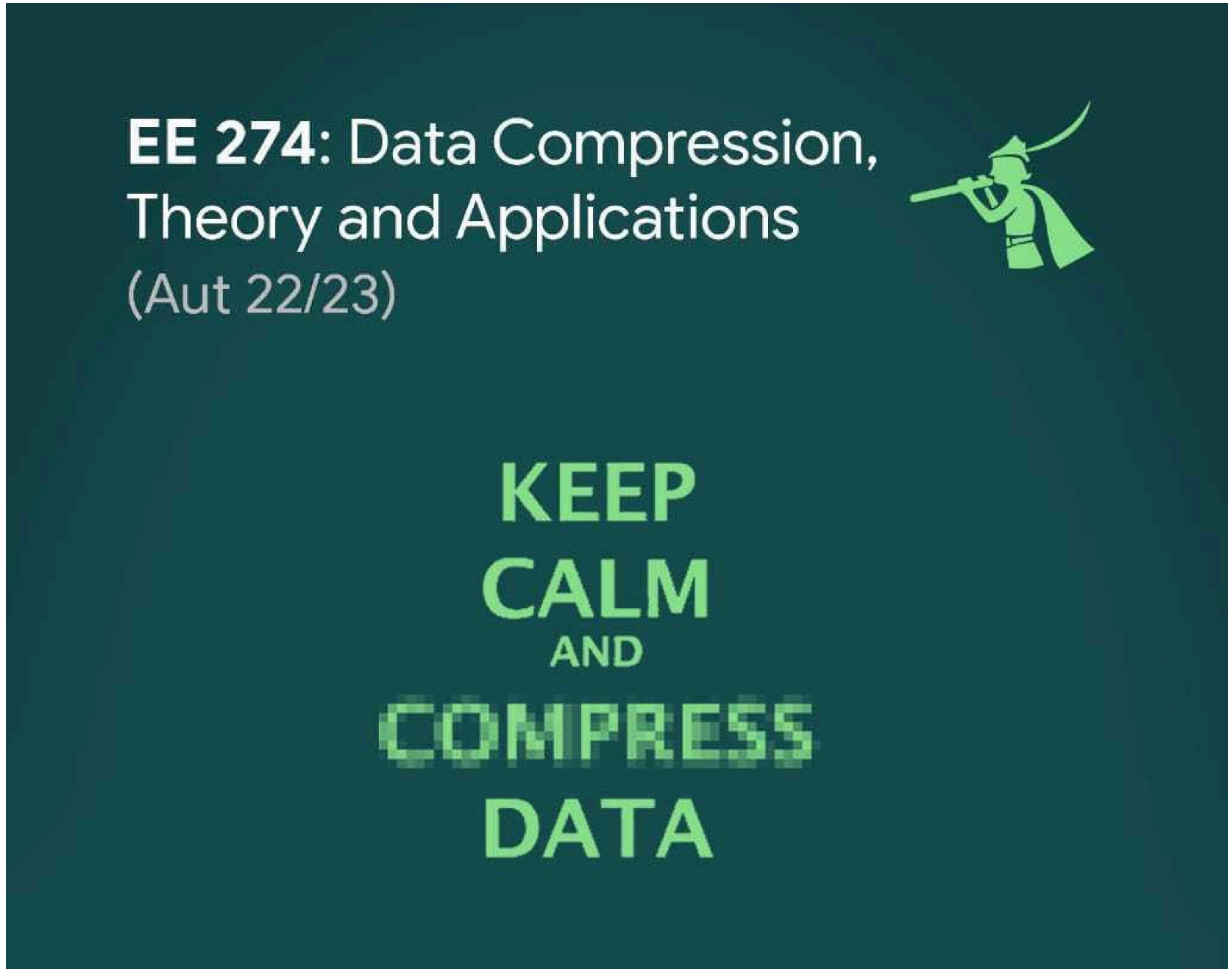


quantization lecture slides

Vector Quantization

A quantizer is a mapping

$$Q : \mathbb{R}^k \longrightarrow \mathcal{C}$$

where $\mathcal{C} = \{\underline{y}_i\}_{i=1}^{N}$ is the "codebook" or "dictionary" comprising N k-dimesional vectors.

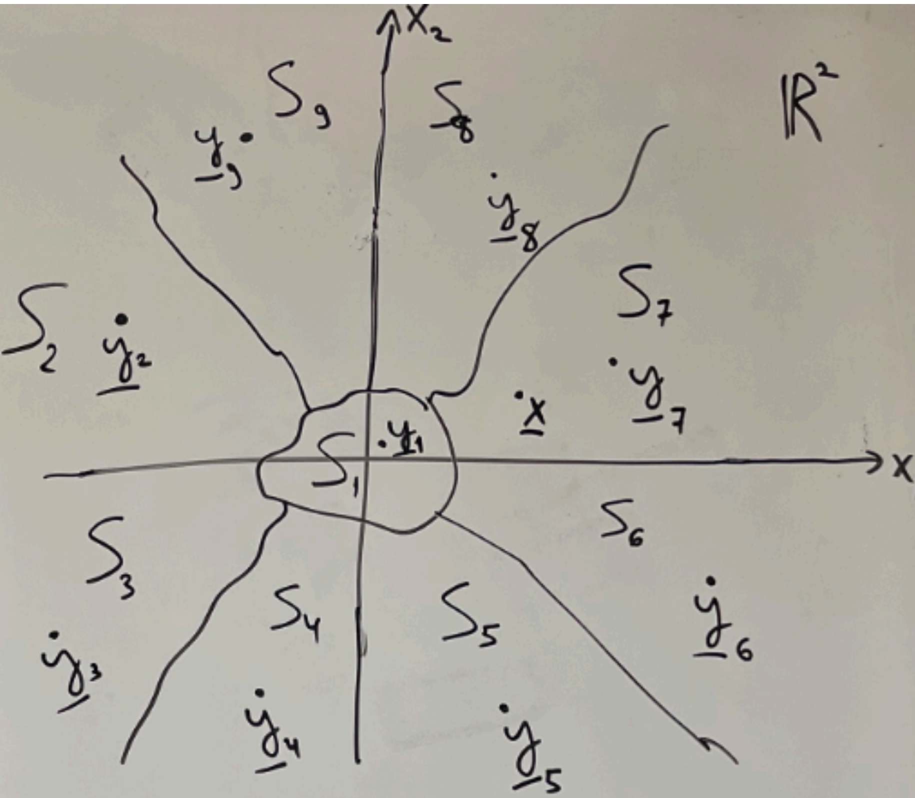
The mapping is defined by:

$$Q(\underline{x}) = \underline{y}_i \quad \text{if} \quad \underline{x} \in S_i$$

where $\{S_i\}_{i=1}^{N}$ is a partition of $\mathbb{R}^k$, i.e.: $\bigcup_{i=1}^{N} S_i = \mathbb{R}^k$, $S_\ell \cap S_m = \phi$ $\ell \neq m$.

The rate is $R = \dfrac{\log N}{k}$ $\dfrac{\text{bits}}{\text{sample}}$; $N = 2^{kR}$

Example:
$k=2$, $N=9$
$Q(\underline{x}) = \underline{y}_7$
$\underline{x} \in S_7$

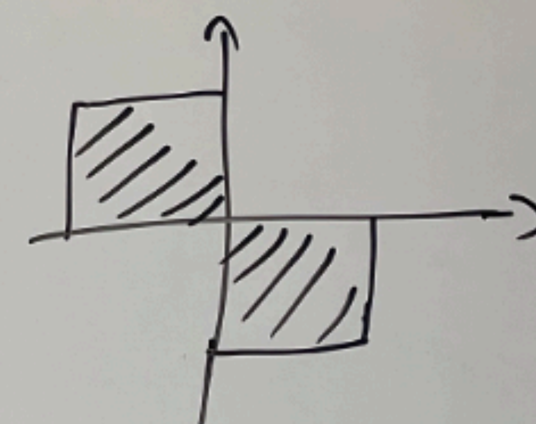


Vector Quantization (cont., board 2)

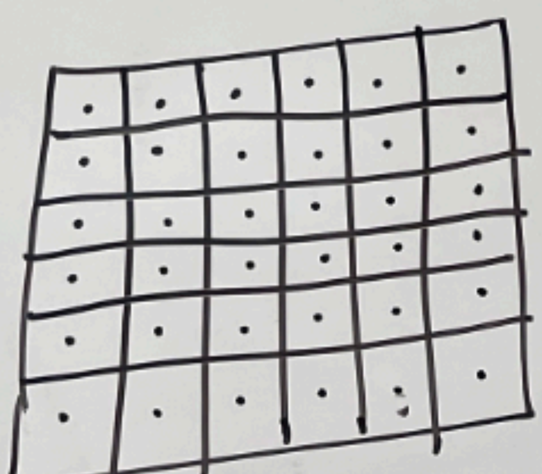
Vector Quanization (VQ) allows to exploit:

1) dependence between vector components
2) more general decision regions
   (than could be obtained via Scalar Quantization (SQ))

Example I: $f(x_1, x_2)$ uniform on

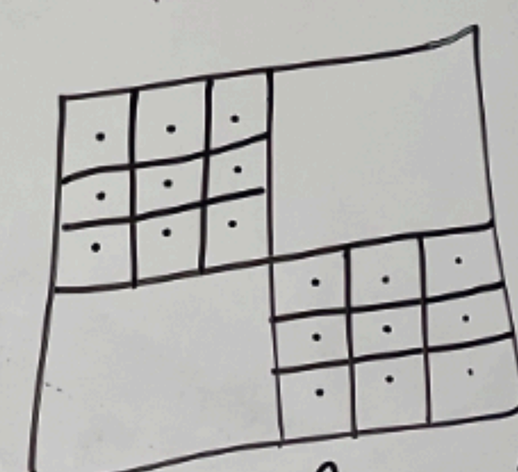


Quanizing each coordinate separately:



$R = \log_2 6 \ \frac{\text{bits}}{\text{sample}}$

while VQ would be:



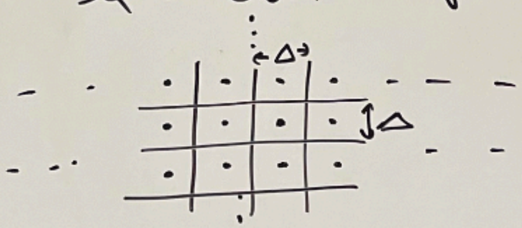
$R = \log_2 6 - \frac{1}{2} \ \frac{\text{bits}}{\text{sample}}$

Vector Quantization (cont., board 3)

Example II: also $k=2$.

IID $\underbrace{\text{components}}_{\text{uniform}}$.

SQ of each component separately would yield decision regions of this form:



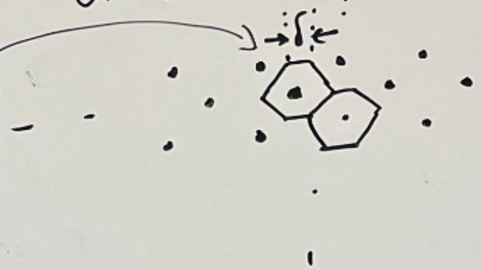
$A_1 = \Delta^2$

$$MSE_1 = \int_{-\Delta/2}^{\Delta/2}\int_{-\Delta/2}^{\Delta/2} (x^2+y^2)\,dx\,dy = \int_{-\Delta/2}^{\Delta/2} \left[\frac{x^3}{3} + y^2 x\right]_{-\Delta/2}^{\Delta/2} dy = \int_{-\Delta/2}^{\Delta/2} \left[\frac{\Delta^3}{12} + y^2\Delta\right] dy = \left[\frac{\Delta^3}{12}y + \frac{y^3}{3}\Delta\right]_{-\Delta/2}^{\Delta/2}$$

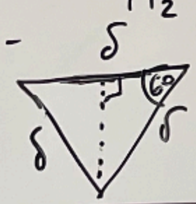
$$= \frac{\Delta^4}{12} + \frac{\Delta^4}{12} = \frac{\Delta^4}{6}$$

Under MSE, the optimal VQ would look like:

Hexagonal decision regions



$$A_2 = \frac{\sqrt{3}}{2}\cdot\delta\cdot\delta\cdot\frac{1}{2}\cdot 6 = \frac{3\sqrt{3}}{2}\delta^2$$



$$MSE_2 = \ldots = \frac{5\sqrt{3}}{8}\delta^4$$

Exercise

For $A_1 = A_2$ we get $\frac{MSE_2}{MSE_1} = 0.962$

Conclusion: even for a simple uniform IID source there is benefit in VQ over SQ.

Vector Quantization (cont., board 4)

Necessary conditions for optimality that are also sufficient for local optimality:

1) Given the dictionary $\{\underline{y}_1, \ldots, \underline{y}_N\}$, the decision regions $\{S_1, \ldots, S_N\}$ must satisfy

$\forall i,\ \underline{x} \in S_i,\ j \neq i$: $d(\underline{x}, \underline{y}_i) \leq d(\underline{x}, \underline{y}_j)$

("nearest neighbor rule" or "Voronoi regions")

2) Given the decision regions $\{S_i\}_{i=1}^{N}$, the dictionary $\{\underline{y}_i\}_{i=1}^{N}$ must satisfy:

$$\forall i : E\left[d(\underline{X}, \underline{y}_i) \mid \underline{X} \in S_i\right] = \min_{\underline{u}} E\left[d(\underline{X}, \underline{u}) \mid \underline{X} \in S_i\right]$$

I.e., $\underline{y}_i$ need be a point in $\mathbb{R}^k$ minimizing the expected distortion in region S_i.

We refer to such a point as the "centroid of S_i": $\text{cent}(S_i) \triangleq \arg\min_{\underline{u}} E\left[d(\underline{X}, \underline{u}) \mid \underline{X} \in S_i\right]$.

E.g., under squared error distortion $\text{cent}(S_i) = E\left[\underline{X} \mid \underline{X} \in S_i\right]$.

These 2 conditions suggest (the following) iterative algorithm for constructing a vector quantizer.

Generalized Lloyd (GL) algorithm:

We specify the (more practical) version pertaining to a given training sequence (rather than a specified distribution)

Given: N (dictionary size), $\varepsilon > 0$ (distortion threshold)

$Y^{(0)}$ - initial dictionary of size N, $\{\underline{\tilde{x}}_j\}_{j=1}^{n}$ - training sequence of n k-dimesional vectors

Initialization: $D_{-1} = \infty$ (large number), $m = 0$ (iteration step)

(1) Given $Y^{(m)}$, find partition $\left(\{S_i^{(m)}\}_{i=1}^{N}\right)$ of training sequence according to nearest neighbor:

$S_i^{(m)}$ comprises all (the) $\underline{\tilde{x}}_j$ s that are closest to $\underline{y}_i^{(m)}$

(2) Compute average distortion for this iteration: $D^{(m)} = \frac{1}{n} \sum_{j=1}^{n} \min_{1 \le i \le N} d(\underline{\tilde{x}}_j, \underline{y}_i^{(m)})$

(3) If $\frac{D^{(m-1)} - D^{(m)}}{D^{(m)}} < \varepsilon$, end.

Otherwise:

(4) Construct $Y^{(m+1)}$ from $Y^{(m)}$ by replacing each $\underline{y}_i^{(m)}$ by $\underline{y}_i^{(m+1)} = \mathrm{Cent}(S_i^{(m)}) \underset{\substack{\uparrow \\ \text{for squared error}}}{=} \frac{1}{|S_i^{(m)}|} \sum_{\underline{x} \in S_i^{(m)}} \underline{x}$,

where $|S_i^{(m)}|$ denotes the number of vectors in $S_i^{(m)}$ (cardinality).

(5) $m \to m+1$ and back to (1).

Generalized Lloyd (GL) algorithm (Cont.):

- ε-Convergence to local minimum guaranteed
- Choice of Y^0 is consequential in practice, with various heuristics
- rule of thumb: $n \gtrsim 50\,N$

  n: size of training set

  N: dictionary size

historical note

- first proposed by Stuart Lloyd in 1957 (motivated by audio compression) at Bell Labs

- was widely circulated but formally published only in 1982

- independently developed and published by Joel Max in 1960

- therefore sometimes referred to as the Lloyd-Max algorithm

- Generalized Lloyd specialized to squared error is the K-means clustering algorithm widely used in Machine Learning