Updated 12/1/2024

Part 3: Applications
L21/L22: Distortion Theory

Wolfgang Gatterbauer, Javed Aslam

cs7840 Foundations and Applications of Information Theory (fa24)

https://northeastern-datalab.github.io/cs7840/fa24/

11/20/2024 & 11/25/2024

Pre-class conversations

- Are you having fun with Python files?
- Intended Topics & Feedback

- **Lecture 19 (Wed 11/13):**
  Logistic Regression (2/2) [Luce's choice axiom, Bradley-Terry model]
  Maximum Entropy (2/2) [Occam, Kolmogorov, Minimum Description Length (MDL)]
- **Lecture 20 (Mon 11/18):**
  Channel capacity [Cover Thomas'06: Ch 7]
- **Lecture 21 (Wed 11/20):**
  Rate Distortion Theory [Cover Thomas'06: Ch 10]
- **Lecture 22 (Mon 11/25):**
  Information Bottleneck Theory R.D
- **(Wed 11/27): no class (Fall break)**
- **Lecture 23 (Mon 12/2):** Probability and Entropy Axioms IB
- **Lecture 24 (Wed 12/4):** Placeholder IB

Project presentations

- **Lecture 25 (Mon 12/9): P4 Project presentations**
- **Lecture 26 (Wed 12/11): P4 Project presentations**

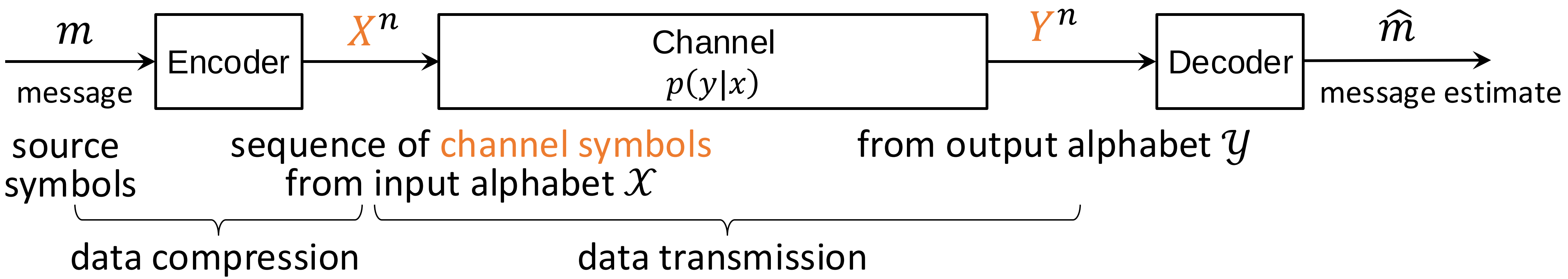
- Rate Distortion & Information bottleneck theory
  - [Cover,Thomas'06] Elements of Information Theory. 2nd ed, 2006: Ch 10 Rate distortion theory
  - [Tishby+'99] Tishby, Pereira, Bialek. The information bottleneck method. The 37th annual Allerton Conference on Communication, Control, and Computing. pp. 368–377.
  - [Harremoes,Tishby'07] The Information Bottleneck Revisited or How to Choose a Good Distortion Measure. International Symposium on Information Theory, 2007.
  - [Zaslavsky+'18] Zaslavsky, Kemp, Regier, Tishby. The Efficient compression in color naming and its evolution. PNAS, 2018.
  - [Webb+'24] Webb, Frankland, Altabaa, Segert, Krishnamurthy, Campbell, Russin, Giallanza, Dulberg, OReilly, Lafferty, Cohen. The Relational Bottleneck as an Inductive Bias for Efficient Abstraction. Trends in Cognitive Science, 2024.
  - [Segert'24] Maximum Entropy, Symmetry, and the Relational Bottleneck: Unraveling the Impact of Inductive Biases on Systematic Reasoning. PhD thesis, Neuroscience @ Princeton, 2024.
  - [Ren,Li,Leskovec'20] Graph Information Bottleneck, NeurIPS, 2020.

- Today:
  - Rate Distortion (basically lossy transmission)

Channel Capacity

Largely based on chapter 7 of
[Cover, Thomas'06] Elements of Information Theory, 2006. https://www.doi.org/10.1002/047174882X

Channel Capacity C = highest rate R



"Information" channel capacity $C = \max_{p(x)} I(X;Y)$

"Operational" channel capacity is the highest rate R (in bits) per channel use

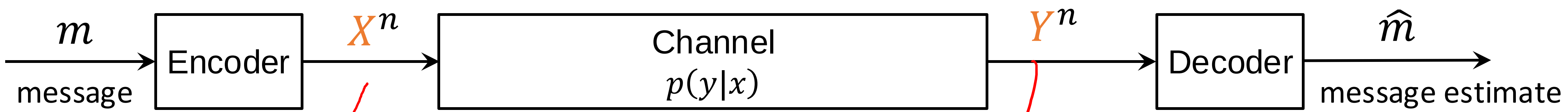
Shannon's channel coding theorem: both are identical, i.e. the channel capacity can be achieved in the limit by using codes with a long block length.

Theorem 7.7.1 (*Channel coding theorem*) *For a discrete memoryless channel, all rates below capacity C are achievable. Specifically, for every rate $R < C$, there exists a sequence of $(2^{nR}, n)$ codes with maximum probability of error $\lambda^{(n)} \to 0$.*

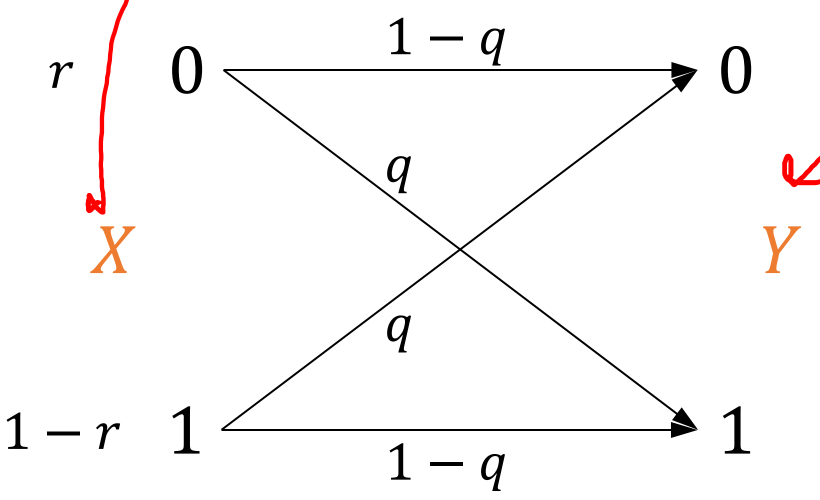
Conversely, any sequence of $(2^{nR}, n)$ codes with $\lambda^{(n)} \to 0$ must have $R \le C$.

Source: [Cover, Thomas'06] Elements of Information Theory, 2006. Chapter 7 channel capacity, https://www.doi.org/10.1002/047174882X

Capacity of binary symmetric channel



channel capacity $C = \max_{p(x)} I(X;Y)$



How do we calculate the channel capacity C ?

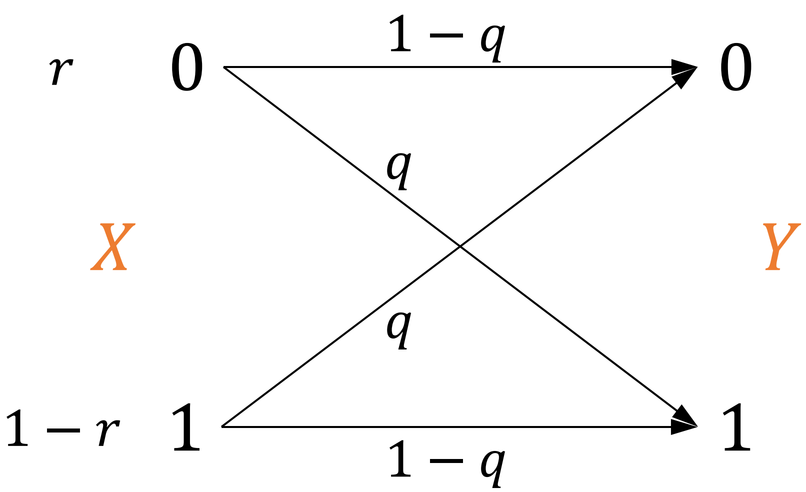
Hence, capacity for binary symmetric channel is $C = 1 - H(q)$

Source: [Cover, Thomas'06] Elements of Information Theory, 2006. Chapter 7 channel capacity, https://www.doi.org/10.1002/047174882X
Gatterbauer, Aslam. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Capacity of binary symmetric channel

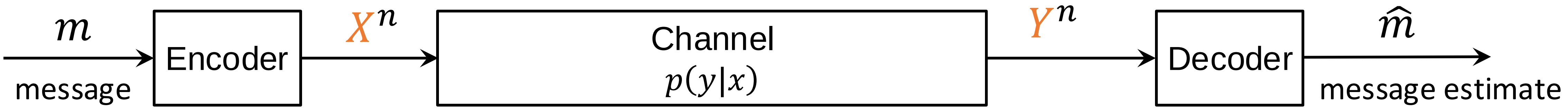
channel capacity $C = \max_{p(x)} I(X;Y)$

$I(X;Y) =$?

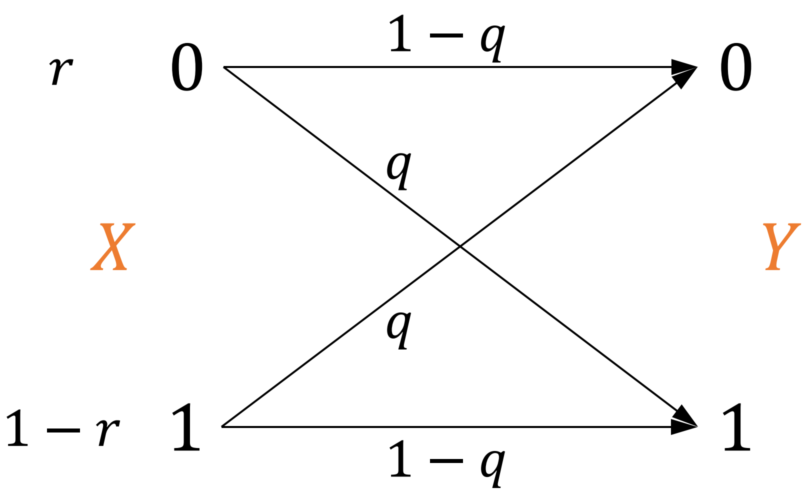


Source: [Cover, Thomas'06] Elements of Information Theory, 2006. Chapter 7 channel capacity, https://www.doi.org/10.1002/047174882X
Gatterbauer, Aslam. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Capacity of binary symmetric channel



channel capacity $C = \max_{p(x)} I(X;Y)$



$I(X;Y) = H(Y) - H(Y|X)$

$\sum_x p(x)\, H(Y|X=x)$

$\sum_x p(x)\, H(q) = H(q)$

$= H(Y) - H(q)$

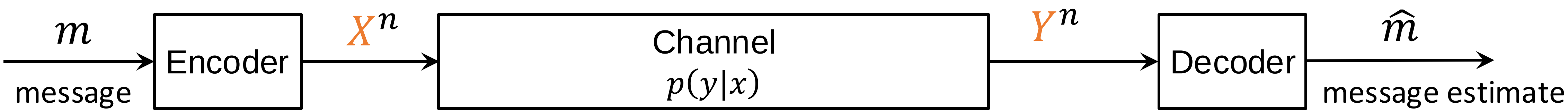
Max of $H(Y) = 1$ (thus also max of $I(X;Y)$)
achieved for $p(Y{=}0) = r(1-q) + (1-r)q = 0.5$,
thus $r = 0.5$ uniform input and output distrib.

$\leq 1 - H(q)$

Hence, capacity for binary symmetric channel is $C = 1 - H(q)$

Source: [Cover, Thomas'06] Elements of Information Theory, 2006. Chapter 7 channel capacity, https://www.doi.org/10.1002/047174882X
Gatterbauer, Aslam. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Capacity of binary symmetric channel



channel capacity $C = \max_{p(x)} I(X;Y)$

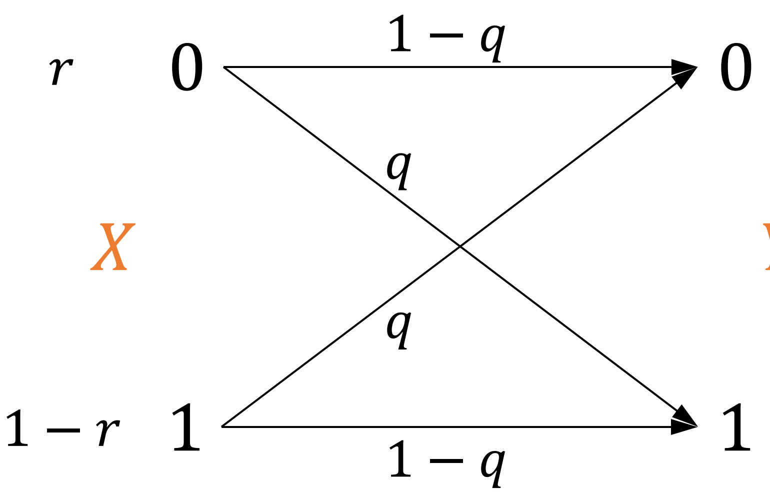
Trying to do it the other way around should work but becomes far more complicated ☹

$$I(X;Y) = H(X) - H(X|Y)$$

$H(X) = H(r)$

$$H(X|Y) = H(X|Y{=}0)\cdot p(Y{=}0) + H(X|Y{=}1)\cdot p(Y{=}1)$$

$p(Y{=}0) = (1-q)r + q(1-r)$ $\quad$ $p(Y{=}1) = rq + (1-r)(1-q)$



$$p(X{=}0|Y{=}0) = \frac{p(Y{=}0|X{=}0)\cdot p(X{=}0)}{p(Y{=}0)} = \frac{(1-q)r}{(1-q)r+q(1-r)}$$

$$p(X{=}0|Y{=}1) = \frac{p(Y{=}1|X{=}0)\cdot p(X{=}0)}{p(Y{=}1)} = \frac{qr}{rq+(1-r)(1-q)}$$

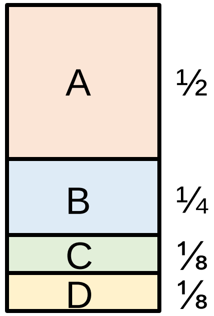
$$I(X;Y) = H(r) - H\left(\frac{(1-q)r}{(1-q)r+q(1-r)}\right)\big((1-q)r + q(1-r)\big)$$
$$- H\left(\frac{qr}{rq+(1-r)(1-q)}\right)\big(rq + (1-r)(1-q)\big)$$

Source: [Cover, Thomas'06] Elements of Information Theory, 2006. Chapter 7 channel capacity, https://www.doi.org/10.1002/047174882X
Gatterbauer, Aslam. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Capacity of binary symmetric channel

rate $R = I(X;Y) = H(Y) - H(q)$

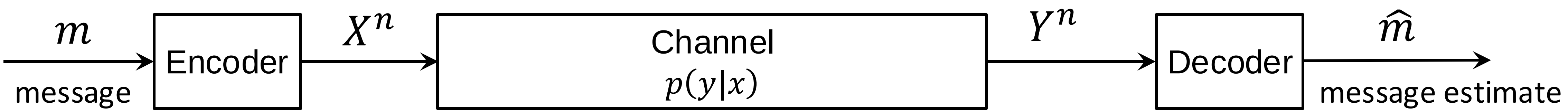
Source symbols & frequency



What is the maximal achievable rate? Assume $q = 0.2$: ?

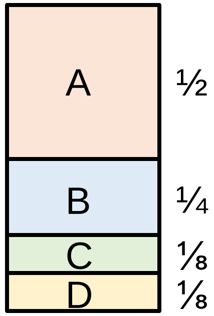
Can you create a code that achieves that rate?
How much does this optimized code buys us over the most naive code you can imagine? ?

Capacity of binary symmetric channel

rate $R = I(X;Y) = H(Y) - H(q)$

Source symbols & frequency



What is the maximal achievable rate? Assume $q = 0.2$:

channel capacity $C = \max_{p(x)} I(X;Y) = C = 1 - H(q)$

$$= 1 - H(0.2)$$

$$= 1 - 0.722 = 0.278$$

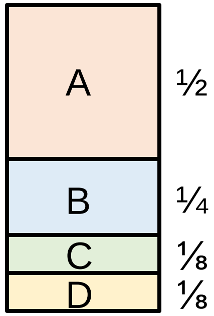
Can you create a code that achieves that rate?
How much does this optimized code buys us over the most naive code you can imagine?

?

Capacity of binary symmetric channel

rate $R = I(X;Y) = H(Y) - H(q)$

Source symbols & frequency



What is the maximal achievable rate? Assume $q = 0.2$:

channel capacity $C = \max_{p(x)} I(X;Y) = C = 1 - H(q)$

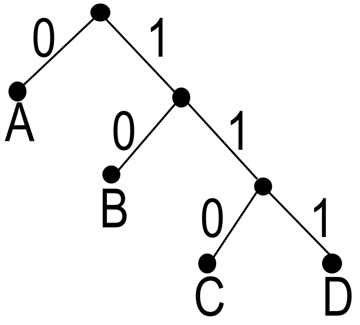
$= 1 - H(0.2)$

$= 1 - 0.722 = 0.278$

symbol/codeword/frequency

Code opt.:

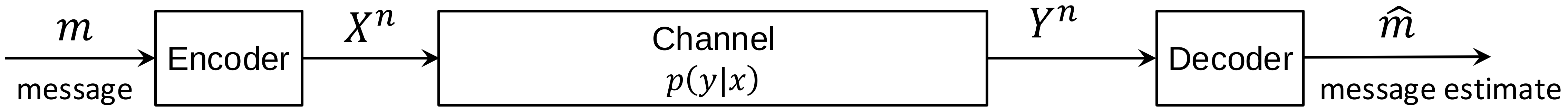
A	0	½
A	10	¼
C	110	⅛
D	111	⅛



Code opt: $R_2 = 1.000 - 0.722 = 0.278$

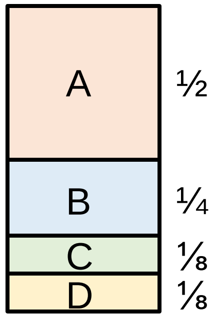
Compare achievable rates for optimized vs. most naive code

Capacity of binary symmetric channel



rate $R = I(X;Y) = H(Y) - H(q)$

Source symbols & frequency



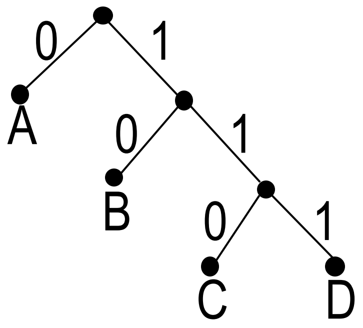
Code naive:

?

symbol/codeword/frequency

Code opt.:

A	0	½
A	10	¼
C	110	⅛
D	111	⅛



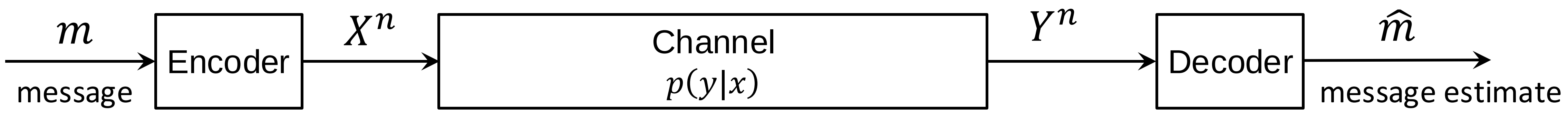
Code naive:

Achievable rate ?

Code opt:

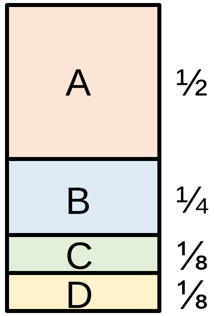
$R_2 = 1.000 - 0.722 = 0.278$

Capacity of binary symmetric channel



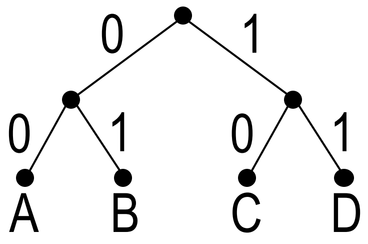
rate $R = I(X;Y) = H(Y) - H(q)$

Source symbols & frequency



Code naive:

A	00	½
B	01	¼
C	10	⅛
D	11	⅛



Code naive:

Fraction of 0's in X: $\frac{11}{16} = 0.688$

... in Y: $\frac{11}{16}0.8 + \frac{5}{16}0.2 = 0.613$

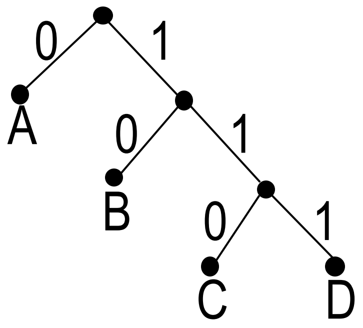
$R_1 = H(0.613) - H(0.2)$

$= 0.963 - 0.722 = 0.241$

symbol/codeword/frequency

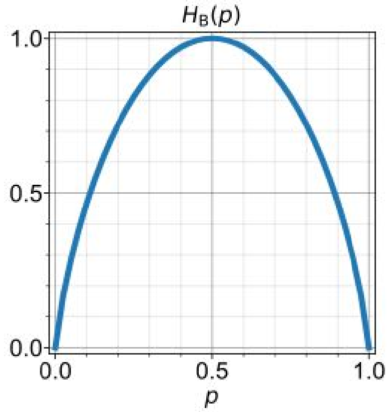
Code opt.:

A	0	½
A	10	¼
C	110	⅛
D	111	⅛



Code opt:

$R_2 = 1.000 - 0.722 = 0.278$



The distortion hits us far more than an optimal code can buy us

Distortion Theory

Largely based on chapter 10 of
[Cover, Thomas'06] Elements of Information Theory, 2006. https://www.doi.org/10.1002/047174882X

Rate distortion theory

- A finite representation of a continuous RV can never be perfect
- How well can we represent it?
- Requires a notion of "goodness" of a representation
  - Distortion measure: distance between RV and its representation

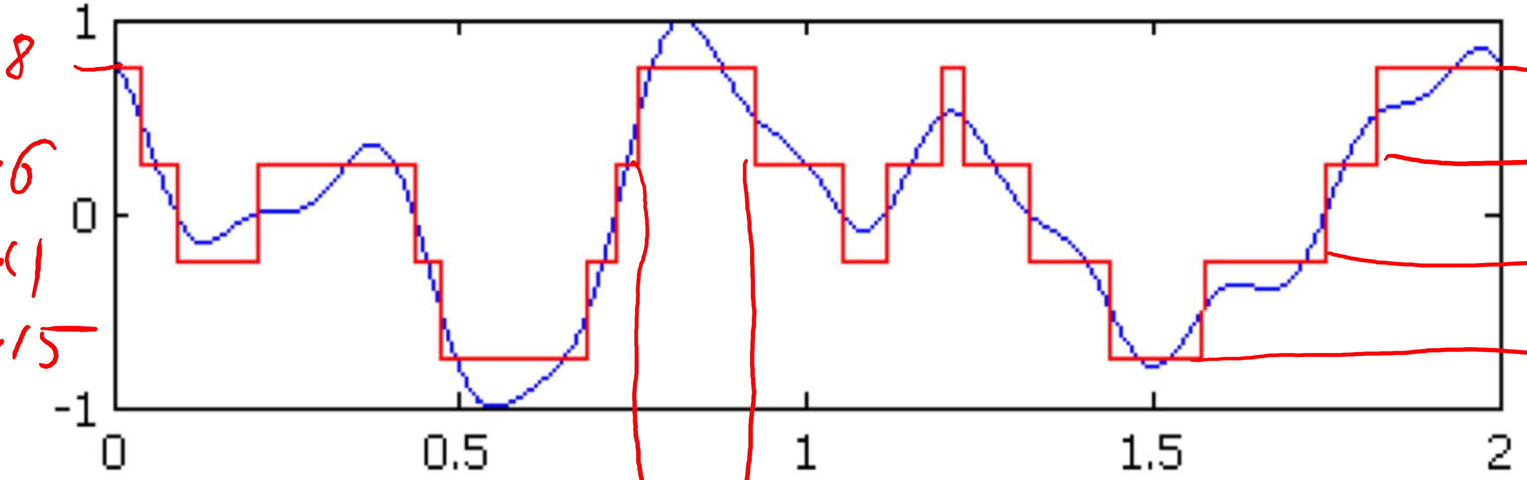
- Rate distortion theory:
  - Given: source distribution p and a distortion measure d
  - Describes: trade-off between communication rate R and distortion d
  - Lossy compression framework with zero-error data compression (earlier topics in class) a special case

Quantization

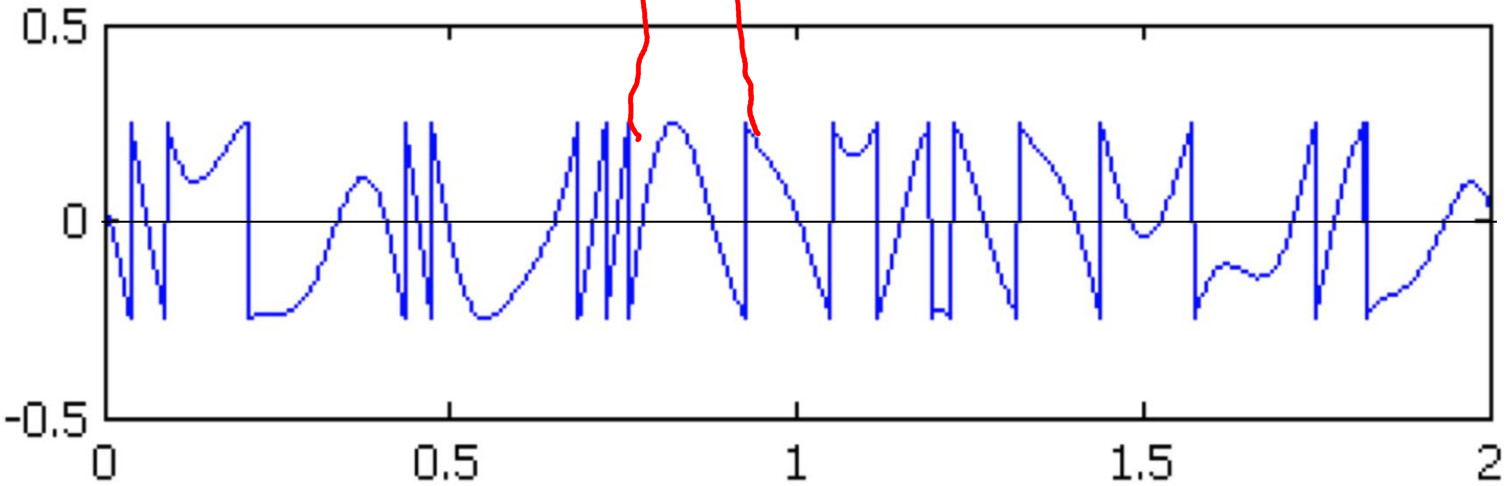
- Let X be a continuous RV (e.g. from a Gaussian distribution)
- We approximate X by $\hat{X}$
- Using R bits to represent X, then $\hat{X}(X)$ has 2^R possible values
  - Example $R = 8$ bits, then then $\hat{X}$ has $2^8 = 256$ possible values

- Goal: find the optimal set of values ("**representatives**") for $\hat{X}$ and associated regions ("**assignment regions**") for each value

Original and Quantized Signal

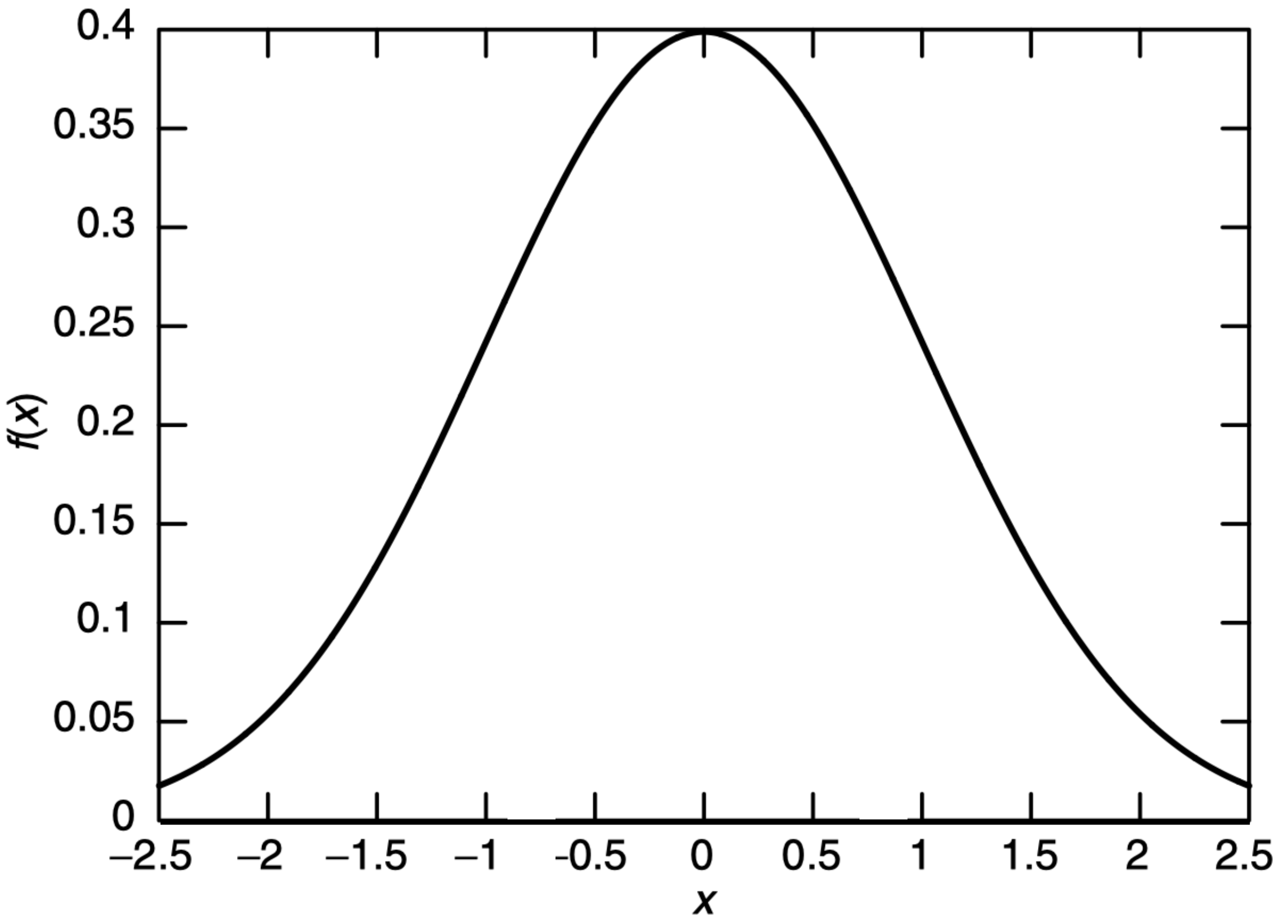


Quantization Error



Source: Bernd Girod, 2012: https://web.stanford.edu/class/ee398a/handouts/lectures/05-Quantization.pdf

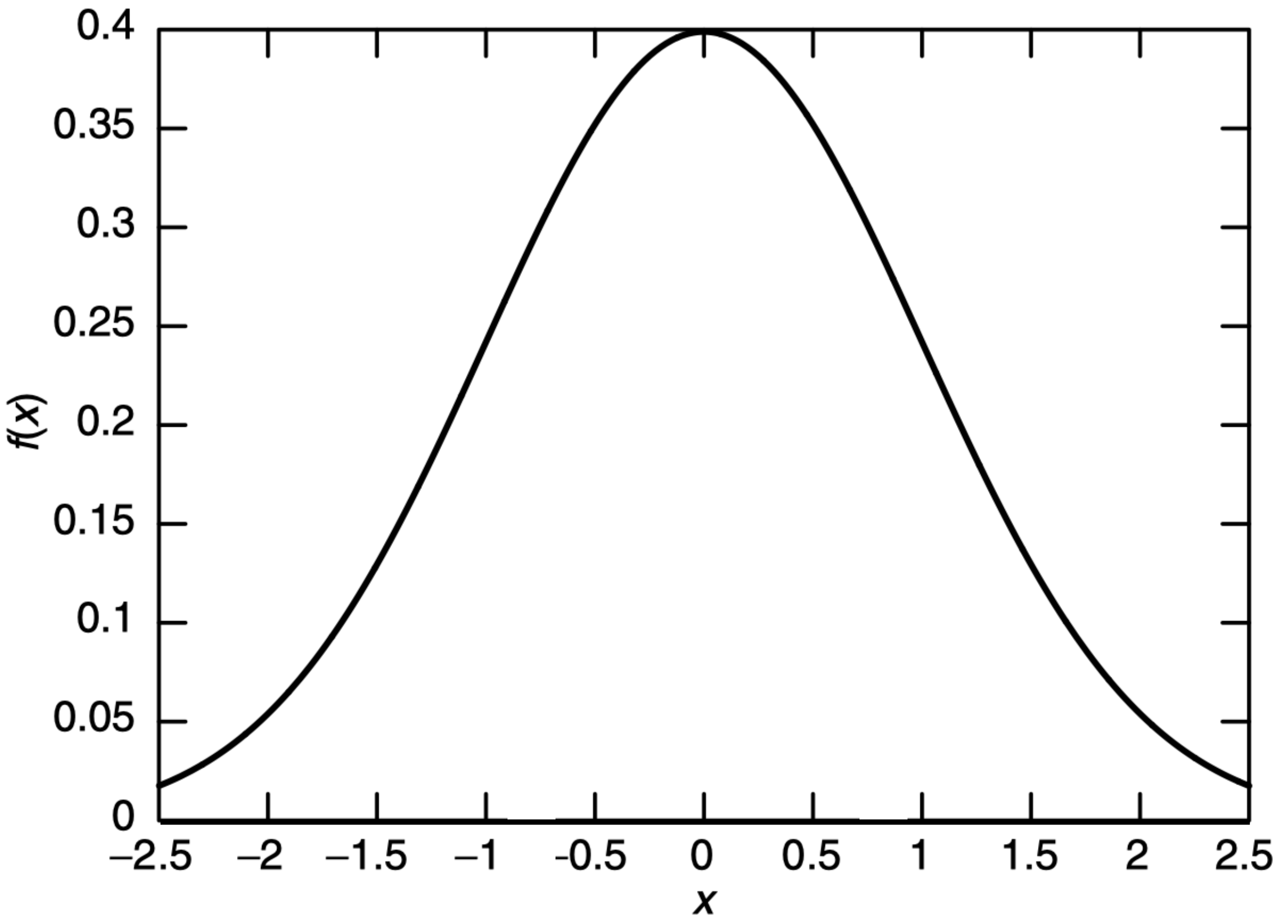
Quantization of a Gaussian



Assume you have $R = 1$ bit (2 values). What is the best way to quantize a Gaussian distribution ?

What is an appropriate measure of distortion ?

Quantization of a Gaussian

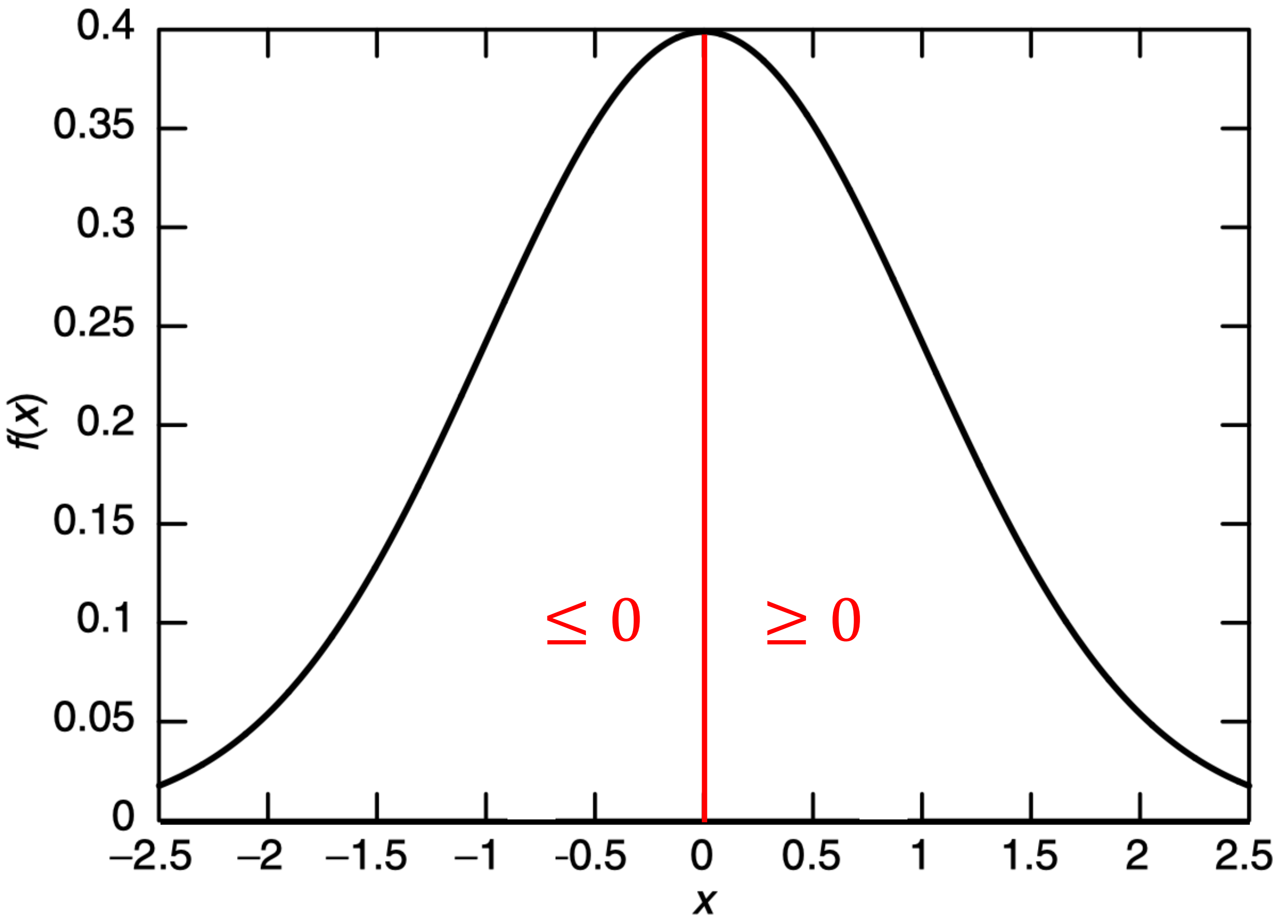


Assume you have $R = 1$ bit (2 values). What is the best way to quantize a Gaussian distribution

?

Assume we like to minimize the mean of squared errors (MSE)

Quantization of a Gaussian

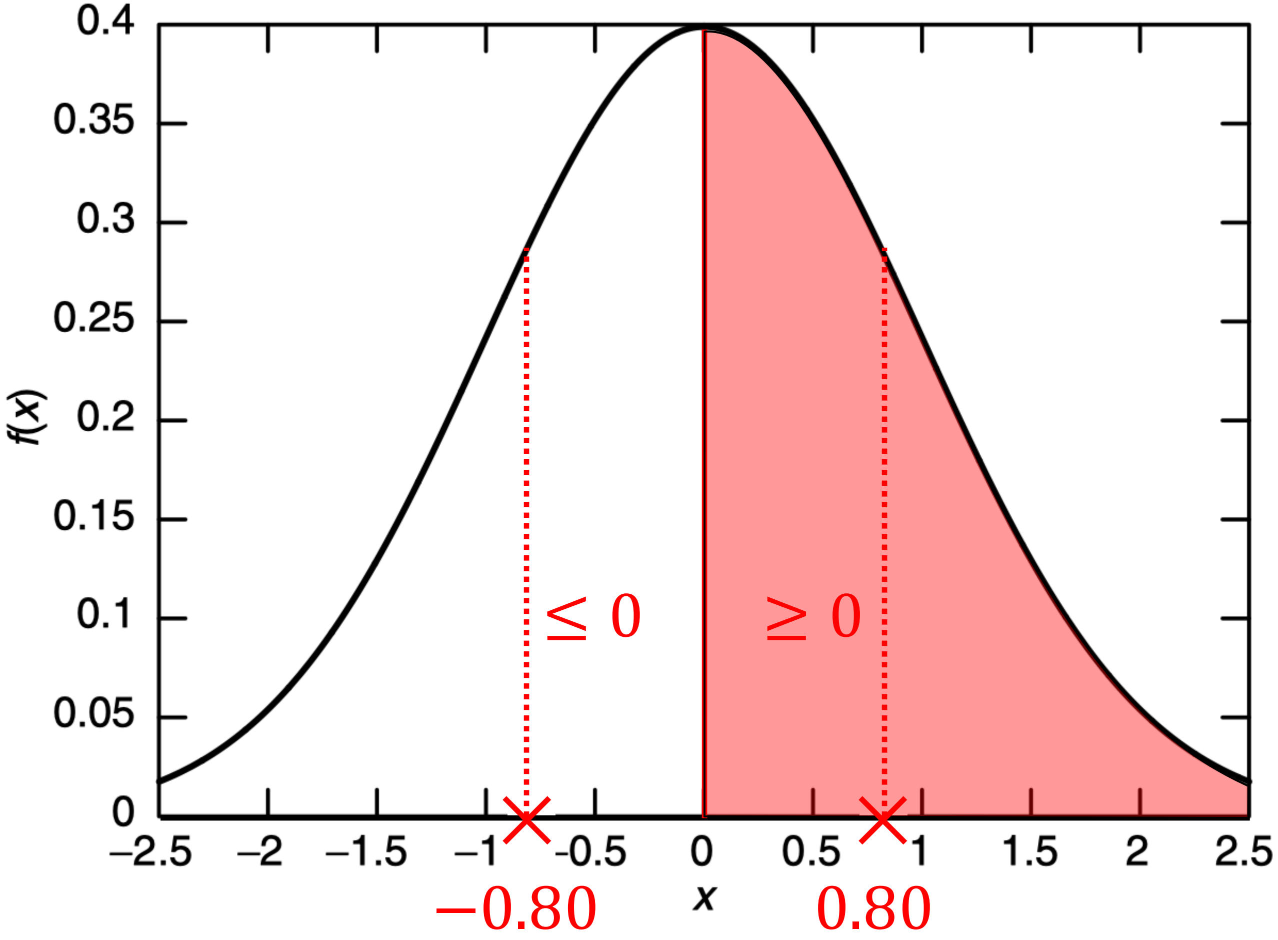


Assume you have $R = 1$ bit (2 values). What is the best way to quantize a Gaussian distribution ?

Assume we like to minimize the mean of squared errors (MSE)

If we have 2 values. It makes sense to choose ≥ 0 and ≤ 0. But what should be the representatives?

Quantization of a Gaussian



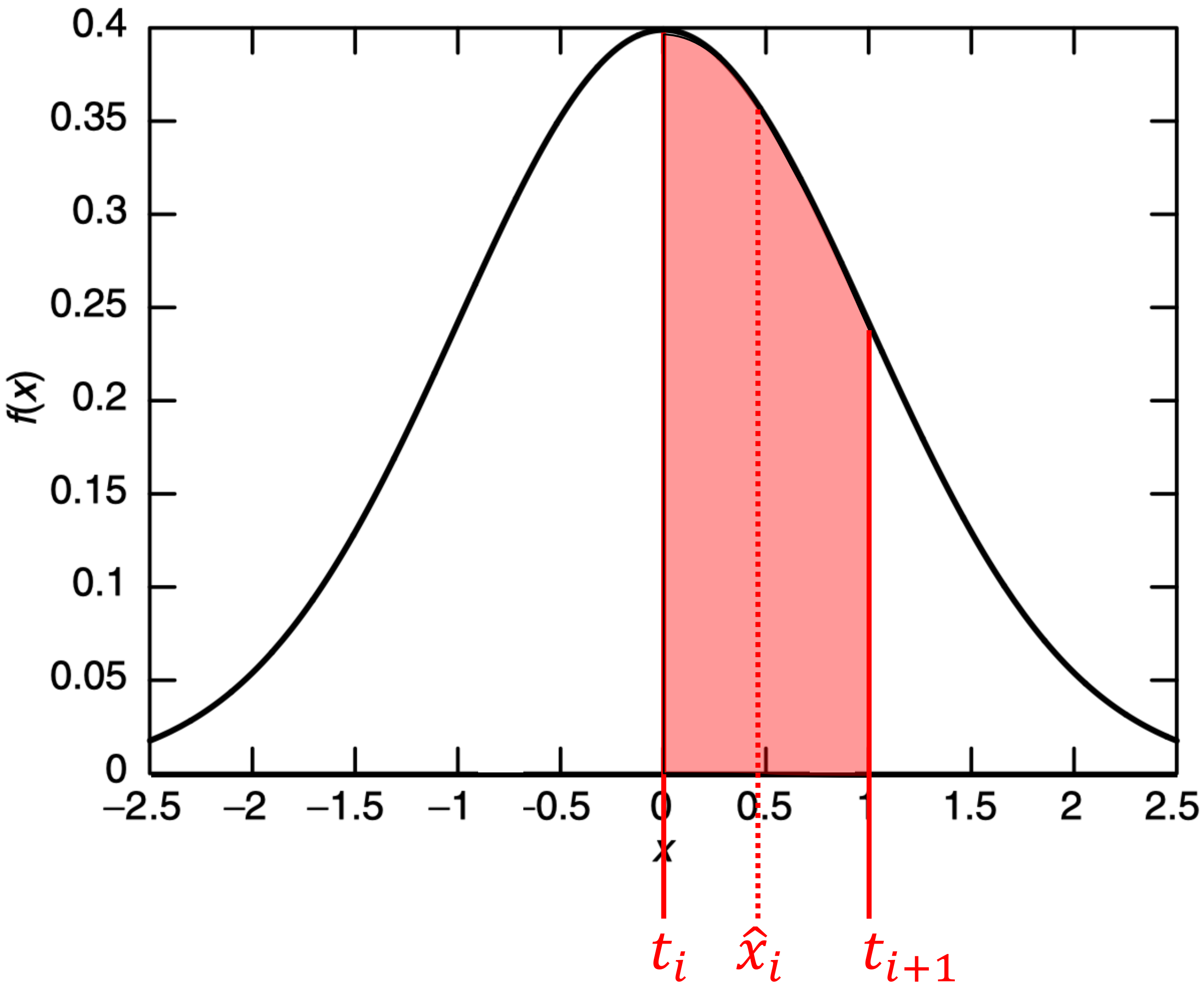
Assume you have $R = 1$ bit (2 values). What is the best way to quantize a Gaussian distribution

Assume we like to minimize the mean of squared errors (MSE)

The (conditional) mean (centroid) of a region minimizes the MSE!

If we have 2 values. It makes sense to choose ≥ 0 and ≤ 0. But what should be the representatives?

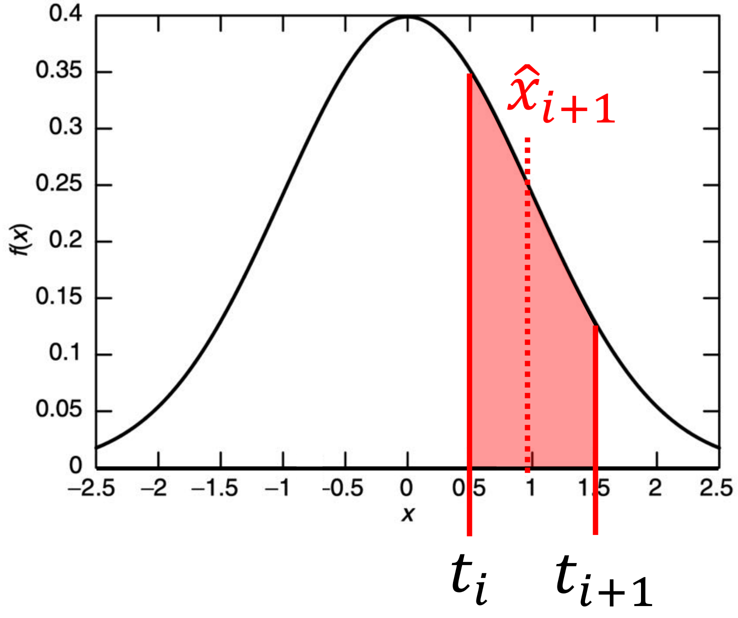
Quantization of a Gaussian



Assume you have $R = 2$ bits (4 values). What is the best way to quantize a Gaussian distribution under MSE?

Now we need to determine 3 boundaries $\{t_i\}$ and 4 reconstruction points $\{\hat{x}_i\}$. But how?

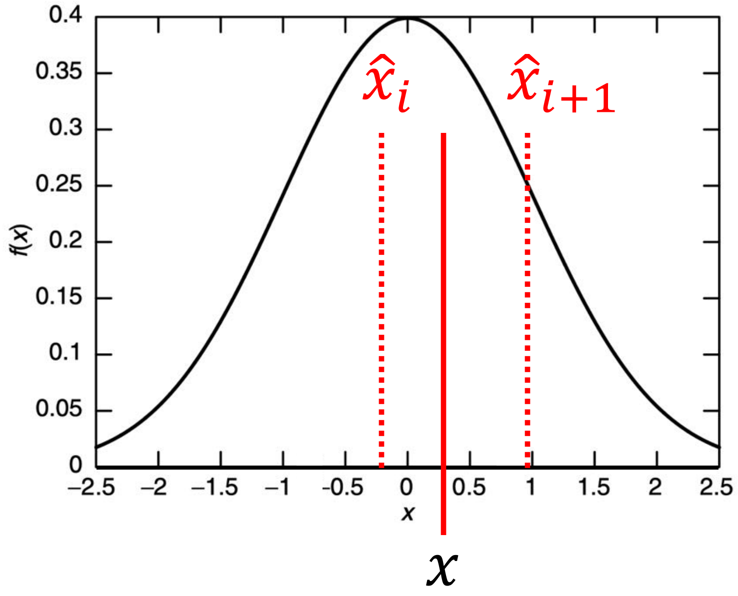
Two properties of optimal boundaries and construction points



$\{t_i\} \Rightarrow \{\hat{x}_{i+1}\}$

Given two thresholds t_i, t_{i+1} marking the boundaries of a region. What is the best representative $\hat{x}_{i+1}$ of the region?

?

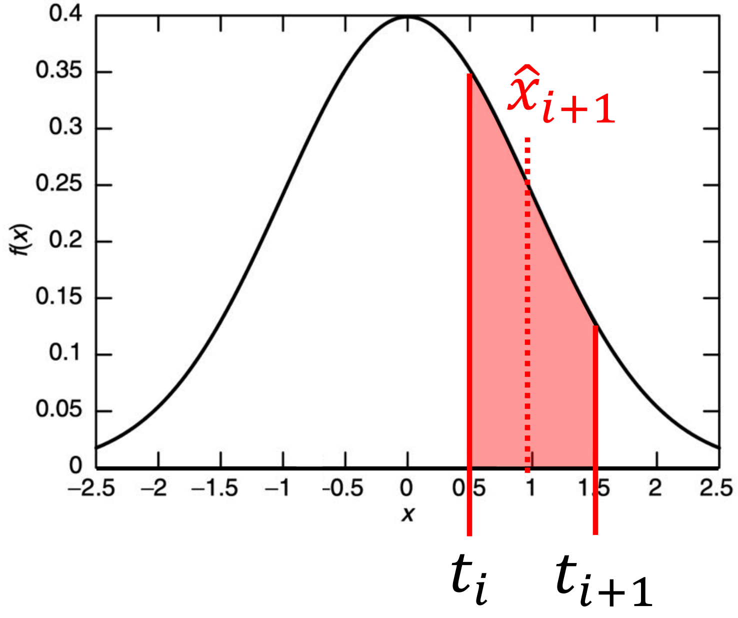


$\{\hat{x}_i\} \Rightarrow \{t_i\}$

Given a set of representative values $\{\hat{x}_{i+1}\}$, which representative should we choose for any given x?

?

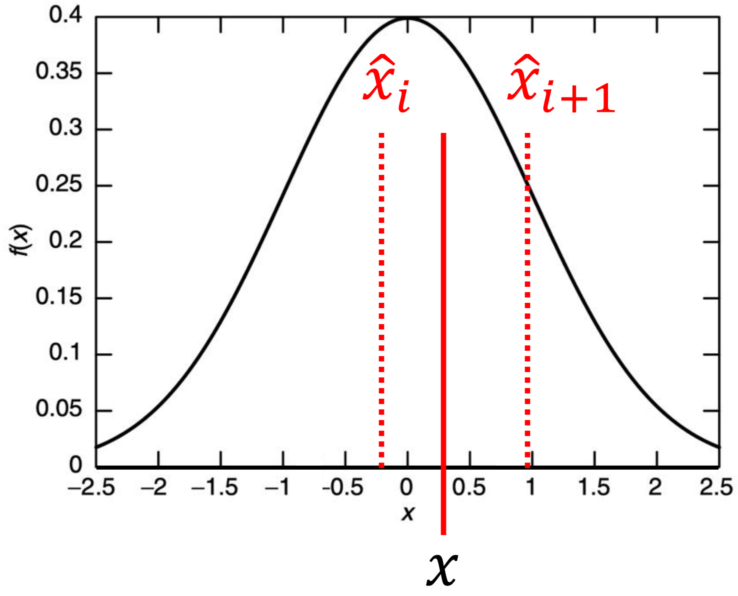
Two properties of optimal boundaries and construction points



$\{t_i\} \Rightarrow \{\hat{x}_{i+1}\}$

Given two thresholds t_i, t_{i+1} marking the boundaries of a region. What is the best representative $\hat{x}_{i+1}$ of the region?

The conditional means (conditioned on the region = centroids) minimize the MSE and should thus be the reconstruction points.



$\{\hat{x}_i\} \Rightarrow \{t_i\}$

Given a set of representative values $\{\hat{x}_{i+1}\}$, which representative should we choose for any given x?

Distortion (MSE) is minimized by assigning values to their closest points. Thus a Voronoi partition gives use the optimal thresholds.

Lloyd-Max scalar quantizer

Problem: For a signal x with given PDF $f_X(x)$ find a quantizer with m representative levels (or "codes" that minimizes

$$d = MSE = \mathbb{E}[(X-\hat{X})^2]$$

Lloyd-Max quantizer

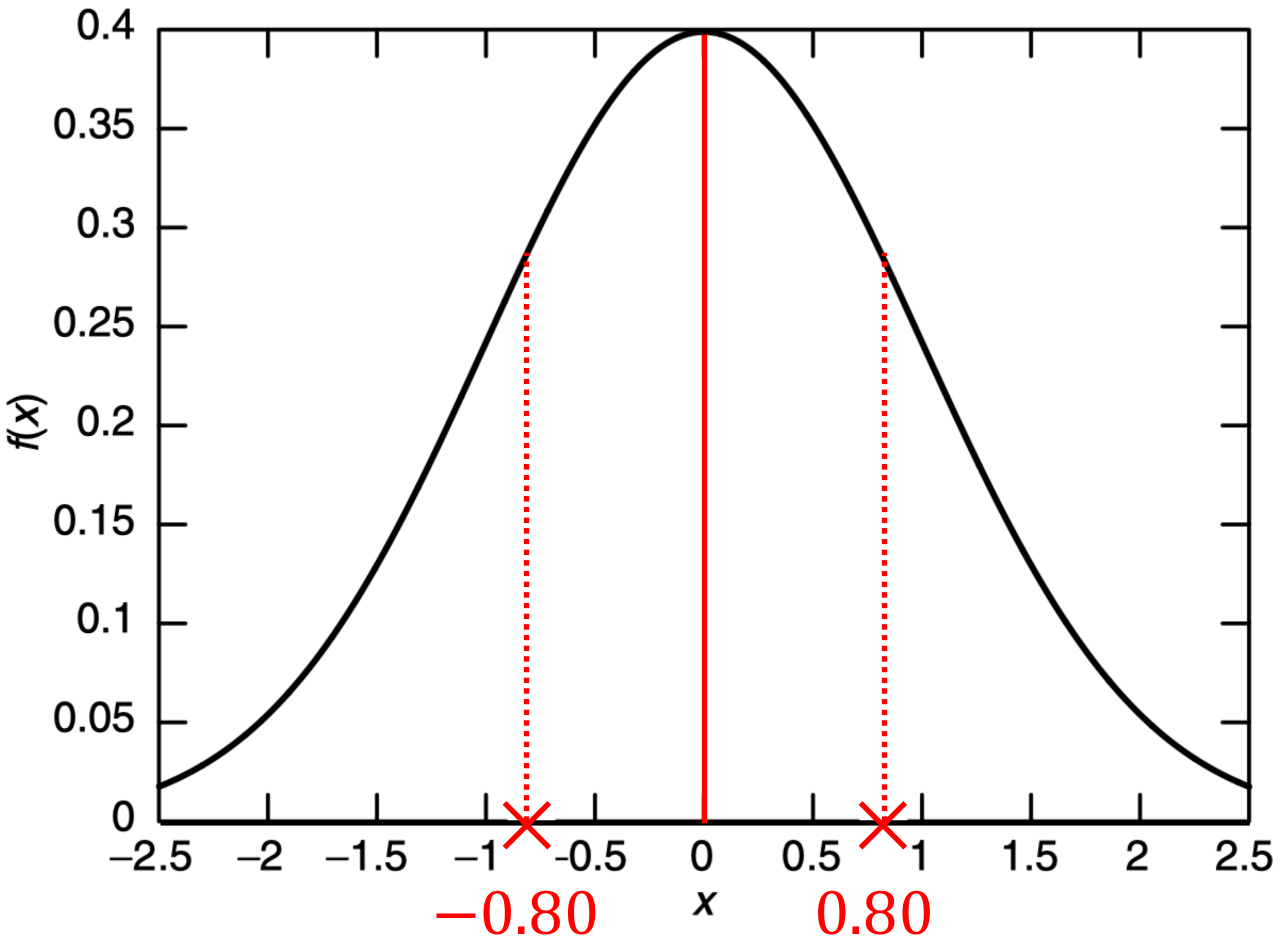
Input: initial vector $\hat{\mathbf{x}}$ of m representative levels
Repeat {

- Create $m-1$ decision thresholds $\mathbf{t}$ exactly half-way between representative levels
- Create m representative levels $\hat{\mathbf{x}}$ as the centroids of PDF between two successive decision thresholds

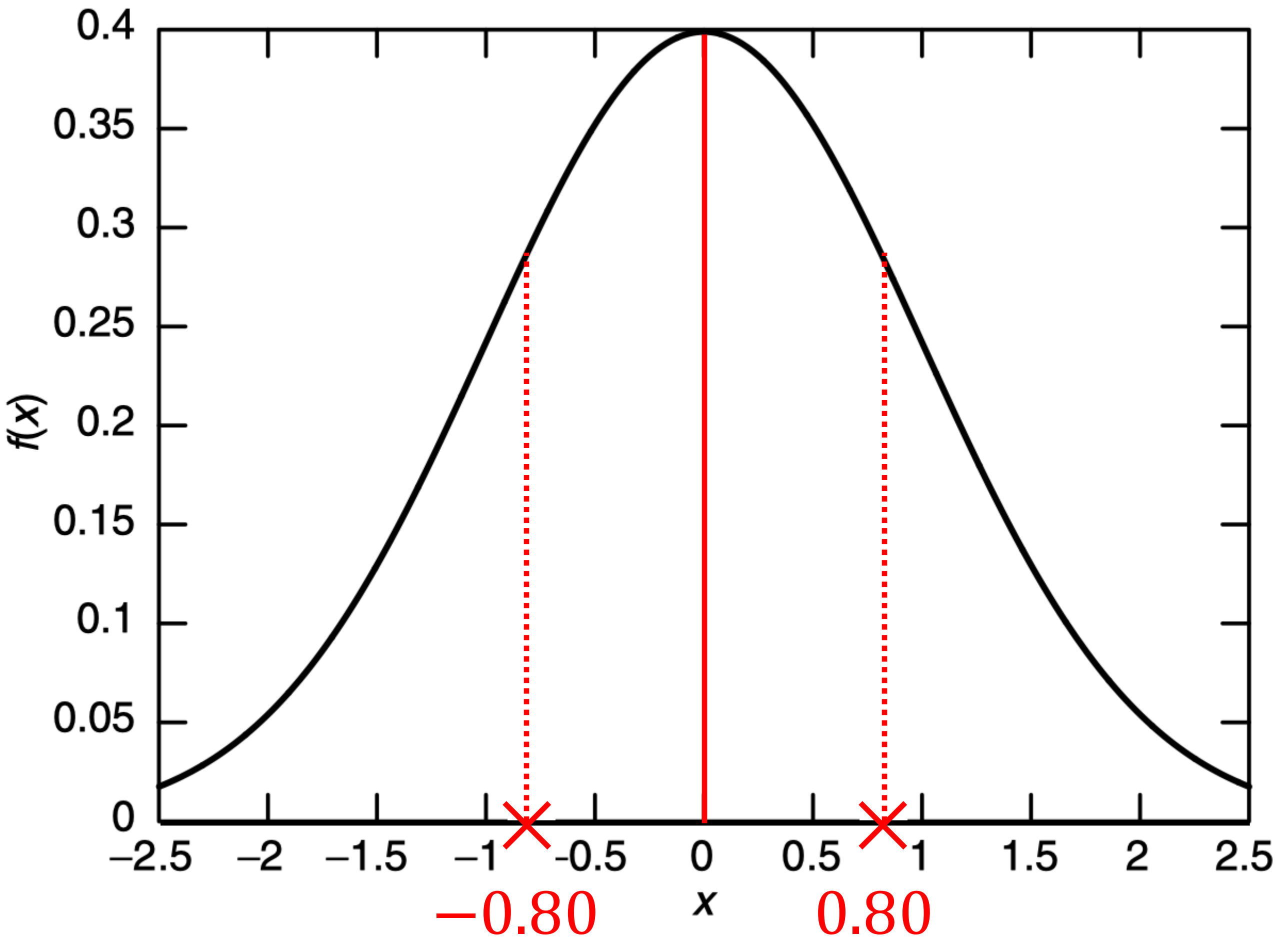
until (likely) convergence}

$$t_i = \frac{\hat{x}_{i-1}+\hat{x}_i}{2}, \quad i = 1, \dots, m-1$$

$$\hat{x}_i = \frac{\int_{t_i}^{t_{i+1}} x \cdot f_X(x) dx}{\int_{t_i}^{t_{i+1}} f_X(x) dx}, \quad i = 0, \dots, m-1$$

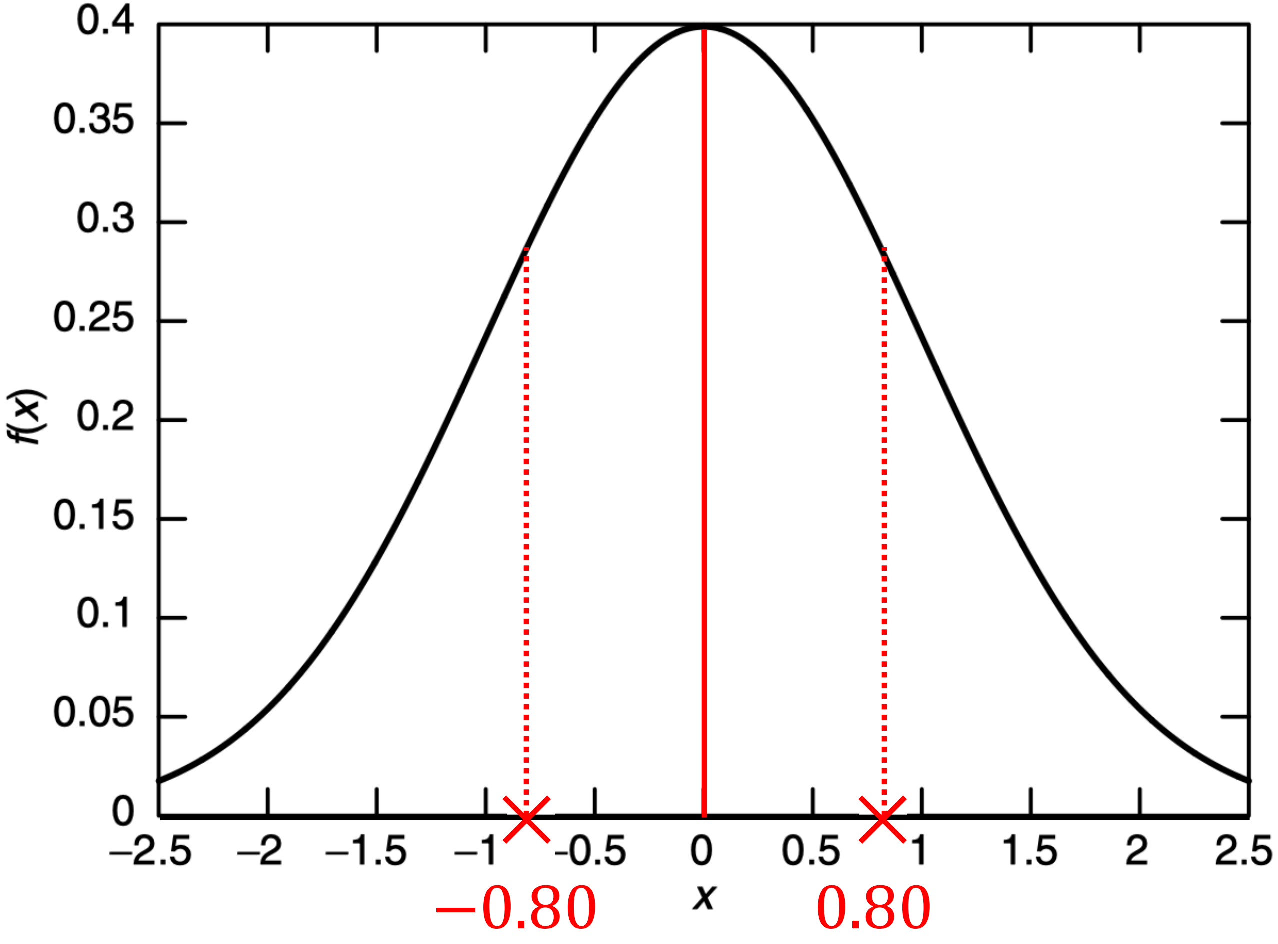



```
 0: representatives: [1 2], d: 1.9414
 1: representatives: [-0.1388  1.9387], d: 0.6019
 2: representatives: [-0.3261  1.4456], d: 0.4686
 3: representatives: [-0.479   1.1851], d: 0.4073
 4: representatives: [-0.5875  1.0354], d: 0.3814
 5: representatives: [-0.661   0.9457], d: 0.3707
 6: representatives: [-0.7095  0.8907], d: 0.3664
 7: representatives: [-0.7411  0.8564], d: 0.3646
 8: representatives: [-0.7616  0.8349], d: 0.3639
 9: representatives: [-0.7747  0.8214], d: 0.3636
10: representatives: [-0.7831  0.8128], d: 0.3635
11: representatives: [-0.7884  0.8074], d: 0.3634
12: representatives: [-0.7919  0.8039], d: 0.3634
13: representatives: [-0.7941  0.8017], d: 0.3634
14: representatives: [-0.7954  0.8003], d: 0.3634
15: representatives: [-0.7963  0.7994], d: 0.3634
16: representatives: [-0.7969  0.7989], d: 0.3634
17: representatives: [-0.7973  0.7985], d: 0.3634
18: representatives: [-0.7975  0.7983], d: 0.3634
19: representatives: [-0.7976  0.7981], d: 0.3634
20: representatives: [-0.7977  0.798 ], d: 0.3634
21: representatives: [-0.7978  0.798 ], d: 0.3634
22: representatives: [-0.7978  0.798 ], d: 0.3634
23: representatives: [-0.7978  0.7979], d: 0.3634
24: representatives: [-0.7979  0.7979], d: 0.3634
25: representatives: [-0.7979  0.7979], d: 0.3634
26: representatives: [-0.7979  0.7979], d: 0.3634
27: representatives: [-0.7979  0.7979], d: 0.3634
28: representatives: [-0.7979  0.7979], d: 0.3634
29: representatives: [-0.7979  0.7979], d: 0.3634
```

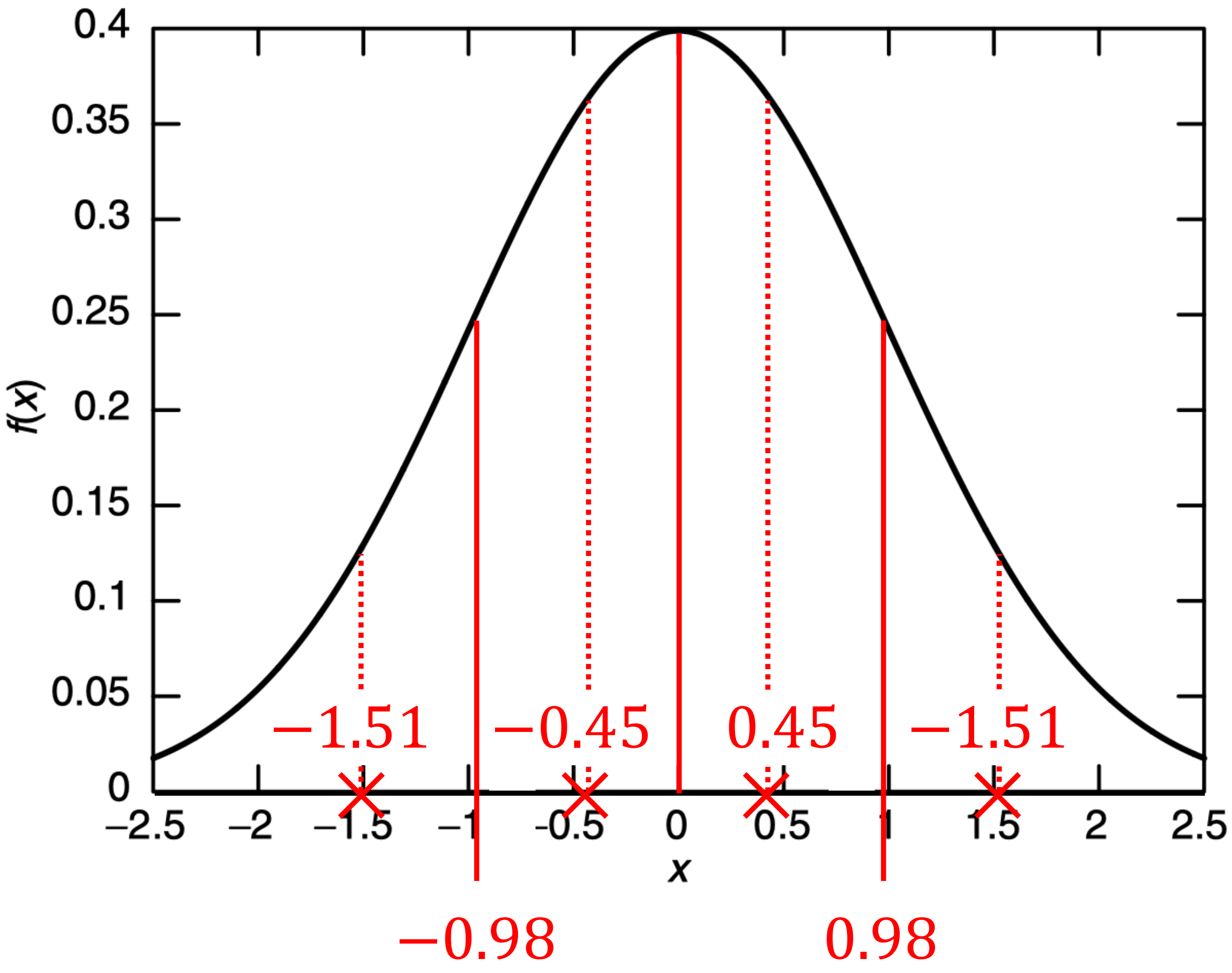



```
 0: representatives: [2 3], d: 4.9960
 1: representatives: [-0.0176  2.8227], d: 0.7932
 2: representatives: [-0.1622  1.8562], d: 0.5800
 3: representatives: [-0.3477  1.404 ], d: 0.4579
 4: representatives: [-0.4948  1.1617], d: 0.4027
 5: representatives: [-0.5984  1.0216], d: 0.3795
 6: representatives: [-0.6682  0.9373], d: 0.3699
 7: representatives: [-0.7143  0.8854], d: 0.3660
 8: representatives: [-0.7442  0.8532], d: 0.3645
 9: representatives: [-0.7635  0.8329], d: 0.3638
10: representatives: [-0.7759  0.8201], d: 0.3636
11: representatives: [-0.7839  0.812 ], d: 0.3635
12: representatives: [-0.789   0.8069], d: 0.3634
13: representatives: [-0.7922  0.8036], d: 0.3634
14: representatives: [-0.7943  0.8015], d: 0.3634
15: representatives: [-0.7956  0.8002], d: 0.3634
16: representatives: [-0.7964  0.7994], d: 0.3634
17: representatives: [-0.7969  0.7988], d: 0.3634
18: representatives: [-0.7973  0.7985], d: 0.3634
19: representatives: [-0.7975  0.7983], d: 0.3634
20: representatives: [-0.7976  0.7981], d: 0.3634
21: representatives: [-0.7977  0.798 ], d: 0.3634
22: representatives: [-0.7978  0.798 ], d: 0.3634
23: representatives: [-0.7978  0.7979], d: 0.3634
24: representatives: [-0.7978  0.7979], d: 0.3634
25: representatives: [-0.7979  0.7979], d: 0.3634
26: representatives: [-0.7979  0.7979], d: 0.3634
27: representatives: [-0.7979  0.7979], d: 0.3634
28: representatives: [-0.7979  0.7979], d: 0.3634
29: representatives: [-0.7979  0.7979], d: 0.3634
```

Python file 232: https://github.com/northeastern-datalab/cs7840-activities/tree/main/notebooks



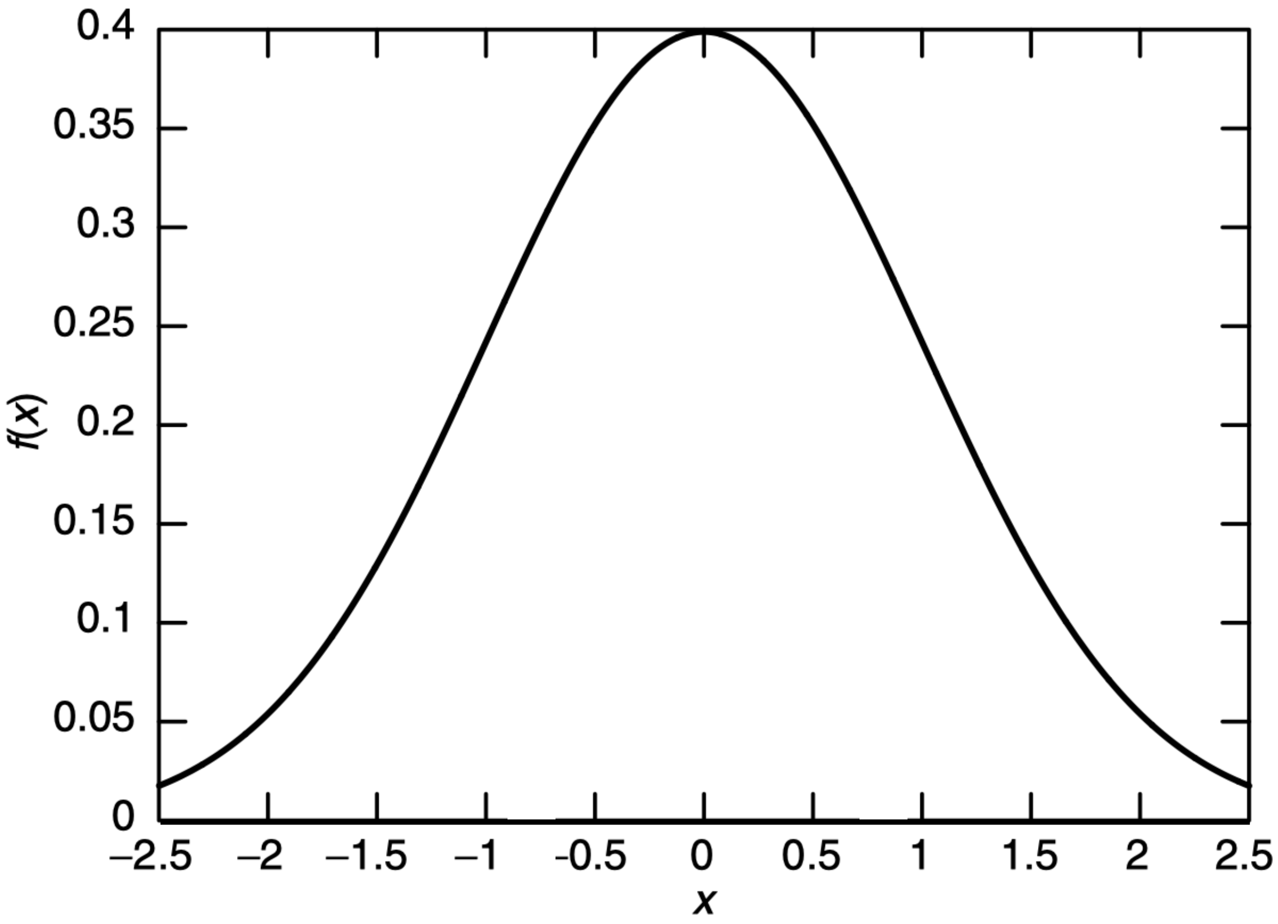

```
 0: representatives: [2 3], d: 4.9960
 1: representatives: [-0.0176  2.8227], d: 0.7932
 2: representatives: [-0.1622  1.8562], d: 0.5800
 3: representatives: [-0.3477  1.404 ], d: 0.4579
 4: representatives: [-0.4948  1.1617], d: 0.4027
 5: representatives: [-0.5984  1.0216], d: 0.3795
 6: representatives: [-0.6682  0.9373], d: 0.3699
 7: representatives: [-0.7143  0.8854], d: 0.3660
 8: representatives: [-0.7442  0.8532], d: 0.3645
 9: representatives: [-0.7635  0.8329], d: 0.3638
10: representatives: [-0.7759  0.8201], d: 0.3636
11: representatives: [-0.7839  0.812 ], d: 0.3635
12: representatives: [-0.789   0.8069], d: 0.3634
13: representatives: [-0.7922  0.8036], d: 0.3634
14: representatives: [-0.7943  0.8015], d: 0.3634
15: representatives: [-0.7956  0.8002], d: 0.3634
16: representatives: [-0.7964  0.7994], d: 0.3634
17: representatives: [-0.7969  0.7988], d: 0.3634
18: representatives: [-0.7973  0.7985], d: 0.3634
19: representatives: [-0.7975  0.7983], d: 0.3634
20: representatives: [-0.7976  0.7981], d: 0.3634
21: representatives: [-0.7977  0.798 ], d: 0.3634
22: representatives: [-0.7978  0.798 ], d: 0.3634
23: representatives: [-0.7978  0.7979], d: 0.3634
24: representatives: [-0.7978  0.7979], d: 0.3634
25: representatives: [-0.7979  0.7979], d: 0.3634
26: representatives: [-0.7979  0.7979], d: 0.3634
27: representatives: [-0.7979  0.7979], d: 0.3634
28: representatives: [-0.7979  0.7979], d: 0.3634
29: representatives: [-0.7979  0.7979], d: 0.3634
```




```
 0: representatives: [-1   0   2   3], d: 0.2671
 1: representatives: [-1.1411  0.2066  1.4723  2.8227], d: 0.1828
 2: representatives: [-1.1172  0.1611  1.303   2.5045], d: 0.1675
 3: representatives: [-1.125   0.1123  1.1789  2.2883], d: 0.1567
 4: representatives: [-1.1458  0.0622  1.0796  2.1395], d: 0.1485
 5: representatives: [-1.1718  0.0131  0.9976  2.0323], d: 0.1420
 6: representatives: [-1.1996 -0.0335  0.9285  1.9514], d: 0.1369
 7: representatives: [-1.2274 -0.0768  0.8698  1.8878], d: 0.1328
 8: representatives: [-1.2542 -0.1165  0.8193  1.8362], d: 0.1296
 9: representatives: [-1.2794 -0.1526  0.7757  1.7935], d: 0.1271
10: representatives: [-1.3027 -0.185   0.7376  1.7575], d: 0.1250
11: representatives: [-1.3241 -0.2142  0.7044  1.7268], d: 0.1234
12: representatives: [-1.3435 -0.2403  0.6752  1.7004], d: 0.1222
13: representatives: [-1.3611 -0.2637  0.6496  1.6775], d: 0.1212
14: representatives: [-1.3771 -0.2846  0.627   1.6576], d: 0.1204
15: representatives: [-1.3914 -0.3032  0.6071  1.6403], d: 0.1198

    ...

67: representatives: [-1.5102 -0.4525  0.4531  1.5107], d: 0.1175
68: representatives: [-1.5102 -0.4525  0.4531  1.5106], d: 0.1175
69: representatives: [-1.5102 -0.4525  0.453   1.5106], d: 0.1175
70: representatives: [-1.5102 -0.4526  0.453   1.5106], d: 0.1175
71: representatives: [-1.5103 -0.4526  0.453   1.5106], d: 0.1175
72: representatives: [-1.5103 -0.4526  0.453   1.5106], d: 0.1175
73: representatives: [-1.5103 -0.4526  0.4529  1.5105], d: 0.1175
74: representatives: [-1.5103 -0.4526  0.4529  1.5105], d: 0.1175
75: representatives: [-1.5103 -0.4527  0.4529  1.5105], d: 0.1175
76: representatives: [-1.5103 -0.4527  0.4529  1.5105], d: 0.1175
77: representatives: [-1.5103 -0.4527  0.4529  1.5105], d: 0.1175
78: representatives: [-1.5103 -0.4527  0.4529  1.5105], d: 0.1175
79: representatives: [-1.5104 -0.4527  0.4529  1.5105], d: 0.1175
```




```
 0: representatives: [-2.5 -1.5 -0.5  0.5  1.5  2.5], d: 0.0849
 1: representatives: [-2.3732 -1.3832 -0.4599  0.4599  1.3832  2.3732], d: 0.0748
 2: representatives: [-2.2658 -1.2991 -0.4292  0.4292  1.2991  2.2658], d: 0.0686
 3: representatives: [-2.182  -1.2349 -0.4059  0.4059  1.2349  2.182 ], d: 0.0647
 4: representatives: [-2.1177 -1.1851 -0.3878  0.3878  1.1851  2.1177], d: 0.0622
 5: representatives: [-2.0683 -1.1461 -0.3734  0.3734  1.1461  2.0683], d: 0.0607
 6: representatives: [-2.0303 -1.1154 -0.362   0.362   1.1154  2.0303], d: 0.0597
 7: representatives: [-2.0009 -1.0912 -0.3529  0.3529  1.0912  2.0009], d: 0.0590
 8: representatives: [-1.9779 -1.0722 -0.3456  0.3456  1.0722  1.9779], d: 0.0586
 9: representatives: [-1.96   -1.0571 -0.3399  0.3399  1.0571  1.96  ], d: 0.0584
10: representatives: [-1.946  -1.0452 -0.3353  0.3353  1.0452  1.946 ], d: 0.0582
11: representatives: [-1.9349 -1.0358 -0.3317  0.3317  1.0358  1.9349], d: 0.0581
12: representatives: [-1.9263 -1.0284 -0.3288  0.3288  1.0284  1.9263], d: 0.0581
13: representatives: [-1.9194 -1.0225 -0.3265  0.3265  1.0225  1.9194], d: 0.0580
14: representatives: [-1.914  -1.0179 -0.3247  0.3247  1.0179  1.914 ], d: 0.0580
15: representatives: [-1.9098 -1.0142 -0.3232  0.3232  1.0142  1.9098], d: 0.0580
16: representatives: [-1.9064 -1.0112 -0.3221  0.3221  1.0112  1.9064], d: 0.0580
17: representatives: [-1.9037 -1.0089 -0.3212  0.3212  1.0089  1.9037], d: 0.0580
18: representatives: [-1.9016 -1.0071 -0.3205  0.3205  1.0071  1.9016], d: 0.0580
19: representatives: [-1.8999 -1.0056 -0.3199  0.3199  1.0056  1.8999], d: 0.0580
20: representatives: [-1.8986 -1.0045 -0.3194  0.3194  1.0045  1.8986], d: 0.0580
21: representatives: [-1.8976 -1.0036 -0.3191  0.3191  1.0036  1.8976], d: 0.0580
22: representatives: [-1.8968 -1.0029 -0.3188  0.3188  1.0029  1.8968], d: 0.0580
23: representatives: [-1.8961 -1.0023 -0.3186  0.3186  1.0023  1.8961], d: 0.0580
24: representatives: [-1.8956 -1.0018 -0.3184  0.3184  1.0018  1.8956], d: 0.0580
25: representatives: [-1.8952 -1.0015 -0.3183  0.3183  1.0015  1.8952], d: 0.0580
26: representatives: [-1.8948 -1.0012 -0.3181  0.3181  1.0012  1.8948], d: 0.0580
27: representatives: [-1.8946 -1.001  -0.3181  0.3181  1.001   1.8946], d: 0.0580
28: representatives: [-1.8944 -1.0008 -0.318   0.318   1.0008  1.8944], d: 0.0580
29: representatives: [-1.8942 -1.0006 -0.3179  0.3179  1.0006  1.8942], d: 0.0580
30: representatives: [-1.8941 -1.0005 -0.3179  0.3179  1.0005  1.8941], d: 0.0580
31: representatives: [-1.894  -1.0004 -0.3178  0.3178  1.0004  1.894 ], d: 0.0580
32: representatives: [-1.8939 -1.0004 -0.3178  0.3178  1.0004  1.8939], d: 0.0580
33: representatives: [-1.8938 -1.0003 -0.3178  0.3178  1.0003  1.8938], d: 0.0580
34: representatives: [-1.8938 -1.0003 -0.3178  0.3178  1.0003  1.8938], d: 0.0580
35: representatives: [-1.8937 -1.0002 -0.3178  0.3178  1.0002  1.8937], d: 0.0580
36: representatives: [-1.8937 -1.0002 -0.3178  0.3178  1.0002  1.8937], d: 0.0580
37: representatives: [-1.8937 -1.0002 -0.3177  0.3177  1.0002  1.8937], d: 0.0580
38: representatives: [-1.8937 -1.0002 -0.3177  0.3177  1.0002  1.8937], d: 0.0580
39: representatives: [-1.8937 -1.0002 -0.3177  0.3177  1.0002  1.8937], d: 0.0580
40: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
41: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
42: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
43: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
44: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
45: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
46: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
47: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
48: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
49: representatives: [-1.8936 -1.0001 -0.3177  0.3177  1.0001  1.8936], d: 0.0580
```

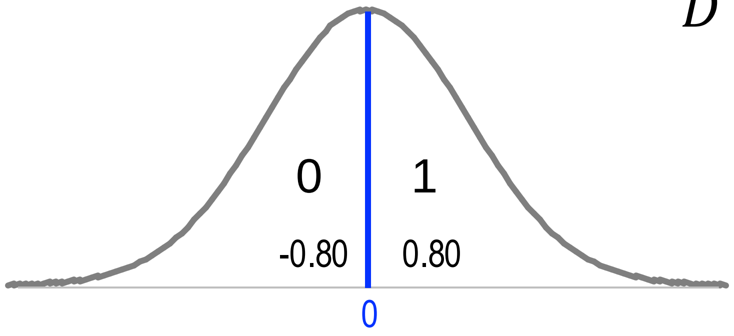
Example: Lloyd-Max quantizers for Gaussian PDF

Data point x	Code length *bits*	Code name $k(x)$	Recon-struction $m(k)$	Reconstruction error d $(m(k(x)) - x)^2$
0.1	1	1	0.80	0.624
	2	10	0.45	0.194
	3	100	0.25	0.058
1.0	1	1	0.80	0.040
	2	11	1.51	0.240
	3	101	0.76	0.058

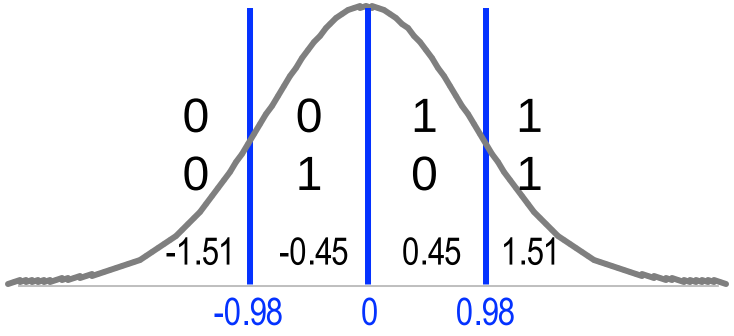
also: index

also: quantized value

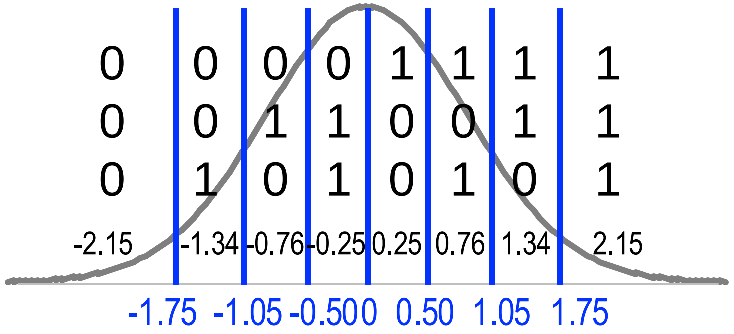
Expected Distortion $D = \mathbb{E}[d]$



0.3634



0.1175



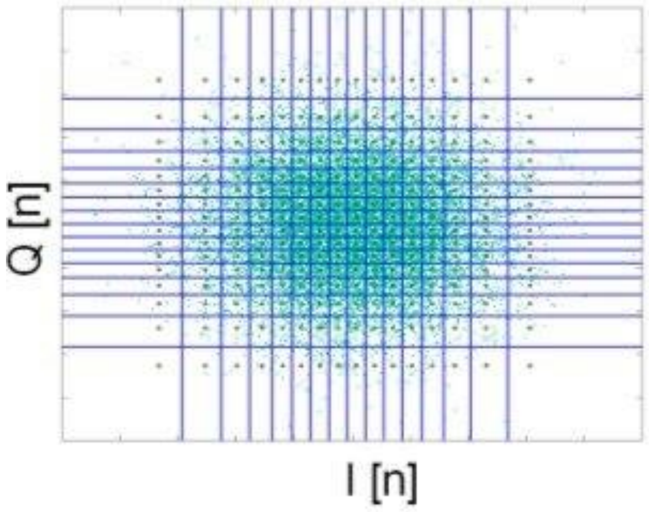
0.0345

Vector quantization:

the geometry of longer block length (higher dimensions): Voronoi tessellations and connection to k-means

The geometry of vector quantization

Independent 4-bit quantization (16 representatives) for $n = 2$ independent Gaussians:



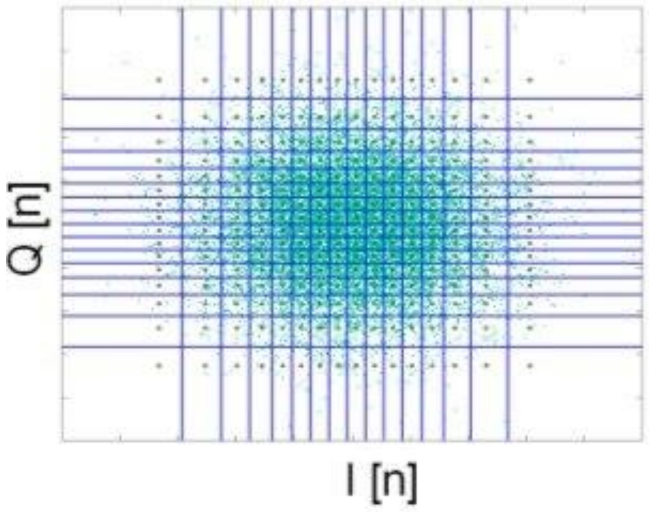
Joint encoding of $n = 2$ independent Gaussians:

?

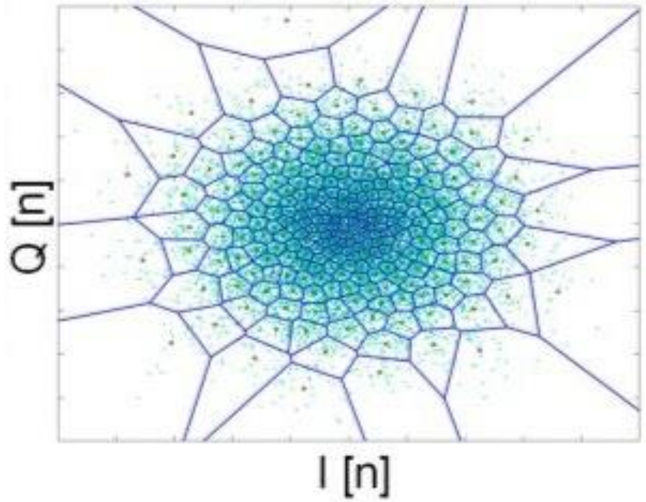
Figure source: https://ieeexplore.ieee.org/document/7767821/

The geometry of vector quantization

Independent 4-bit quantization (16 representatives) for $n = 2$ independent Gaussians:



Joint encoding of $n = 2$ independent Gaussians: 2D vector quantization, i.e. block length $n = 2$ and 4-bit per sample, or 8-bit (and 256 representatives) for two samples together



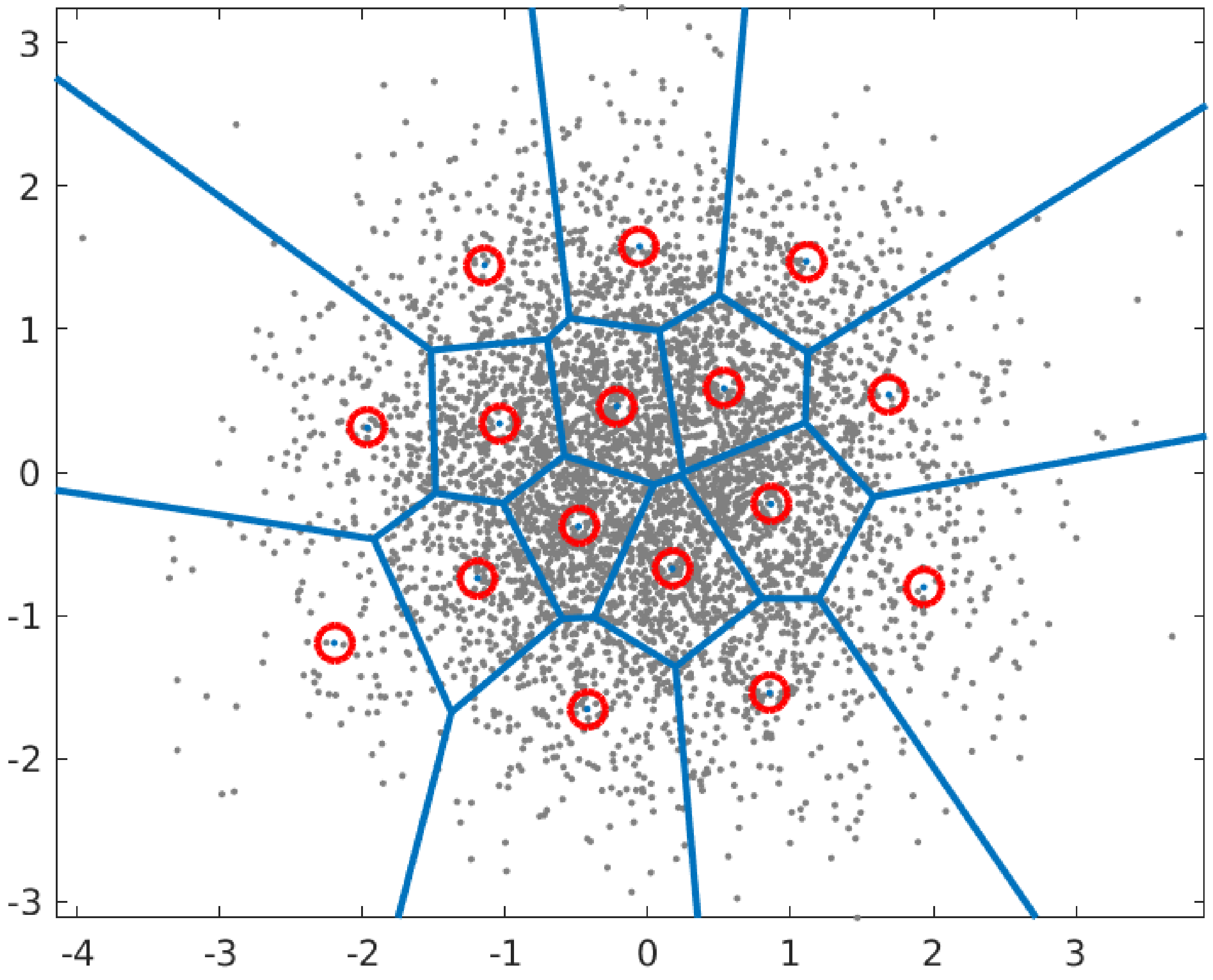


Figure source: https://speechprocessingbook.aalto.fi/Modelling/Vector_quantization_VQ.html

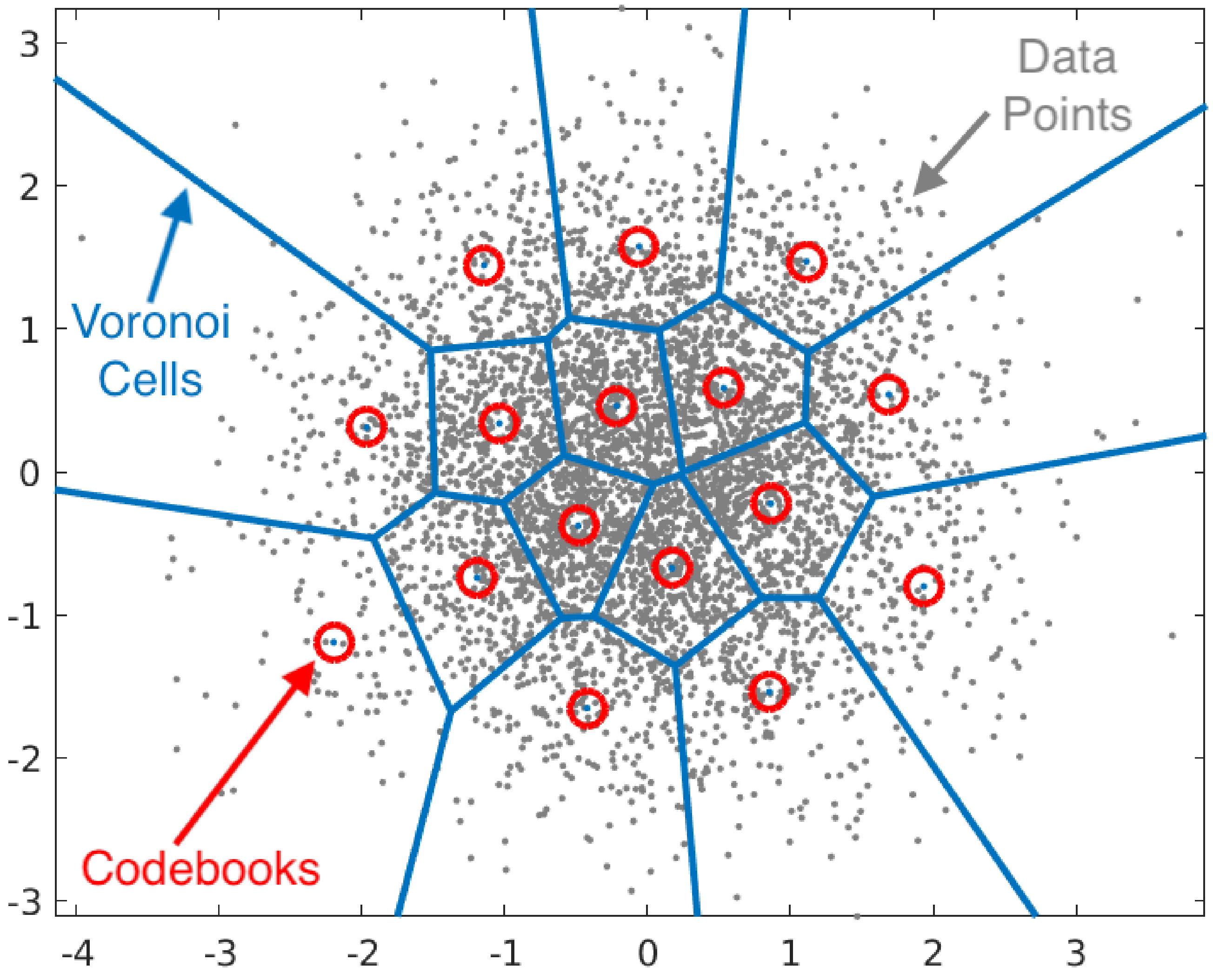


Figure source: https://speechprocessingbook.aalto.fi/Modelling/Vector_quantization_VQ.html

Lloyd's algorithm = k-means

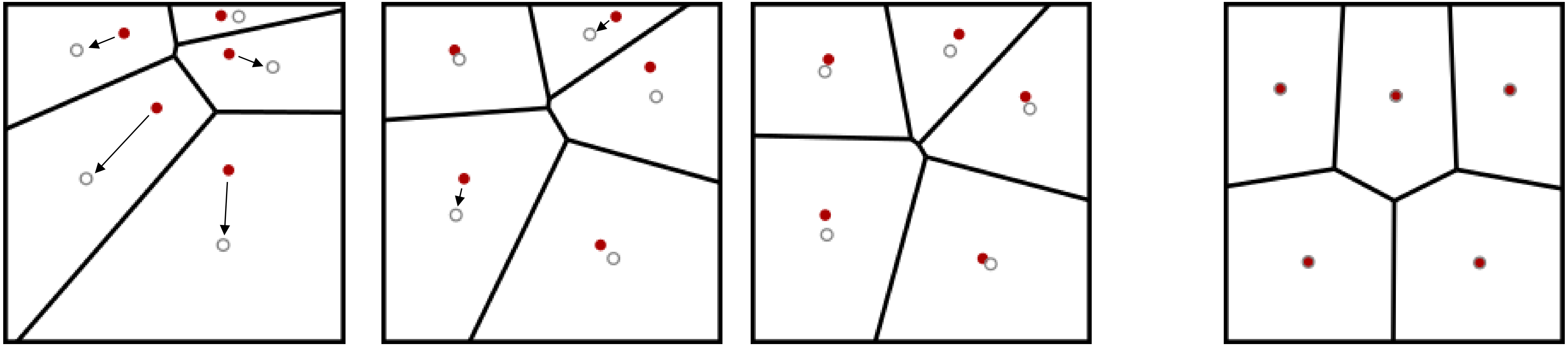
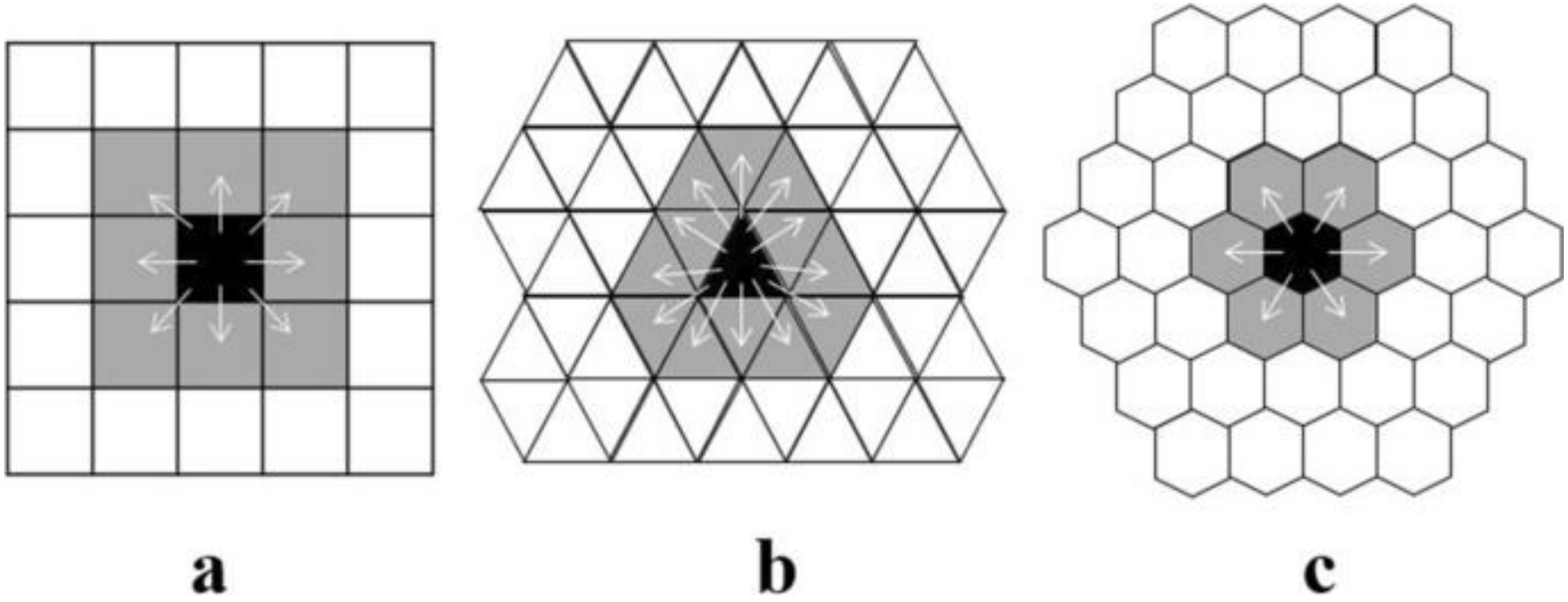


Figure source: https://en.wikipedia.org/wiki/Lloyd's_algorithm

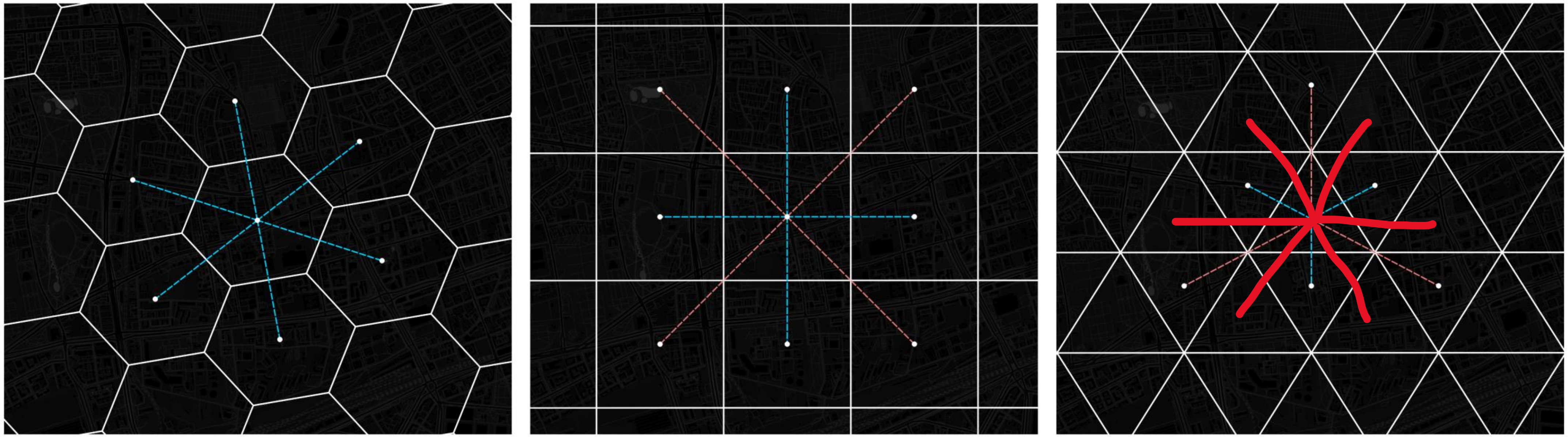
Optimal tessellations



?

Figure source: https://link.springer.com/article/10.1007/s41651-024-00200-5

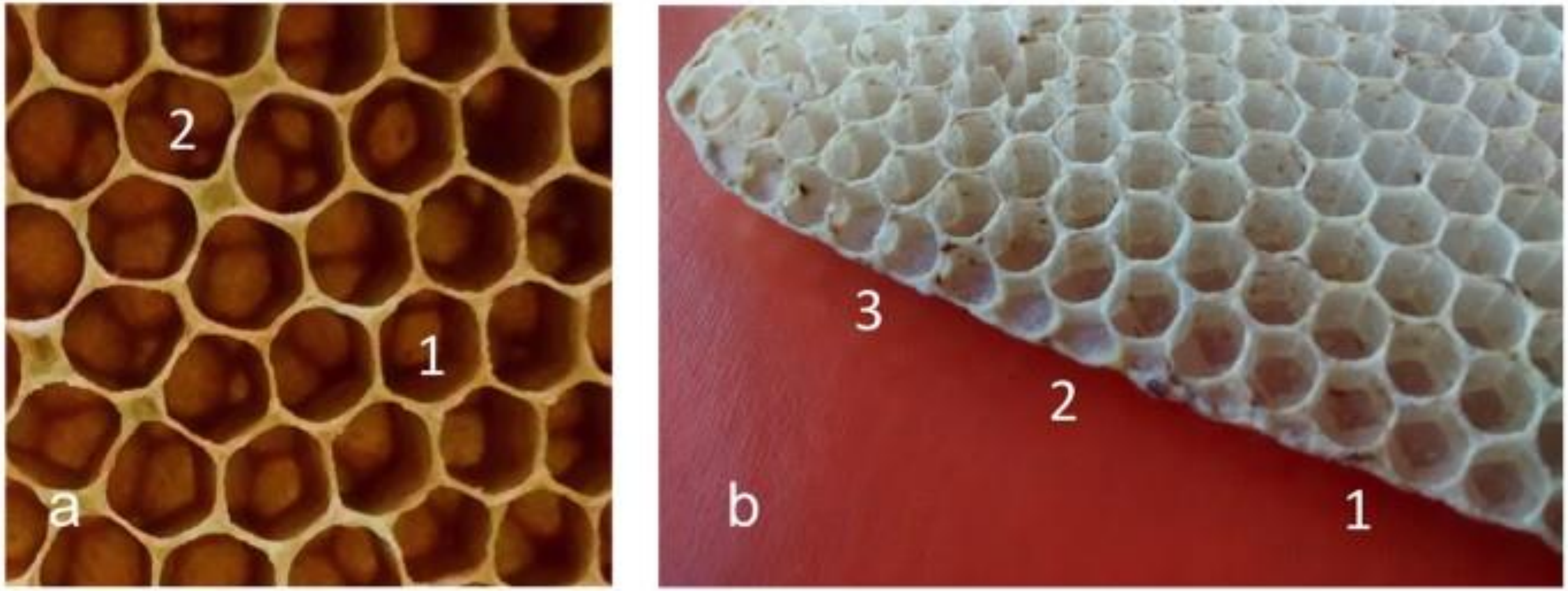
Optimal tessellations



Three types of spatial grids: hexagonal, square, and triangular.
Only the hexagonal grid provides an equal distance between the centers of neighboring cells.
There are at least two different distance categories for other kinds of grids.

Source: https://www.kontur.io/blog/why-we-use-h3/

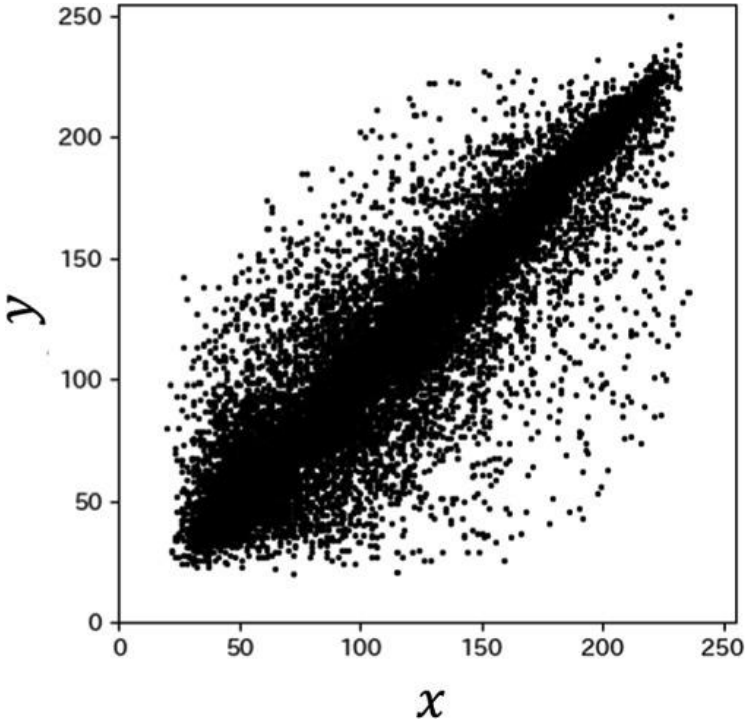
Optimal tessellations



"Early natural philosophers, like Marcus Terentius Varro [37 BC], based on the observation that hexagons possess the highest surface/perimeter ratio, compared to other polygons that can be used for tiling the plane, suggested that honey bees build their hexagonal cells in order to achieve the best economy of material."

Source: Nazzi, "The hexagonal shape of the honeycomb cells depends on the construction behavior of bees", Nature, 2016. https://www.nature.com/articles/srep28341

It even gets better *with* correlations

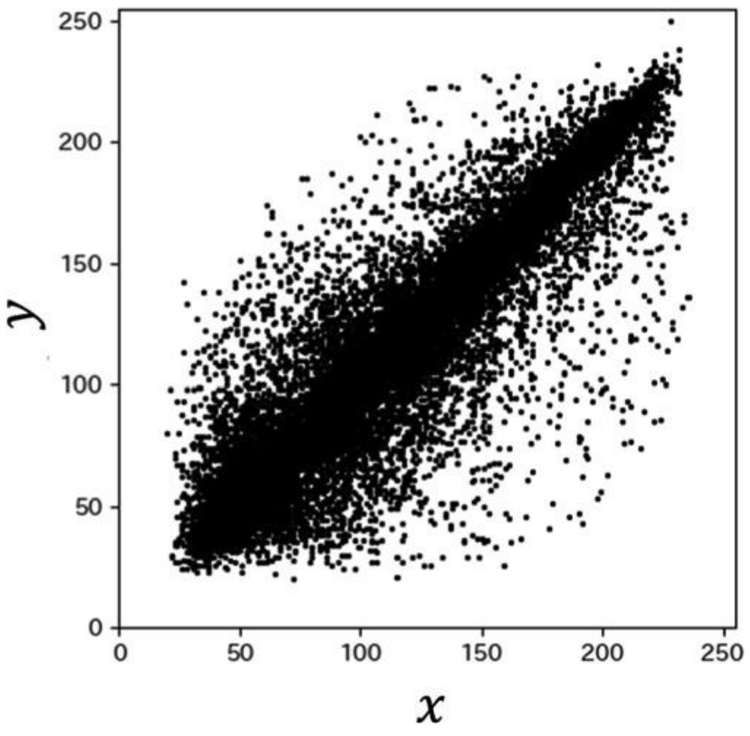


Correlation of neighboring pixels

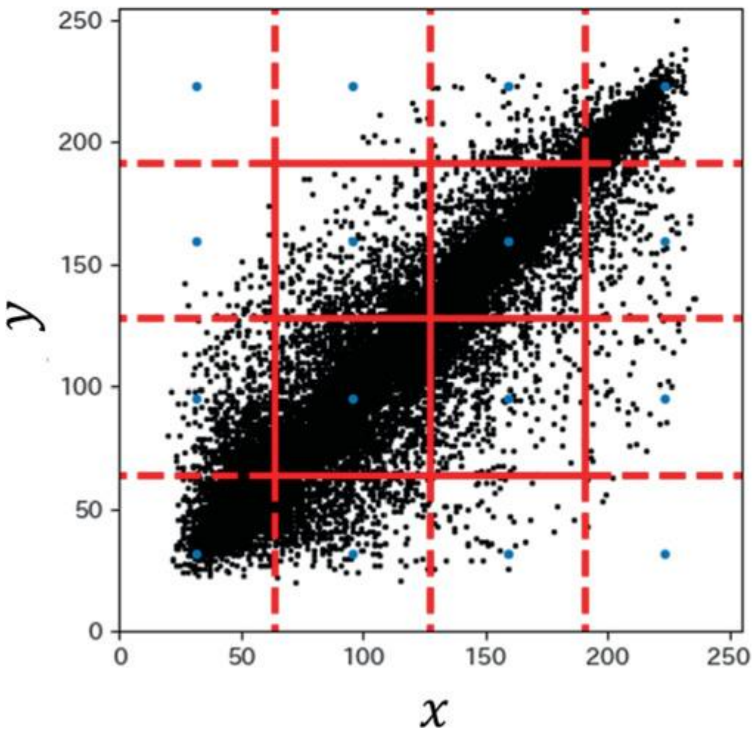
?

Vector space partitioning in
<u>scalar quantization</u> (approximate)

It even gets better *with* correlations



Correlation of neighboring pixels



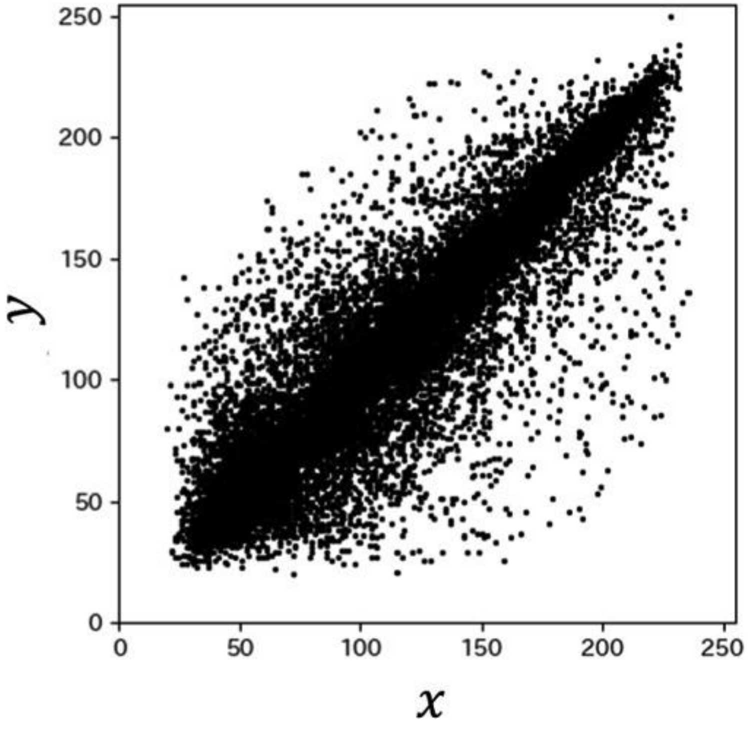
Vector space partitioning in scalar quantization (approximate)

?

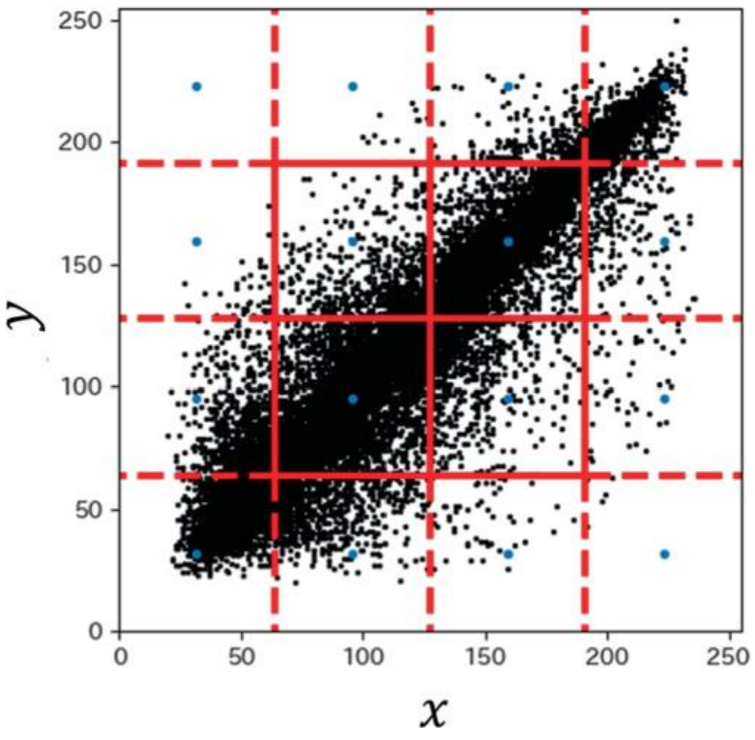
Arrangement of cells with the smallest average quantization error in vector quantization

Figure source: https://link.springer.com/chapter/10.1007/978-3-031-57840-3_40

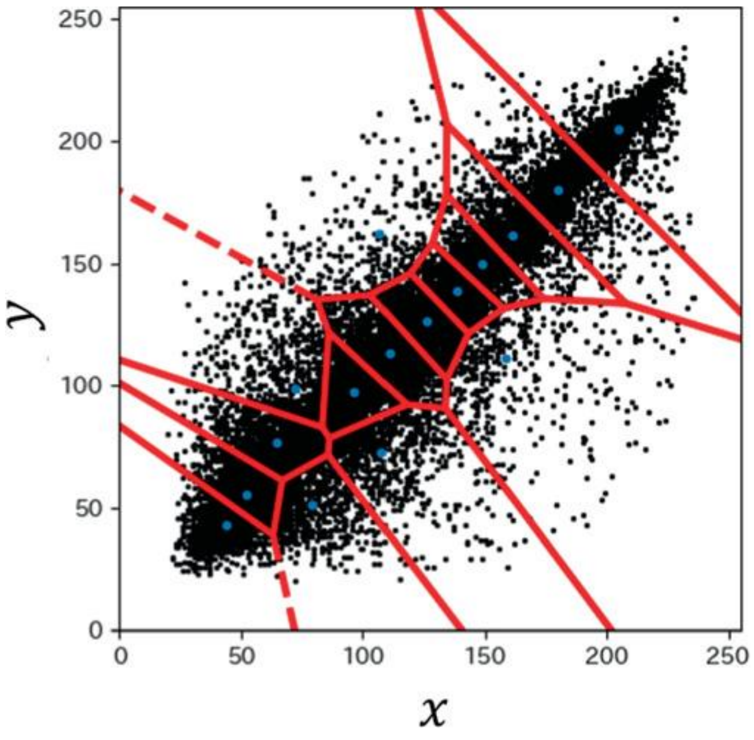
It even gets better *with* correlations



Correlation of neighboring pixels



Vector space partitioning in scalar quantization (approximate)



Arrangement of cells with the smallest average quantization error in vector quantization

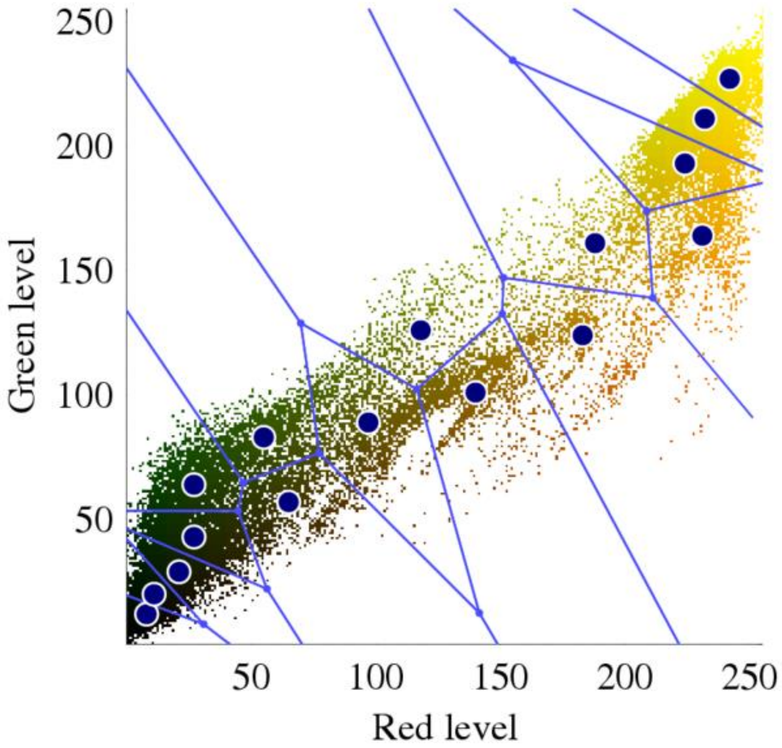
Figure source: https://link.springer.com/chapter/10.1007/978-3-031-57840-3_40
Gatterbauer, Aslam. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Rate-distortion code vs. k-means

$n = 2$ channels per pixel (will be encoded together)

$nR = 4$ bits per pixel (2 bits per channel level), thus 16 representatives

Example image with only red and green channel (for illustration)



Vector quantization of colors present in the image into Voronoi cells using *k*-means

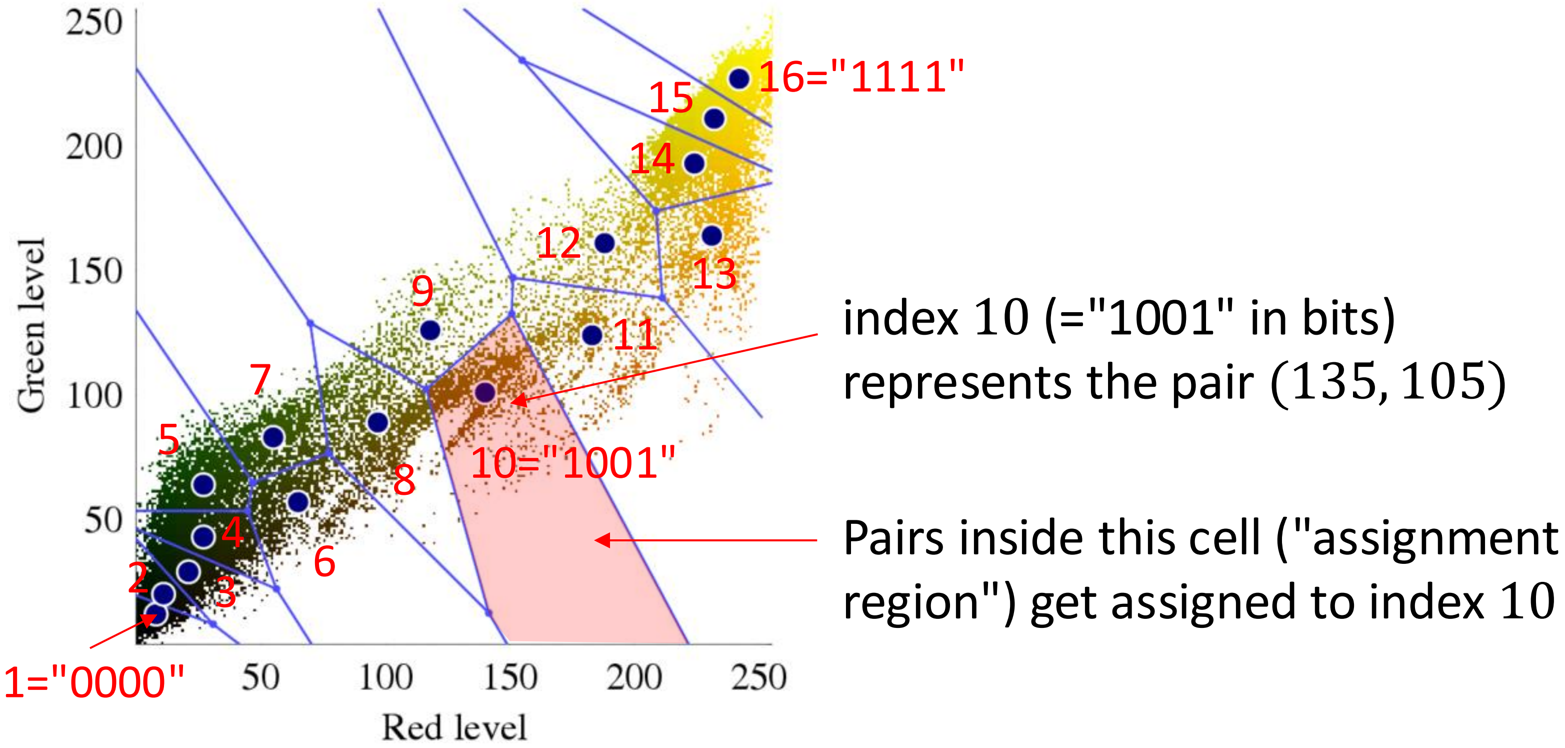
Source: https://en.wikipedia.org/wiki/K-means_clustering

Rate-distortion code vs. k-means

$n = 2$ channels per pixel (will be encoded together)

$nR = 4$ bits per pixel (2 bits per channel level), thus 16 representatives

Example image with only red and green channel (for illustration)



Vector quantization of colors present in the image into Voronoi cells using *k*-means

Source: https://en.wikipedia.org/wiki/K-means_clustering

The magic of vector quantization

- Given a set of n samples (e.g. iid from Gaussian distribution)
- We want to jointly quantize the vector $(X_1, \ldots, X_n)$
- Represent these vectors using nR bits
- Represent the entire sequence by a single index taking 2^{nR} values ("representatives")

- Vector quantization achieves a lower distortion than linear (independent, scalar) quantization

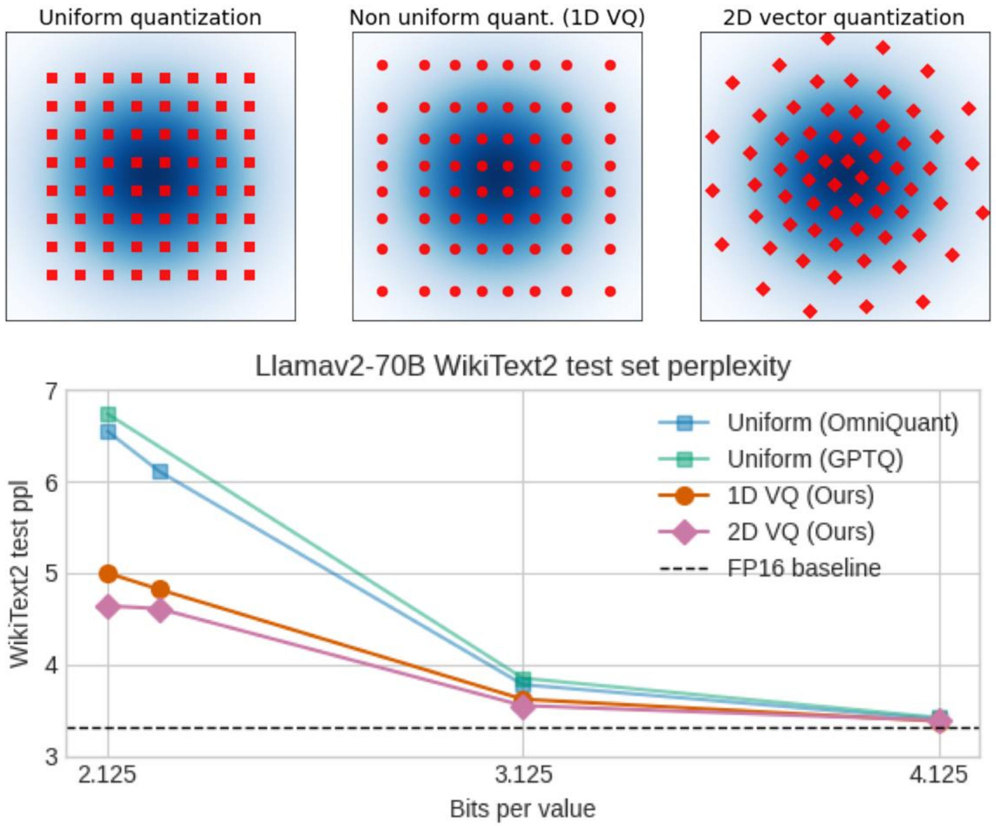


Figure 1. **Top:** An example of how vector quantization can better represent 2D normally distributed data compared to uniform quantization, non-uniform quantization. **Bottom:** Comparing GPTVQ to state-of-the-art uniform quantization on Llama 70B.

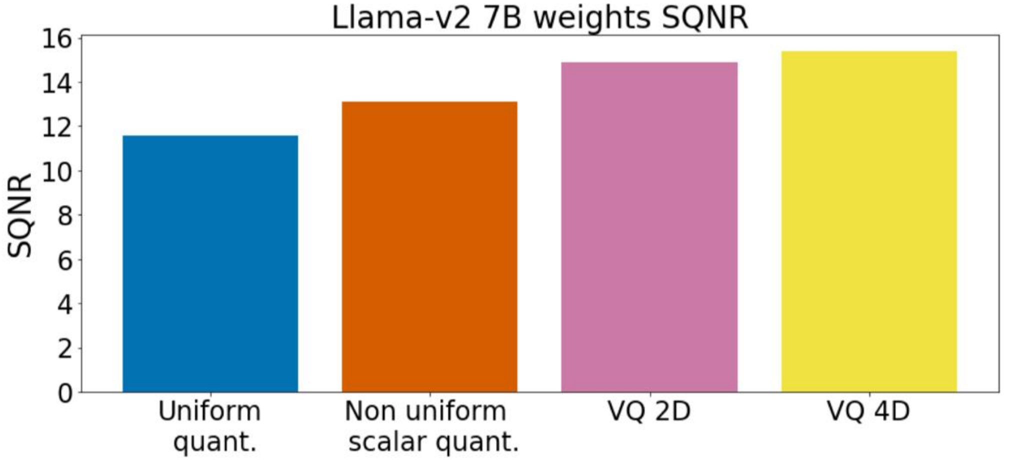
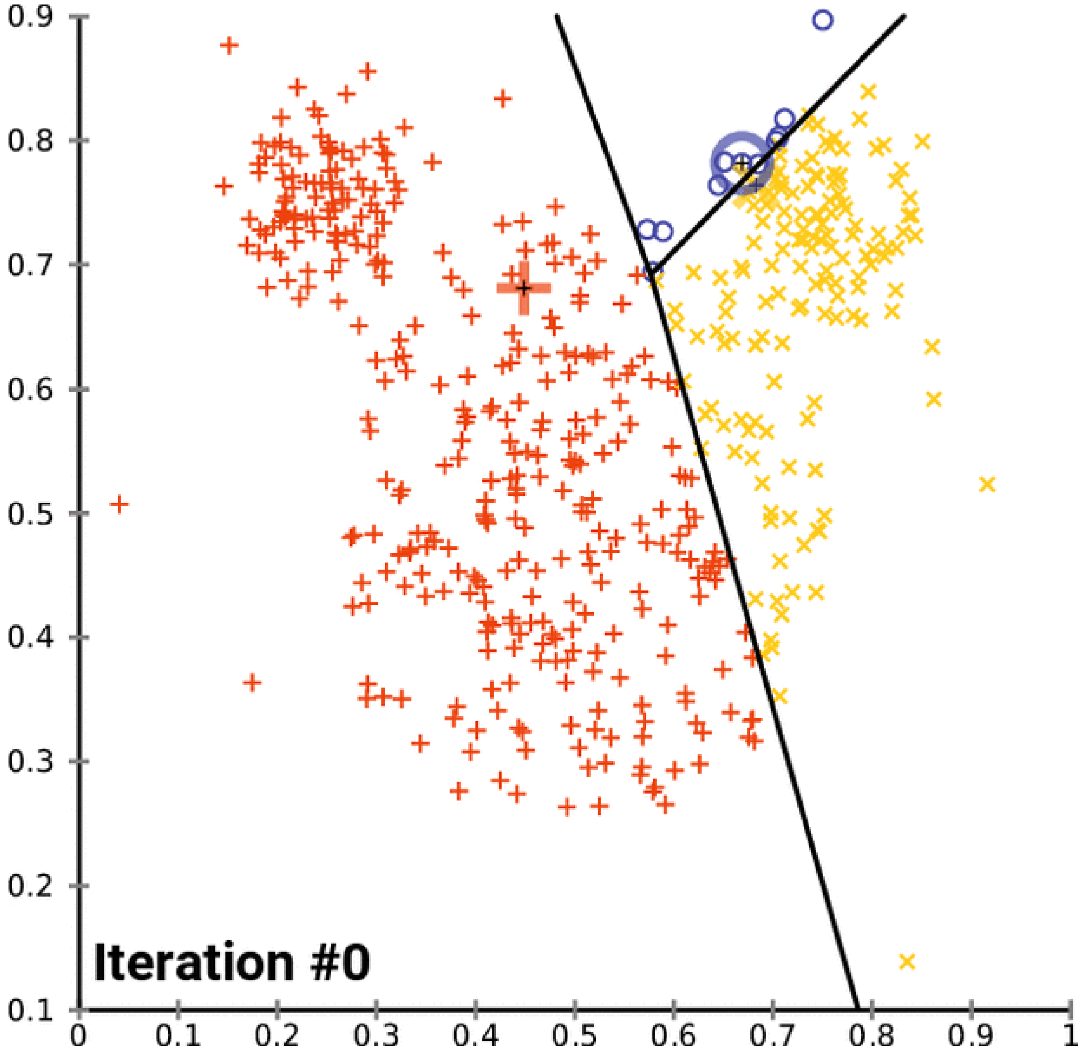


Figure 2. Quantization SQNR depending on the dimensionality for Llama-v2 7B weights. Signal-to-noise ratio increases with quantization dimensionality due to additional flexibility in the quantization grid.

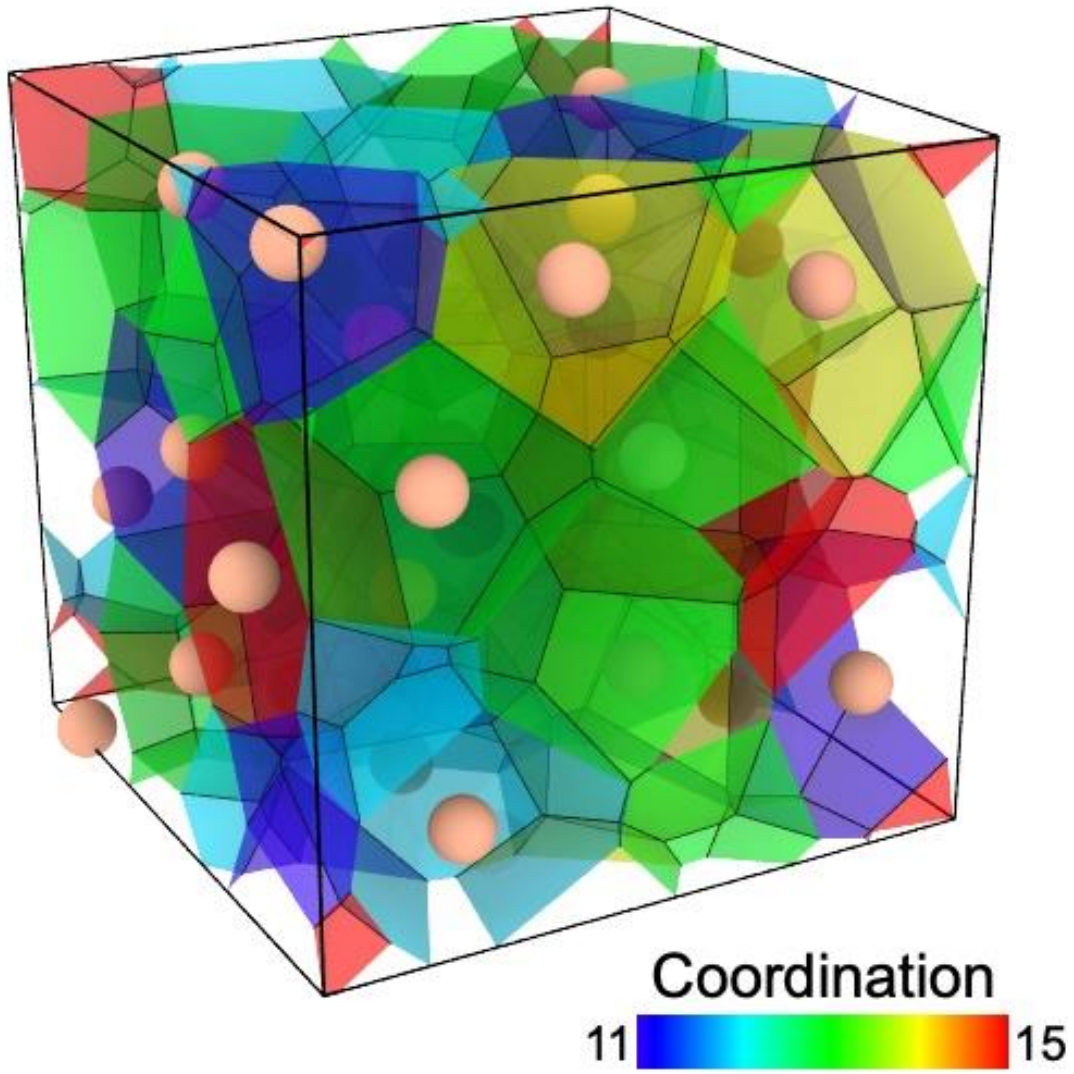
Source: https://arxiv.org/abs/2402.15319

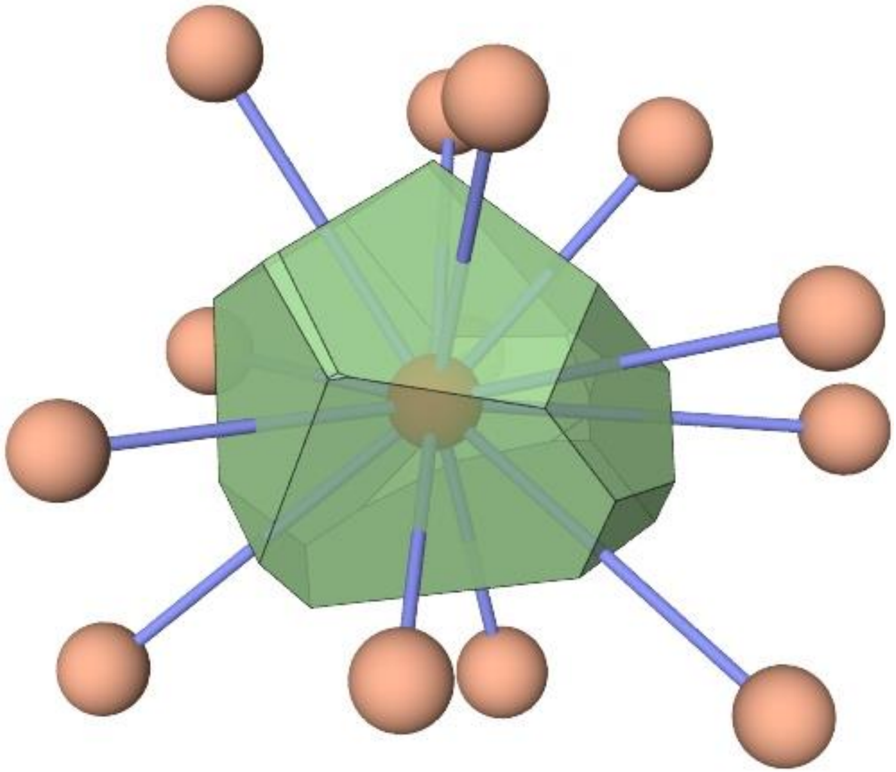
An animation of k-means



Source of animation figure: https://en.wikipedia.org/wiki/K-means_clustering

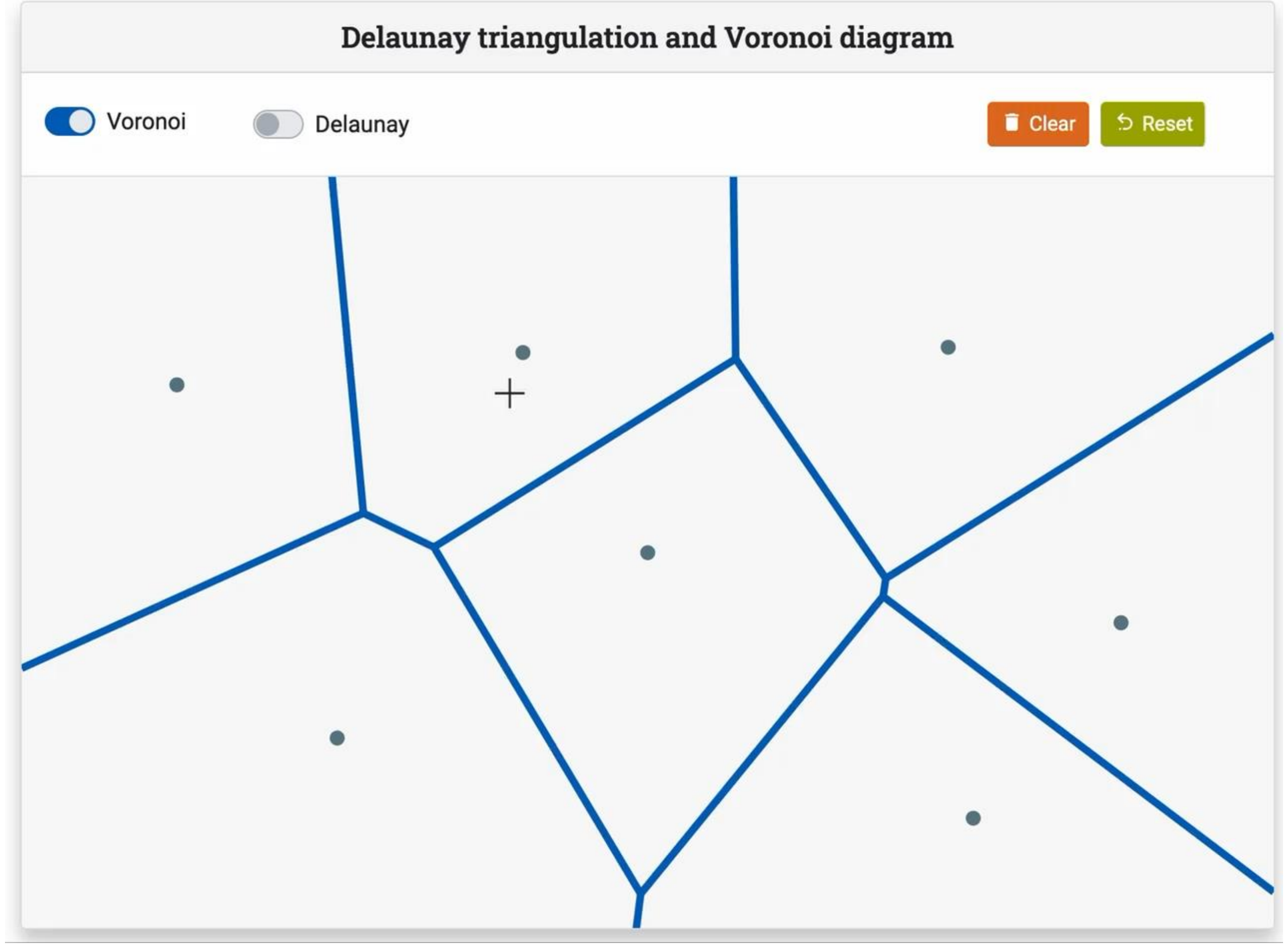
k-means in higher dimensions





Source: https://www.ovito.org/docs/current/reference/pipelines/modifiers/voronoi_analysis.html

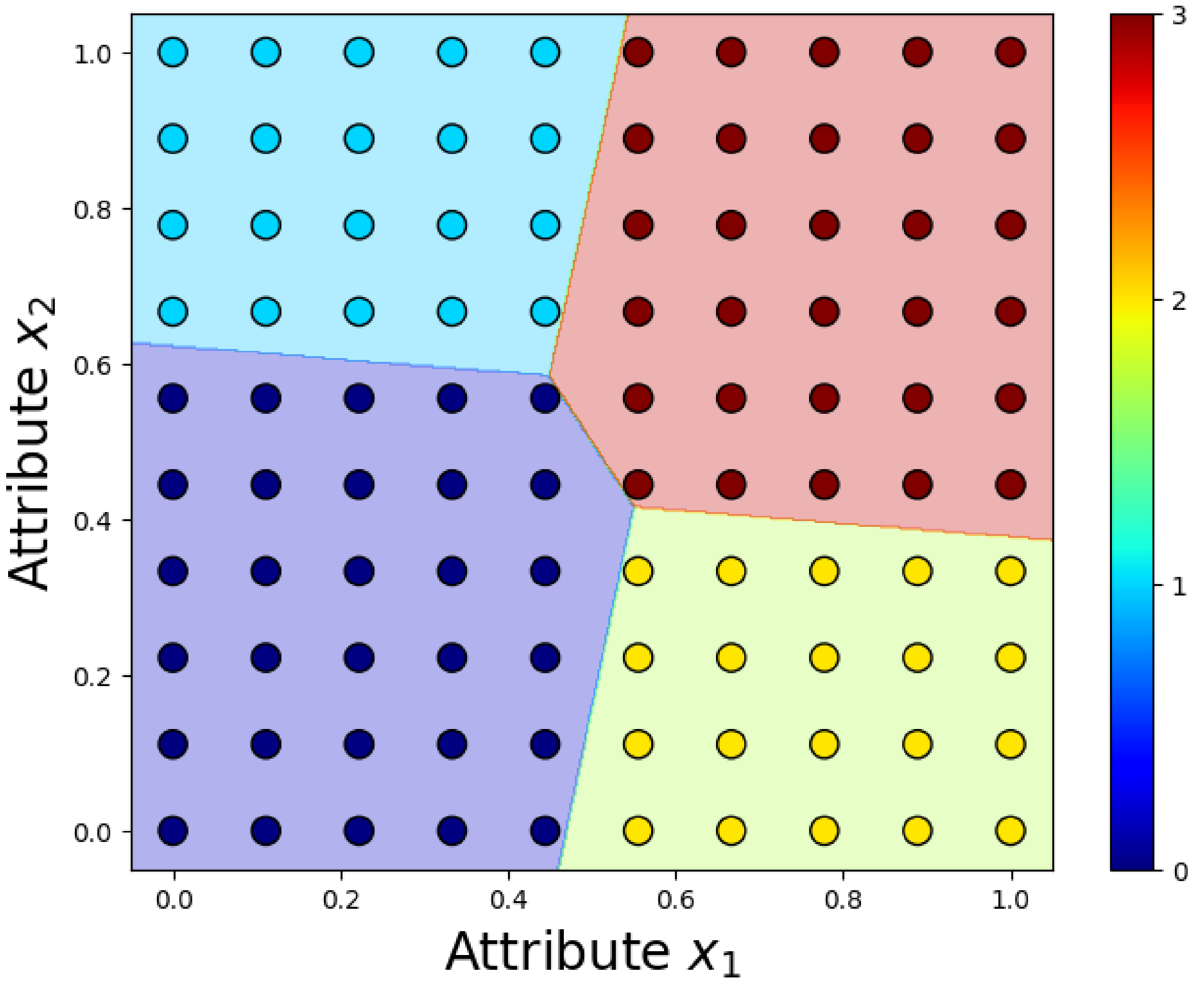
An animation of Voronoi tessellation



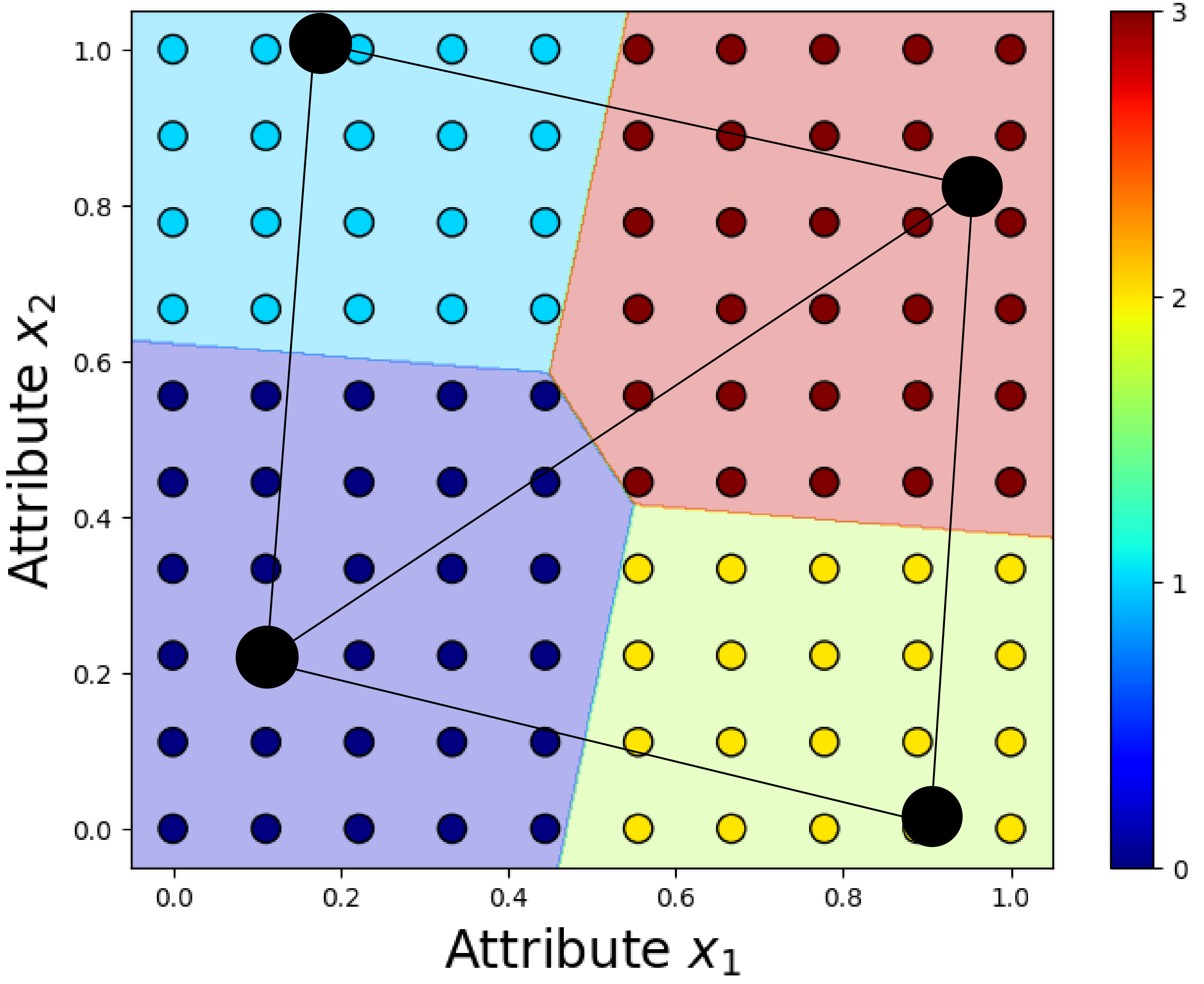
Source: https://cartography-playground.gitlab.io/playgrounds/triangulation-delaunay-voronoi-diagram/

Gatterbauer, Aslam. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Logistic regression vs. (soft) k-means

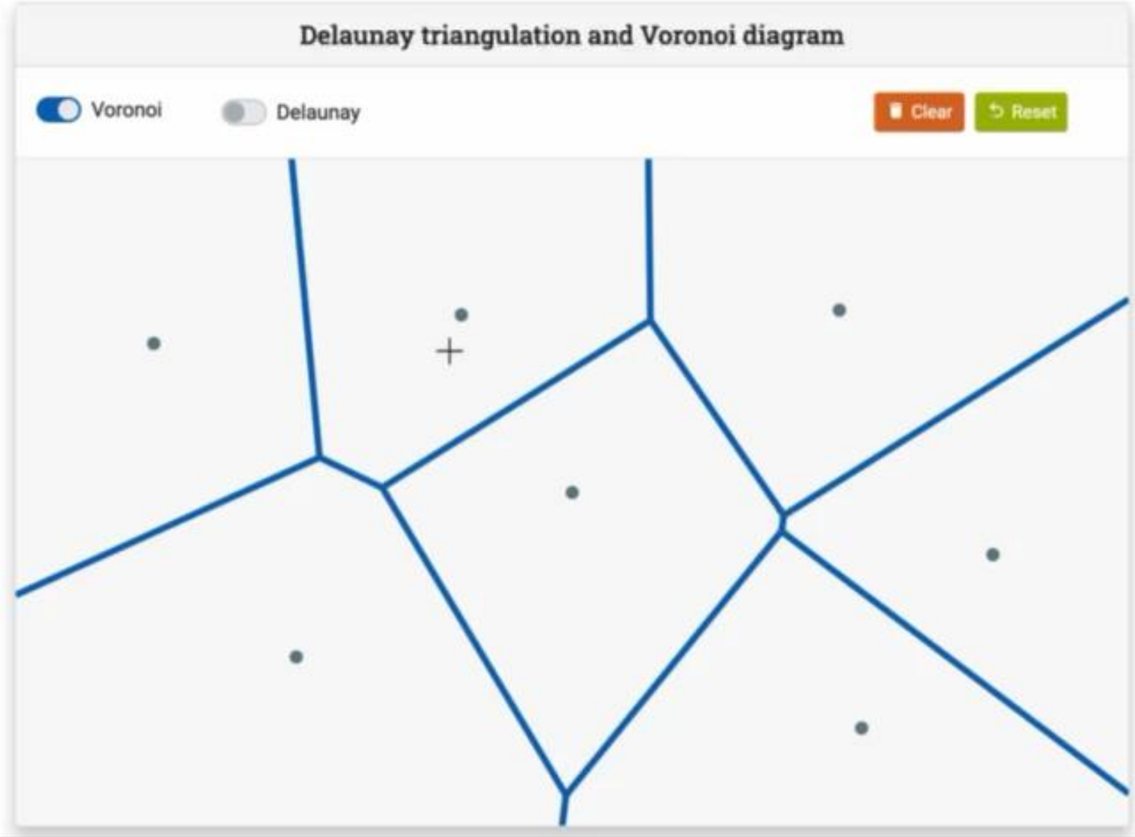


Logistic regression vs. (soft) k-means



Is this always possible ?

2.2 MB instead of 5.2 MB



Source: https://cartography-playground.gitlab.io/playgrounds/triangulation-delaunay-voronoi-diagram/
Gatterbauer, Aslam. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Let's make this more formal (Definitions)

Largely based on chapter 10 of
[Cover, Thomas'06] Elements of Information Theory, 2006. https://www.doi.org/10.1002/047174882X

Distortion theory

- Given: source distribution p, distortion measure d. What is the minimum expected distortion D achievable at a particular transmission rate R (in bits)?
  - In particular: What is the fundamental lower-bound on distortion D for a given rate R?
  - Intuition: more bits available (higher rate R), then fewer errors (smaller distortion D)
- Equivalently: what is the min rate R required to achieve a given distortion D?

- An intriguing aspect of this theory is that joint descriptions (think block codes) are more efficient than individual descriptions, even for independent RVs
  - The reason is found in the geometry: rectangular grid points (arising from independent descriptions) do not fill up the space efficiently (recall the earlier Voronoi diagrams)
  - Instead of representing each RV using R bits, we represent a sequence of n RVs by a single index taking 2^{nR} values. Encoding entire sequences at once achieves a lower distortion D for the same rate than independent quantization of the individual samples

Distortion function d

Distortion function (measure) d:

cost of representing a symbol by its quantized version

$$d: \mathcal{X} \times \hat{\mathcal{X}} \rightarrow \mathbb{R}^+$$

source alphabet

reproduction alphabet

Usually, $\mathcal{X} = \hat{\mathcal{X}}$

We assume the distortion to be bounded:

$$d_{\max} = \max_{x \in \mathcal{X}, \hat{x} \in \hat{\mathcal{X}}} d(x, \hat{x}) \leq \infty$$

What is then the distortion between sequences ?

Distortion function d

Distortion function (measure) d:
cost of representing a symbol by its quantized version

$$d: \mathcal{X} \times \hat{\mathcal{X}} \to \mathbb{R}^+$$

source alphabet (→ $\mathcal{X}$)
reproduction alphabet (→ $\hat{\mathcal{X}}$)

Usually, $\mathcal{X} = \hat{\mathcal{X}}$

We assume the distortion to be bounded:

$$d_{\max} = \max_{x \in \mathcal{X}, \hat{x} \in \hat{\mathcal{X}}} d(x, \hat{x}) \leq \infty$$

Distortion between sequences is the <u>average per symbol</u> distortion:

$$d(x^n, \hat{x}^n) = \frac{1}{n} \sum_i d(x_i, \hat{x}_i)$$

Hamming distortion:

$$d(x, \hat{x}) = \begin{cases} 0 & \text{if } x = \hat{x} \\ 1 & \text{if } x \neq \hat{x} \end{cases}$$

same as "probability of error" distortion ?

Distortion function d

Distortion function (measure) d:
cost of representing a symbol by its quantized version

$$d: \mathcal{X} \times \hat{\mathcal{X}} \to \mathbb{R}^+$$

source alphabet

reproduction alphabet

Usually, $\mathcal{X} = \hat{\mathcal{X}}$

We assume the distortion to be bounded:

$$d_{\max} = \max_{x \in \mathcal{X}, \hat{x} \in \hat{\mathcal{X}}} d(x, \hat{x}) \leq \infty$$

Distortion between sequences is the <u>average per symbol</u> distortion:

$$d(x^n, \hat{x}^n) = \frac{1}{n} \sum_i d(x_i, \hat{x}_i)$$

Hamming distortion:

$$d(x, \hat{x}) = \begin{cases} 0 & \text{if } x = \hat{x} \\ 1 & \text{if } x \neq \hat{x} \end{cases}$$

same as "probability of error" distortion

$$\mathbb{E}[d(X, \hat{X})] = \mathbb{P}[X \neq \hat{X}]$$

Squared-error distortion:

$$d(x, \hat{x}) = (x - \hat{x})^2$$

Why are we always so excited about squared errors? Think "least squares", "sum of squared errors (SSE)", or "mean of squared errors (MSE)", in linear regression, etc...

?

Distortion function d

Distortion function (measure) d:
cost of representing a symbol by its quantized version

$$d: \mathcal{X} \times \hat{\mathcal{X}} \to \mathbb{R}^+$$

source alphabet

reproduction alphabet

Usually, $\mathcal{X} = \hat{\mathcal{X}}$

We assume the distortion to be bounded:

$$d_{\max} = \max_{x \in \mathcal{X}, \hat{x} \in \hat{\mathcal{X}}} d(x, \hat{x}) \leq \infty$$

Distortion between sequences is the average per symbol distortion:

$$d(x^n, \hat{x}^n) = \frac{1}{n} \sum_i d(x_i, \hat{x}_i)$$

Hamming distortion:

$$d(x, \hat{x}) = \begin{cases} 0 & \text{if } x = \hat{x} \\ 1 & \text{if } x \neq \hat{x} \end{cases}$$

same as "probability of error" distortion

$$\mathbb{E}[d(X, \hat{X})] = \mathbb{P}[X \neq \hat{X}]$$

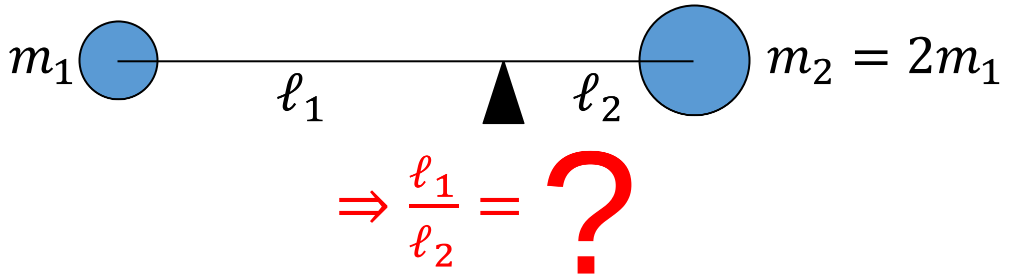
Squared-error distortion:

$$d(x, \hat{x}) = (x - \hat{x})^2$$

Connection to simple **expectations (means)**:

The squared error distortion penalizes large deviations quadratically. The **conditional mean** of X (given some available information) minimizes this penalty.

Distortion function d: Squared-error distortion



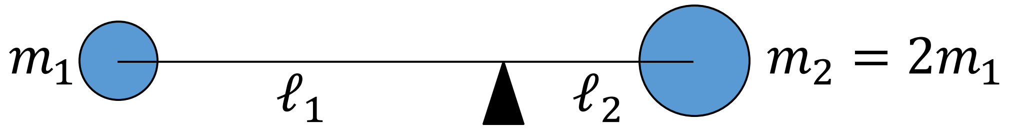
Squared-error distortion:

$$d(x, \hat{x}) = (x - \hat{x})^2$$

Connection to simple **expectations (means)**:

The squared error distortion penalizes large deviations quadratically. The **conditional mean** of X (given some available information) minimizes this penalty.

Distortion function d: Squared-error distortion



$$\Rightarrow \frac{\ell_1}{\ell_2} = 2$$

What does this have to do with squared-error distortion?

?

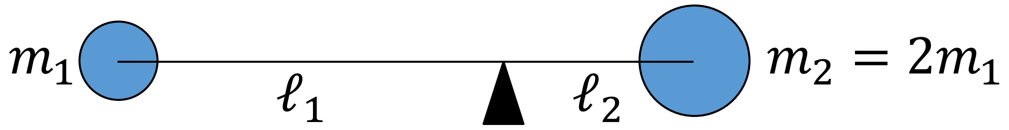
Squared-error distortion:

$$d(x, \hat{x}) = (x - \hat{x})^2$$

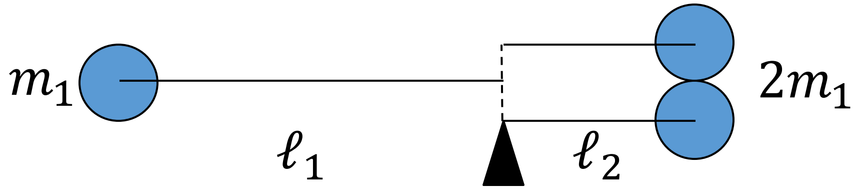
Connection to simple **expectations (means)**:

The squared error distortion penalizes large deviations quadratically. The **conditional mean** of X (given some available information) minimizes this penalty.

Distortion function d: Squared-error distortion



$$\Rightarrow \frac{\ell_1}{\ell_2} = 2$$



$$\min[{\ell_1}^2 + 2{\ell_2}^2], \text{ s.t. to } \ell_1 + \ell_2 = c$$

$$\text{SSE}(\ell_1) = {\ell_1}^2 + 2(c - \ell_1)^2$$

$$\frac{\partial \text{SSE}}{\partial \ell_1} = 2\ell_1 + 2(-2c + 2\ell_1) = 0$$

$$\Rightarrow \ell_1 = \frac{2c}{3} \qquad \Rightarrow \frac{\ell_1}{\ell_2} = 2 \qquad ☺$$

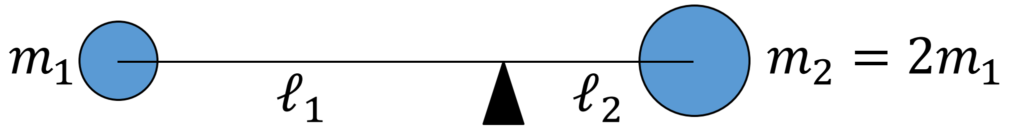
Squared-error distortion:

$$d(x, \hat{x}) = (x - \hat{x})^2$$

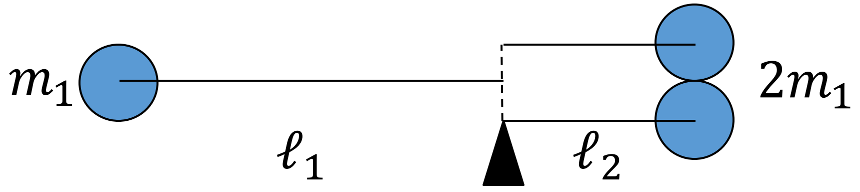
Connection to simple **expectations (means)**:

The squared error distortion penalizes large deviations quadratically. The **conditional mean** of X (given some available information) minimizes this penalty.

Distortion function d: Squared-error distortion



$$\Rightarrow \frac{\ell_1}{\ell_2} = 2$$



$$\min[{\ell_1}^2 + 2{\ell_2}^2], \text{ s.t. to } \ell_1 + \ell_2 = c$$

$$\Rightarrow \frac{\ell_1}{\ell_2} = 2$$ ☺

$$\min[\ell_1 + 2\ell_2], \text{s.t. to } \ell_1 + \ell_2 = c$$?

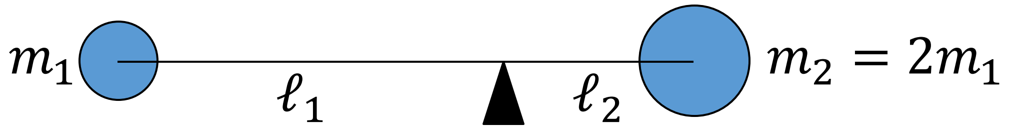
Squared-error distortion:

$$d(x, \hat{x}) = (x - \hat{x})^2$$

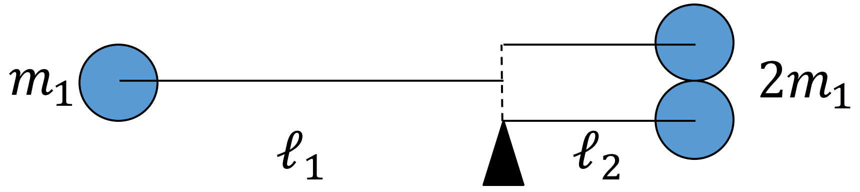
Connection to simple **expectations (means)**:

The squared error distortion penalizes large deviations quadratically. The **conditional mean** of X (given some available information) minimizes this penalty.

Distortion function d: Squared-error distortion



$$\Rightarrow \frac{\ell_1}{\ell_2} = 2$$

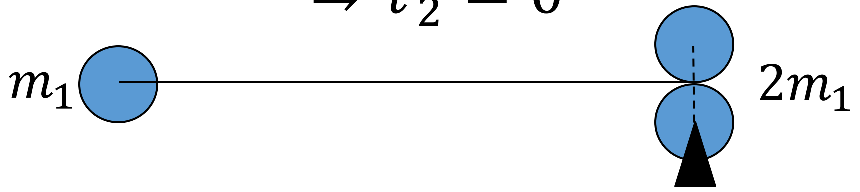


$$\min[{\ell_1}^2 + 2{\ell_2}^2], \text{ s.t. to } \ell_1 + \ell_2 = c$$

$$\Rightarrow \frac{\ell_1}{\ell_2} = 2$$ ☺

$$\min[\ell_1 + 2\ell_2], \text{s.t. to } \ell_1 + \ell_2 = c$$

$$\Rightarrow \ell_2 = 0$$



☹

The arithmetic mean is the "center" ("centroid" or center of mass) of the distribution that balances the squared error!

Squared-error distortion:

$$d(x, \hat{x}) = (x - \hat{x})^2$$

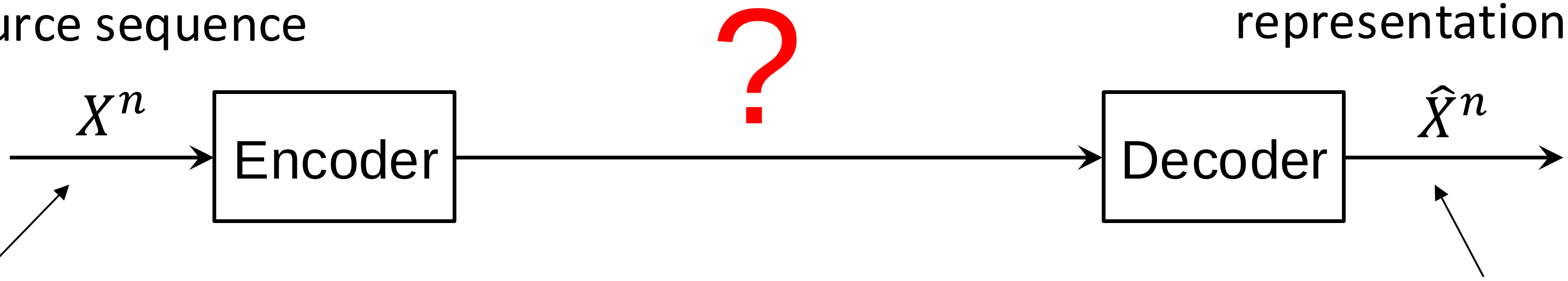
Connection to simple **expectations (means)**:

The squared error distortion penalizes large deviations quadratically. The **conditional mean** of X (given some available information) minimizes this penalty.

Rate-distortion code

source sequence

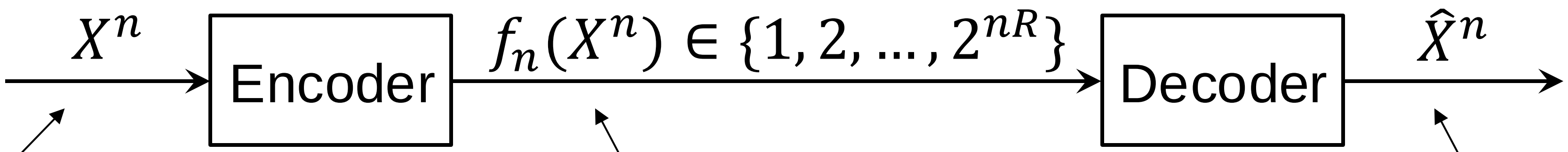
vector quantization, reproduction, representation, reconstruction, ...



A source produces an iid sequence $X_1, X_2, \ldots, X_n$ with $X_i \sim p(X)$ and X taken from a source alphabet $\mathcal{X}$

The **representation** of X is $\hat{X}(X)$. The decoder represents X^n by an estimate $\hat{X}^n \in \hat{\mathcal{X}}^n$ with $\hat{\mathcal{X}}$ being the reproduction alphabet

Rate-distortion code



A source produces an iid sequence $X_1, X_2, \dots, X_n$ with $X_i \sim p(X)$ and X taken from a source alphabet $\mathcal{X}$

We are given R bits to represent X. Thus the function $\hat{X}$ can take on 2^R different values

The **representation** of X is $\hat{X}(X)$. The decoder represents X^n by an estimate $\hat{X}^n \in \hat{\mathcal{X}}^n$ with $\hat{\mathcal{X}}$ being the reproduction alphabet

The encoder describes the source sequence X^n via an **encoding function** that maps X^n to an index

$$f_n: \mathcal{X}^n \rightarrow \{1, 2, \dots, 2^{nR}\}$$

The **decoding function** maps an index to a reconstructed sequence

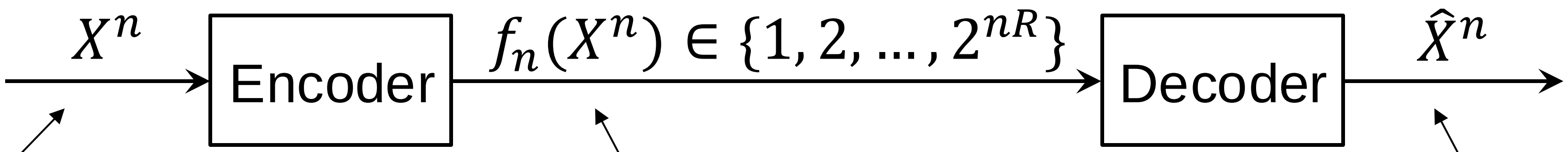
$$g_n: \{1, 2, \dots, 2^{nR}\} \rightarrow \hat{\mathcal{X}}^n$$

Rate-distortion code

source sequence

index

vector quantization, reproduction, representation, reconstruction, ...



A source produces an iid sequence $X_1, X_2, \dots, X_n$ with $X_i \sim p(X)$ and X taken from a source alphabet $\mathcal{X}$

We are given R bits to represent X. Thus the function $\hat{X}$ can take on 2^R different values

The **representation** of X is $\hat{X}(X)$. The decoder represents X^n by an estimate $\hat{X}^n \in \hat{\mathcal{X}}^n$ with $\hat{\mathcal{X}}$ being the reproduction alphabet

The encoder describes the source sequence X^n via an **encoding function** that maps X^n to an index

$$f_n : \mathcal{X}^n \rightarrow \{1, 2, \dots, 2^{nR}\}$$

The **decoding function** maps an index to a reconstructed sequence

$$g_n : \{1, 2, \dots, 2^{nR}\} \rightarrow \hat{\mathcal{X}}^n$$

A $(2^{nR}, n)$-**rate distortion code** consists of f_n and g_n.

$g_n(1), \dots, g_n(2^{nR})$: **codebook**

$f_n^{-1}(1), \dots, f_n^{-1}(2^{nR})$: **assignment regions**

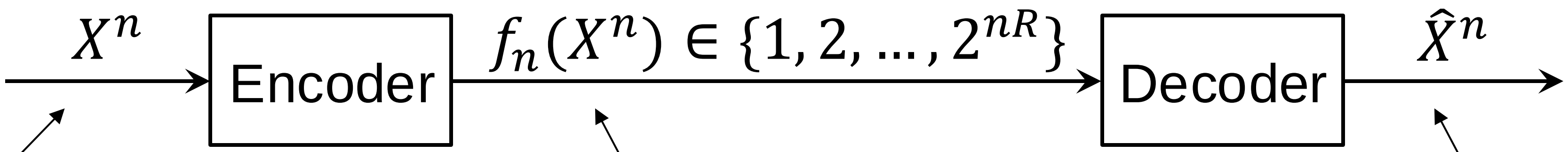
What is its associated distortion ?

Rate-distortion code

vector quantization, reproduction, representation, reconstruction, ...

source sequence

index



A source produces an iid sequence $X_1, X_2, \ldots, X_n$ with $X_i \sim p(X)$ and X taken from a source alphabet $\mathcal{X}$

We are given R bits to represent X. Thus the function $\hat{X}$ can take on 2^R different values

The **representation** of X is $\hat{X}(X)$. The decoder represents X^n by an estimate $\hat{X}^n \in \hat{\mathcal{X}}^n$ with $\hat{\mathcal{X}}$ being the reproduction alphabet

The encoder describes the source sequence X^n via an **encoding function** that maps X^n to an index

$$f_n: \mathcal{X}^n \to \{1, 2, \ldots, 2^{nR}\}$$

The **decoding function** maps an index to a reconstructed sequence

$$g_n: \{1, 2, \ldots, 2^{nR}\} \to \hat{\mathcal{X}}^n$$

A $(2^{nR}, n)$-**rate distortion code** consists of f_n and g_n.

$g_n(1), \ldots, g_n(2^{nR})$: **codebook**

$f_n^{-1}(1), \ldots, f_n^{-1}(2^{nR})$: **assignment regions**

Its associated distortion is:

$$D = \mathbb{E}_{X\sim p}[d(X^n, \overbrace{g_n(f_n(X^n))}^{\hat{X}})]$$

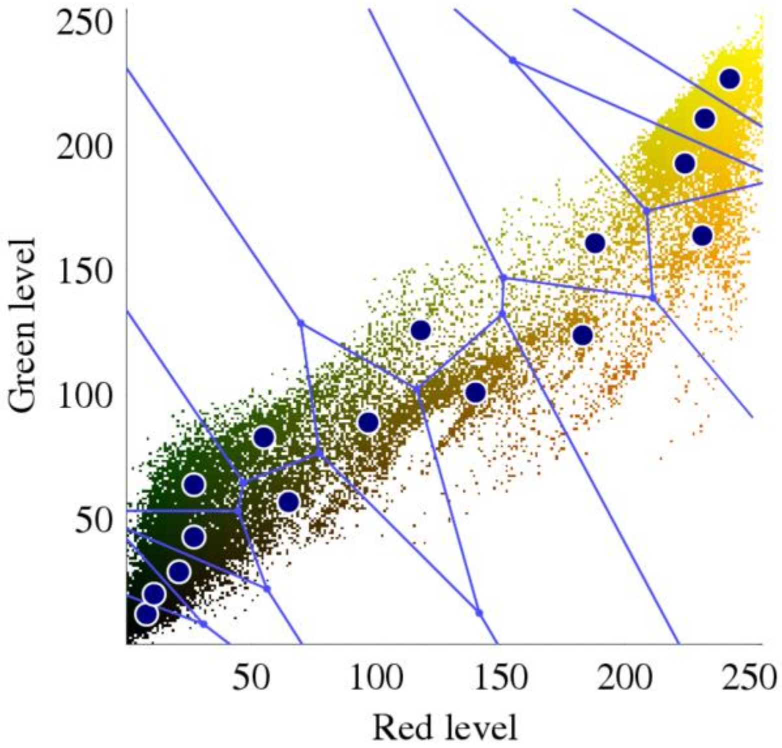
$$= \sum_{x^n} p(x^n) \cdot d(x^n, g_n(f_n(x^n)))$$

Rate-distortion code vs. k-means

$\mathcal{X} = \hat{\mathcal{X}} = \{0,1,\dots,255\}$ thus 8 bit resolution

$n = 2$ channels per pixel (will be encoded together), 16 bits per pixel

Example image with only red and green channel (for illustration)



Vector quantization of colors present in the image into Voronoi cells using *k*-means

Source: https://en.wikipedia.org/wiki/K-means_clustering

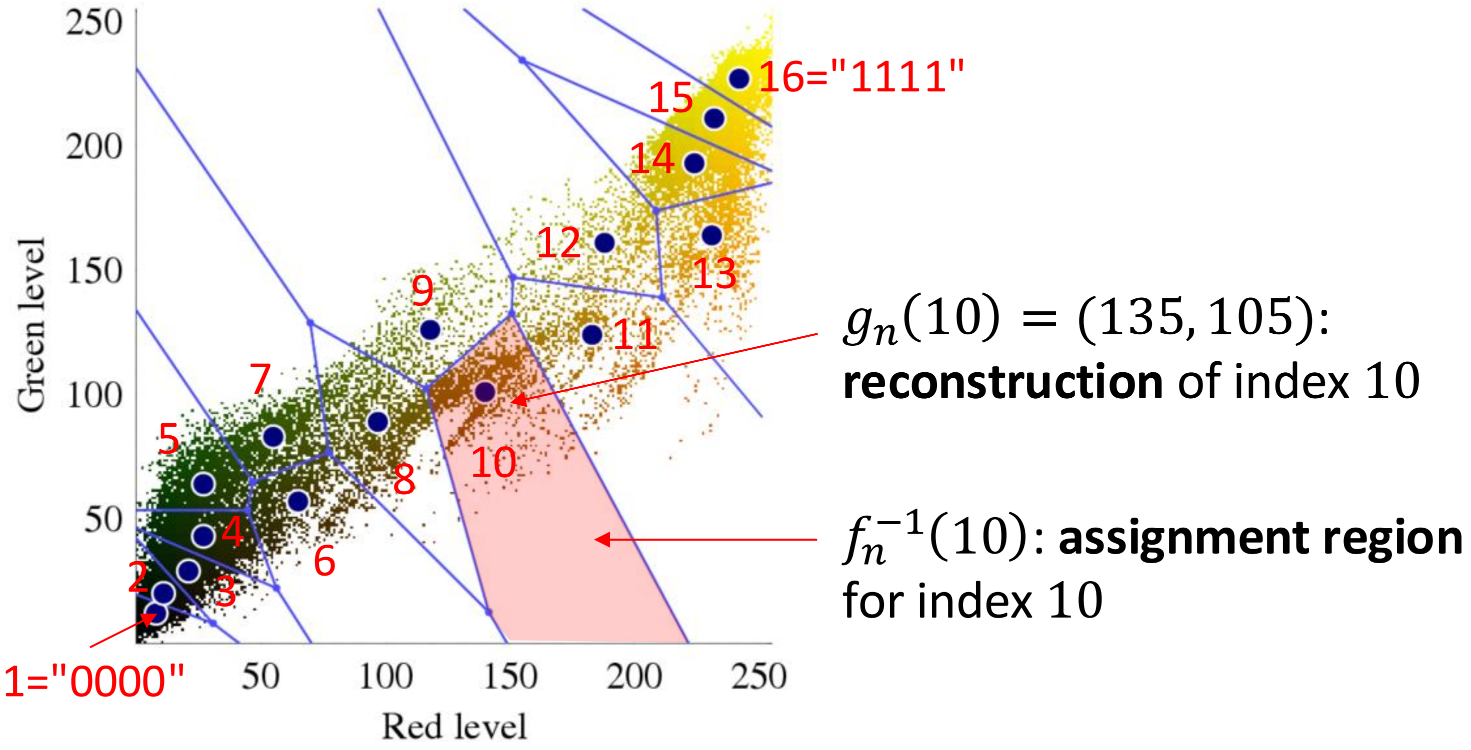
Rate-distortion code vs. k-means

$\mathcal{X} = \hat{\mathcal{X}} = \{0,1,\ldots,255\}$ thus 8 bit resolution

$n = 2$ channels per pixel (will be encoded together), 16 bits per pixel

$nR = 4$ bits per pixel (2 bits per channel level), thus 16 representatives

Example image with only red and green channel (for illustration)



Vector quantization of colors present in the image into Voronoi cells using *k*-means

Source: https://en.wikipedia.org/wiki/K-means_clustering

Main theorem of Rate-distortion theory

A **rate distortion pair** (R, D) is **achievable** if there exists a sequence of $(2^{nR}, n)$-rate distortion code (f_n, g_n) with

$$\lim_{n\to\infty} \mathbb{E}_{X\sim p}[d(X^n, g_n(f_n(X^n)))] \leq D$$

A **rate distortion region** for a source is the closure of the set of achievable distortion pairs (R, D).

The **rate distortion** $R(D)$ is the infimum of rates R s.t. (R, D) is in the rate distortion region of the source for given distortion D.

THEOREM: The rate distortion $R(D)$ for an iid source $X \sim p$ and bounded distortion $d(X, \hat{X})$ is

reconstruction of X

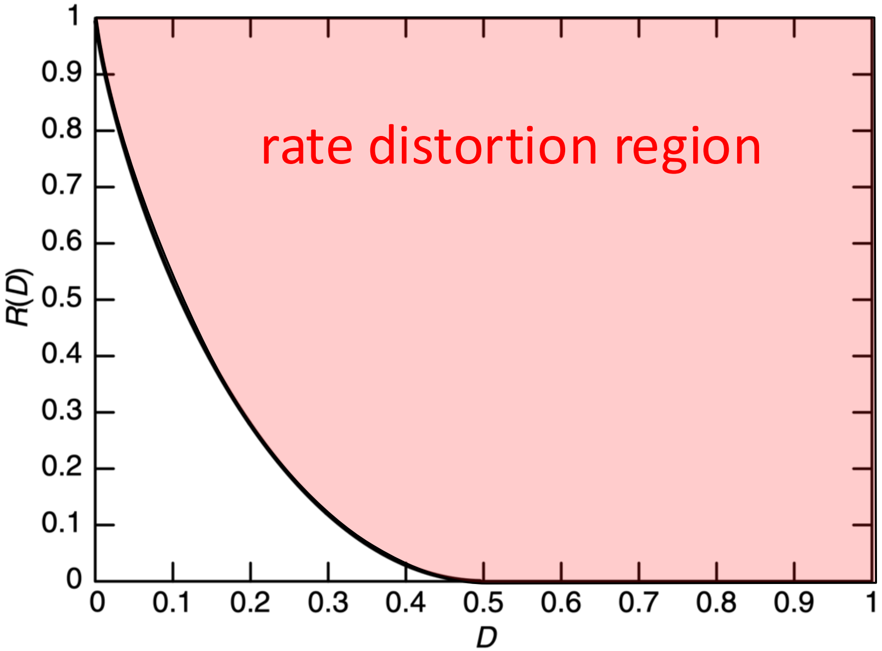
$$R(D) = \min_{p(\hat{X}|X):\ \mathbb{E}[d(X,\hat{X})] \leq D} I(X; \hat{X})$$

maximum allowable distortion

$\sum_{(x,\hat{x})} p(x,\hat{x}) \cdot d(x,\hat{x})$

$p(x) \cdot p(\hat{x}|x)$

rate distortion function for Bernoulli p with Hamming distortion



Rate Distortion function $R(D)$ Channel capacity C

reconstruction of X

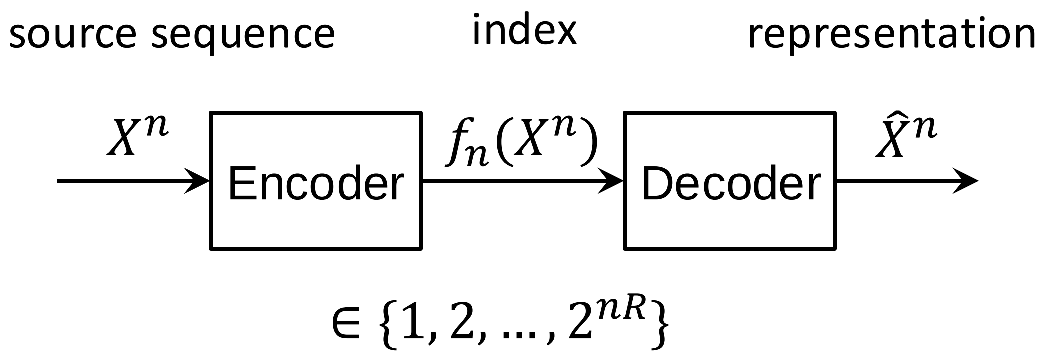
$$R(D) = \min_{p(\hat{X}|X):\ \mathbb{E}[d(X,\hat{X})] \le D} I(X;\hat{X})$$

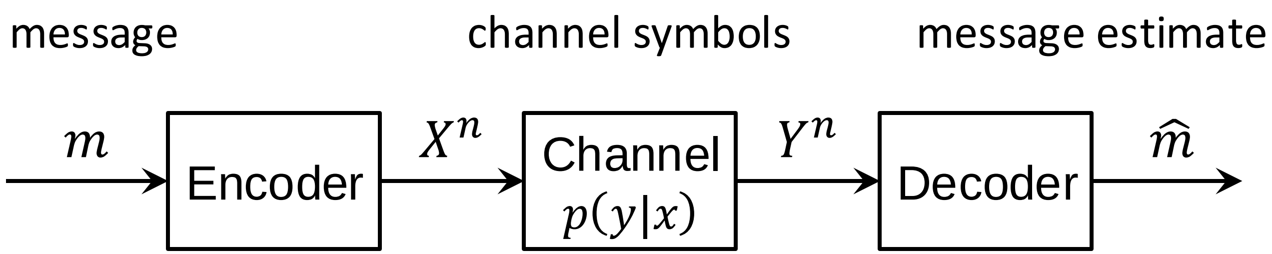
maximum allowable distortion

$$C = \max_{p(X)} I(X;Y)$$

RATE-DISTORTION THEORY   CHANNEL CODING THEORY

Why is one minimizing, the other maximizing mutual information ?





Rate Distortion function $R(D)$

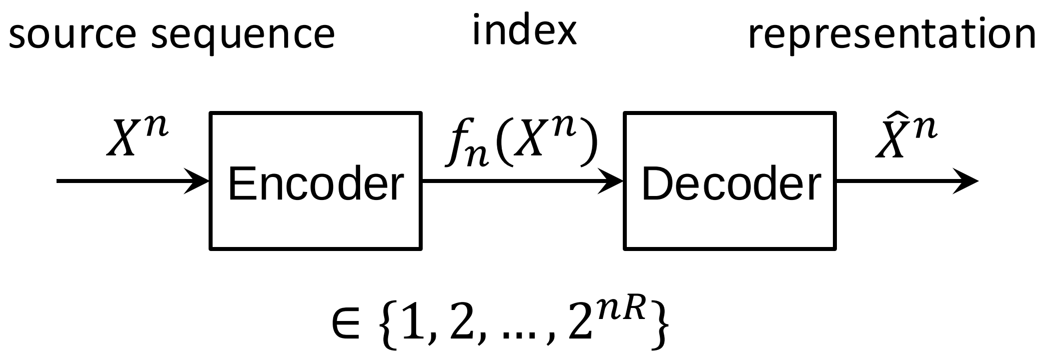
reconstruction of X

$$R(D) = \min_{p(\hat{X}|X):\ \mathbb{E}[d(X,\hat{X})] \leq D} I(X;\hat{X})$$

maximum allowable distortion

RATE-DISTORTION THEORY

- compress data X into a small representation $\hat{X}$ while satisfying a given distortion constraint $\leq D$ (and thus achieve a certain level of fidelity)

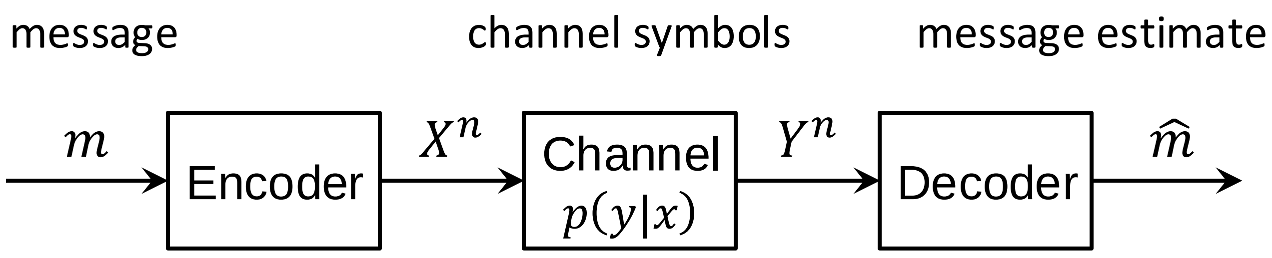


Channel capacity C

$$C = \max_{p(X)} I(X;Y)$$

CHANNEL CODING THEORY

- encode the information (via its input distribution $p(X)$) as to maximize the amount of information successfully transmitted through the channel



Rate Distortion function $R(D)$

reconstruction of X

$$R(D) = \min_{p(\hat{X}|X):\ \mathbb{E}[d(X,\hat{X})] \leq D} I(X;\hat{X})$$

maximum allowable distortion

RATE-DISTORTION THEORY

- compress data X into a small representation $\hat{X}$ while satisfying a given distortion constraint $\leq D$ (and thus achieve a certain level of fidelity)
- find the minimum communication rate $R = I(X;\hat{X})$ necessary to satisfy distortion $\leq D$
- Optimization (Minimization) over $p(\hat{X}|X)$ reflects the search for the most efficient encoding that meets the distortion D.

Channel capacity C

$$C = \max_{p(X)} I(X;Y)$$

CHANNEL CODING THEORY

- encode the information (via its input distribution $p(X)$) as to maximize the amount of information successfully transmitted through the channel
- find the maximum reliable communication rate $R = I(X;Y)$ that a channel can support (its capacity C)
- Optimization (Maximization) over $p(X)$ reflects the search for the input distribution that makes best use of the channel's capacity to transmit information.

2 Examples

Largely based on Ch10 of [Cover, Thomas'06] Elements of Information Theory, 2006. https://doi.org/10.1002/047174882X , and Ch 8 of [Yeung'08] Information Theory and Network Coding. https://doi.org/10.1007/978-0-387-79234-7

Rate Distortion for Bernoulli p with Hamming distortion

Consider a binary source $X \sim$ Bernoulli(p):

$$p(X=1) = p$$
$$p(X=0) = 1-p$$

WLOG, assume $p \leq 0.5$.

Assume a Hamming distortion measure

$$d(x,\hat{x}) = \begin{cases} 0 & \text{if } x = \hat{x} \\ 1 & \text{if } x \neq \hat{x} \end{cases}$$

If we had to guess x, should we rather guess x=0 or x=1?

?

$$\mathbb{P}[X=0] \geq 0.5$$

Our minimum expected distortion between X and a constant estimate of x=0 is:

?

$$D_{\max} = \mathbb{E}[d(X,0)] = \mathbb{P}[X=1] = p$$

Rate Distortion for Bernoulli p with Hamming distortion

Consider a binary source $X \sim \text{Bernoulli}(p)$:

$$p(X = 1) = p$$
$$p(X = 0) = 1 - p$$

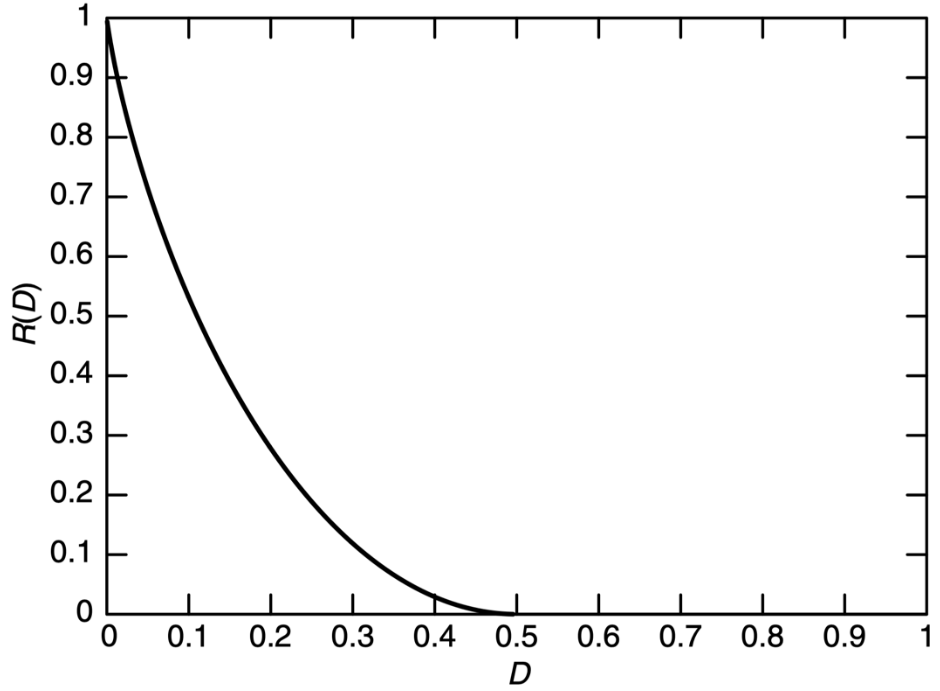
WLOG, assume $p \leq 0.5$.

Assume a Hamming distortion measure:

$$d(x, \hat{x}) = \begin{cases} 0 & \text{if } x = \hat{x} \\ 1 & \text{if } x \neq \hat{x} \end{cases}$$

What is the description rate $R(D)$ required to describe X with an expected proportion of errors less than or equal to D?

rate distortion function



$$R(D) = \begin{cases} H(p) - H(D), & 0 < D < p \\ 0, & \text{else} \end{cases}$$

Two steps (instead of minimizing $I(X; \hat{X})$ directly): We first find a lower bound. We then show that this lower bound is achievable.

Rate Distortion for Bernoulli p with Hamming distortion

Lower bound:

For any joint distribution satisfying the distortion constraint, we know:

$$I(X;\hat{X}) = H(X) - H(X|\hat{X})$$

$$= H(p) - H(Y|\hat{X})$$

$$\geq H(p) - H(Y)$$

$$\geq H(p) - H(D)$$

Let Y denote $d(X, \hat{X})$, or $(Y = 1) \Leftrightarrow (X \neq \hat{X})$. Then conditioning on $\hat{X}$, X and Y determine each other, and thus the uncertainty (entropy H) is the same if we consider X or Y: $H(X|\hat{X}) = H(Y|\hat{X})$

$H(Y|\hat{X}) \leq H(Y)$: our uncertainty can only reduce by conditioning (i.e. learning additional information)

since $\mathbb{P}[Y] = \mathbb{P}[X \neq \hat{X}] = \mathbb{E}[d(X \neq \hat{X})] \leq D$ for $D \leq p$, and $H(x)$ increases with $x \leq 0.5$

We thus have:

$$R(D) \geq H(p) - H(D)$$

Rate Distortion for Bernoulli p with Hamming distortion

We now show that the lower bound is actually the rate distortion function by finding a joint distribution $(X, \hat{X})$ that meets the distortion constraint and has $R(D) = I(X; \hat{X})$.

Concretely, for $0 \leq D \leq p$, we can achieve value $H(p) - H(D)$ for the rate distortion function $R(D)$ by choosing $(X; \hat{X})$ to have the joint distribution given by the following binary symmetric channel:

Recall that for a Binary Symmetric Channel
$I(X;Y) = H(Y) - H(p)$.

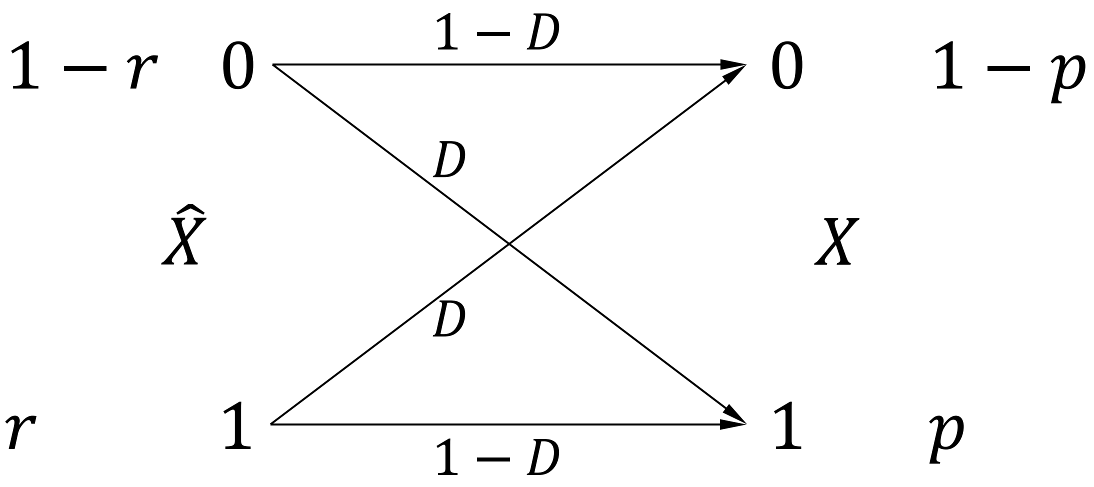
Here just p corresponds to D and Y to X:
$I(X;\hat{X}) = H(p) - H(D)$.

We need to find an appropriate $r_{\hat{X}}$ of $\hat{X}$ at the input of the channel s.t. the output distribution of X is the specified p_X.

Let $r = \mathbb{P}[\hat{X} = 1]$. Then choose r s.t.

$$r(1-D) + (1-r)D = p$$

$$\Rightarrow \quad r = \frac{p-D}{1-2D}$$



Rate Distortion for Bernoulli p with Hamming distortion

If $D \leq p \leq 0.5$, then:

- $\mathbb{P}[\hat{X} = 1] \geq 0$ and $\mathbb{P}[\hat{X} = 0] \geq 0$
- $I(X; \hat{X}) = H(X) - H(X|\hat{X}) = H(p) - H(D)$

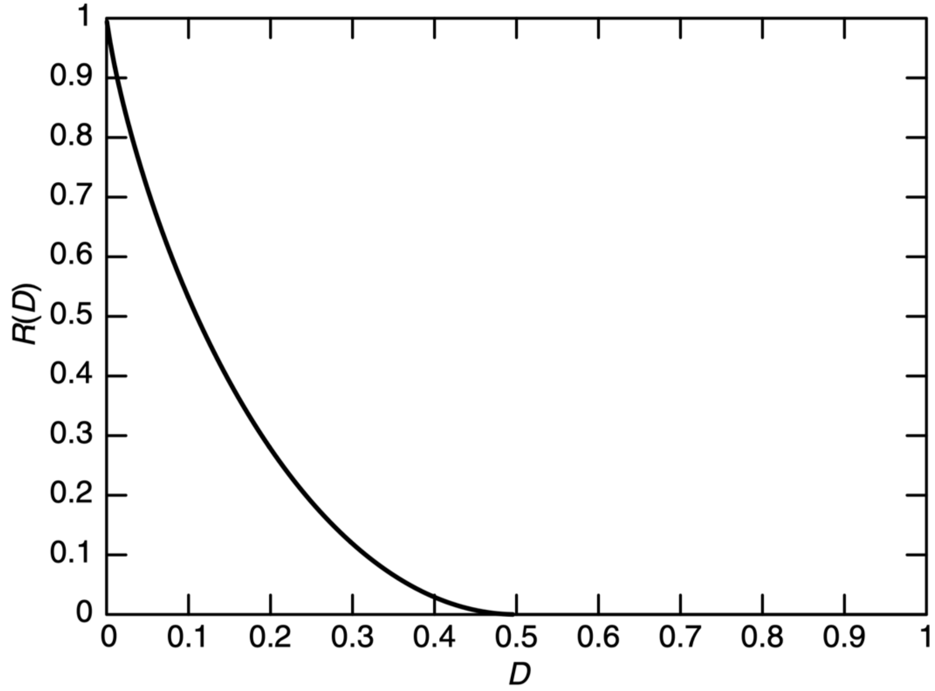
and the expected distortion is $\mathbb{P}[X \neq \hat{X}] = D$.

Indeed, the uncertainty of X when $\hat{X}$ is known is D, hence $H(X|\hat{X}) = H(D)$.

If $D \geq p$, then:

- We can achieve $R(D) = 0$ by letting $\hat{X} = 0$ with probability 1

rate distortion function



$$R(D) = \begin{cases} H(p) - H(D), & 0 < D < p \\ 0, & \text{else} \end{cases}$$

Rate Distortion for Gaussian source with squared error distortion

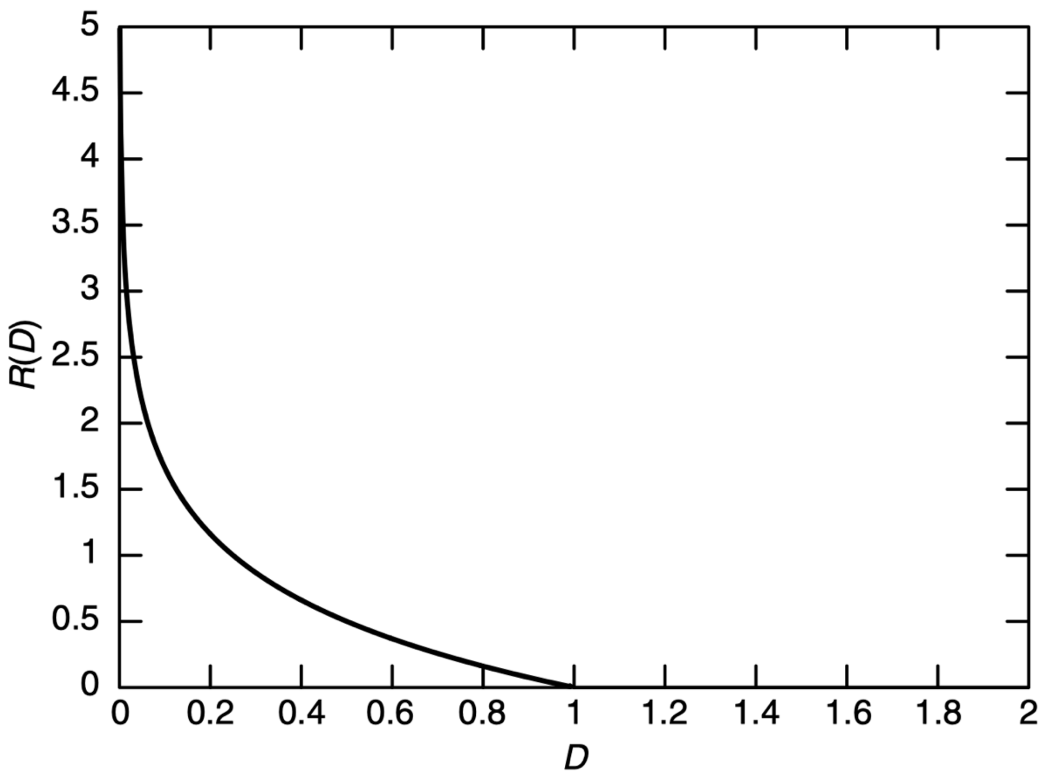
Consider a Gaussian source $X \sim \mathcal{N}(0, \sigma^2)$.

Assume a squared error distortion

$$d(x, \hat{x}) = (x - \hat{x})^2$$

WLOG, assume $p \leq 0.5$

Then the description rate $R(D)$ required to describe X with an expected proportion of errors less than or equal to D can be shown to be as follows:



$$R(D) = \begin{cases} \frac{1}{2}\ln\left(\frac{\sigma^2}{D}\right), & 0 \leq D \leq \sigma^2 \\ 0, & \text{else} \end{cases}$$

Proof: see book

Rate Distortion for Gaussian source with squared error distortion

We can rewrite $R(D)$ to express the distortion D in terms of the rate R: $D(R) = \sigma^2 2^{-2R}$

Each bit of description reduces the expected distortion by a factor of 4.

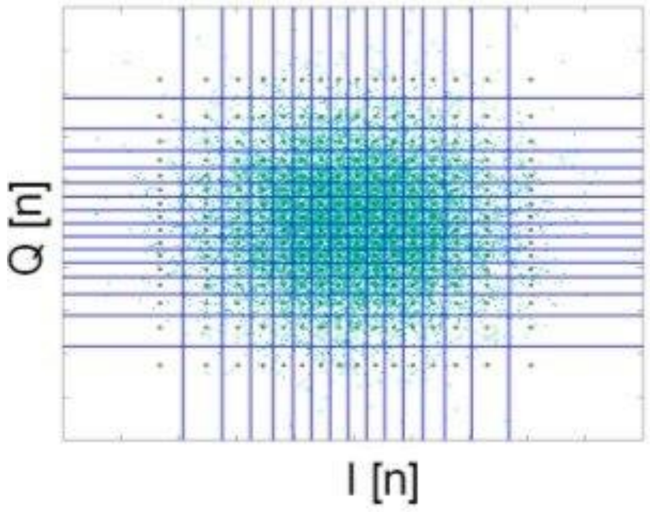
With a 1-bit description, the best expected square error is $0.25\sigma^2$.

Our simple 1-bit quantization from earlier can be calculated to be $0.36\sigma^2$.

The rate distortion limit $R(D)$ is achieved by considering several distortion problems in succession (longer block lengths) instead of considering each problem separately.

Geometry of longer block lengths:

Independent 4-bit quantization:



Blocklength $n = 2$ and 4-bit per sample

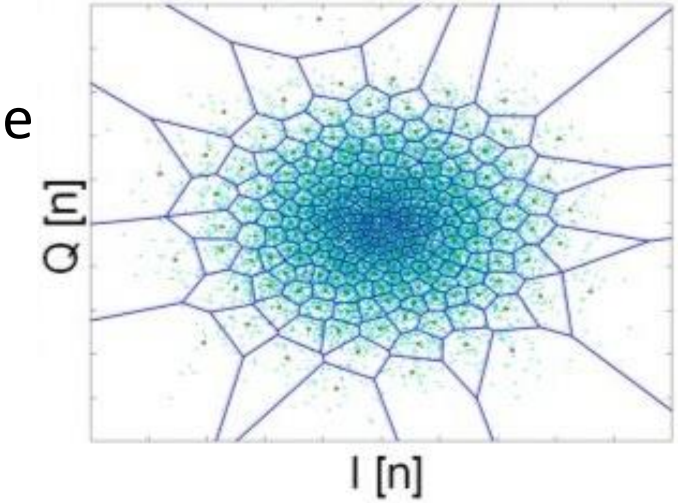


Figure source: https://ieeexplore.ieee.org/document/7767821/