Achieving capacity on AWGN channel (Part 2)

Our channel

$X_i \longrightarrow \oplus \longrightarrow Y_i$ (with Z_i added at $\oplus$)

$Z_i \sim \mathcal{N}(0, \sigma) \Rightarrow \sigma^2 = N$

$E[X_i^2] = P$

$C = \frac{1}{2} \log_2\left(1 + \frac{P}{N}\right)$ where $X_i \sim \mathcal{N}(0, P)$ is maximizing distn.

Ex: $\frac{P}{N} = 1000 \Rightarrow \sim 30\,dB$

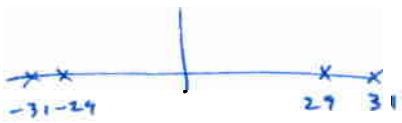
$C = \frac{1}{2} \log_2(1001) \simeq 5$ bits / channel use.

Transmit 5 bits / channel use: Let $X_i \in \{\pm 1, \pm 3, \ldots, \pm 31\}$

encoder:

Input	X_i
00000	−1
00001	−3
⋮	
11111	+31

Signal set



Evaluate performance: possible to show

$$P_e = 2\,Q\left(\sqrt{\frac{d^2}{4N}}\right)$$

$$Q(x) = \int_x^\infty \frac{e^{-x^2/2}}{\sqrt{2\pi}}\,dx$$

d^2 = minimum distance between points in signal set ($d^2 = 4$ in above case)

N = noise power (noise variance)

Can rewrite as:

$$P_e = 2\,Q\left(\sqrt{\frac{d^2}{4\rho}\,\frac{\rho}{N}}\right) \qquad \rho = \text{signal power}$$

(*) $\frac{d^2}{\rho}$ is figure of merit

(*) For a fixed SNR $\frac{\rho}{N}$, we need to find a coding scheme with large $\frac{d^2}{\rho}$.

(*) For AM scheme above (assume equally likely inputs)

$d^2 = 4$

$$\rho = E[X_i^2] = \frac{2}{32}\sum_{i=0}^{15}(2i+1)^2$$

$$= \frac{1}{16}\sum_{i=0}^{15} 4i^2+2i+1$$

↑ use $\sum_{i=1}^{N} i = \frac{N(N+1)}{2}$, $\sum_{i=1}^{N} i^2 = \frac{N(N+1)(2N+1)}{6}$

$$= 85$$

~~$P_e = 2\,Q\left(\sqrt{\frac{}{}}\right)$~~

FOM: $\frac{d^2}{\rho} = \frac{1}{98}$

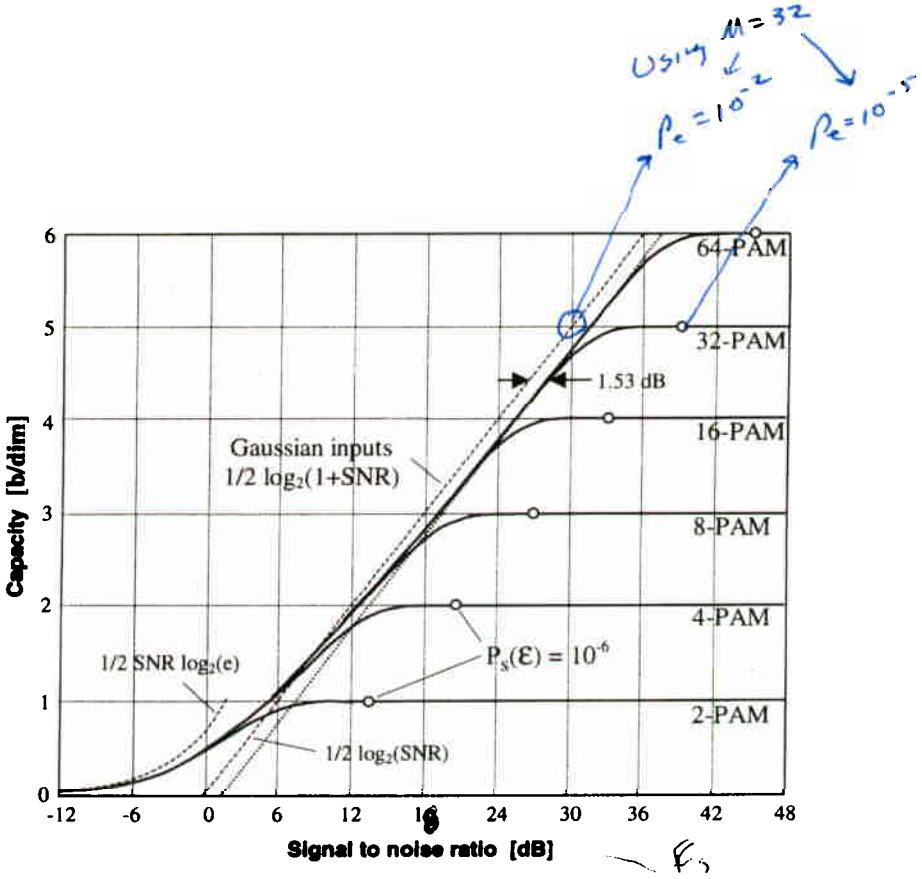
$$P_e = 2\,Q\left(\sqrt{\frac{1}{4(98)}\cdot 1000}\right) = 2\,Q(1.55) = .05$$

↑ Not too bad but we should be able to do much better.

⇒ Shannon says we should be able to get P_e arbitrarily small → $P_e = 10^{-6}$

⇒ at SNR = 40dB $P_e = 2\,Q(5.05) \leq 10^{-6}$

Ques:



Ques:

1) For fixed SNR $\frac{P}{N}$ how to improve system (P_e)?

2) For fixed P_e how can be reduce $\frac{P}{N}$ required?

Ans: ⇒ for 1) & 2) need to develop modulation schemes that have better $\frac{d^2}{P}$ $\quad P_b = 2Q\left(\sqrt{\frac{d^2}{4P}\frac{P}{N}}\right)$

⇒ Do this by encoding in blocks!!

⇒ Instead of encoder producing 1 channel symbol at a time, let encoder produce a block of symbols

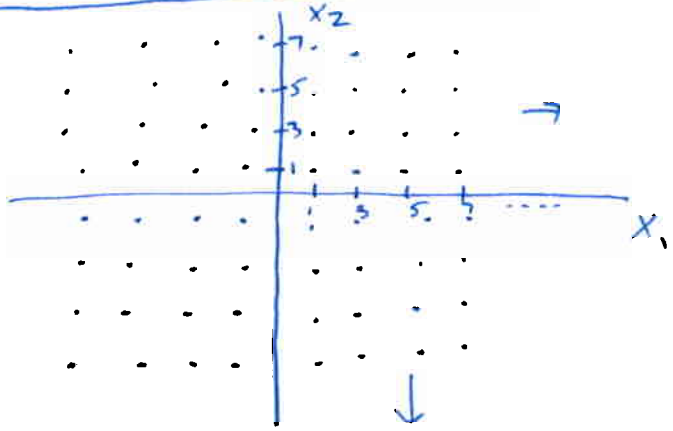
Basic idea:

$(X_1, X_2) \rightarrow \oplus \rightarrow (Y_1, Y_2)$

$\uparrow$

(Z_1, Z_2)

(*) If we use (32-AM) twice



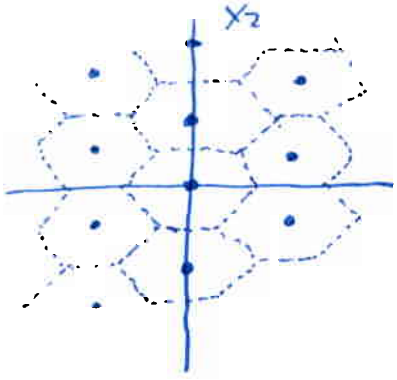
→ integer cubic lattice

$\Rightarrow$ This modulation scheme will produce same result as 1-d 32-AM except

$\rightarrow P_e = 4\,Q\left(\sqrt{\frac{d^2}{4\rho}\frac{\rho}{N}}\right)$

$\rightarrow \frac{d^2}{\rho}$ is same as in 1-d 32-AM

(*) If we design a smarter signal constellation in 2-d



It is possible to show this increases $\frac{d^2}{\rho}$ by a factor 1.15

$P_e = 6\,Q\left(\sqrt{\frac{d^2}{\rho}(1.15)\frac{\rho}{N}}\right) \Rightarrow$ This has net effect of either 1) improved P_e or 2) being able to operate at smaller $\frac{\rho}{N}$

↑ old scheme. ↑ (called coding gain)

(*) Conclusion: go to higher dimensional constellations, improve $\frac{d^2}{P}$ as you go to ∞ dimensions.

(*) It is possible to show that PAM using an ∞-dimension hyper-sphere achieve capacity.

(*) ∞-dimensional hyper spheres are ∞-ly complex.

(*) Practical methods for improving $\frac{d^2}{P}$

2 basic methods: trellis codes & multi level codes

Ungerböck IBM Zurich 1976

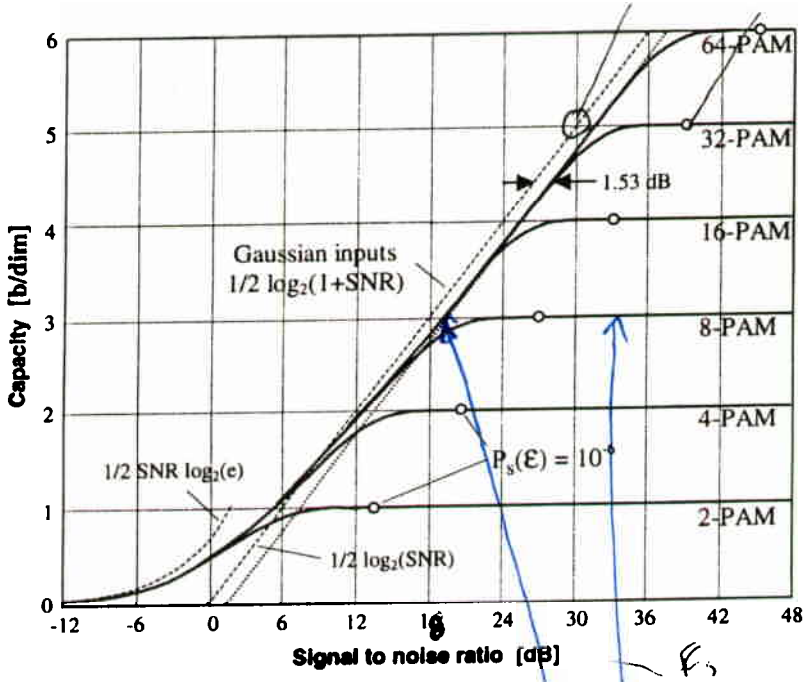
Imai & Hyrakawa (late 70's)

Huber Wachsman (late 90's)

Trellis codes :- heart of all high efficiency modems (phone line etc).

- A method of constructing a high dimensional constellation using only low dimensional constellations.

- Motivation: page (3) says that to get close to capacity, use redundancy in M to obtain better performance.



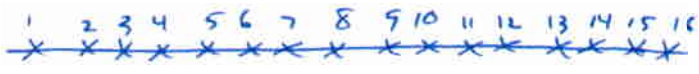
Achieve 3 bits/channel use w/ $M = 8$ levels

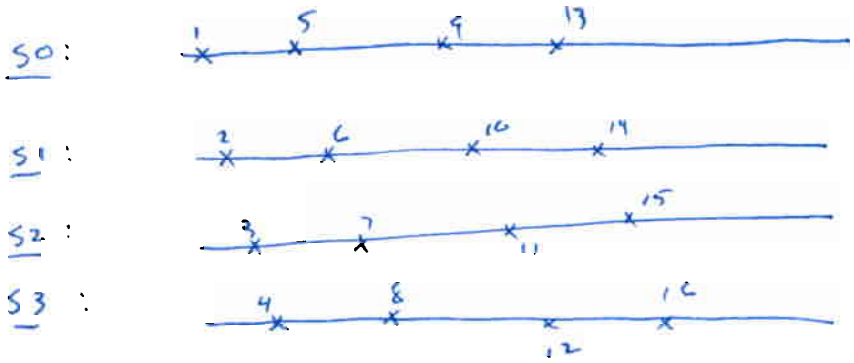
Achieve 3 bits/channel use w/ $M = 16, 32$ levels.

⇒ Idea: → expand # of symbols to be sent
→ use error control code to select a subset.

Basics: Assume we want to send 3 bits/channel use
⇒ Normally we use $M = 8$ levels.
⇒ Consider an $M = 16$ level approach

16-PAM constellation

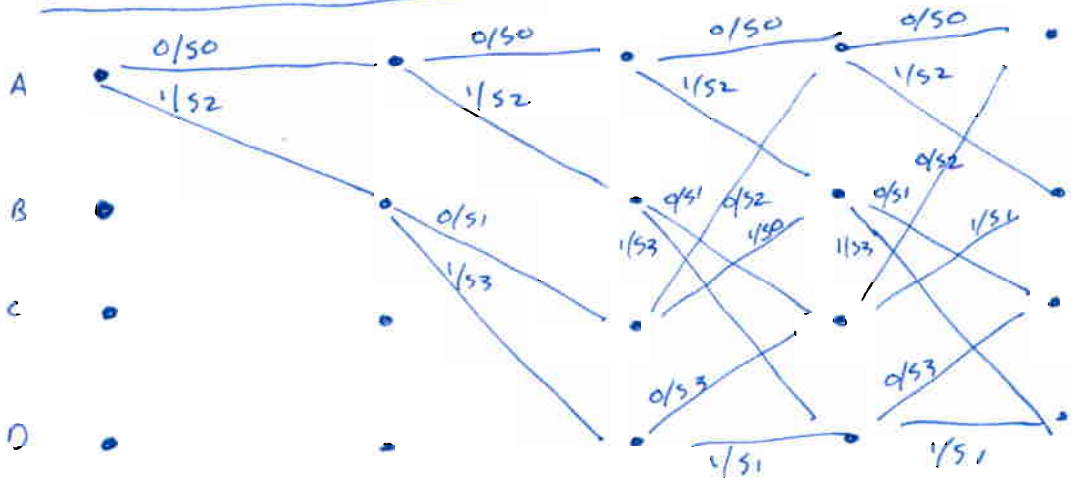




Naive approach: 2 b.ts select S0/S1/S2/S3 } encodes 4 b.ts, we need only encode 3
2 b.ts select point inside S0/S1/S2/S3

Coded approach: 1 b.t selects S0/S1/S2/S3 ← use redundancy here!!
2 b.t select point inside. S0/S1/S2/S3

Selection of signal sets:



where trellis is labeled w/ b_0/S_i

Encoding procedure: at each time instant we encode
3 b.ts ⇒ 1 symbol
$b_0 b_1 b_2$

1) b_0 selects constellation S0/S1/S2/S3
2) $b_1 b_2$ select point inside S_i

Comments:

1) Signal sets $S0/S1/S2/S3$ are constrained from a sequence standpoint (Not all sequences of signal sets are allowed)

2) 'Trellis' comes from $R = 1/2$ convolutional code

3) Code is decoded using Viterbi algorithm

4) n stages of the trellis produce a channel symbols (from an n-dimensional space).

5) Turns out for large # of states in the trellis, you can easily get 3-6dB of coding gain

6) Modems use this

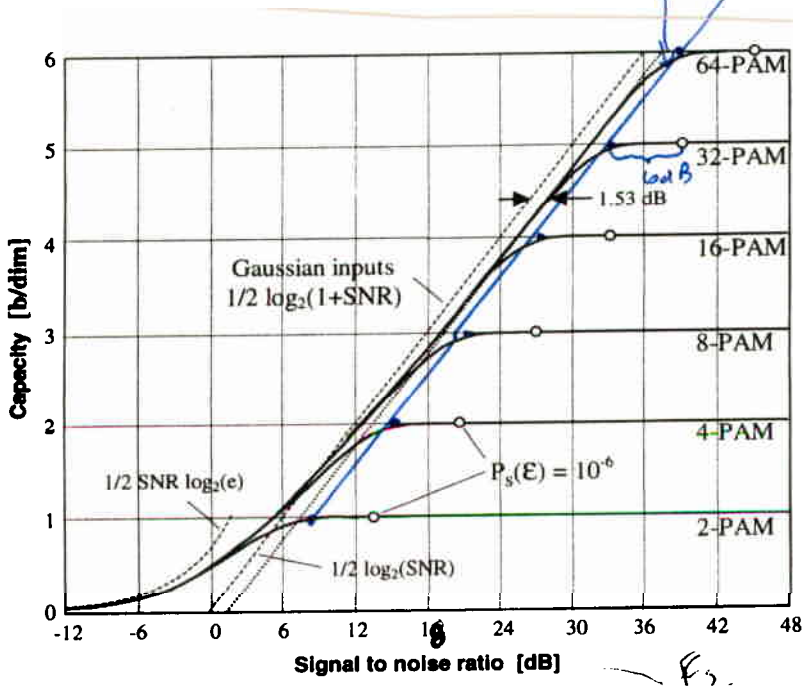
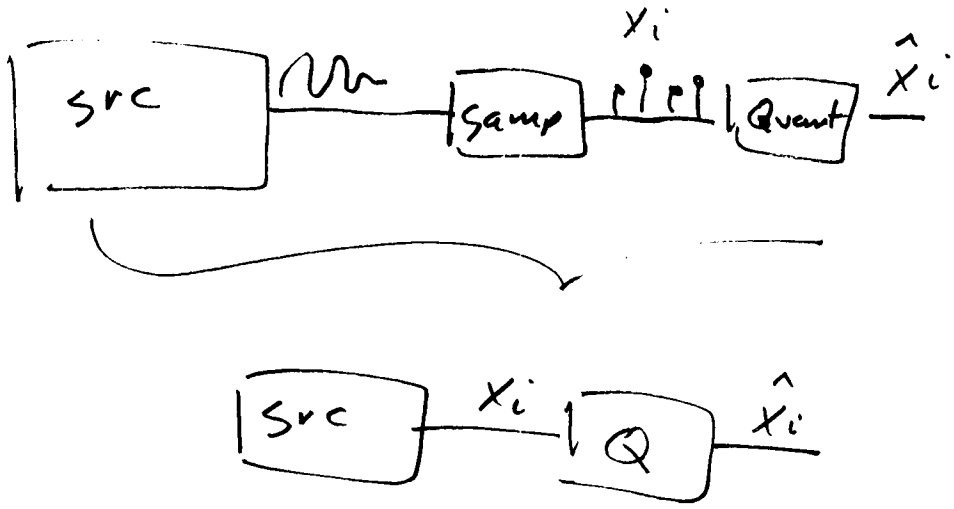


Fig.

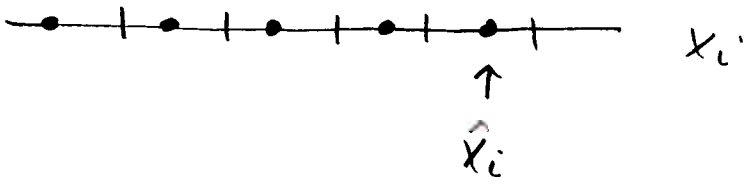
Summary on $R(D)$



$D = E[(x - \hat{x})^2]$ "distortion"

Quantizer:

$SNR_Q = 2^{2R}$



Fund question:

1) For fixed D, what is fewest # bits to represent x?
2) For fixed R_s, what is smallest D?

$$R(D) = \min_{f(\hat{x}|x)} I(X;\hat{X})$$

$$E[(x-\hat{x})^2] \le D$$

Last time: for $X_i \sim \mathcal{N}(0, \sigma^2)$

$$R(D) = \min_{\substack{f(\hat{x}|x) \\ \leq D}} I(X;\hat{X}) \geq \frac{1}{2}\log_2 \frac{\sigma^2}{D}$$

Question: how do we achieve lower bound

Answer: "test channel"

let $X = \hat{X} + Z$ $\quad Z \sim \mathcal{N}(0, D)$, $\hat{X} \sim \mathcal{N}(0, \sigma^2 - D)$ } indep.

$$\hat{X} \longrightarrow \oplus \longrightarrow X \qquad Z \uparrow$$

$\Rightarrow$ $X \sim \mathcal{N}(0, D + \sigma^2 - D) = \mathcal{N}(0, \sigma^2)$

Easy to show: $I(X;\hat{X}) = \frac{1}{2}\log_2 \frac{\sigma^2}{D}$

so: $f(\hat{x}|x) = \frac{f(\hat{x}, x)}{f(x)} = \frac{f(x|\hat{x})f(\hat{x})}{f(x)}$

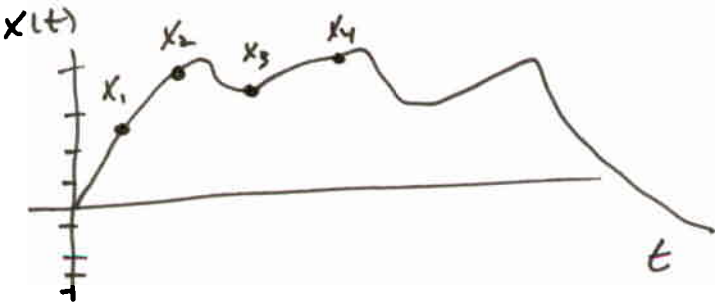
Answer to 2:

How do we achieve lower limits on R_s?

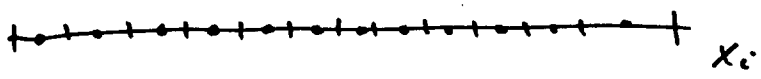
Namely, we want to encode a source in a few bits as possible while maintaining fidelity D. How is this done? Encoding in blocks of symbols!! (Vector Quantization)

(*) Quantization: Scalar vs. vector

Consider a source to be scalar quantized.



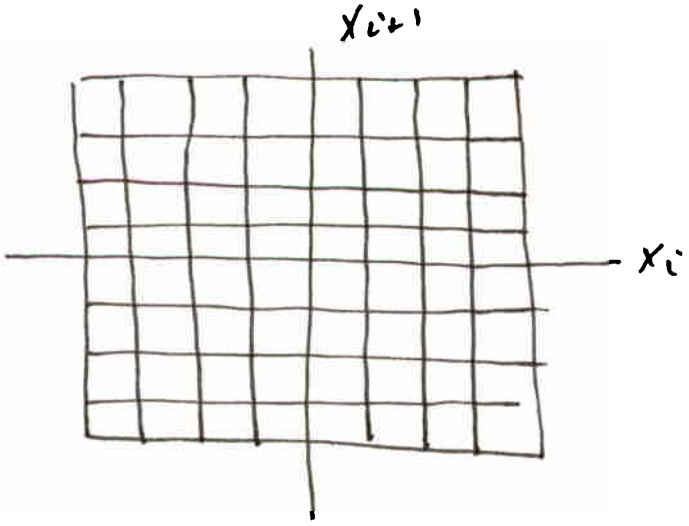
Use scalar quantizer to quantize X_i



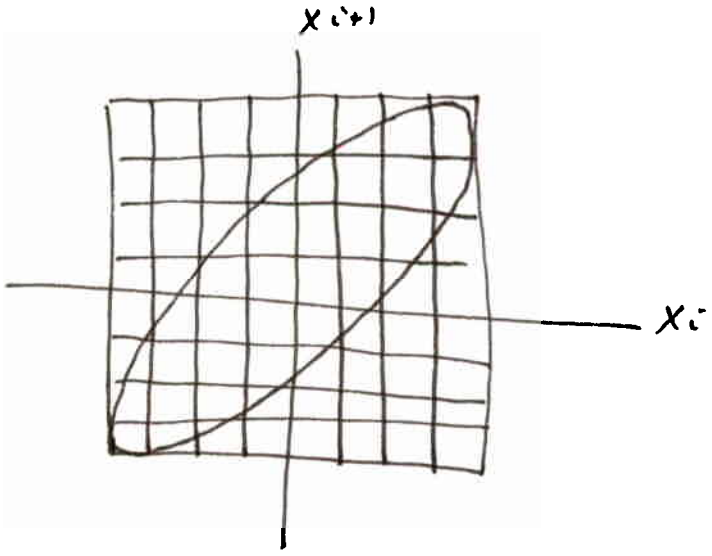
$2^{+2L} = SNR$

For fixed L, the only way to improve SNR is use a vector quantizer.

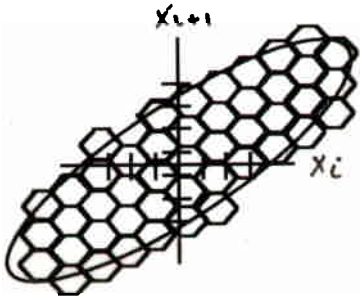
consider (X_i, X_{i+1}) pair : use scalar quantizer twice



but (X_i, X_{i+1}) vector is likely to be concentrated near $x=y$ line (if correlated



Better yet: (A vector quantizer !!)



3 gains: 1) Cell shape gain

2) Region shape gain

3) Memory gain

1) Cell shape gain: use "spheres" instead of cubes in high dimension

gain → 1.53 dB in SNR

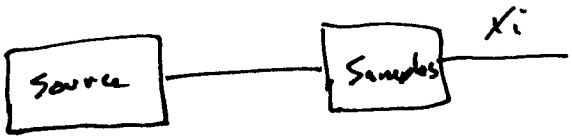
2) Region shape gain: have region match support region of source. Gain depends on source ~ 2 – 6dB

3) Memory gain: have quantizer extract memory (or correlation) in source

gain: as much as 12 dB

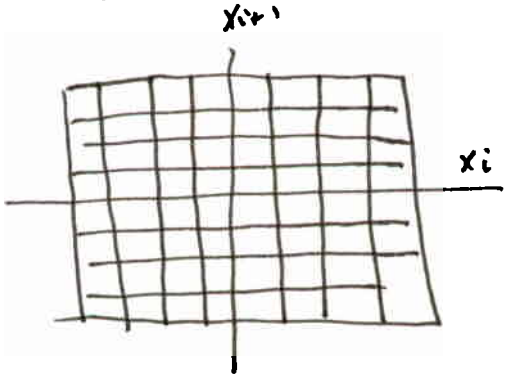
Better yet: A well-designed VQ can achieve the rate-distortion function. (A VQ is optimum).

Other questions: what if source has no memory. Is there still gain using a VQ? Of course there is!!

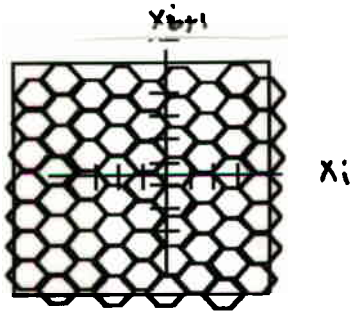


Assume x_i are i.i.d uniform: - use a uniform scalar quantizer:

- use a VQ



vs.



cell shape gain only

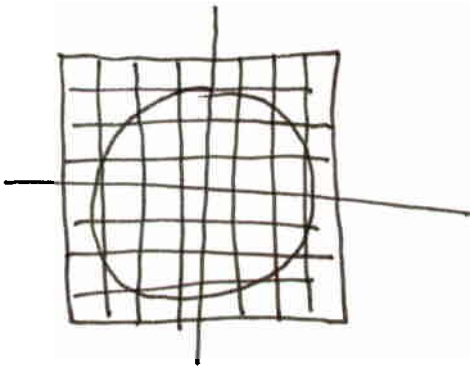
1.53 dB is about .25 bits

⇒ quite a bit!!

Assume X_i are iid Gaussian (mean 0, var σ_x^2, σ_y^2)

$$p(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right)}$$

Scalar quantizer 2x



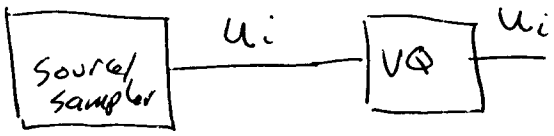
vs

For this source one gets

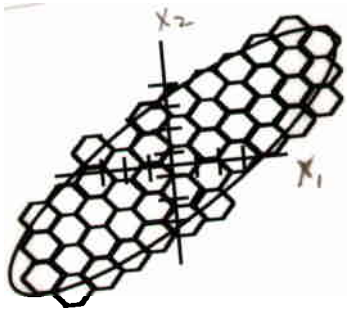
1) cell shape ~ 1.53 dB
2) region shape ~ 3 dB

by using a VQ

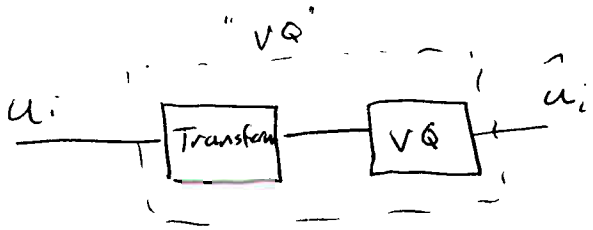
How does one get all the gains associated with a VQ? (cell shape, region shape, memory).

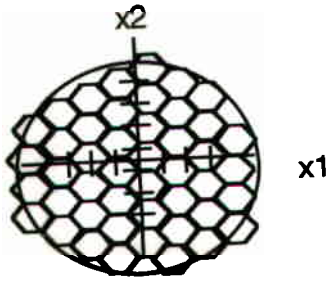


Option 1): - Design a full VQ:
- Instead of using a scalar quantizer use a high dimensional ($k > 20$) VQ to get the gain and achieve $R(D)$.



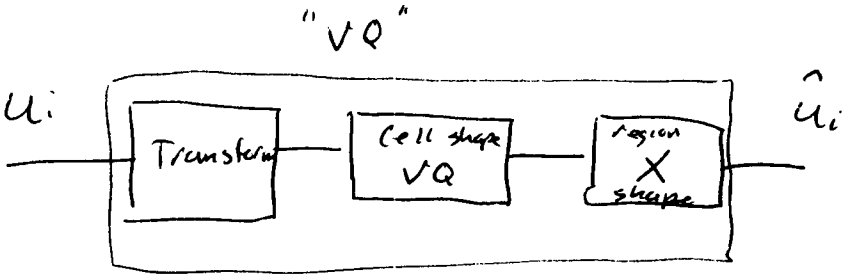
Option 2) - Apply a good transformation to the data to completely decorrelate it.
- Design a VQ that gets cell and region shape.





Transforms: - Karhunen Loeve
- Fourier
- Discrete cosine Transform

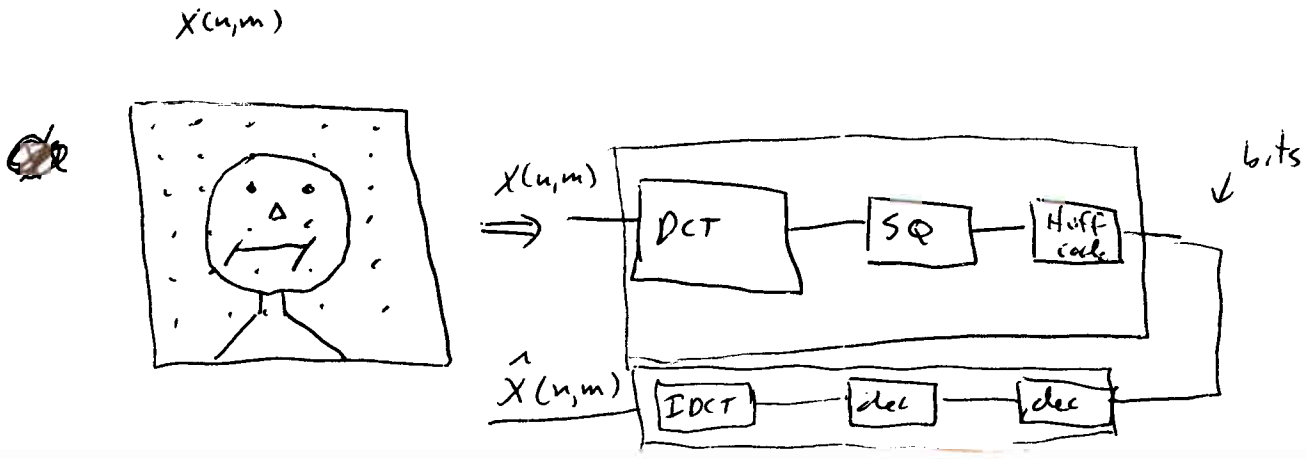
Option 3):
- Apply Transform to decorrelated
- Apply quantizer to get cell shape
- Apply "X" to get region shape



- "X" is a Huffman code (or any other lossless encoder)

- This tandem approach to quantizing is guaranteed to achieve R(D), if Transform perfectly decorrelates, as VQ gets all cell shape and Huffman code gets all region shape.

Example: JPEG image compression standard



- DCT is discrete cosine transform
- SQ is scalar quantizer

What's wrong with this approach?

- No cell shape gain.
- Can never pick up 1.53 dB that cell shaper sets.
- One could save .25 bits w/o any loss in performance if one used a VQ to set cell shape gain instead of SQ.

- Analogy: JPEG at .5 bits ⇒ distortion D

  Another scheme w/ cell shape gain at .25 bits ⇒ same distortion D