

# Shaping QAM Constellations Using Lattices

Brian M. Kurkoski

Japan Advanced Institute of Science and Technology

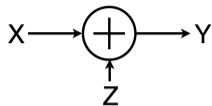
2025 August 26

IEEE 2025 7th Symposium on Future Telecommunication Technologies

Lombok, Indonesia

## AWGN Channel Capacity

The additive-white Gaussian noise (AWGN) channel has input  $X$ , output  $Y$  and noise  $Z \sim \mathcal{N}(0, N)$ .



- ▶ **The most celebrated result in information theory:** The capacity of the AWGN channel in bits per channel use is:

$$C = \frac{1}{2} \log_2 \left( 1 + \frac{P}{N} \right)$$

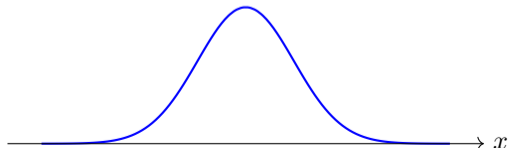
- ▶ The capacity is the highest possible communications rate for a channel.
- ▶ **Key Fact:** The **capacity-achieving input distribution is Gaussian**,  $X \sim \mathcal{N}(0, P)$ .

The channel input  $x_1, x_2, \dots, x_n$  satisfies:

$$\frac{1}{n} \sum_{i=1}^n x_i^2 \leq P,$$

the AWGN power constraint.

Optimal Input Distribution

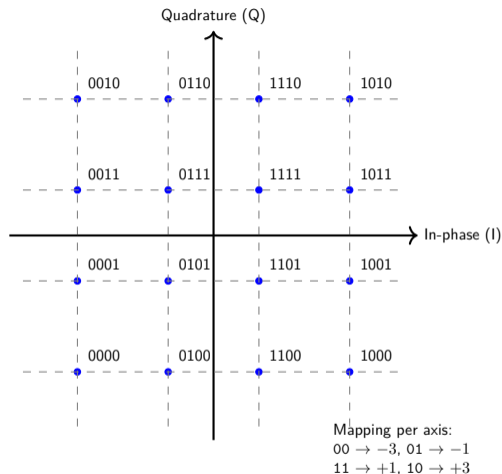


# Quadrature Amplitude Modulation (QAM) Constellations

- ▶ Square  $M$ -QAM:  $M = L^2$  equally likely points on an  $L \times L$  grid (in-phase & quadrature components)
- ▶ Rate  $R = \frac{1}{2} \log M$  bits/dimension.
- ▶ Average symbol energy (i.e., power for unit symbol time):

$$E_s = \frac{(\sqrt{M} - 1)(\sqrt{M} + 1)}{3}.$$

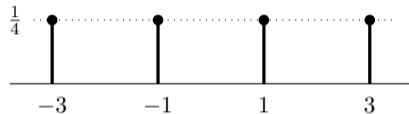
Modulation	$M$	$R$	$E_s$
4-QAM (QPSK)	4	1	2
16-QAM	16	2	10
64-QAM	64	3	42



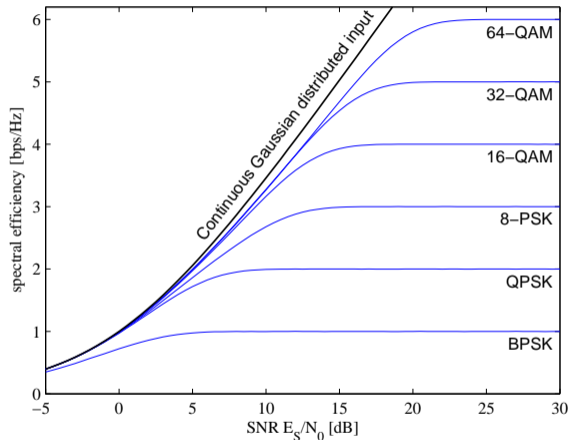
16-QAM with gray mapping

# Finite Constellations vs. Gaussian Input

- ▶ QAM constellations induce a *uniform* input distribution.



- ▶ Figure shows achievable rates are strictly below the Gaussian-input capacity.
- ▶ In the worst case, the QAM loss is 1.53 dB
- ▶ **This motivates shaping** – techniques to approximate the Gaussian input distribution.



Pfletschinger, Stephan. (2005). Multicarrier BICM with adaptive bit-loading and iterative decoding.

# Outline: Shaping QAM Constellations Using Lattices

## 1. Motivation

- ▶ Shaping reduces transmit power while maintaining rate
- ▶ Benefits: efficiency, reduced distortion, lower energy use
- ▶ **Being strongly considered for inclusion in 6G standards**

## 2. Background on Lattices

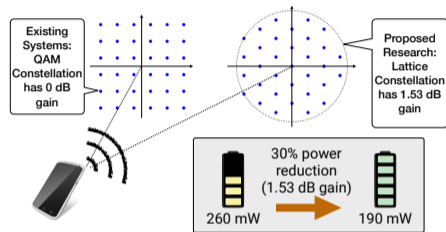
- ▶ Lattices and their applications
- ▶ Lattice quantization and Construction A
- ▶ Shaping gain of convolutional code-based lattices

## 3. Shaping QAM

- ▶ Compatible with existing QAM mappers — adds a shape/deshape operation
- ▶ Shaping is performed using lattices
- ▶ Show shaping of 0.6 dB out of theoretical maximum 1.53 dB

## Part 1: Motivation — Reducing Transmit Power

- ▶ **Shaping Means** Using a non-uniform input distribution to reduce average power
- ▶ **Why shaping?** Finite constellations (e.g., QAM) are non-Gaussian  $\Rightarrow$  require higher  $E_s/N_0$  than Gaussian-input capacity.
- ▶ **Shaping gain:** Approximating a Gaussian input via lattice/probabilistic shaping yields up to  $\approx 1.53$  dB gain (large-constellation limit).
- ▶ **Direct power savings:** Same rate/BER at lower  $E_s \Rightarrow$  reduced TX power (e.g., 30% power reduction shown).

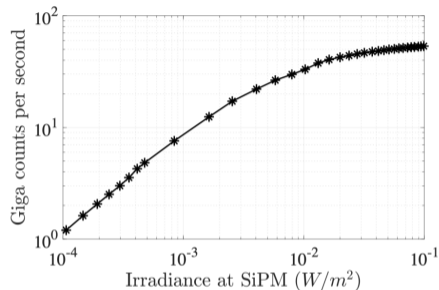


Existing (QAM): 260 mW  
Shaped (lattice): 190 mW

*Note:* The classic shaping limit is 1.53 dB; actual gain depends on constellation size, rate, and implementation.

# Motivation: Operate in the Linear Region of an Optical Receiver

- ▶ Silicon photomultipliers (SiPM) exhibit a **nonlinear response**:
  - ▶ At low input power: response is approximately linear.
  - ▶ At high input power: saturation due to recovery time / dead time.
- ▶ **Motivation for shaping**:
  - ▶ Reduces average transmit power.
  - ▶ Keeps more of the received signal energy within the linear regime.
  - ▶ Improves detection fidelity and SNR.



He, Cuiwei, Zubair Ahmed, and Steve Collins. "Signal pre-equalization in a silicon photomultiplier-based optical OFDM system." IEEE Access 9 (2021).

# Two Routes to a Gaussian-like Input: Probabilistic vs. Geometric Shaping

## Probabilistic Shaping (PS)

- ▶ **Idea:** Use the same QAM alphabet (e.g.  $\pm 3, \pm 1$ ), but change the symbol probabilities to approximate Gaussian
- ▶ **How:** Distribution matcher (Huffman code, CCDM) with systematic ECC.
- ▶ **Pros:**
  - ▶ Fine-grained rate adaptation.
  - ▶ Works with standard BICM.
- ▶ **Cons:**
  - ▶ DM complexity
  - ▶ finite-length rate loss
  - ▶ error propagation possible

## Geometric Shaping (GS)

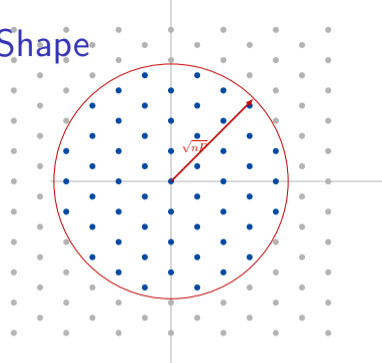
- ▶ **Idea:** Symbol probabilities are uniform, but projection of shape is Gaussian-like.
- ▶ **How:** Non-rectangular constellations; shell mapping; lattice shaping.
- ▶ **Pros:**
  - ▶ Can reduce PAPR / help nonlinear front-ends via geometry.
- ▶ **Cons:**
  - ▶ New labeling/demapper may be needed.

## Geometric Shaping: An $n$ -Ball Is the Ideal Shape

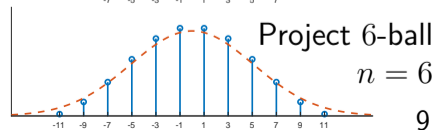
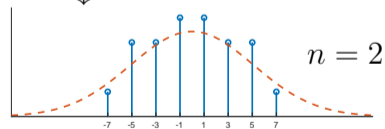
- ▶ AWGN power constraint is an  $n$ -ball:

$$\frac{1}{n} \sum_{i=1}^n x_i^2 \leq P.$$

- ▶ This is ideal shaping region — but what is relationship with Gaussians?
  - ▶  $n = 2$ , projection is nonuniform, but not close to Gaussian
  - ▶  $n = 6$ , looks close to Gaussian but much more shaping gain is possible.
  - ▶  $n \rightarrow \infty$  the projection approaches a Gaussian!
- ▶ Unfortunately, it is hard to map information to codewords when  $n$  gets large  $\rightarrow$  use lattices!

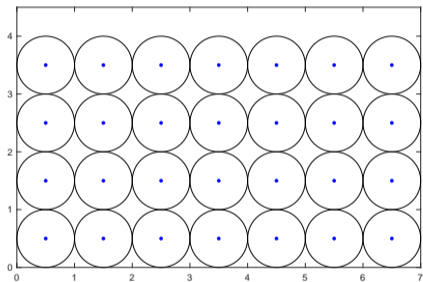


↓ Project to  $x_1$ -axis

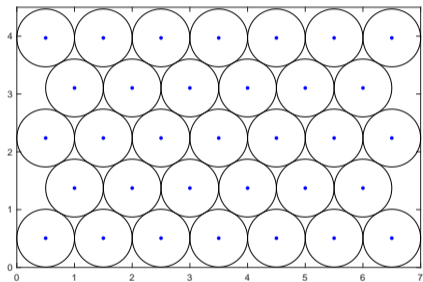


## Part 2: Lattices — Efficient Arrangement of Circles in Two Dimensions

- ▶ Given a large space (like a huge box), what is largest number of balls that can fit?
- ▶ Large space allows us to ignore boundary effects.
- ▶ The answer leads to the most efficient arrangement of balls.



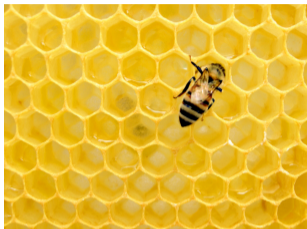
Rectangular



Hexagonal

- ▶ Hexagonal packing is most efficient in 2 dimensions.
- ▶ Ball centers are the lattice points, and all balls have the same size.

## 2D Applications — Efficient Arrangement in a Plane



- ▶ Bees build hexagonal honeycombs.
- ▶ Hexagons maximize honey storage.
- ▶ Bees know hexagons are the best 2-D lattice!



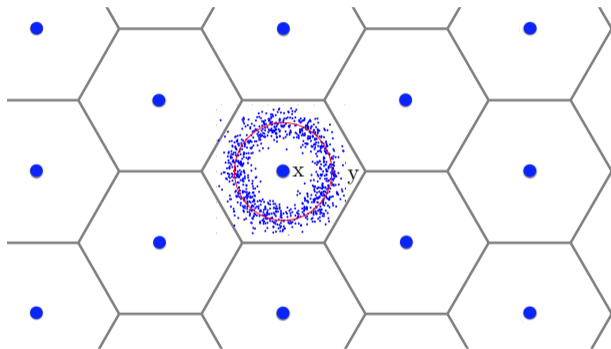
Smartphone has  
rectangular  
arrangement of icons



Smartwatch has a  
small screen – uses  
more efficient  
hexagonal placement  
of icons

## Application — Lattices for Communications

- ▶ Lattice points are codewords.
- ▶ Gaussian noise is spherically symmetric.
- ▶ Efficient arrangement means higher rates.



# Lattices

Lattices are defined in the  $n$ -dimensional Euclidean space  $\mathbb{R}^n$  with vector addition.

## Definition

An  $n$ -dimensional *lattice*  $\Lambda$  is a discrete additive subgroup of  $\mathbb{R}^n$ .

## Linear Algebra Representation

- ▶ A lattice  $\Lambda$  can be spanned by a set of  $n$  linearly independent basis column vectors  $\mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_n$ .
- ▶ A vector  $\mathbf{x}$  is a lattice point if it is a linear combination using integers  $b_i \in \mathbb{Z}$ :

$$\mathbf{x} = \mathbf{g}_1 b_1 + \mathbf{g}_2 b_2 + \dots + \mathbf{g}_n b_n.$$

- ▶ The matrix-form representation is:

$$\mathbf{x} = \mathbf{G}\mathbf{b},$$

where  $\mathbf{b}$  is a vector of integers and  $\mathbf{G}$  is an  $n$ -by- $n$  matrix called a *generator matrix*.

## Example $n = 2$

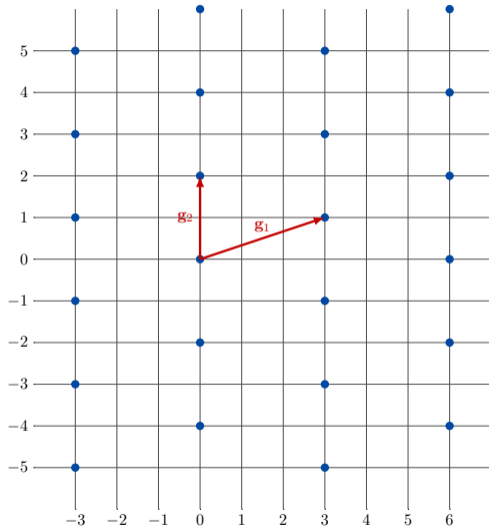
For  $n = 2$ , consider the generator vectors:

$$\mathbf{g}_1 = \begin{bmatrix} 3 \\ 1 \end{bmatrix} \text{ and } \mathbf{g}_2 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

Then any lattice point  $\mathbf{x}$  is:

$$\mathbf{x} = \mathbf{G}\mathbf{b}$$
$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 & 0 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

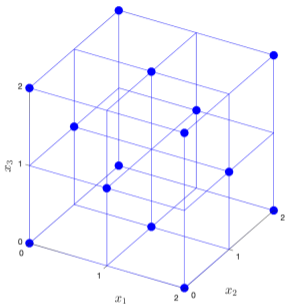
where  $b_1, b_2$  are integers  $\mathbb{Z}$ .



## Example $n = 3$



(a)



(b)

Generator matrix

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 2 \end{bmatrix}$$

(c)

- ▶ In  $n = 3$  dimensions, stacking cannonballs is a packing of spheres.
- ▶ The three representations (a), (b) and (c) are equivalent.
- ▶ Lattices exist in  $n \geq 4$  dimensions, but cannot easily draw pictures.

## Lattice Quantization

### Definition

For any  $\mathbf{y} \in \mathbb{R}^n$ , *nearest-neighbor quantization* is the lattice point  $\mathbf{x} \in \Lambda$  which is closest to  $\mathbf{y}$  in Euclidean distance:

$$\mathbf{x} = Q_{\Lambda}(\mathbf{y}) = \arg \min_{\mathbf{z} \in \Lambda} \|\mathbf{y} - \mathbf{z}\|^2$$

### Definition

The *Voronoi region*  $\mathcal{V}$  is the region of space closer to  $\mathbf{x}$  than to any other point in  $\Lambda$ :

$$\mathcal{V}(\mathbf{x}) = \{\mathbf{u} \in \mathbb{R}^n \mid Q(\mathbf{u}) = \mathbf{x}\}$$

The volume  $V(\Lambda)$  of the Voronoi region  $\mathcal{V}$  for a lattice with generator matrix  $\mathbf{G}$ :

$$V(\Lambda) = |\det(\mathbf{G})|.$$

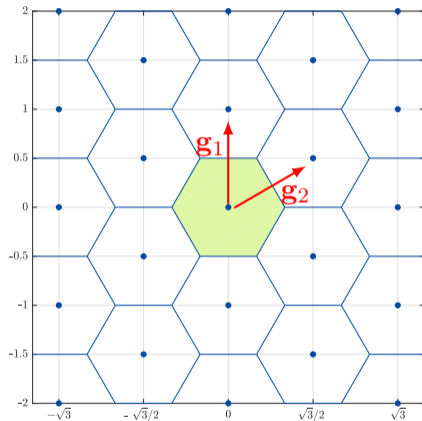
## Example: Hexagonal $A_2$ Lattice and Its Voronoi Region

The  $n = 2$  hexagonal lattice, denoted  $A_2$ , has generator matrix:

$$\mathbf{G} = \begin{bmatrix} \frac{\sqrt{3}}{2} & 0 \\ \frac{1}{2} & 1 \end{bmatrix}$$

The  $A_2$  lattice, the basis vectors, and the hexagonal Voronoi region are shown in the figure. The volume is

$$V(\Lambda) = |\det(\mathbf{G})| = \sqrt{3}/2.$$



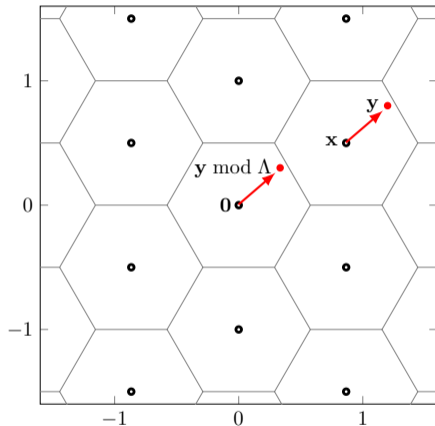
## Lattice Modulo

The *lattice modulo* operation is finding the image of  $\mathbf{y}$  inside the Voronoi region  $\mathcal{V}$ .  
Written using mod:

$$\mathbf{u} = \mathbf{y} - Q_{\Lambda}(\mathbf{y}) = \mathbf{y} \bmod \Lambda$$

The vector  $\mathbf{u} = \mathbf{y} \bmod \Lambda$  is in the 0-centered Voronoi region,  $\mathbf{u} \in \mathcal{V}$ .

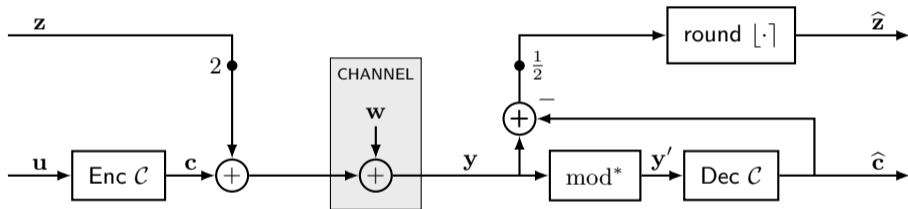
Example: Consider the  $A_2$  lattice.  
 $\mathbf{y} = [1.2, 0.8]^t$  modulo the lattice is shown with the red arrow.



## Construction A: Lattices from Binary Codes

*Construction A* Let  $\mathcal{C}$  be an  $(n, k)$  binary linear code. A vector  $\mathbf{x}$  is a point of lattice  $\Lambda_A$  if and only if  $\mathbf{x}$  is congruent modulo  $q$  to a codeword of  $\mathcal{C}$ . In other words:

$$\Lambda = \mathcal{C} + q\mathbb{Z}^n$$



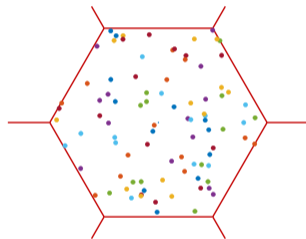
- ▶ Powerful lattices can be constructed from powerful binary codes
- ▶ Convolutional codes are good because of optimal Viterbi decoding
- ▶ All lattices used in the next section are Construction A

## Quantization Using Lattices. Average Error $\approx$ Average Power

- ▶ Lattices can be used for vector quantization — key metric is **average error**
- ▶ Consider random samples  $\mathbf{y}$  uniformly distributed over Voronoi region  $\mathcal{V}$ .

The **average error** of approximating  $\mathbf{y}$  using  $\mathbf{x}$  is:

$$\frac{1}{nV(\mathcal{V})} \int_{\mathcal{V}} \|\mathbf{y}\|^2 d\mathbf{y},$$



- ▶ Later, lattices are used for shaping a code — key metric is **average power**
- ▶ Average error and average power are similar metrics.

## Shaping Gain — Improvement in Quantization Using Lattices

- ▶ Scalar quantization is quite simple — e.g. rounding to integers.
- ▶ Vector quantization is generally better than scalar quantization
- ▶ Vector quantization using lattices is expected to reduce the average error.

How much better is vector quantization than scalar quantization?

The *shaping gain*  $\gamma_s(\mathcal{V})$  can be roughly understood as:

$$\gamma_s(\mathcal{V}) = \frac{\text{Average Error Using Scalar Quantization}}{\text{Average Error Using } \mathcal{V}},$$

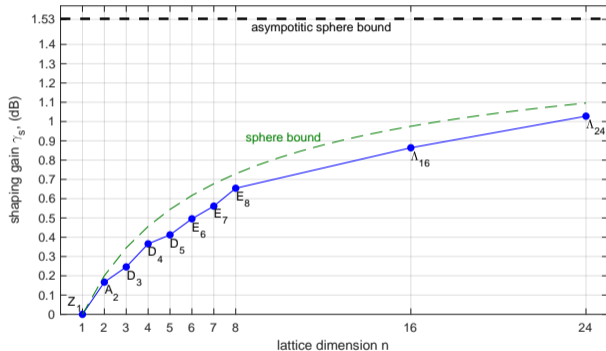
with suitable normalization.

## Shaping Gain for Good Lattices — Dimensions 1 to 24

- ▶ Best possible shape for a quantizer is an  $n$ -ball.
- ▶ **Sphere bound** the shaping gain for an  $n$ -ball:

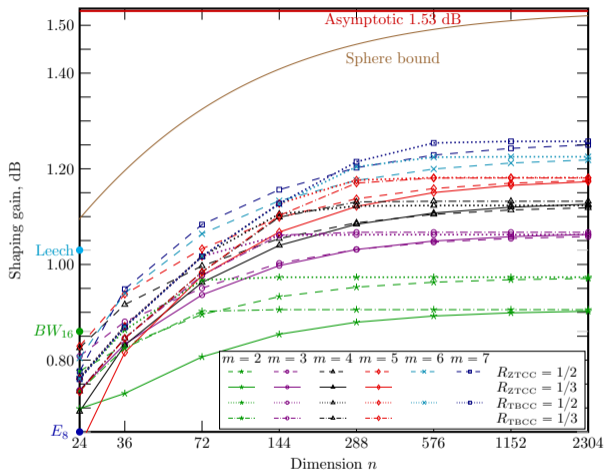
$$\gamma_s = \frac{\pi(n+2)}{12\Gamma(\frac{n}{2}+1)^{2/n}}$$

- ▶ Lattice Voronoi regions are not  $n$ -balls,
- ▶ But, shaping gain for good lattices is close to sphere bound  $\Rightarrow$  Good lattices have sphere-like Voronoi regions.



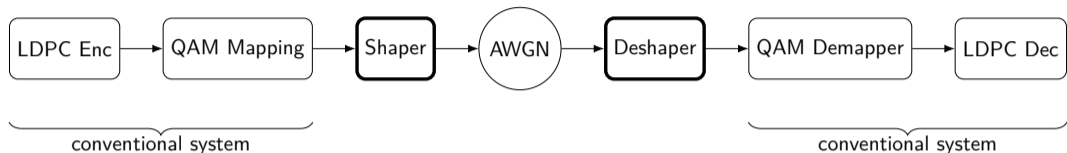
# Shaping Gain for Convolutional Code Lattices — Dimensions 24 to 2304

- ▶ Asymptotic shaping gain is 1.53 dB.
- ▶ Rate  $\frac{1}{2}$  Construction A lattices using convolutional codes with memory  $m$ .
- ▶  $m = 2$ : good shaping gain, low complexity
- ▶  $m = 7$ : best-known shaping gain of 1.25 dB.
- ▶ Viterbi algorithm is key for quantization.



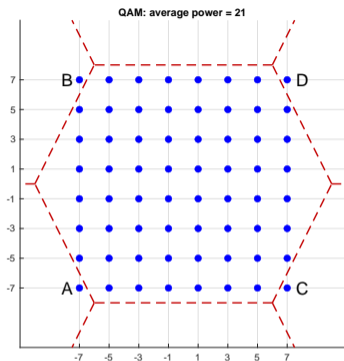
Fan Zhou and Brian Kurkoski, *Construction  $D'$  Lattices for Power-Constrained Communications*, *IEEE Trans. on Communications*, 2022.

## Part 3: Shaping QAM Using Lattices

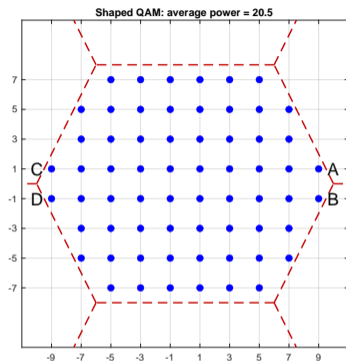


- ▶ Many proposed probabilistic and geometric shaping methods require modifying the conventional system, e.g. labeling/demapper.
- ▶ Our goal is to modify current systems as little as possible.
- ▶ Insert shaper/deshaper so that the conventional system “sees” a QAM constellation.

## Central idea: Shaping QAM using Lattices



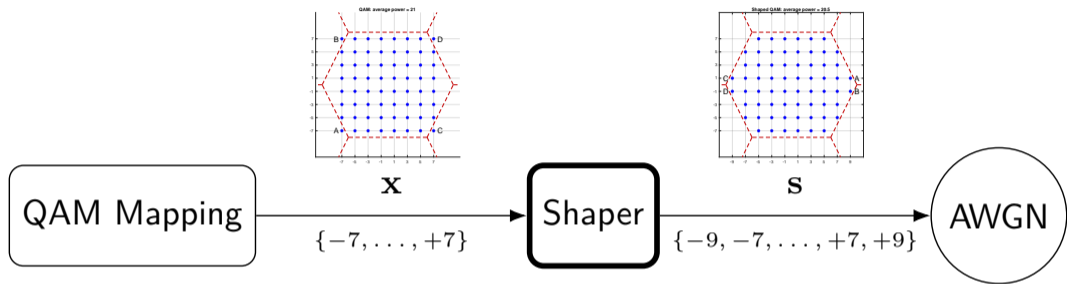
Conventional QAM,  $P_{QAM} = 21$



Shaped QAM,  $P_{Shaped} = 20.5$

- ▶ Before transmission, shape QAM using lattice modulo  $\Lambda_S$ .
- ▶ Shaped constellation has lower average power due to lattice code
- ▶ Number of points is preserved = Rate is not changed

## Before Transmission: Shape QAM Using Lattice Modulo



- ▶ Shaper operation is lattice modulo:

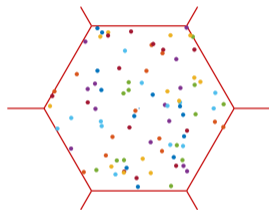
$$\mathbf{s} = \mathbf{x} \bmod \Lambda_s = \mathbf{x} - Q(\mathbf{x})$$

where  $Q(\cdot)$  is lattice decoding/quantization.

# Shaped Constellation Has Lower Average Power Due to Lattice Code

## How to Find Average Power?

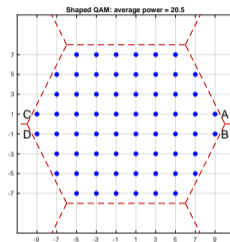
### Average Quantization Error



$$\text{Ave. error} = \frac{1}{nV(\mathcal{V})} \int_{\mathcal{V}} \|\mathbf{y}\|^2 d\mathbf{y},$$

- ▶ Average quantization error  $\approx$  average power.
- ▶ Approximations get better as rate increases.
- ▶ **Known shaping gain of existing lattices estimates transmit power.**

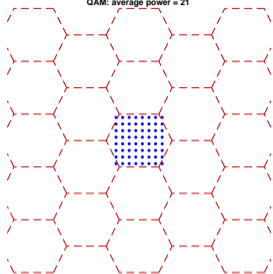
### Average Power



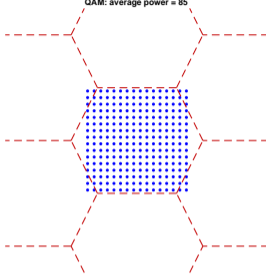
$$\text{Ave. power} = \frac{1}{n} \sum_{\mathbf{y} \in \mathcal{C}} \|\mathbf{y}\|^2,$$

# Lattice Scaling Must Match QAM Size

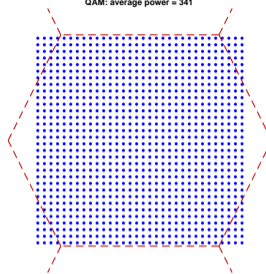
QAM: average power = 21



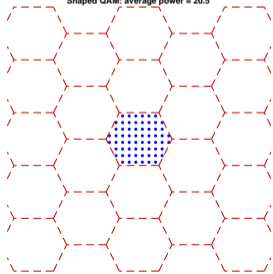
QAM: average power = 85



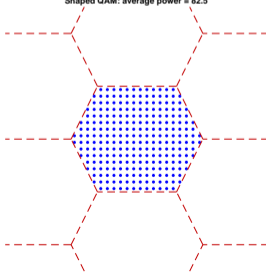
QAM: average power = 341



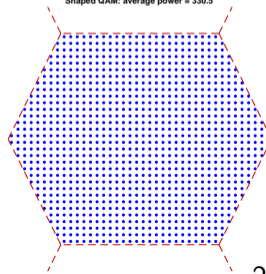
Shaped QAM: average power = 20.5



Shaped QAM: average power = 82.5



Shaped QAM: average power = 330.5



## Design of Shaping Lattice

- ▶ 2D lattices ( $n = 2$ ) yield only *small* shaping gains.
- ▶ Instead of 2D constellations, take dimension  $n$  *large* to improve the shaping gain.
- ▶ For large  $n$ , take sequence of  $n/2$  QAM symbols  $\Rightarrow$  embed in  $\mathbb{R}^n$  and shape.
- ▶ Construction A with convolutional code gives these designs:

$m$	Lattice scale Factor	Code Rate	Maximum Shaping Gain
3	4	1.5	0.95 dB
	8	2.5	
	16	3.5	
4	4	1.5	1.02 dB
	8	2.5	
	16	3.5	
5	4	1.5	1.07 dB
	8	2.5	
	16	3.5	

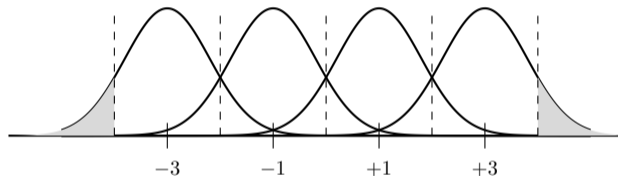
Convolutional code parameters:

- ▶ Memory  $m = 3, 4, 5$ .
- ▶ Block length  $n = 256$ , code rate  $\frac{1}{2}$ .
- ▶ Shaped QAM has a fractional rates, 1.5, 2.5, ...:
  - ▶ Caused by lattice structure.

# Shaped QAM: Small Error-Rate Loss Overcome by Power Gain

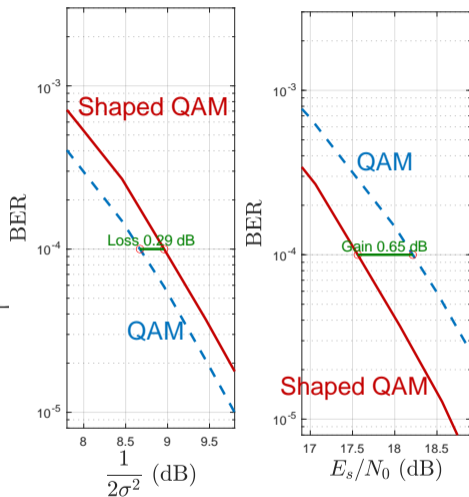
Error rate increase:

- ▶ Shaped QAM has small error-rate loss
- ▶ Due “wrap around” errors
- ▶ Horizontal axis is  $\frac{1}{2\sigma^2}$ ; ignores power



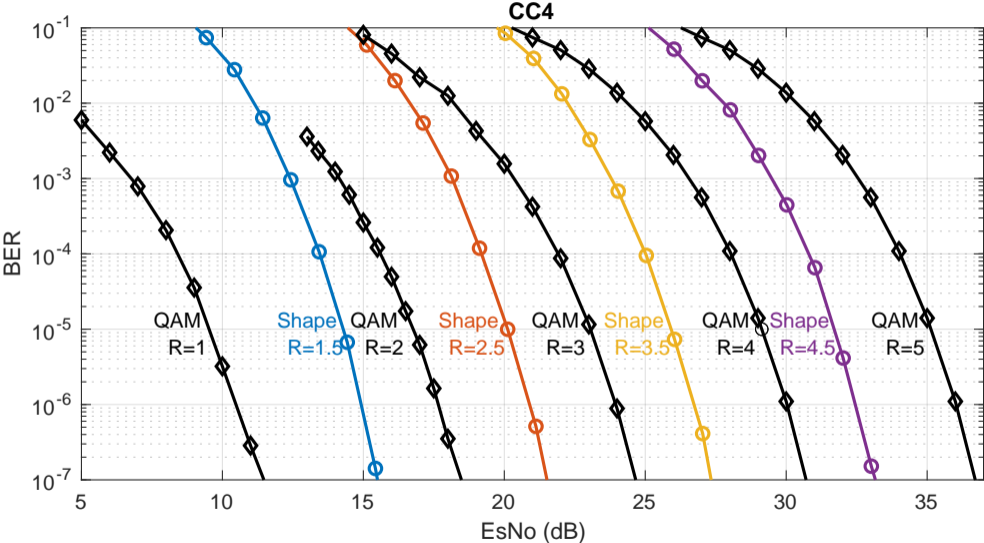
Good news: The benefit of reduced average power is bigger than the error-rate loss!

$E_s/N_0$  includes benefits of reduced transmit power.

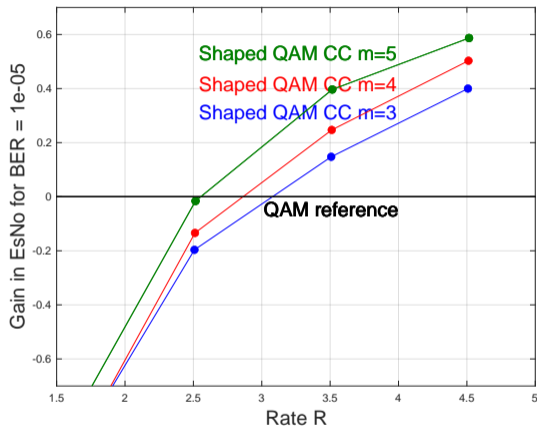
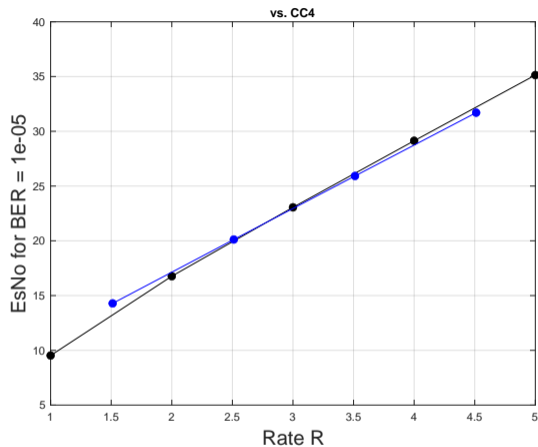


Shaping lattice:  $8D_4$ , Rate  $R = 2.25$ .

# Shaping Using $n = 256, m = 4$ Convolutional Code vs. QAM

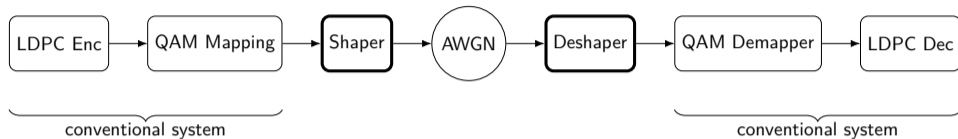


# EsNo Gain vs. Rate $R$ at BER = $10^{-5}$



At high rate,  $m = 5$  code obtains 0.6 dB shaping gain (out of 1.07 dB possible for this code).

## Complexity Considerations



### ▶ **Shaper:**

- ▶ Implemented using Viterbi algorithm.
- ▶ Complexity grows as  $O(2^m)$ , where  $m$  is the convolutional code memory.

### ▶ **Deshaper:**

- ▶ Implemented efficiently via back substitution on sparse matrix.
- ▶ Complexity is linear in block length  $n$ .
- ▶ Possible implementation using convolutional encoder.

## Conclusion and Future Work

### Summary:

- ▶ Shaping reduces transmit power, improves linearity, and lowers energy consumption.
- ▶ Existing shaping methods require modifying the labeling/demapper, etc.
- ▶ Proposed geometric method called Shaped QAM:
  - ▶ Lattice shaping via Construction A and convolutional codes is practical.
  - ▶ Compatible with existing QAM detection implementations
  - ▶ Demonstrated shaping gain of up to **0.6 dB** using convolutional-code lattices.

### Future Work:

- ▶ Increase shaping gain by higher-memory convolutional codes
- ▶ Coset lattice cosets to increase shaping gain.
- ▶ Reduce BER penalty associated with shaping operations.
- ▶ Investigate tradeoffs between complexity, shaping gain, and error performance.

# IEEE Information Theory Society — Chapter Activities for Indonesia

- ▶ **Distinguished Lecturer (DL) Program**
  - ▶ IEEE DLs are internationally recognized experts who give high-quality talks.
  - ▶ Chapters can invite DLs with full travel support from ITSoc.
  - ▶ ITSoc DL information: <https://www.itsoc.org/distinguished-lecturers>
- ▶ **Chapter Initiatives Program** (max. 3000 USD) for local activities<sup>1</sup>:
  - ▶ Host seminars or workshops on information theory.
  - ▶ Invite guest speakers from academia or industry.
  - ▶ Cover venue, publicity, or catering for networking events.
  - ▶ Submit your funding proposal.
- ▶ **Creation of New Chapter**
  - ▶ Petition requires signatures of at least **12 IEEE IT Society members**.
  - ▶ Chapters may be **joint** with other IEEE Societies.



---

<sup>1</sup>Proposed for 2026