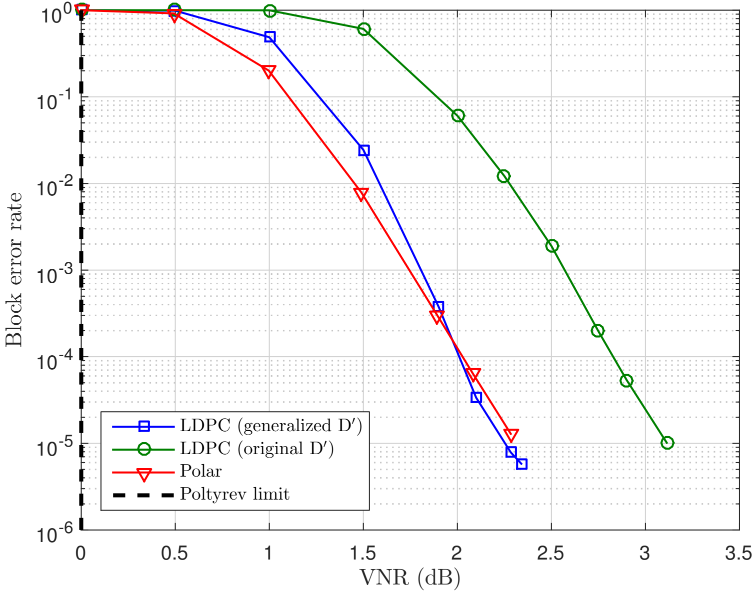
Power-Unconstrained AWGN Channel: Design

- Generalized Construction D′ with $L = 2$ coded levels
- Parameters from [Yan-Liu-Ling-Wu'14]: $n = 1024$, $P_e(\Lambda, \sigma^2) \leq 10^{-5}$
- Equal error probability rule:

$$P_e(\Lambda, \sigma^2) \leq P_e(\mathcal{C}_0, \sigma^2) + P_e(\mathcal{C}_1, (\sigma/2)^2) + P_e(4\mathbb{Z}^n, (\sigma/4)^2)$$

- LDPC component codes:
  - Variable-regular with $d_v = 3$
  - Triangular PEG-based check splitting for linear-time encoding
  - Rates $R_0 = 0.2383$ and $R_1 = 0.9043$
- Comparison with:
  - Polar lattices [Yan-Liu-Ling-Wu'14]
  - (Original) Construction D′ LDPC lattices [Sadeghi *et al.*'06]

Power-Unconstrained AWGN Channel: Results



Power-Constrained AWGN Channel

- Channel model:

$$\mathbf{x} \in \mathcal{X} = (\Lambda + \mathbf{d}) \cap \mathcal{V}(\Lambda') \quad \longrightarrow \quad \mathbf{y} = \mathbf{x} + \mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(0, \sigma^2)$$

  where $\Lambda' = 2^L \mathbb{Z}^n$, and $\mathbf{d} \in \mathbb{R}^n$ is a shift vector (or dither) chosen such that $\mathcal{X}$ lies in a zero-mean 2^L-PAM constellation

- Modulo-lattice transformation for lattice decoding [Erez-Zamir'04]:

$$\mathbf{r} = \alpha \mathbf{y} - \mathbf{d} \bmod \Lambda' = \mathbf{c} + \mathbf{z}_{\text{eff}} \bmod 2^L$$

  gives an equivalent channel with effective noise

$$\mathbf{z}_{\text{eff}} = (\alpha - 1)\mathbf{x} + \alpha \mathbf{z}$$

- Then, decode $\mathbf{c} \in \mathcal{C}$ on the modulo-2^L channel, with σ^2 replaced by

$$\sigma_{\text{eff}}^2 = (\alpha - 1)^2 P + \alpha^2 \sigma^2$$

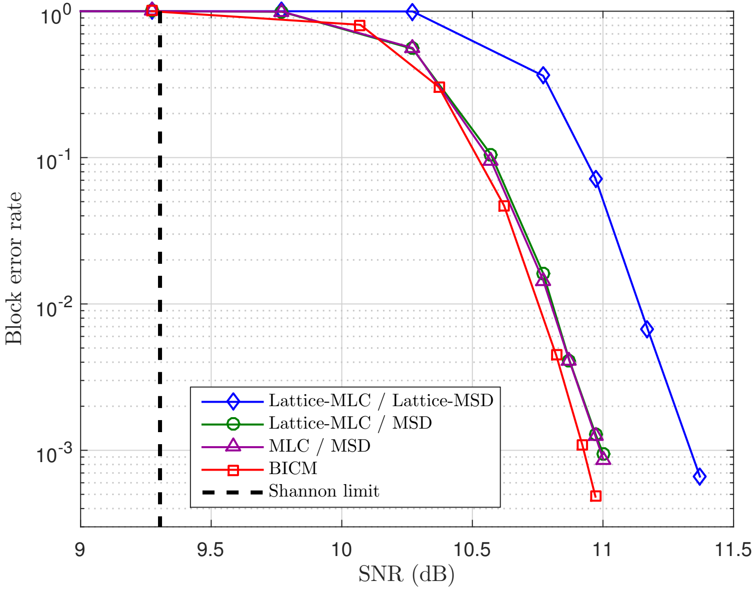
Power-Constrained AWGN Channel: Design

- Generalized Construction D′ with $L = 2$ coded levels (4-PAM modulation)
- Parameters: $n = 2048$, $P_e \leq 10^{-3}$, $R = 1.5$ bits per symbol
- Equal error probability rule:

$$P_e(\Lambda, \sigma^2) \leq P_e(\mathcal{C}_0, \sigma^2) + P_e(\mathcal{C}_1, (\sigma/2)^2)$$

- LDPC component codes:
  - Variable-regular with $d_v = 3$
  - Triangular PEG-based check splitting for linear-time encoding
  - Rates: $R_0 = 0.5244$ and $R_1 = 0.9756$
- Comparison with:
  - Conventional (non-lattice) MLC with conventional (non-lattice) MSD
  - BICM scheme with Gray labeling ($n = 4096$, $R = 3/4$)

Power-Constrained AWGN Channel: Results



Conclusions

Conclusions

- Lattice codes may provide significant gains for network information theory, but their practical implementation is still challenging
- Multilevel lattices are promising since they can be AWGN-good and only require encoding/decoding of binary codes
- Construction D$'$ LDPC lattices admit efficient encoding and decoding and do not require nested matrices (just nested codes)
- Encouraging examples with competitive performance

Open Problems

Ongoing work:

- Include (nested lattice) shaping
- Design irregular LDPC lattices

Open problems:

- Can we prove AWGN-goodness under linear complexity?
- Do quantization-good Construction D/D$'$ lattices exist?
- Is compute-and-forward with probabilistic shaping possible?

Thank You!