SUMMARY

• DISCRETE RANDOM VARIABLES:

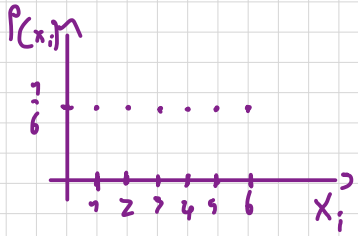
- LET X BE A RANDOM VARIABLE ~> R.V.
  - -> X CAN BE DEFINED AS: $(X, \Omega_X, P(x))$
    - $\Omega_X = \{x_1, x_2, \dots, x_n\}$: ALPHABET OF X -> ALL POSSIBLE OUTCOMES
    - $P(x) = \{p(x_1), p(x_2), \dots, p(x_n)\} \;/\; P(x_i) = p_i = P(X \overset{\sim>\text{R.V.}}{=} x_i)$ (i: OUTCOME i)

• UNIFORM DISTRIBUTION:

- X : $P(x) = \{P(x_1), \dots, P(x_n)\} \;/\; P(x_1) = \dots = P(x_n)$

eg. DICE

-> $\left(X, \Omega_X = \{1,2,3,4,5,6\}, P(x) = \{\frac{1}{6}, \dots, \frac{1}{6}\}\right)$



eg. COIN TOSS: n^0 OF TAILS (N_T) IN A 3 COINS TOSS

	N_T
H H H	0
H H T	1
H T H	1
H T T	2
T H H	1
T H T	2
T T H	2
T T T	3

-> $\Omega_X = \{0, 1, 2, 3\}$

-> $P(X = N_T) = \frac{1}{2^N}\binom{N}{i}$ -> $P(x) = \{\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8}\}$

• BINOMIAL DISTRIBUTION:

- $P(X = k) = \binom{N}{k} p^k (1-p)^{N-k}$

-> GIVEN $X = \{x_1, \dots, x_n\} \;/\; X_i \sim \text{BERNOULLI}(p)$ -> $X_i = \begin{cases} 0, & 1-p \\ 1, & p \end{cases}$

- L> k OF ALL X_i SATISFY THE PROPERTY WITH PROBABILITY p,
- L> $n-k$ OF ALL X_i DO NOT SATISFY THE PROPERTY

• GEOMETRIC DISTRIBUTION:

- LET X = n^0 OF TIME BEFORE 1 SUCCESS / p OF FAILURE = P; p OF SUCCESS = $1-p$

-> $P(X = i) = p^{i-1}(1-p)$ ~> eg. $P(X = 3 \to F, F, S) = p^2(1-p)$

- **EVENT**: ANY SUBSET OF Ω_X

  -> EVENT A: $P(A) = \sum_{x_i \in A} p(x_i)$

  - **INTERSECTION**: $P(A \cap B) = P(A, B) = \sum p(x_i) \,/\, x_i \in A$ AND $x_i \in B$
  - **UNION**: $P(A \cup B) = P(A) + P(B) - P(A \cap B)$

- **TOTAL PROBABILITY LAW**:

  LET Ω_X DECOMPOSE AS UNION OF DISJOINT EVENTS

  $\left(\rightarrow \Omega_X = \bigcup_i B_i \,/\, B_i \cap B_j = \phi \,,\, i \neq j\right)$

  $\rightarrow P(A) = \sum_i P(A, B_i)$

- **CONDITIONAL PROBABILITY**:

  - $P(A \mid B) = \dfrac{P(A, B)}{P(B)}$

  ex.

  $B = \{2, 4, 6\} \,/\, P(B) = \frac{1}{2}$, $A = \{4\} \,/\, P(A) = \frac{1}{2}$

  $\rightarrow P(A \mid B) = \dfrac{P(A, B)}{P(B)} = \frac{1}{6} / \frac{1}{2} = \frac{1}{3}$

  -> TOTAL PROBABILITY LAW: $P(A) = \sum_i P(A \mid B) P(B_i)$

- **BAYES THEOREM**:

  - $P(B \mid A) = \dfrac{P(A \mid B) \cdot P(B)}{P(A)}$

- **EXPECTATIONS:**
  - LET $F(X) \in \mathbb{R} \rightarrow E[F(X)] = \sum_{x_i \in \Omega_X} p(x_i) F(x_i)$
- **MOMENTS:**
  - $\mu_1 = \mu = E[X] = \sum_{x_i} p_{x_i} \cdot x_i$
  - $\mu_2 = E[X^2] = \sum_{x_i} x_i^2 \, p(x_i)$

→ VARIANCE: $\sigma^2 = E[(X-\mu)^2] = E[x^2 - 2x\mu + \mu^2] = E[x^2] - 2\mu^2 + \mu^2 =$

$= E[x^2] - \mu^2 = \mu_2 - \mu^2$

ex. DICE

- $E[X] = \sum_{x_i} p_{x_i} x_i$ / $X = \{1, 2, 3, 4, 5, 6\}$, $p_{x_i} = \frac{1}{6}$
  - → $E[X] = \frac{1}{6} \cdot \sum_{x_i} x_i = 1 \cdot \frac{1}{6} + \dots + 6 \cdot \frac{1}{6} = 3,5$
  - → $E[X^2] = \frac{1}{6} \sum_{x_i} x_i^2 = 1^2 \cdot \frac{1}{6} + \dots + 6^2 \frac{1}{6} = 15,17$
  - → $\sigma^2 = \mu_2 - \mu^2 = 15,17 - 3,5^2 = 2,92$

- **JOINT PMF (PROBABILITY MASS FUNCTION):**

LET X, Y : $p(x, y) = P(X = x, Y = y)$

- MARGINALIZATION: GIVEN A PMF → $P(x, y) \longmapsto P(X), P(Y)$

→ $p(x_i) = \sum_{y \in \Omega_Y} p(x_i, y)$

ex. WEATHER

	Temp < 25	Temp ≥ 25
Sunny	0.4	0.2
Cloudy	0.35	0.05

→ $P(X = \text{SUNNY}) = P(\text{SUNNY}, \text{TEMP} < 25) + P(\text{SUNNY}, \text{TEMP} \geq 25) = 0,4 + 0,2 = 0,6$

- **STATISTICAL INDEPENDENCE:**
  - X, Y ARE STAT. INDEP. $\Leftrightarrow p(x, y) = p(x)\, p(y)$, $\forall x, y$
- **CONDITIONAL PMF:**
  - $p(x \mid y) = P(X = x \mid Y = y) = \frac{p(x, y)}{p(y)}$

• INTRODUCTION TO INFORMATION THEORY:

• INFORMATION CONTENT:

REDUCIBLE VIDEO
https://youtu.be/B3y0RsVCyrw

LET $X : (X, \Omega_X, P(X))$

- INFORMATION IS INVERSELY PROP TO PROBABILITY → $\downarrow p \rightarrow \uparrow$ INFORMATION
- $p = 1 \rightarrow$ NO INFORMATIONS
- IF A, B STAT. INDIP. → $INFO_{TOT} = INFO(A) + INFO(B)$

→ SHANNON, 1948: DISCOVERED INFORMATION THEORY, SATISFYING ALL 3 CONDITIONS

- **INFORMATION:** $h(A) = \log_2\left(\frac{1}{P_A}\right)$
- **ENTROPY:** $H(X) = E[h(X)] = \sum_{x_i} p(x_i) \log_2\left(\frac{1}{p(x_i)}\right)$ (X ⇝ R.V.)

→ ENTROPY DESCRIBES THE AVERAGE INFORMATION CONTAINED IN A R.V.

- PROPERTY OF $\log$: $\sum_{x_i} p(x_i) \log_2\left(\frac{1}{p(x_i)}\right) = -\sum_{x_i} p(x_i) \log_2(p(x_i))$
- $H(X) \geq 0$
  - ↳ $H(X) = 0 \iff \exists i \,/\, p_i = 1 \quad (\rightarrow p_j = 0 \,/\, i \neq j)$ ("DEPENDS BY")
- $H(x)$ ⇝ PMF, c/ OUTCOMES
- WHY $\log_2$? → IN INFORMATION THEORY, INFO IS COUNTED IN bits
- WHAT IF $P_A = 0$? → WE CONSIDER $\lim_{p(x_i) \to 0} p(x_i) \log_2\left(\frac{1}{p(x_i)}\right) = 0$
  - ↳ $p_i = 0$ MUST BE DISCARDED: $H(p_1, 0, p_3) = H(p_1, p_3)$
- **ENTROPY INEQUALITIES:** $0 \leq H(X) \leq \log_2 N$, $X = \{x_1, \dots, x_N\}$ ($N = |\Omega_X|$)
  - PROOF:
    - WE HAVE TO PROVE $H(X) - \log_2 N \leq 0$

→ $\sum_i p_i \log_2 \frac{1}{p_i} - \log_2 N = \sum_i p_i \log_2 \frac{1}{p_i} - \sum_i p_i \log_2 N$ ($\sum_i p_i = 1$)

$\rightarrow$

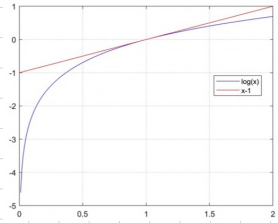
$$= \sum_i p_i \left(\log_2 \frac{1}{p_i} - \log_2 N \right) = \sum_i p_i \left(\log_2 \frac{1}{p_i} + \log_2 \frac{1}{N} \right) =$$

$$= \sum_i p_i \left[\log_2 \frac{1}{p_i N} \right] \overset{\text{CAMBIO BASE}}{=} \sum_i p_i \left[\frac{\ln 1/p_i N}{\ln 2} \right] =$$

$\log_a b = \frac{\log_k b}{\log_k a}$ (CAMBIO BASE)

$\frac{1}{\ln 2} = \frac{\ln e}{\ln 2} = \log_2 e$ (CAMBIO BASE)

$$= \log_2 e \cdot \sum_i p_i \ln \frac{1}{p_i N}$$



- LOG INEQUALITY: $\ln x \leq x - 1$

$$\rightarrow \log_2 e \cdot \sum_i^N p_i \ln \frac{1}{p_i N} \leq \log_2 e \sum p_i \left(\frac{1}{p_i N} - 1 \right)$$

$$\rightarrow \log_2 e \sum_i^N p_i \left(\frac{1}{p_i N} - 1 \right) = \log_2 e \left[\underbrace{\sum_i^N \frac{1}{N}}_{=1} - \underbrace{\sum_i p_i}_{=1} \right] = 0$$

$$\rightarrow H(X) - \log_2 N \leq 0$$

es. ENTROPY OF A BINARY R.V.

$$X: \Omega_X = \{0, 1\}, \quad P(X) = \{P(0) = p, P(1) = 1 - p\}$$

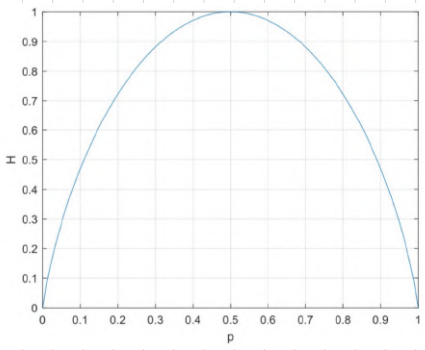
$$\rightarrow H(X) = \sum_i p_i \log_2 \frac{1}{p_i} = p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{1-p}$$

$\rightarrow$

- SYMMETRIC (LABEL INVARIANCE)
- $H(X) = 0 \iff p = 1$ OR $p = 0$
- $\max (H(X)) \iff p = 0,5$

$\rightarrow h(x) = \log_2 \frac{1}{\frac{1}{2}} = \log_2 2 = 1$

$\rightarrow$ IN $p = 0,5$ $H(X)$ IS MAX BECAUSE THE UNCERTAINTY IS MAXIMUM



• LAGRANGE OPTIMIZATION : ~> "LAGRANGE MULTIPLIERS"

IT'S A METHOD TO FIND min/max OF A CURVE, GIVEN SOME CONSTRAINT

↳ es. FIND min/max OF $f(x,y) = xy + 1$, GIVEN $x^2 + y^2 = 1$

• SET :

- $F(p_1, \dots, p_m)$: FUNTION TO FIND max/min
- $y_i(p_1, \dots, p_m)$: COSTRAINT FUNCTION
- λ_i : LAGRANGE MULTIPLIERS

USEFUL VIDEO ON ~> LAGRANGE MULTIPLIERS

https://youtu.be/8mjcnxGMwFo

https://www.youtube.com/watch?v=5A39Ht9Wcu0

-> WE DEFINE $\Delta \triangleq F + \sum_i \lambda_i y_i$

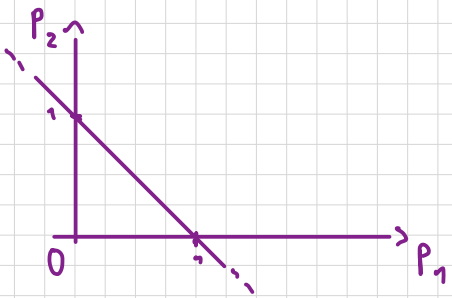
↳ UNDER THE COSTRAINT $y_i = 0$ -> A MAXIMUM FOR Δ IS A MAXIMUM FOR F

• WE CAN FIND max/min OF Δ WITH :

-> $\nabla \Delta(p_1, \dots, p_m, \lambda_1, \dots, \lambda_i) = \vec{0}$ -> $\left[\frac{\partial \Delta}{\partial p_i} = 0 \ , \ \frac{\partial \Delta}{\partial \lambda_i} = 0 \right]$

es.

FIND THE POINT AT MINIMUM DISTANT FROM $O(0,0)$, ON THE LINE $p_1 + p_2 = 1$



->

• DISTANCE FROM O : $F(p_1, p_2) = p_1^2 + p_2^2$

• COSTRAINS y : $p_1 + p_2 = 1$ -> $y = p_1 + p_2 - 1 = 0$

-> $\Delta(p_i, \lambda_i) = \Delta(p_1, p_2, \lambda_0) = F(p_1, p_2) + \lambda_0 y =$

$= p_1^2 + p_2^2 + \lambda_0(p_1 + p_2 - 1)$

• GRADIENT $\nabla \Delta$:

- $\frac{\partial \Delta}{\partial p_1} = 0$ -> $2p_1 + \lambda_0 = 0$
- $\frac{\partial \Delta}{\partial p_2} = 0$ -> $2p_2 + \lambda_0 = 0$

$\Big\}$ $p_1 = p_2 = -\frac{1}{2}\lambda_0$

-> F è min (=) $p_1 = p_2 = \frac{1}{2}$

- $\frac{\partial \Delta}{\partial \lambda_0} = 0$ -> $p_1 + p_2 - 1 = 0$ -> $\lambda_0 = -1$

· LAGRANGE OPTIMIZATION ON ENTROPY:

· WE CAN USE LAGRANGE OPT. TO OPTIMIZE $H(X)$, GIVEN $\sum_i p_i = 1$

->

$$\Delta(p_1, \dots, p_m, \lambda_0) = -\sum_i p_i \log_2 p_i + \lambda_0\left(\sum_i p_i - 1\right)$$

· $\frac{\partial \Delta}{\partial p_i} = -\log_2 p_i - p_i \frac{1}{p_i \ln 2} + \lambda_0 = -\log_2 p_i - \underbrace{\log_2 e}_{c} + \lambda_0 = 0$

($\frac{\partial(\sum_i p_i)}{\partial p_i} = \frac{\partial p_i}{\partial p_i}$, $\therefore \frac{\partial p_j}{\partial p_i} = 0$, FOR $i \neq j$; $\frac{1}{\log_a b} = \log_b a$)

$$\to -\log_2 p_i - c + \lambda_0 = 0 \to \log_2 p_i = \lambda_0 - c \to p_i = 2^{\lambda_0 - c}$$

↳ $\forall i$ -> UNIFORM DISTRIB.

· $\frac{\partial \Delta}{\partial \lambda_0} = \sum_i^N p_i - 1 = 0 \to \sum_i^N p_i = 1 \to p_i = \frac{1}{N}$

-> FOR ANY CARDINALITY N, $H(X)$ IS MAXIMIZED $\Leftrightarrow p_1 = \dots = p_m = \frac{1}{N}$

↳ $P(x)$ ~ UNIFORM DISTRIBUTION

$$\to H(X) = \sum_i p_i \log_2 \frac{1}{p_i} \overset{p_i = 1/N}{=} \sum_i p_i \log_2 N = \log_2 N \cdot \underbrace{\sum_i p_i}_{=1}$$

$$\to \max(H(X)) = \log_2 N$$

· PRINCIPLE OF MAXIMUM $H(x)$:

WE HAVE TO ESTIMATE A PROBABILITY DISTRIBUTION, ONLY HAVING A FEW DATA (ex. μ)

-> WE HAVE TO IDENTIFY THE PMF WITH THAT DATA AS CONSTRAINT AND MAXIMUM H

↳ SOLVABLE WITH LAGRANGE OPT.

->

· CONSIDER THE MEAN VALUE μ AS CONSTRAINT FOR THE DISTRIBUTION

$$\to \sum_i^N x_i p_i = \mu \quad \to \quad \psi_2(p_1, \dots, p_m) = \sum_i^N x_i p_i - \mu = 0$$

$$\to \Delta = H(X) + \lambda_0 \psi_1 + \lambda_1 \psi_2 \quad / \psi_1 : \sum_i p_i = 1$$

ex. DICE, WITH $\mu = 4$

· $\Delta = -\sum_i p_i \log_2 p_i + \lambda_0\left(\sum_i^N p_i - 1\right) + \lambda_1\left(\sum_i^N i\, p_i - \mu\right)$

->

① $\frac{\partial \Delta}{\partial p_i} = -\log_2 p_i - p_i \frac{1}{p_i} \log_2 e + \lambda_0 + i\lambda_1 = 0$

② $\frac{\partial \Delta}{\partial \lambda_0} = \sum_i^6 p_i - 1 = 0$

③ $\frac{\partial \Delta}{\partial \lambda_1} = \sum_i^6 i\, p_i - \mu = 0$

① -> $\frac{\partial \Delta}{\partial p_i} = -\log_2 p_i - c + \lambda_0 + i\lambda_1 = 0 \rightarrow p_i = 2^{\lambda_0 + i\lambda_1 - c} = 2^{\lambda_0 - c} \cdot 2^{i\lambda_1}$ ($c \rightsquigarrow \log_2 c$)

$= \alpha \beta^i \;/\; \alpha = 2^{\lambda_0 - c}, \; \beta = 2^{\lambda_1}$

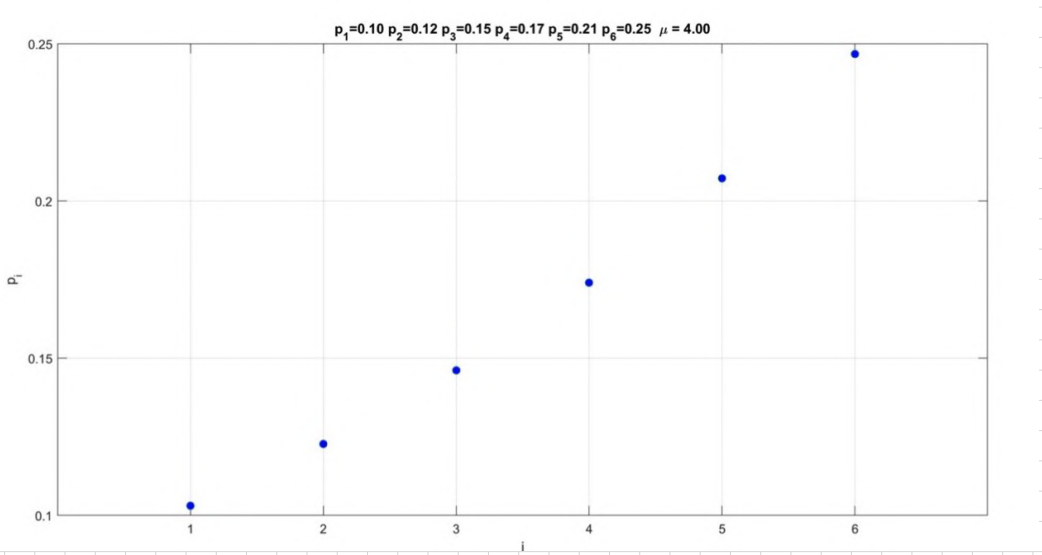
② -> $\frac{\partial \Delta}{\partial \lambda_0} = \sum_i^6 p_i = 1 \rightarrow \sum_i^6 \alpha \beta^i = 1 \rightarrow \alpha = \frac{1}{\sum_i^6 \beta^i}$

-> $p_i = \alpha \beta^i = \frac{\beta^i}{\sum_i^6 \beta^i}$

③ -> $\frac{\partial \Delta}{\partial \lambda_1} = \sum_i^6 i\, p_i = \mu \rightarrow \sum_i^6 \frac{i\,\beta^i}{\sum_i^6 \beta^i} = \mu$

-> $\sum_i^6 i\,\beta^i = \mu \sum_i^6 \beta^i$

- 1 SOL. : $\beta = 0$
- OTHER SOL. NUMERICALLY ... -> $\beta = 1{,}79$



- $\bar{n}$ AND $H(n)$:

LET $X = \{x_1, \dots, x_m\}$

- WE WANT TO REPRESENT EACH OUTCOME WITH A BINARY VECTOR
  → CARDINALITY N: $\log_2 N$ bits TO REPRESENT N OUTCOMES
- ROBERT FANO: STUDENT IN POLITO, THEN PROFESSOR AT MIT
  ↳ "YOU CAN DIVIDE THE SPACE TO HAVE ALMOST EQUAL SIZE"

ex.

	$p_1 = 0,5$	$p_2 = 0,25$	$p_3 = 0,125$	$p_4 = 0,125$
DIVISION 1:	0	1	1	1
DIVISION 2:		0	1	1
DIVISION 3:			0	1
	0	10	110	111
n° OF bits:	$n_1 = 1$	$n_2 = 2$	$n_3 = 3$	$n_4 = 3$

→ IT SATISFIES THE "PREFIX CONDITION": ANY BINARY CODE CANNOT BE THE INITIAL PART OF ANOTHER CODE

- FANO'S SOLUTION IT'S NOT OPTIMAL (HUFFMAN'S CODE)
- AVERAGE n° OF bits: $\bar{n} = \sum_i^N p_i n_i$

ex. * $\bar{n} = \sum_1^4 p_i n_i = 0,5 \cdot 1 + 0,25 \cdot 2 + 0,125 \cdot 3 + 0,125 \cdot 3 = 1,75$

- $H(x) = \sum_{i=1}^4 p_i \log_2 \frac{1}{p_i} = 0,5 \log_2 2 + 0,25 \log_2 4 + 2 \cdot 0,125 \log_2 8 =$
  $= 0,5 + 2 \cdot 0,25 + 3 \cdot 2 \cdot 0,125 = 1,75$

→ $\bar{n} = H(x)$

→ IN GENERAL: $H(x) \leq \bar{n} \leq H(x) + 1$

- **ENTROPY OF $F(X)$:**

LET $X / \Omega_X = \{x_1, \dots, x_n\}$

- WHAT IS THE RELATION BETWEEN $H(X)$ AND $F(H(X))$?

→ IF F IS INJECTIVE → $H(X) = H(F(X))$

↳ f: $\forall x \in \text{dom}(f) \rightarrow f(x_i) = f(x_j) \Leftrightarrow i = j$

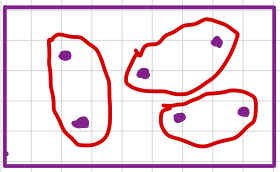
→ WHEN 2 OR MORE X_i ARE MAPPED ON THE SAME Y → $H(F(X)) < H(X)$

↳ ∵ WHEN $F(x_1 \text{ with } p = p_1) = F(x_2 \text{ with } p = p_2) = k \rightarrow p_{TOT} = p_1 + p_2$

$$\rightarrow \begin{cases} H(X) = p_1 \log_2 \frac{1}{p_1} \\ H(F(X)) = (p_1 + p_2) \log_2 \frac{1}{p_1 + p_2} \end{cases}$$

IF $p = p_1 = p_2$ → $H(F(X)) < H(X)$

ex. CLUSTERING



→ $H(X) > H(F(X))$

- **JOINT ENTROPY:**

GIVEN 2 R.V. X AND Y

$$\rightarrow H(X,Y) = \sum_{x,y \in \Omega_X \times \Omega_Y} p(x,y) \log_2 \frac{1}{p(x,y)}$$

- $0 \leq H(X,Y) \leq \log_2(N_X \cdot N_Y)$
- $H(X,Y) \leq H(X) + H(Y)$

→ $H(X,Y) = H(X) + H(Y) \Leftrightarrow X, Y$ STAT. INDIP.

↳ PROOF ON SLIDES

• CONDITIONAL ENTROPY :

WE KNOW :

$$\begin{cases} H(X,Y) = \sum_{x,y \in \Omega_X \times \Omega_Y} p(x,y) \log_2 \frac{1}{p(x,y)} \\ p(x,y) = p(x|y)\, p(y) \end{cases}$$

$$\rightarrow H(x,y) = \sum_{x,y} p(x,y) \log_2 \frac{1}{p(x|y)p(y)} =$$

$$= \sum_{x,y} p(x,y) \log_2 \frac{1}{p(x|y)} + \sum_{x,y} p(x,y) \log_2 \frac{1}{p(y)} =$$

$$= \sum_{x,y} p(x,y) \log_2 \frac{1}{p(x|y)} + \sum_{y} p(y) \log_2 \frac{1}{p(y)} =$$

$$= H(X|Y) + H(Y)$$

$$\rightarrow H(X|Y) = \sum_{x,y} p(x,y) \log_2 \left(\frac{1}{p(x|y)} \right)$$

• ENTROPY CHAIN :

$$\rightarrow H(X,Y,Z) = H(X|YZ) + H(YZ) =$$

CONSIDERING $H(x,y)$ ↑

$$= H(X|YZ) + H(Y|Z) + H(Z)$$

→

- GIVEN X, BEFORE WE KNOW ITS OUTCOME, THE UNCERTAINTY IS $H(x)$
  - → AFTER WE KNOW THE OUTCOME OF X, THE UNCERTAINTY IS 0
  - → AFTER WE KNOW THE OUTCOME OF Y, THE UNCERTAINTY IS $H(X|Y)$
    - → $H(X|Y)$ IS THE REMAINING UNCERTAINTY OF X, AFTER Y OUTCOME
- $0 \leq H(X|Y) \leq H(X)$

→ $H(X|Y) = H(X) \iff X, Y$ STAT. INDIP.

- **MUTAL INFORMATION (INFORMATION GAIN):**

→ VIDEO EXPAINING MEANING OF $I(X;Y)$
https://youtu.be/U9h1xkNELvY

- $I(X;Y) = H(X) - H(X|Y)$
- $0 \leq I(X;Y) \leq H(X)$
- $I(X;Y) = H(X) - H(X|Y) + (H(Y) - H(Y))$
  - SINCE $H(X,Y) = H(X|Y) + H(X) = H(Y|X) + H(X)$

  $\rightarrow I(X;Y) = H(X) + H(Y) - H(X,Y)$
- $I(X;Y) = H(X) + H(Y) + H(X,Y) / H(X,Y) = H(Y,X)$

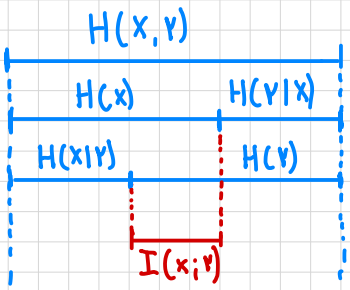
  $\rightarrow I(X;Y) = H(X) + H(Y) + H(Y|X) - H(X) =$

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

- **PROPERTIES SUMMARY:**

$I(X;Y)$ IS SYMMETRIC

- $0 \leq I(X;Y) \leq H(X)$
- $I(X;Y) = H(X) + H(Y) - H(X,Y)$
- $I(X;Y) = H(Y) - H(Y|X) = H(X) - H(X|Y)$



- **POINT WISE MUTUAL INFORMATION:**

$$\log_2\left(\frac{p(x,y)}{p(x)\,p(y)}\right) \Big/ I(X,Y) = \sum p(x,y) \log_2\left(\frac{p(x,y)}{p(x)\,p(y)}\right)$$

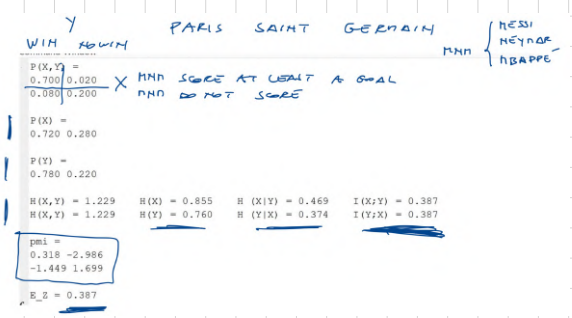
→ GIVEN A PAIR OF OUTCOME (x,y), IT COMPARES THE TRUE JOINT PROBABILITY AGAINST THE PROBABILITY WE WOULD HAVE IF THE 2 EVENTS WHERE STAT. INDIP.

- IF $p(x,y) = p(x)\,p(y)$ → X, Y STAT. INDIP.

  $\rightarrow \frac{p(x,y)}{p(x)\,p(y)} = 1 \rightarrow \log_2(\dots) = 0$

07. PARIS SAINT GERMAIN

- Y : {PSG WIN, PSG LOSE}
- X = {MNM GOAL >1, MNM NO GOAL}



-> THE INFORMATION OF THE R.V. Y IS $H(Y) = 0,76$

WHEN X IS REVEALED, IT AFFECTS $H(Y)$ -> WE CAN COMPUTE HOW MUCH THE REVELATION OF X AFFECTS $H(Y)$, COMPUTING $H(Y|X)$

-> THE FINAL INFORMATION GAIN IS $I(X;Y) = H(Y) - H(Y|X)$

- WHEN WE OBSERVE THE OUTCOME OF A R.V., IT AFFECTS THE INFORMATION GIVEN BY THE OTHERS R.V. -> IT REDUCE THE UNCERTANTY
- PMI (POINTWISE MUTUAL INFO) CAN BE A MEASURE OF HOW MUCH THE CORRELATION OF THE 2 R.V. AFFECTS EACH OTHER
- <u>KULLBACK-LEIBLER DISTANCE:</u>

SUPPOSE A R.V. X, $\mathcal{R}_X = \{x_1, \dots, x_m\}$, $\begin{cases} P(X) = \{p_1, \dots, p_m\} \\ Q(X) = \{q_1, \dots, q_m\} \end{cases}$

-> P, Q ARE 2 DISTRIBUTIONS DEFINED ON X

- IN ORDER TO COMPARE THEM, WE COMPUTE THE KL DISTANCE:

$$D_{KL}(P||Q) = \sum_i p_i \log_2 \frac{p_i}{q_i}$$

IF $p_i = q_i$ -> $D_{KL}(P||Q) = 0$

IF $p_i \neq q_i$ -> $D_{KL}(P||Q) > 0$

- PROOF $D_{KL} \geq 0$:

$$-D_{KL} \leq 0 \rightarrow -D_{KL} = \sum_i p_i \log_2 \frac{q_i}{p_i} = \log_2 e \sum_i p_i \log_2 \frac{q_i}{p_i} =$$

$$= \log_2 e \sum_i p_i \log_2 \frac{q_i}{p_i} \leq \log_2 e \sum_i p_i \left(\frac{q_i}{p_i} - 1\right)$$

$$\sum_i q_i - \sum_i p_i = 0 \rightarrow D_{KL} \geq 0$$

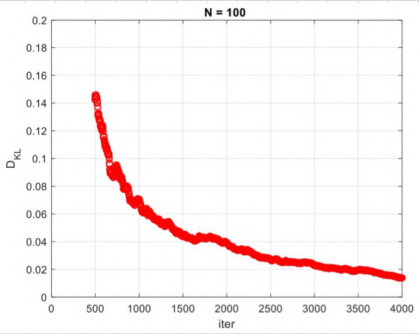
- THE KL DISTANCE IS USED TO MEASURE THE DISTANCE BETWEEN 2 DISTRIBUTION

-> TYPICAL EXAMPLE:

- $P(X)$ IS THE EMPIRICAL DISTRIBUTION OF THE OBSERVED DATA
- $Q(X)$ IS A DISTRIBUTION WE USE TO MODEL $P(X)$

eg.

- $D_{KL}(P||Q)$ WHERE:
  - $P(X) \sim U[1; N]$
  - $Q(X) \sim$ {RANDOM GENERATED DISTRIBUTION BETWEEN 1 e N



-> GRAPH: $D_{KL} = f(n^{\circ}\text{ITERAZIONI})$, WHEN $N=100$, $D_{KL} \longrightarrow 0$ ITER $\to \infty$

- KL DISTANCE AND MUTUAL INFORMATION:

$$I(X;Y) = \sum_{x,y} p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)} = D_{KL}\left(p(x,y) \,||\, p(x)p(y)\right)$$

$$\to D_{KL}(P||Q) = \sum_{x,y} p_i \log_2 \frac{p_i}{q_i}$$

- IF $P(X)$ = TRUE DISTRIBUTION OF X, $Q(X)$ = DISTRIBUTION USED TO MODEL $P(X)$

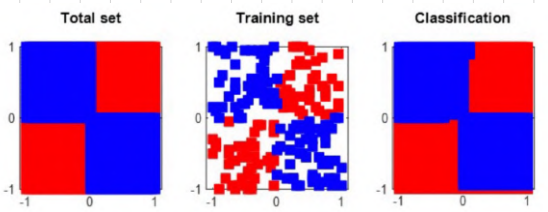
-> D_{KL}: {INFORMATION GAIN WE HAVE WHEN WE USE $P(X)$ INSTEAD OF THE MODEL $Q(X)$
INFORMATION LOSS WE HAVE WHEN WE USE $Q(X)$ INSTEAD OF THE TRUE $P(X)$

- N.B: D_{KL} IS NOT A REAL MATHEMATIC DISTANCE

$$\to D_{KL}(P||Q) \neq D_{KL}(Q||P)$$

• APPLICATION OF INFORMATION THEORY TO CLASSIFIERS:

• CLASSIFIERS:



- GIVEN:
  - FEATURES (OR VARIABLES): $\{X_1, \dots, X_m\}$
  - ALPHABETS: $\{\Omega_1, \dots, \Omega_m\}$
  - VECTOR (OR INSTANCE/OBSERVATION): $\vec{v} = (x_1, \dots, x_N) \;/\; x_i \in \Omega_i$
    - $A_{\vec{v}}$ (TOTAL ALPHABET OF $\vec{v}$) $\subseteq A_{TOT} = \Omega_1 \times \dots \times \Omega_m$ (CARTESIAN PRODUCT)
  - CLASS (OR CATEGORY) C / ALPHABET Ω_C

→ A FUNCTION F IS DEFINED: $F: A_V \longmapsto C \rightarrow F: \vec{v} \mapsto C$ (SINGLE CLASS)

→ THE TOTAL FUNCTION F IS UNKNOWN, BUT THERE IS A SUBSET TS (TRAINING SET) / $\forall \vec{v} \in TS \rightarrow F(\vec{v}) = C$ IS KNOWN

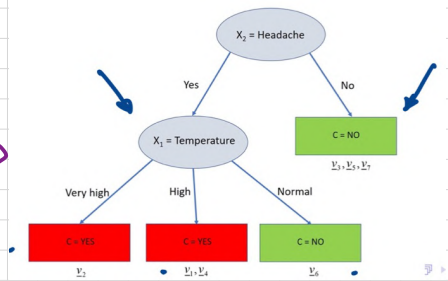
- STARTING FROM TS, WE WANT TO BUILD A FUNCTION y / $y: A_V \mapsto C$, y HAS THE GOAL OF $y(\vec{v}) = F(\vec{v})$, $\forall \vec{v} \notin TS$
  → IN GENERAL:

$$y: \begin{cases} \forall w \in TS \rightarrow y(w) = F(w) \\ \forall v \notin TS \rightarrow y(v) = F(v) \end{cases}$$ ~> THIS IS OUR GOAL

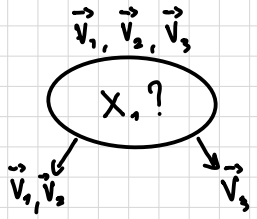
→ WE WANT y TO PROVIDE THE TRUE CLASS, $\forall \vec{v}$

ex. DATA ⟶ PATIENT ILL OR HEALTY

	Attributes			Decision
	Temperature	Headache	Nausea	Flu
1	high	yes	no	yes
2	very_high	yes	yes	yes
3	normal	no	no	no
4	high	yes	yes	yes
5	high	no	yes	no
6	normal	yes	no	no
7	normal	no	yes	no



- **DECISION TREE CLASSIFIERS:** ⇝ FLOW CHART



ON EACH NODE WE MAKE A TEST ON A GIVEN FEATURE
→ THIS DIVIDES THE SET OF VECTOR UNDER ANALYSIS IN THAT NODE INTO SUBSETS
→ AT THE END WE HAVE LEAVES WHERE THE FINAL DECISION ON THE CLASS IS TAKEN

- **INFORMATION GAIN APPLICATION:**

WE USE THE INFORMATION GAIN TO DECIDE THE FEATURE TO BE TESTED

- WE ALWAYS CHOOSE THE THE FEATURE X_i / $I(C; X_i)$ IS MAX
  → IT REDUCE THE UNCERTAINTY ABOUT THE CLASS

- **INFORMATION GAIN RATIO:** ⇝ WE WILL WORK WITH THIS INSTEAD OF IG, ∵ IT'S LIKE A NORMALIZATION, IT TAKES INTO ACCOUNT THAT DIFFERENT FEATURES CAN HAVE DIFFERENT CARDINALITIES

$$IGR(C; \vec{X}_i) = \frac{I(C; X_i)}{H(X_i)}$$ (FEATURE)

→ THE ALGORITHM BASED ON THESE IDEAS IS THE ID3 (ITERATIVE DICHOTOMIZER 3)

- **ID3 ALGORITHM:**

1. GIVEN A SET S OF VECTORS $\vec{V} = \{x_1, \dots, x_N\}$ AND THEIR CLASSES $C = F(\vec{V})$
   → COMPUTE THE FEATURE X_i / $IGR(C; X_i)$ IS MAX
2. CREATE A DECISION NODE FOR THIS FEATURE X_i AND SPLIT THE SET S INTO SUBSETS ACCORDING TO IT
3. RECURSIVELY REPEAT ON SUBSETS BY USING THE REMAINING FEATURES

ex. DECISION TREE FOR PATIENT ILL / HEALTHY

	X_1 Temperature	X_2 Attributes Headache	X_3 Nausea	C Decision Flu
1	high	yes	no	yes
2	very_high	yes	yes	yes
3	normal	no	no	no
4	high	yes	yes	yes
5	high	no	yes	no
6	normal	yes	no	no
7	normal	no	yes	no

1.

- COMPUTE THE ENTROPY OF THE CLASS:

$$H(C) = \sum_i p_i \log_2\left(\frac{1}{p_i}\right)$$

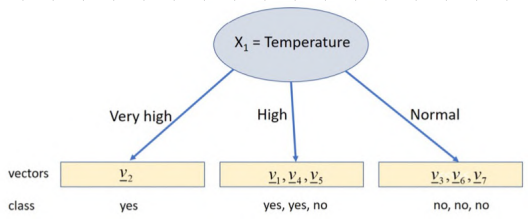
$/ i = 1, i = 2 \rightarrow p_1 = P(c = YES) = \frac{3}{7}, \; p_2 = P(c = NO) = \frac{4}{7}$

$\rightarrow H(C) = 0,9852$

- COMPUTE $IGR(C; X_i) \; / \; i = 1,2,3$:

$\rightarrow H(C)$

- X_1 = TEMPERATURE:



IS ONLY COMPUTABLE ON A R.V., NOT ON HIS OUTCOMES

ex. $H(X_1 = H)$ DOES NOT HAVE A MEANING

↳ IT EXISTS $h(X_1 = H)$

$$\cdot H(C \mid X_1 = \overbrace{VERY\ HIGH}^{VH}) = H(C = \{YES, NO\} \mid X_1 = VH) =$$

$$= \sum_c p(c, VH) \log_2 \frac{1}{p(c \mid VH)} =$$

$$= p(YES, VH) \overbrace{\log_2 \frac{1}{p(YES, VH)}}^{=0} + \overbrace{p(NO, VH)}^{=0} \log_2 \frac{1}{p(NO \mid VH)} = 0$$

$$\cdot H(C \mid X_1 = \overbrace{HIGH}^{H}) = H(C = \{YES, NO\} \mid X_1 = H) =$$

IT'S NOT A REAL CONDITIONAL ENTROPY

↳ $\because X_1$ HAS AN OUTCOME

MATHEMATICALLY WRONG, BUT TO UNDERSTAND...

$$= H(C = YES \mid X_1 = H) + H(C = NO \mid X_1 = H) =$$

$$= \underbrace{p(YES \mid H)}_{=2/3} \log_2\left(\frac{1}{p(YES \mid H)}\right) + \underbrace{p(NO \mid H)}_{=1/3} \log_2\left(\frac{1}{p(NO \mid H)}\right) = 0,9183$$

$$\cdot H(C \mid X_1 = NORMAL) = H(C = \{YES, NO\} \mid X_1 = N) =$$

$$= H(C = YES \mid X_1 = N) + H(C = NO \mid X_1 = N) =$$

$$= \underbrace{p(YES \mid N)}_{=0} \log_2\left(\frac{1}{p(YES \mid N)}\right) + \underbrace{p(NO \mid N)}_{=1} \log_2\left(\frac{1}{p(NO \mid N)}\right) = 0$$

- $H(X_1) = H(X_1 = \{\vec{1}VH, \vec{2}H, \vec{3}N\}) = H(X_1 = VH) + H(X_1 = H) + H(X_2 = N) =$ ($P_{VH} = 1/7$, $P_H = 3/7$, $P_N = 3/7$)

$= 0,4011 + 0,5239 + 0,5239 = 1,4688$

- $H(C|X_1) = H(C = \{\vec{1}YES, \vec{2}NO\} \mid X_1 = \{\vec{1}VH, \vec{2}H, \vec{3}N\}) =$

$= \sum_{c} \sum_{X_{1,j}} p(c, x_{1,j}) \log_2 \left(\frac{1}{p(c|x_{1,j})} \right) =$

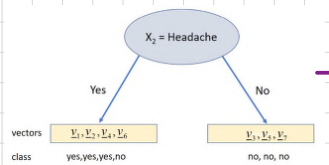
$= \underbrace{p(YES, VH)}_{=1/7} \log_2 \frac{1}{\underbrace{p(YES|VH)}_{=1}} + \underbrace{p(YES, H)}_{=3/7} \log_2 \frac{1}{\underbrace{p(YES|H)}_{=3/3}} + \underbrace{p(YES, N)}_{=0} \log_2 \frac{1}{\underbrace{p(YES|N)}_{=0}} +$

$+ \underbrace{p(NO, VH)}_{=0} \log_2 \frac{1}{\underbrace{p(NO|VH)}_{=0}} + \underbrace{p(NO, H)}_{=1/7} \log_2 \frac{1}{\underbrace{p(NO|H)}_{=1/3}} + \underbrace{p(NO, N)}_{=3/7} \log_2 \frac{1}{\underbrace{p(NO|N)}_{=1}} =$

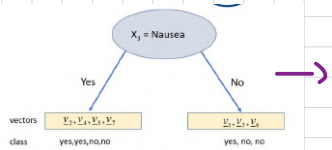
$= 0 + 0,1677 + 0 + 0 + 0,2264 = 0,3936$

- $I(C; X_1) = H(C) - H(C|X_1) = 0,9852 - 0,3936 = 0,5916$
- $IGR(C; X_1) = \frac{I(C; X_1)}{H(X_1)} = \frac{0,5916}{1,4688} = 0,4083$

-> REPEAT THE SAME FOR X_2, X_3:



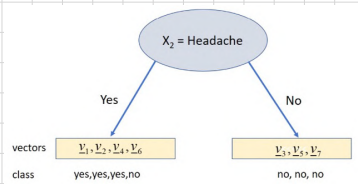
$H(C) = 0.9852$
$H(C|X_2 = \text{Yes}) = 0.8113$
$H(C|X_2 = \text{No}) = 0$
$H(C|X_2) = 0.4636$
$I(C; X_2) = 0.5216$
$H(X_2) = 0.9852$
$IGR(C, X_2) = 0.5294$



$H(C) = 0.9852$
$H(C|X_3 = \text{Yes}) = 1$
$H(C|X_3 = \text{No}) = 0.9183$
$H(C|X_3) = 0.9650$
$I(C; X_3) = 0.0202$
$H(X_3) = 0.9852$
$IGR(C, X_3) = 0.0205$

2. $IGR(C, X_2)$ IS THE HIGHEST

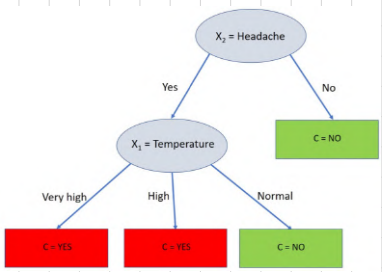
-> WE CHOOSE X_2 AS 1° NODE



3. WE REPEAT (1.) FOR $\vec{V}_1, \vec{V}_2, \vec{V}_4, \vec{V}_6$ (CONSIDERING ONLY X_1, X_3)

-> $\vec{V}_3, \vec{V}_5, \vec{V}_7$ IS ALREADY A FINAL LEAF

FINAL DECISION TREE:



- **ID3 STOPPING CRITERIA:**
  - WHEN A SUBSET CONTAINS ONLY VECTOR OF SAME CLASS → LEAF
  - WHEN A SUBSET CONTAINS VECTOR OF DIFFERENT CLASSES BUT ALL THE FEATURE HAVE ALREADY BEEN CONSIDERED → LEAF / LABEL: MOST COMMON CLASS IN $\vec{V}_i$
  - WHEN A SUBSET IS EMPTY → LEAF / LABEL: MOST COMMON CLASS OF PARENT'S SUBSET

- **NUMERICAL FEATURE:**
  - FIX A THRESHOLD t AN SPLIT $X_i \leq t$, $X_i > t$
  - COMPUTE IGR
  - CHANGE t
  - CONSIDER t / IGR IS MAX

- **N.B:** NUMERICAL FEATURES CAN BE CONSIDERED AGAIN, WITH A DIFFERENT t

- **ENTROPY FOR DATA SERIES:**

  - SHANNON'S ENTROPY CANNOT BE APPLIED TO DATA SERIES

  eg.

  1. 1 0 1 1 0 1 0 1 0 1 0 1 0 → $N_0 = N_1 = 7$
  2. 1 0 1 1 0 1 1 1 0 1 0 0 0 0 → $N_0 = N_1 = 7$

  → 1. & 2 HAVE SAME H, BUT SECOND IS MORE RANDOM

- **KOLMOGOROV DEFINITION OF COMPLEXITY:**

  MINIMUM n^o OF BITS OF A MASCHINE TO COMPLETELY REPRESENT THE SEQUENCE

- **APPROXIMATE AND SAMPLE ENTROPY:**

  LET $\bar{V} = (V_1, \dots, V_m)$, M = LENGTH OF A PATTERN,

  - $\bar{X} = (X_1, \dots, X_m)$, $\bar{Y} = (Y_1, \dots, Y_m)$ / $d_c(\bar{X}, \bar{Y}) = \max_i \{|X_i - Y_i|\}$
    ↳ CHEBISCEV

  - TOTAL LENGTH - M PATTERNS INSIDE $\bar{V}$:

    PATTERN

    $$T_M = \{\bar{X}_m(i)\} \,/\, \bar{X}_m(i) = \{V_i, \dots, V_{i+N-1}\}$$

  → TO COMPUTE H:

  1. CONSIDER ALL $\bar{X}_n, \bar{Y}_n \in T_m$
  2. COMPUTE $d_c(X_n, Y_n) \leq t$ ~> THRESHOLD

- **PERMUTATION ENTROPY:**

- WE WILL CONSIDER AN EXAMPLE TO UNDERSTAND IT (EXPLANATION: https://www.aptech.com/blog/permutation-entropy/)

->

$$\bar{V} = \{2, 8, 6, 7, 11, 14, 16\}$$

- WE CHOOSE N_R : N_R WILL BE THE N° OF ROW OF OUR MATRIX

1. EACH ROW WILL BE A SUBSET OF LENGTH N_R, EACH SUBSET IS COMPUTED SHIFTING THE SLIDING WINDOW BY 1 POSITION EACH TIME

-> $\bar{V} = \{2, 8, 6, 7, 11, 14, 16\}$ (1°, 2°, 3°, ...) MATRIX: ⟶

1°	2°	3°	...	
2	8	6	7	11
8	6	7	11	14
6	7	11	14	16

2. EACH COLUMN OF THE MATRIX WILL BE MAPPED TO A SEQUENCE OF SUCCESSIVE NUMBERS, STARTING FROM LOWEST TO HIGHEST

ex. $\{15^{2°}, 27^{3°}, 30^{4°}, 12^{1°}\}$ (HIGHEST, SMALLEST) MAPPING ⟶ $\{2, 3, 4, 1\}$

ex. $\{15^{4°}, 7^{1°}, 9^{3°}, 7^{2°}\}$ MAPPING ⟶ $\{4, 1, 3, 2\}$

->

$$\begin{bmatrix} 2 & 8 & 6 & 7 & 11 \\ 8 & 6 & 7 & 11 & 14 \\ 6 & 7 & 11 & 14 & 16 \end{bmatrix} \longmapsto \begin{bmatrix} 1 & 3 & 1 & 1 & 1 \\ 3 & 1 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 & 3 \end{bmatrix}$$

3. p_i WILL BE THE FREQUENCY OF EACH COLUMN IN THE MATRIX

-> $\tilde{\pi}_1, \tilde{\pi}_2, \tilde{\pi}_3, \tilde{\pi}_3, \tilde{\pi}_3$

$$\begin{bmatrix} 1 & 3 & 1 & 1 & 1 \\ 3 & 1 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 & 3 \end{bmatrix}$$

PERMUTATIONS:

-> $p_{\tilde{\pi}_1} = 1/5$, $p_{\tilde{\pi}_2} = 1/5$, $p_{\tilde{\pi}_3} = 3/5$

4. $H(X)$ CAN NOW BE COMPUTED AS: $H_p(X) = p_{\tilde{\pi}_i} \sum_i \log_2 \left(\frac{1}{p_{\tilde{\pi}_i}}\right)$

- **RENYI ENTROPY:**

$$H_\alpha(X) = \frac{1}{1-\alpha} \log_2 \left[\sum_i p_i^\alpha \right] \quad , \alpha \neq 1, \ \alpha \geq 0$$

- IF $\alpha = 1$: SHANNON'S ENTROPY

$$\lim_{\alpha \to 1} H_\alpha(X) = -\sum_i p_i \log_2 p_i \overset{\sim}{=} H(X)$$

- OTHERS α:
  - $\alpha = 0$: $H_0(X) = \log_2 N$ ~> CARDINALITY $|\Omega_X|$
  - $\alpha = 2$: $H_2(X) = -\log_2 \left[\sum_i p_i^2 \right]$
  - $\alpha \to \infty$: $H_\infty(X) = -\log_2 \left[\max_i \{ p_i \} \right]$

- PROPERTIES OF $H(X)$ STILL HOLDS FOR $p_i = 1/N$

$\to H(X)$ MAX $\iff p = \frac{1}{N}$ $\to H(X) = \log_2 N$

- **INFORMATION SOURCES:**

THEY ARE DEVICES GENERATING DATA STREAMS

- CHARACTERIZED BY A SEQUENCE OF RANDOM VARIABLES X_n, $n \in \mathbb{Z}$ (n INTEGER)
- A COMPLETE DESCRIPTION OF INFORMATION SOURCES REQUIRES KNOWLEDGE OF ALL POSSIBLE PROBABILITIES $P(X_S = x_S)$ / S = SUBSET OF $\mathbb{Z}$
- $X_S = x_S$ CORRESPONDS TO $X_n = x_n$, $\forall n \in S$

eg.

- $X = \{0, 1, 2\}$, $S = \{-2, 0, 2\}$

→ SAMPLE PROBABILITY: $P(X_S = \{2, 1, 0\}) = P(X_{-2} = 2, X_0 = 1, X_2 = 0)$

- **STATIONARITY OF AN INFORMATION SOURCE:**

FOR A FIXED DETERMINISTIC VECTOR X_S OF $|S|$ VALUES:

→ $P(X_S = x_S) = P(X_{S+\Delta} = x_S)$, $\Delta \in \mathbb{Z}$

→ PMF DOESN'T CHANGE OVER TIME → WE CAN USE PROBABILITY TO MAKE PREDICTIONS

- IF $|S| = 1$: $P(X_n = x) = P(X_0 = x) = P_X(x)$ ~> PMF

- **MARKOV SOURCES:**

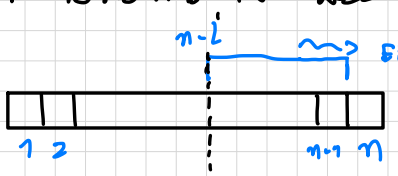
AN INFORMATION SOURCE $X_1, X_2, X_3, \ldots$ IS SAID TO BE A MARKOV SOURCE WITH MEMORY L IF THE FOLLOWING PROPERTY HOLDS FOR EVERY $n > L$:

$$P(X_n = x_n \mid X_{1:n-1} = x_{1:n-1}) = P(X_n = x_n \mid X_{n-L:n-1} = x_{n-L:n-1})$$

(CONDITIONING; $X_{1:n-1}$: PREVIOUS SOURCES BEFORE n; $X_{n-L:n-1}$: LAST L SOURCES BEFORE n)

→ THE ONLY EFFECTIVE PART ARE THE L LAST SYMBOLS

→ X_i



INFORMATION SOURCES

- **TIME INVARIANCE:**

  PROPERTY FOR MARKOV SOURCES, $n > L$:

$$P\left(X_n = x_n \,\middle|\, X_{n-L:n-1} = x_{n-L:n-1}\right) = P\left(X_{L+1} = x_{L+1} \,\middle|\, X_{1:L} = x_{1:L}\right)$$

- STATE OF THE SOURCE AT TIME n:

$$\Sigma_n \overset{\text{SYMBOLS}}{\rightsquigarrow} \triangleq X_{n-L:n-1}$$

- IF THE SYMBOL ALPHABET IS $\mathcal{X}$ $\rightarrow$ STATE ALPHABET: $\mathcal{X}^L$

- TIME-INVARIANT MARKOV SOURCES ARE CHARACTERIZED BY:

  - DISTRIBUTION OF THE INITIAL STATE: $\Sigma_{L+1} = X_{1:L}$

  - CONDITIONAL PROBABILITIES: $P\left(\Sigma_{L+2} = \sigma' \,\middle|\, \Sigma_{L+1} = \sigma\right)$

    $\rightarrow$ σ, σ' ARE STATE VALUES FROM $\mathcal{X}^L$ $\quad \sigma = x_{2:L+1} \quad \sigma' = x_{1:L}$

- $P\left(\Sigma_{L+2} = \sigma' \,\middle|\, \Sigma_{L+1} = \sigma\right)$ ARE USUALLY ARRANGED INTO A TRANSITION PROBABILITY MATRIX P

$$\rightarrow (P)_{\sigma', \sigma} = P\left(\Sigma_{L+2} = \sigma' \,\middle|\, \Sigma_{L+1} = \sigma\right)$$

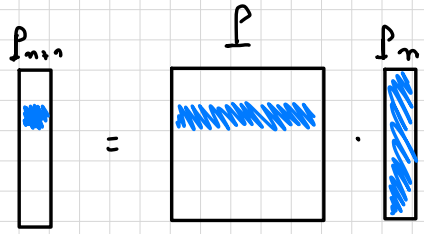
  - THE STATE PROBABILITY VECTOR IS DEFINED AS A COLUMN VECTOR:

$$(p_n)_\sigma = P(\Sigma_n = \sigma)$$

  $\rightarrow$ THE EVOLUTION OF THE STATE PROBABILITY VECTOR IS GOVERNED BY:

  - $p_{L+1} = \alpha_0$
  - $p_{n+1} = P \cdot p_n$ ($P \rightsquigarrow$ MATRIX, $p_n \rightsquigarrow$ COLUMN VECTOR)

→



$$\rightarrow p_{n+1} = P \cdot p_n$$

$$\rightarrow (p_{n+1})_\sigma = \sum_{\sigma'} P_{\sigma,\sigma'} \cdot (p_n)_{\sigma'}$$

$$\rightarrow P(\Sigma_{n+1} = \sigma) = \sum_{\sigma'} P(\Sigma_{n+1} = \sigma \mid \Sigma_n = \sigma') \cdot P(\Sigma_n = \sigma')$$

ES.

- 4 STATES: $1, 2, 3, 4$
- TRANSITION PROBABILITY MATRIX: $P_{i,j} = P(\Sigma_{n+1} = i \mid \Sigma_n = j)$
- IF WE WANT $\Sigma_{n+1} = 1$:

$$\rightarrow P(\Sigma_{n+1} = 1, \Sigma_n = 1) + P(\Sigma_{n+1} = 1, \Sigma_n = 2) +$$

$$P(\Sigma_{n+1} = 1, \Sigma_n = 3) + P(\Sigma_{n+1} = 1, \Sigma_n = 4) = P(\Sigma_{n+1} = 1)$$

$$/ \; P(\Sigma_{n+1} = i, \Sigma_n = j) = P(\Sigma_{n+1} = i \mid \Sigma_n = j) \cdot P(\Sigma_n = j)$$

→

$$\begin{pmatrix} P(\Sigma_{n+1}=1) \\ P(\Sigma_{n+1}=2) \\ P(\Sigma_{n+1}=3) \\ P(\Sigma_{n+1}=4) \end{pmatrix} = \begin{pmatrix} P(\Sigma_{n+1}=1 \mid \Sigma_n=1) & P(\Sigma_{n+1}=1 \mid \Sigma_n=2) & P(\Sigma_{n+1}=1 \mid \Sigma_n=3) & P(\Sigma_{n+1}=1 \mid \Sigma_n=4) \\ P(\Sigma_{n+1}=2 \mid \Sigma_n=1) & P(\Sigma_{n+1}=2 \mid \Sigma_n=2) & P(\Sigma_{n+1}=2 \mid \Sigma_n=3) & P(\Sigma_{n+1}=2 \mid \Sigma_n=4) \\ P(\Sigma_{n+1}=3 \mid \Sigma_n=1) & P(\Sigma_{n+1}=3 \mid \Sigma_n=2) & P(\Sigma_{n+1}=3 \mid \Sigma_n=3) & P(\Sigma_{n+1}=3 \mid \Sigma_n=4) \\ P(\Sigma_{n+1}=4 \mid \Sigma_n=1) & P(\Sigma_{n+1}=4 \mid \Sigma_n=2) & P(\Sigma_{n+1}=4 \mid \Sigma_n=3) & P(\Sigma_{n+1}=4 \mid \Sigma_n=4) \end{pmatrix} \begin{pmatrix} P(\Sigma_n=1) \\ P(\Sigma_n=2) \\ P(\Sigma_n=3) \\ P(\Sigma_n=4) \end{pmatrix}$$

$$\boldsymbol{p}_{n+1} = \boldsymbol{P} \cdot \boldsymbol{p}_n$$

- # MARKOV CHAINS:

THEY ARE THE EVOLUTION OF PROBABILITY VECTORS OVER TIME

- A MARKOV CHAIN NEEDS TO SATISFY 2 PROPERTY:
  - **IRREDUCIBILITY**: POSSIBILITY TO GO WITH $p_i > 0$ FROM ANY STATE TO AN OTHER IN n FINITE STEPS
    $\rightarrow P^n$ HAS ONLY POSITIVE ELEMENTS, FOR $n >> 0$
  - **APERIODICITY**: STARTING FROM STATE i, WE DON'T KNOW WHEN WE WILL RETURN TO STATE i AFTER SOME TRANSITIONS

- IF A FINITE-STATE MARKOV CHAIN IS IRREDUCIBLE AND APERIODIC.

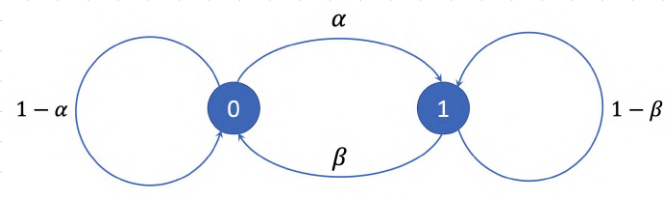
$\rightarrow \exists!\ p_\infty$ (STATIONARY STATE PROBABILITY DISTRIBUTION) $/\ p_\infty = \lim_{n\to\infty} p_n = \lim_{n\to\infty} P^{n \cdot L \cdot 1}\, \alpha_0\ ,\ \forall\, \alpha_0$

$\rightarrow p_\infty = \begin{pmatrix} p_0 \\ \vdots \\ p_n \end{pmatrix} / P_\infty = P \cdot p_\infty$

es.

- 2 STATES: 0, 1
- MATRIX: $P = \begin{bmatrix} 1-\alpha & \beta \\ \alpha & 1-\beta \end{bmatrix}, \quad \alpha, \beta \in [0,1]$

$\rightarrow$ GRAPH:



- $p_\infty = \begin{pmatrix} p_0 \\ p_1 \end{pmatrix}$ CAN BE OBTAINED BY:

$\rightarrow \begin{pmatrix} p_0 \\ p_1 \end{pmatrix} = \begin{pmatrix} 1-\alpha & \beta \\ \alpha & 1-\beta \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \end{pmatrix}$

->

$$\begin{cases} p_0 = (1-\alpha)p_0 + \beta p_1 \\ p_1 = \alpha p_0 + (1-\beta)p_1 \quad (\rightsquigarrow \text{REDUNDANT}) \\ p_0 + p_1 = 1 \end{cases}$$

-> THE SOLUTION IS : $p_0 = \frac{\beta}{\alpha+\beta}$, $p_1 = \frac{\alpha}{\alpha+\beta}$

ex. 2

- BINARY SOURCE : STATES 0, 1
- $L = 2$
-

$$\begin{cases} P(X_n = 0 \mid X_{n-1} + X_{n-2} = 0) = p_0 \\ P(X_n = 0 \mid X_{n-1} + X_{n-2} = 1) = p_1 \\ P(X_n = 0 \mid X_{n-1} + X_{n-2} = 2) = p_2 \end{cases}$$

->

- WE NEED TO FIND THE MATRIX P
- WE DEFINE:
  - STARTING STATE : X_{n-2}, X_{n-1}
  - ENDING STATE : X_{n-1}, X_n

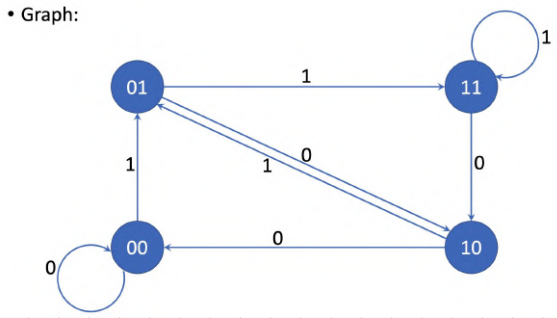
-> STATE EVOLUTION SUMMARY:

Starting state	Current symbol	Ending state	Probability
X_{n-2}, X_{n-1}	X_n	X_{n-1}, X_n	
00	0	00	p_0
00	1	01	$1-p_0$
01	0	10	p_1
01	1	11	$1-p_1$
10	0	00	p_1
10	1	01	$1-p_1$
11	0	10	p_2
11	1	11	$1-p_2$

- GRAPH:

Starting state	Current symbol	Ending state	Probability
X_{n-2}, X_{n-1}	X_n	X_{n-1}, X_n	
00	0	00	p_0
00	1	01	$1-p_0$
01	0	10	p_1
01	1	11	$1-p_1$
10	0	00	p_1
10	1	01	$1-p_1$
11	0	10	p_2
11	1	11	$1-p_2$

⟼



- MATRIX:

ENDING STATE

$$\boldsymbol{P} = \begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{pmatrix} p_0 & 0 & p_1 & 0 \\ 1-p_0 & 0 & 1-p_1 & 0 \\ 0 & p_1 & 0 & p_2 \\ 0 & 1-p_1 & 0 & 1-p_2 \end{pmatrix} \end{array} \rightsquigarrow \text{STARTING STATE}$$

↝ n EIGENV $\neq 0$: n° LINES IN GRAPH

- p_∞ : $\begin{pmatrix} p_{00} \\ p_{01} \\ p_{10} \\ p_{11} \end{pmatrix}$

↝

$$\begin{aligned} p_{00} &= p_0 p_{00} + p_1 p_{10} \\ p_{01} &= (1-p_0) p_{00} + (1-p_1) p_{10} \\ p_{10} &= p_1 p_{01} + p_2 p_{11} \\ 1 &= p_{00} + p_{01} + p_{10} + p_{11} \end{aligned}$$

SOLUTION ⟼

$$\begin{aligned} p_{00} &= \alpha p_1 p_2 \\ p_{01} &= p_{10} = \alpha(1-p_0)p_2 \\ p_{11} &= \alpha(1-p_0)(1-p_1) \\ \alpha &= \frac{1}{p_1 p_2 + 2(1-p_0)p_2 + (1-p_0)(1-p_1)} \end{aligned}$$

- IN MATLAB / PYTHON:

N = n. of states

$P \cdot \underline{p}_\infty = \underline{p}_\infty \qquad \underline{1}^T \underline{p}_\infty = 1$

$(P - I_N) \cdot \underline{p}_\infty = \underline{0} \rightarrow p_\infty = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$, ∵ rank() is max

$(P - I_N)_{1:N-1} \rightarrow$ first (N-1) rows of $(P - I_N)$

$\begin{pmatrix} (P - I_N)_{1:N-1} \\ \underline{1}^T \end{pmatrix} \underline{p}_\infty = \begin{pmatrix} \underline{0}_{N-1} \\ 1 \end{pmatrix}$ ↝ LAST ROWS OF $(P - I_N)$ REPLACED WITH ONES.

In matlab:

```
p_inf = [P(1:N-1,:) - eye(N-1,N); ...
         ones(1,N)] \ ...
        [zeros(N-1,1); 1];
```

- **ENTROPY RATE OF AN INFORMATION SOURCE:**

  - ENTROPY RATE: $\bar{H} \triangleq \lim_{n \to \infty} \frac{H(X_{1:n})}{n}$

  -> IF AN INFORMATION SOURCE IS STATIONARY: $\bar{H} = H(X)$

  -> FOR A STATIONARY SOURCE: $\bar{H} = \lim_{n \to \infty} H(X_n | X_{1:n-1})$

  - FOR A MARKOVIAN SOURCE:

$$\bar{H} = H(X|\Sigma) = -\sum_{\sigma \in X^L} P_\infty(\sigma) \left(\sum p(x|\sigma) \log_2 p(x|\sigma) \right) =$$

P_∞ : COLUMN VECTOR / INDEX: SYMBOL σ

$$= \sum_\sigma \underbrace{P(\Sigma = \sigma)}_{P_\infty} \underbrace{H(X|\Sigma=\sigma)}_{\text{COLUMN VECTOR / INDEX: SYMBOL } \sigma}$$

$$/ \; H(X|\Sigma=\sigma) = -\sum_x P(X=x|\Sigma=\sigma) \log_2 \left(P(X=x|\Sigma=\sigma) \right)$$

* ES. 1

$$\bar{H} = \underbrace{\frac{\beta}{\alpha+\beta}}_{P_0} H(\bar{P}_0) + \underbrace{\frac{\alpha}{\alpha+\beta}}_{P_1} H(\bar{P}_1)$$

* ES. 2

COLUMN 0 OF P: $\begin{bmatrix} p_0 \\ 1-p_0 \\ \cdot \\ 0 \end{bmatrix}$ $\xrightarrow{\hat{2}}$ $\begin{bmatrix} 0 \\ 0 \\ p_1 \\ 1-p_1 \end{bmatrix}$ $\xrightarrow{\hat{2}}$

$$\bar{H} = p_{00} \cdot H(\bar{P}_{[:,0]}) + p_{01} \cdot H(\bar{P}_{[:,1]}) +$$

$$+ p_{10} H(\bar{P}_{[:,2]}) + p_{11} \cdot H(\bar{P}_{[:,3]})$$

- **SOURCE CODING:**
  - GOAL: REDUCING AMOUNT OF DATA NECESSARY TO STORE LARGE SET OF SYMBOLS
    → DATA COMPRESSION
  - IT MUST BE AN INVERTIBLE OPERATION: ORIGINAL DATA $\Longleftrightarrow$ ENCODED DATA
  - 3 TYPES OF SOURCE CODING ALGORITHM: FIXED-TO-FIXED, FIXED TO VARIABLE, VARIABLE-TO-FIXED
  - **FIXED TO FIXED:**
    - BLOCK OF N SOURCE SYMBOLS FROM ALPHABET X, CARDINALITY $M = |X|$
    - M^N POSSIBLE BLOCKS: 1 BLOCK UNIQUELY IDENTIFIED BY A NUMBER IN $(0, M^N - 1)$
      - IF $M = 2$: n bits SUFFICIENT TO REPRESENT INTEGERS IN $(0, 2^n - 1)$

        $\rightarrow 2^n \geq M^N \rightarrow n \geq N \log_2 M \;/\; v = \lceil N \log_2 M \rceil$

        → MIN NUM. $\in \mathbb{Z}$ DI BIT
    - BITS PER ENCODED SYMBOL: $\bar{n} = \frac{v}{N} = \frac{\lceil N \log_2 M \rceil}{N} \simeq \log_2 M$
    - ENCODING IS EASY: $\forall$ SYMBOL → v BITS
    - DECODING IS EASY: $\forall$ v BITS → $\exists!$ SYMBOL
  - **FIXED TO VARIABLE:**
    - ADVANTAGEOUS IF SOURCE SYMBOLS HAVE DIFFERENT PROBABILITIES
      - $\uparrow p$ → SHORT ENCODING
      - $\downarrow p$ → LONG ENCODING
    - ASSUME SYMBOLS GENERATED BY STATIONARY SOURCE, FROM ALPHABET X
    - $p_i = P(X = \mathcal{E}_i), \quad i = 0, 1, \dots, (M = |X|)$
    - AVERAGE BITS PER ENCODED SYMBOL: $\langle v \rangle = \frac{1}{N} \sum_{i=1}^{M} N_i v_i$

      → IT'S A WEIGHTED AVERAGE
      - v_i: CODEWORD LENGTH OF SYMBOL $\mathcal{E}_i$
      - N: n° OF SYMBOLS IN STRING
      - N_i: n° OF OCCURRENCES OF $\mathcal{E}_i$

- FOR A VERY LONG STRING: $\frac{N_i}{N} \simeq p_i$

$\rightarrow \bar{V} = \sum_{i=1}^{M} p_i V_i$

EX.

- CONSIDER $\frac{N_i}{N} \simeq p_i$
- SOURCE STRING: $\varepsilon_1\ \varepsilon_1\ \varepsilon_2\ \varepsilon_1\ \varepsilon_1\ \varepsilon_3\ \varepsilon_2\ \varepsilon_1 \rightarrow \begin{cases} N_1 = 5 \\ N_2 = 2 \\ N_3 = 1 \end{cases}$

$N = 8$

$\rightarrow$

LET: $\varepsilon_i \rightarrow V_i$ BITS

- TOT n° OF BITS AFTER ENCODING:

$\bar{V} = \sum_{i=1}^{M} N_i V_i = 5 V_1 + 2 V_2 + 1 V_3$

- IF $\downarrow \bar{V} \rightarrow$ BETTER SOURCE CODING

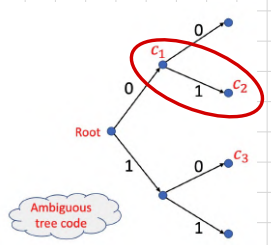
$\rightarrow$ WE WANT TO MINIMIZE $\bar{V}$

- DECODING IS NOT EASY: WE DON'T KNOW THE LENGTH OF EACH SYMBOL
  - $\rightarrow$ NO KNOWN POSITION OF SEPARATION $\rightarrow$ DECODING AMBIGUITY

- DECODING AMBIGUITY

IT IS SOLVED BY TREE STRUCTURES: IF A CODEWORD IS IN THE PATH OF AN OTHER CODEWORD $\rightarrow$ AMBIGUITY

$\rightarrow$ THEY NEED TO SATISFY THE PREFIX-FREE CONDITION



- KRAFT INEQUALITY:

- A NODE AT DEPTH V_i CONTROLS $2^{V_{max} - V_i}$ NODES

$\rightarrow$ WE DEFINE THE INEQUALITY: $\sum_{i=1}^{M} 2^{V_{max} - V_i} \leq 2^{V_{max}}$

$\rightarrow$ DIVIDING BOTH SIDES BY $2^{V_{max}} \rightarrow$ KRAF INEQUALITY: $\sum_{i=1}^{M} 2^{-V_i} \leq 1$

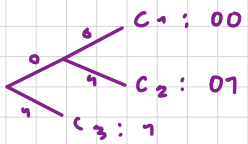
- IT CAN BE PROVED THAT: (~> PROOF SKIPPED: MINIMIZATION USING LAGRANGE MULTIPLIERS, WITH KRAFT'S INEQUALITY AS CONSTRAINT)
  - LOWER BOUND TO $\bar{V}$: $H(X)$
    -> $\bar{V} \geq H(X)$
  - UPPER BOUND TO $\bar{V}$: $H(X) + 1$
    -> $\bar{V} \leq H(X) + 1$
- SHANNON'S THEOREM: ~> IT IS THE LINK BETWEEN SOURCE CODING AND ENTROPY

$\exists$ DECODABLE SOURCE CODE / $H(X) \leq \bar{V} \leq H(X) + 1$

  - LOWER BOUND: NECESSARY FOR $\exists$ OF THE CODE
  - UPPER BOUND: SUFFICIENT

ex.

- $\bar{p} = \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{2}\right)$ ->



-> $\bar{V} = (2, 2, 1)$ -> KRAFT: $2^{-2} + 2^{-2} + 2^{-1} = 1 \leq 1$ ✓

- CROSS ENTROPY:

LET p_i BE THE TRUE PROBABILITY OF A DISTRIBUTION, q_i ITS ESTIMATION

- OPTIMUM SOURCE CODE ON q_i HAS $V_i = -\log_2 q_i$
- $\bar{V} = H(p, q) = -\sum_i p_i \log_2 q_i$

-> $H(p, q) \geq H(p, p)$: THE MISMATCH BETWEEN p AND q INCREASES $\bar{V}$

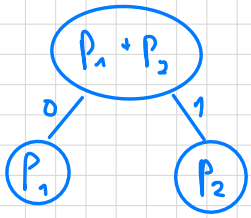
- KULLBACK-LEIBLER DIVERGENCE: $D(p \| q) = H(p, q) - H(p, p) = \sum_i p_i \log_2 \left(\frac{p_i}{q_i}\right)$
- FOR MARKOVIAN SOURCES: $H(X)\big|_{p = p(x|s)} \leq \bar{V}_s \leq H(X)\big|_{p = p(x|s)} + 1$

- **HUFFMAN CODES:**

PREFIX-FREE TREE CODES THAT MINIMIZE $\bar{V}$ -> OPTIMAL SOURCE CODES

- ALGORITHM:

1. SELECT THE 2 NODES WITH MINIMUM PROBABILITY p_1 e p_2
2. BUILD A SUB-TREE: LEAVES: 2 MINIMUM NODES CHOSEN; ARCS: 0/1; NEW NODE: SUM OF THE 2 p



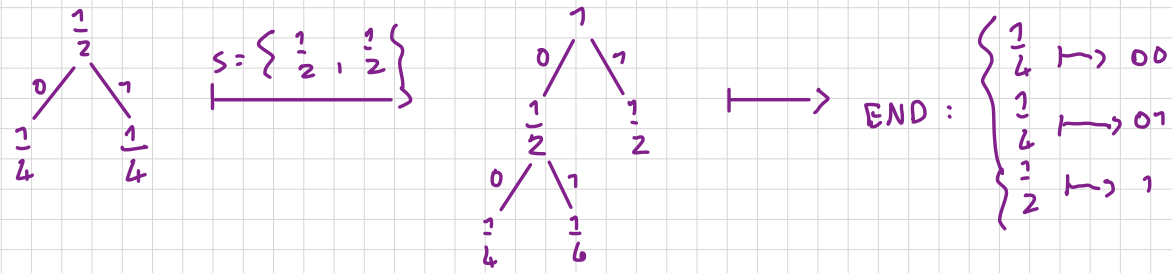
3. INSERT THE NEW PROBABILITY $p_1 + p_2$ AMONG THE SET OF SELECTABLE p
4. RESTART FROM 1.

-> TERMINATION: NO MORE p SELECTABLE

ex.

$$\bar{p} = \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{2}\right), \quad S = \{\text{SET OF SELECTABLE } p\} = \left\{\frac{1}{4}, \frac{1}{4}, \frac{1}{2}\right\}$$

->



- $\bar{V} = \frac{1}{4} \cdot V_1 + \frac{1}{4} \cdot V_2 + \frac{1}{2} \cdot V_3 = \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 + \frac{1}{2} \cdot 1 = 1.5$
- $H(X) = -2 \cdot \frac{1}{4} \log_2 \frac{1}{4} - \frac{1}{2} \log_2 \frac{1}{2} = 1 + \frac{1}{2} = 1.5$

-> $H(X) \leq \bar{V}$ ✓ -> $H(X) = \bar{V}$ -> IT'S OPTIMAL

- **STORAGE OF THE COMPRESSED DATA:**

REQUIRES 2 TYPES OF DATA:

- CODE TREE WITH SYMBOLS AND THEIR BIT ENCODING
- SEQUENCE OF ENCODED BITS

ex.

- Consider the source sequence "ABCCCCBABABACCCC" (16 symbols)
- Its encoding is "000111110100010001001111" (24 bits)
- The stored data are:
  - The tree: 00,A,01,B,1,C
  - The encoded bits: 000111110100010001001111
- Usually, the storage of the tree code is negligible with respect to the storage of the encoded bits
- In this example, the actual number of bits per symbol required is $\bar{v} = \frac{24}{16} = 1.5$
- The source entropy is $\bar{H} = H(0.25, 0.25, 0.5) = 1.5 = \bar{v}$

/ A : 00 , $p_A = 0.25$
B : 01 , $p_B = 0.25$
C : 1 , $p_C = 0.5$

- **CALCULATING $\bar{V}$ FOR HUFFMAN CODES:**

- IT CAN BE CALCULATED RECURSIVELY:

$$\bar{V} = \phi(\bar{p}) \quad / \quad \phi(p_1, p_2, p_3, \dots, p_n) = p_1 + p_2 + \phi(p_1 + p_2, p_3, \dots, p_n)$$

$p_1 \leq p_2 \leq p_3 \leq \dots \leq p_n$

ex.

$$\bar{p} = (\overset{1}{0.1}, \overset{2}{0.1}, \overset{3}{0.2}, \overset{4}{0.2}, \overset{5}{0.4})$$

-> 0000 0.1 0 / 0001 0.1 1 -> 0.2 / 001 0.2 0 / 01 0.2 1 -> 0.4 / 1 0.4 1 -> 0.6 / 1 -> 1

->

$$\phi(\bar{p}) = \overset{2}{\boxed{2 \cdot 0.1}} + \phi_2(2 \cdot 0.1 + 0.2 + 0.2 + 0.4) =$$

$$= 0.2 + \overset{3}{\boxed{2 \cdot 0.2}} + \phi_3(0.2 + 2 \cdot 0.2 + 0.4) =$$

$$= 0.2 + 0.4 + \overset{3}{\boxed{0.2 + 0.4}} + \phi(0.6 + 0.6) = 2.2$$

- $\bar{v} = 0.1 \cdot 4 + 0.1 \cdot 4 + 0.2 \cdot 3 + 0.2 \cdot 2 + 0.4 \cdot 1 = 2.2$ (?)
- $H(X) \simeq -2 \cdot 0.1 \log_2 0.1 - 2 \cdot 0.2 \log_2 0.2 - 0.4 \log_2 0.4 = 2.12$

- DICTIONARY METHODS:

- COMPLEMENTARY METHOD TO FIXED-TO-VARIABLE

- VARIABLE LENGTH SOURCE STRINGS ENCODED INTO FIXED LENGTH STRINGS
  - SET OF THESE: DICTIONARY

-> { FIXED-TO-VARIABLE: STATISTICAL METHODS -> QUALITY OF COMPRESSION (·) GOODNESS OF MODEL
   { DICTIONARY: STATIC OR DYNAMIC

- DICTIONARY METHODS: SELECT STRINGS OF SYMBOLS AND ENCODE EACH STRING AS A TOKEN USING A DICTIONARY

- STATIC DICTIONARY: EX. ENGLISH LANGUAGE DICTIONARY

->
  IF MATCH FOUND IN DICTIONARY:
    -> INDEX TO THE DICTIONARY WRITTEN IN OUTPUT STREAM
  ELSE:
    -> UNCOMPRESSED WORD IS WRITTEN

- COMPRESSED AND UNCOMPRESSED DATA CAN BE DISTINGUISHED BY EXTRA BITS AT THE BEGINNING

- ENGLISH DICTIONARY METHOD:

- ≈ 500 000 WORDS -> $2^{19} = 524\,288 > 500\,000$ -> 19 BITS NECESSARY

->
  - IF WORD IN DICTIONARY:
    -> 1 EXTRA BIT + 19 BITS = 20 BITS
  ELSE: $0 \div 255 = 2^8 - 1$
    -> 1 EXTRA BIT + 8 BITS DEFINING Nº OF CHARS + CHAR ENCODING

- IMPROVEMENT -> 2 EXTRA BITS:
  - 00. WORD IN FIRST $1024 = 2^{10}$ MOST FREQUENT WORDS -> 2 + 10 BITS
  - 01. WORD IN FIRST $16384 = 2^{14}$ MOST FREQUENT WORDS -> 2 + 14 BITS
  - 10: REMAINING WORDS -> 2 + 19 BITS
  - 11: WORD NOT IN DICTIONARY -> VARIABLE Nº OF BITS

- EFFICIENCY (·) GOODNESS OF STAT. MODELS, LANGUAGE

- ADAPTIVE METHODS:
  - STARTS WITH A STATIC DEFAULT DICTIONARY
  - ADDS WORDS WHEN FOUND FROM SOURCE
  - DELETES OLD UNUSED WORDS → KEEP SIZE STABLE, SEARCH TIME SHORT

- STRING COMPRESSION:
  - SOURCE CODES BASED ON STRING COMPRESSION ARE MORE EFFICIENT THAN THOSE BASED ON SYMBOLS

→ APPLYING HUFFMAN CODING TO PROGRESSIVELY LONGER STRINGS, WITH LENGTH m:

→

m	$\frac{\bar{v}}{m}$
1	1
2	0.9050
3	0.9087
4	0.8918
5	0.8890
6	0.8882

- $\frac{\bar{v}}{m}(m)$ CONVERGES TO

$$H_b(0.7) = 0.8813$$

↳ ENTROPY OF A BINARY R.V

$P(X) = \{0.7, 0.3\}$

  - SLOW CONVERGENCE ! BECAUSE m IS FIXED
  - n^o POSSIBLE STRINGS 2^m GROWS W.J m:

- LEMPLEV-ZIV CODES: (NO NEED TO KNOW DETAILED EXPLANATION)
  - DICTIONARY METHODS IN 1970, FIRST ALGORITM: LZ 77
  - LZ77 ALGORITM:

- The principle of LZ77 is to fill the dictionary with parts of the input stream as it is read off
- The method is based on a sliding window divided in two parts:
  - Search buffer
  - Look-ahead buffer
- The search buffer contains already encoded symbols
- The look-ahead buffer contains symbols to be encoded
- In practical implementations, the search buffer is long a few thousands symbols and the look-ahead buffer a few tens
- The encoded output consists of tokens represented by three components:
  - Offset = distance of the first encoded symbol to the end of the search buffer
  - Length = length of the encoded string
  - Next symbol = first symbol after the encoded string
- The two buffers are implemented as sliding windows
- Initially,
  - the search buffer is empty
  - the symbols to be encoded fill the look-ahead buffer
- Repeat until the look-ahead buffer is empty:
  - Scan the search buffer to find the longest string matching the first symbols in the look-ahead buffer
  - If nothing is found
    - Output token <0,0,first symbol in look-ahead buffer>
  - else
    - Output token <offset, length, next symbol after encoded string in look-ahead buffer>
  - Move the buffer boundary to the right up to the next symbol to encode

eg.

LZ77 example: "cat cat catering"

- Assume buffer length is 8
- The $ sign is used to terminate the string to encode

Search	Look-ahead	Token
"	"cat cat "	<0,0,"c">
" c"	"at cat c"	<0,0,"a">
" ca"	"t cat ca"	<0,0,"t">
" cat"	" cat cat"	<0,0," ">
" cat "	"cat cate"	<4,4,"c">
"at cat c"	"atering$"	<4,2,"e">
"cat cate"	"ring$	" <0,0,"r">
"at cater"	"ing$	" <0,0,"i">
"t cateri"	"ng$	" <0,0,"n">
" caterin"	"g$	" <0,0,"g">
"catering"	"$	" <0,0,"$">

eg.

Another example: "sir sid eastman easily teases sea sick seals"

" --- "sir sid " --- <0,0,"s">	"an easil" --- "y teases" --- <0,0,"y">	
" s" --- "ir sid e" --- <0,0,"i">	"n easily" --- " teases " --- <7,1,"t">	
" si" --- "r sid ea" --- <0,0,"r">	"easily t" --- "eases se" --- <8,3,"e">	
" sir" --- " sid eas" --- <0,0," ">	"ly tease" --- "s sea si" --- <2,1," ">	
" sir " --- "sid east" --- <4,2,"d">	" teases " --- "sea sick" --- <4,2,"a">	
" sir sid" --- " eastman" --- <4,1,"e">	"ases sea" --- " sick se" --- <4,2,"i">	
"ir sid e" --- "astman e" --- <0,0,"a">	"s sea si" --- "ck seals" --- <0,0,"c">	
"r sid ea" --- "stman ea" --- <6,1,"t">	" sea sic" --- "k seals$" --- <0,0,"k">	
"sid east" --- "man easi" --- <0,0,"m">	"sea sick" --- " seals$ " --- <5,2,"e">	
"id eastm" --- "an easil" --- <4,1,"n">	" sick se" --- "als$ " --- <0,0,"a">	
" eastman" --- " easily " --- <8,4,"i">	"sick sea" --- "ls$ " --- <0,0,"l">	
"man easi" --- "ly tease" --- <0,0,"l">	"ick seal" --- "s$ " --- <4,1,"$">	

- IN LATER YEARS: * ZIP METHODS, 1ST: PKZIP, 1987, KATZ → CORE METHOD: DEFLATE
  → DEFLATE: MODIFIED LZ77 + HUFFMAN CODES

• LOSSY COMPRESSION:

- IF $n \in \mathbb{R}$ $\rightarrow$ ∞ NUMBER OF BITS TO REPRESENT IT

$\rightarrow$ WE CAN OBTAIN AN APPROXIMATE DESCRIPTION USING A FINITE NUMBER OF BITS

$\rightarrow$ IT PRODUCES AN ERROR CALLED DISTORTION

- THE RATE DISTORTION THEORY (RDT) DESCRIBES THE RELATIONSHIP BETWEEN THE RATE $\frac{\#bits}{symbol}$ AND AVERAGE DISTORTION

- QUANTIZATION:

CONSIDER A R.V. $X \sim N(0,1)$ AND $\hat{X}$ ITS QUANTIZED REPRESENTATION OVER $R = 1$ BIT

$$\rightarrow \hat{X} = \begin{cases} a, & X \geq 0 \\ -a, & X < 0 \end{cases}$$

- MEAN SQUARE ERROR AS DISTORTION MEASURE: $D = E[(X - \hat{X})^2]$

$\rightarrow$ IF $a = \sqrt{\frac{2}{\pi}} \approx 0.79$ $\rightarrow$ D IS MINIMUM: $D = 1 - \frac{2}{\pi} = 0.3634$

- IF $R > 1$: CALCULATION IS DIFFICULT

$\rightarrow$ ∃ LLOYD ALGORITHM

LLOYD ALGORITHM :

- CONSIDER :
  - X R.V. / $(N-1)$ THRESHOLDS $t = (t_1, \dots, t_{N-1})$, N QUANTIZED VALUES $x = (x_1, \dots, x_N)$

$$\rightarrow D(t, x) = \sum_{i=1}^{N} \int_{t_{i-1}}^{t_i} (x - x_i)^2 f_X(x)\, dx \quad / \quad t_0 \triangleq -\infty,\ t_N \triangleq \infty$$

- A NECESSARY CONDITION FOR THE MINIMUM IS THAT $\frac{\partial D}{\partial t_i} = 0$:
  - $$\frac{\partial D(t,x)}{\partial t_i} = \left[(t_i - x_i)^2 - (t_i - x_{i+1})^2 \right] f_X(x) = 0$$

$$\rightarrow (t_i - x_i)^2 - (t_i - x_{i+1})^2 = (2t_i - x_i - x_{i+1})(x_i - x_{i+1}) = 0$$

$$\rightarrow \text{WE GET : } t_i = \frac{x_i + x_{i+1}}{2}, \quad i = 1, \dots, N$$

- THEN WE CONSIDER $\frac{\partial D}{\partial x_i} = 0$:
  - $$\frac{\partial D(t,x)}{\partial x_i} = \int_{t_{i-1}}^{t_i} 2(x_i - x) f_X(x)\, dx = 0$$

$$\rightarrow \text{WE GET : } x_i = \frac{\int_{t_{i-1}}^{t_i} x f_X(x)\, dx}{\int_{t_{i-1}}^{t_i} f_X(x)\, dx} = E[X \mid X \in (t_{i-1}, t_i)] \quad (i = 1, \dots, N)$$

- ALGORITHM :
  1. INITIALIZE THE QUANTIZED VALUES : $x^{(0)} = x_0$
  2. REPEAT FOR $n = 0, 1, 2, \dots$ ~> n^0 OF STEPS FOR ALGORITHM

$$t_i^{(n+1)} = \frac{x_i^{(n)} + x_{i+1}^{(n)}}{2} \quad (i = 1, \dots, N-1)$$

$$x_i^{(n+1)} = E[X \mid X \in (t_{i-1}^{(n+1)}, t_i^{(n+1)})] \quad (i = 1, \dots, N)$$

  3. REPEAT UNTIL $\| x^{(n+1)} - x^{(n)} \|^2 < \text{THRESHOLD}$

es.

- $X \sim N(0,1) \;/\; f_X(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$
- $R = 2$ (4 LEVELS), BY SIMMETRY: $t = (-b, 0, t)$, $x = (-x_2, -x_1, x_1, x_2)$
  - $t^{(m+1)} = \frac{x_1^{(m)} + x_2^{(m)}}{2}$
  - $x_1^{(m+1)} = \frac{\int_{t^{(m+1)}}^{\infty} x \cdot \exp\left(-\frac{x^2}{2}\right) dx}{\sqrt{2\pi}\left(\frac{1}{2} - Q\left(t^{(m+1)}\right)\right)}$
  - $x_2^{(m+1)} = \frac{\exp\left(-\frac{[t^{(m+1)}]^2}{2}\right)}{\sqrt{2\pi} \cdot Q\left(t^{(m+1)}\right)}$

$Q(x) = P(N(0,1) > x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}} du$

- DISTORTION:

$$D = 2\int_0^t (x - x_1)^2 \left[\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}\right] dx + 2\int_0^{\infty} (x - x_2)^2 \left[\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}\right] dx$$

$$= 1 - 2\sqrt{\frac{2}{\pi}}\, x_1 + x_1^2 - 2\sqrt{\frac{2}{\pi}}\, e^{-\frac{t^2}{2}}(x_2 - x_1) + 2\left(x_2^2 - x_1^2\right) Q(x)$$

Iter.	x_1	t	x_2	D	Iter.	x_1	t	x_2	D
0	1.0000	1.5000	2.0000						
1	0.6220	1.5000	1.9387	0.1627	11	0.4538	0.9842	1.5125	0.1175
2	0.5582	1.2803	1.7540	0.1342	12	0.4534	0.9831	1.5117	0.1175
3	0.5169	1.1561	1.6515	0.1235	13	0.4531	0.9825	1.5112	0.1175
4	0.4913	1.0842	1.5930	0.1196	14	0.4530	0.9821	1.5109	0.1175
5	0.4758	1.0422	1.5590	0.1182	15	0.4529	0.9819	1.5107	0.1175
6	0.4665	1.0174	1.5391	0.1178	16	0.4529	0.9818	1.5106	0.1175
7	0.4609	1.0028	1.5274	0.1176	17	0.4528	0.9817	1.5105	0.1175
8	0.4576	0.9942	1.5205	0.1175	18	0.4528	0.9817	1.5105	0.1175
9	0.4557	0.9890	1.5164	0.1175	19	0.4528	0.9816	1.5104	0.1175
10	0.4545	0.9860	1.5139	0.1175	20	0.4528	0.9816	1.5104	0.1175

- **LBG ALGORITHM:**

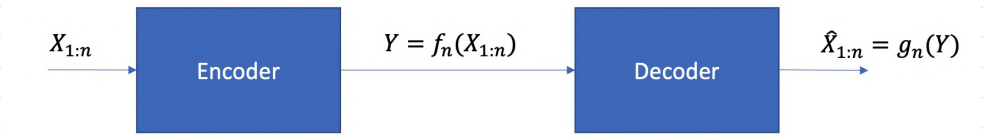
MULTIDIMENSIONAL GENERALIZATION OF LLOYD'S ALGORITHM

- REGIONS R_i, $\hat{x}_i = E[x \mid x \in R_i]$

- **RATE DISTORTION FUNCTION:**

CONSIDERING THE SOURCE ENCODING / DECODING PROBLEM WITH R.V. $X_{1:n}$ VARIABLES ARE REPRESENTED BY nR BITS / $R = \text{bits}/\text{symbol}$

- SEQUENCE $X_{1:n}$ IS ENCODED BY INDEXES IN $[1, 2^{nR}]$
- WE ASSUME $p(x)$ IS THE PROB. DISTRIBUTION AND $X_i \in \mathcal{X} = \{\varepsilon_1, \dots, \varepsilon_M\}$



- DISTORTION FUNCTION: $0 \leq d(X_{1:n}, \hat{X}_{1:n}) \leq \infty$
- HAMMING DISTORTION FUNCTION: $d_H(x, \hat{x}) = \begin{cases} 1 & , x \neq \hat{x} \\ 0 & , x = \hat{x} \end{cases}$
- SQUARED ERROR DISTORTION FUNCTION: $d_s(x, \hat{x}) = (x - \hat{x})^2$
- AVERAGE HAMMING DISTORTION: $E[d_H(X, \hat{X})] = P(X \neq \hat{X})$
- SEQUENCE DISTORTION: $d(x_{1:n}, \hat{x}_{1:n}) = \frac{1}{n} \sum_{i=1}^{n} d(x_i, \hat{x}_i)$
- ENCODING FUNCTION: $f_n : X^n \mapsto \{1 : 2^{nR}\}$
- DECODING FUNCTION $g_n : \{1 : 2^{nR}\} \mapsto X^n$

- SOURCE CODE DISTORTION:

$$D_n = E\left[d\left(X_{1:n}, \underbrace{g_n}_{\text{DECODING}}\left(\underbrace{f_n}_{\text{ENCODING}}\left(X_{1:n}\right)\right)\right)\right]$$

- THE RATE DISTORTION PAIR (R, D) ACHIEVABLE $\Leftrightarrow \exists\, f_n, g_n \;/\; \lim_{n\to\infty} D_n \leq D$

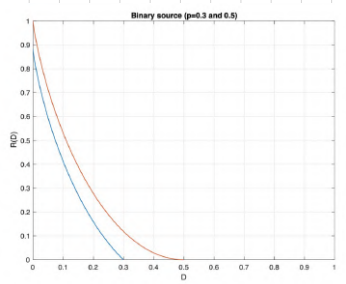
  → $R(D)$: RATE-DISTORTION FUNCTION ⇝ GIVES MINIMUM R TO OBTAIN D

  → $D(R)$: DISTORTION-RATE FUNCTION

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

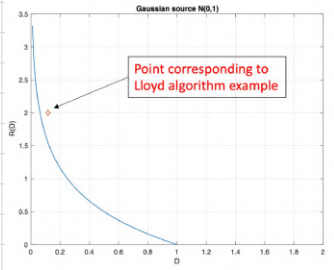
- $R(D)$ OF BINARY SOURCE, WITH HAMMING DISTORTION, , SYMBOLS GENERATED WITH $p < \frac{1}{2}$:

  → $R(D) = \max\{0,\ H_b(p) - H_b(D)\}$



- $R(D)$ OF GAUSSIAN SOURCE $N(0, \sigma^2)$, , WITH SQUARED ERROR DISTORTION:

  → $R(D) = \max\left\{0,\ \frac{1}{2}\log_2\left(\frac{\sigma^2}{D}\right)\right\}$



- INTERPRETATION OF RESULTS:

  CONSIDER A STATIONARY BINARY SOURCE WITH $p = 0.3$

  - IF WE WANT LOSSLESS COMPRESSION: $R(D=0) = H_b(0.3) = 0.8873$
  - IF $D = 0.1$ → $R(0.1) = \underbrace{H_b(0.3)}_{0.8813} - \underbrace{H_b(0.1)}_{0.4690} = 0.4123$

  → REDUCTION IN n° OF COMPRESSED BITS UP TO: $\frac{0.4690}{0.8873} = 53.22\,\%$

- BINARY SOURCE: $\left(\rightsquigarrow p < \frac{?}{2}\right)$

- $R(D) = \min_{P(X \neq \hat{X}) \leq D} \{ I(X;\hat{X}) \}$

- IF SYMBOLS ARE 0,1 $\rightarrow P(X \neq \hat{X}) = E[X \oplus \hat{X}]$:

$$\rightarrow I(X;\hat{X}) = H(X) - H(X|\hat{X}) \overset{\text{CONDITION CLAUSE}}{=} H_b(p) - H(X \oplus \hat{X} | \hat{X}) =$$

$$= H_b(p) - H(X \oplus \hat{X} | \hat{X}) \overset{\text{CONDITIONING REDUCES ENTROPY}}{\geq} H_b(p) - H_b(X \oplus \hat{X}) \geq$$

$$\geq H_b(p) - H_b(X \oplus \hat{X}) \geq H_b(p) - H_b(D)$$

$\downarrow$ $P(X \oplus \hat{X} = 1) \leq D$

- WE MUST SHOW THAT $\exists$ JOINT DISTR. OF X AND $\hat{X}$ / $I(X;\hat{X}) = H_b(p) - H_b(D)$ IF $p < D < \frac{?}{2}$

  - WE ASSUME:

    - $P(X=0) = \frac{1-p-D}{1-2D}$, $P(X=1) = \frac{p-D}{1-2D}$

    - $\hat{X}$ IS THE OUTPUT OF A BINARY SYMMETRIC CHANNEL WITH ERROR PROBABILITY D

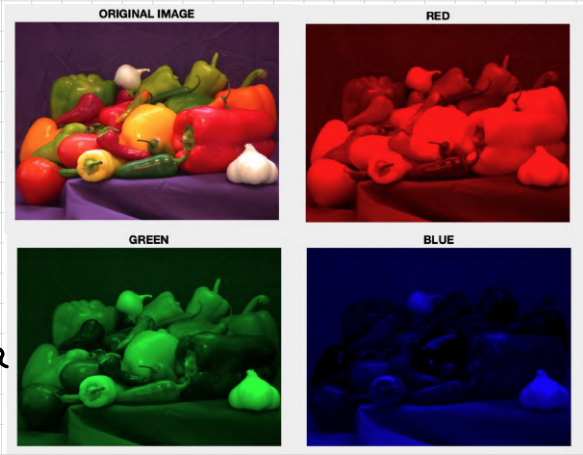
$$\rightarrow P(\hat{X}=0) = \frac{1-p-D}{1-2D} \cdot (1-D) + \frac{p-D}{1-2D}(D) = 1-p$$

$\rightarrow$ WITH THIS DISTRIBUTION: $I(X;\hat{X}) = H(\hat{X}) - H(\hat{X}|X) = H_b(p) - H_b(D)$

- # IMAGE COMPRESSION :

EACH IMAGE IS COMPOSED BY PIXELS

- EACH PIXEL CAN BE REPRESENTED IN 2 WAYS:
  - BINARY NUMBERS IN $[0; 2^n - 1]$ ON n BITS INDICATING THE SHADE OF GREY (GREYSCALE)
  - ARRAYS OF 3 (-> R, G, B) COMPONENT / EACH ONE IS A BINARY NUMBER IN $[0; 2^n - 1]$. ON $3n$ BITS, FOR COLOR IMAGES



- ## PNG FORMAT :
  - PNG SUPPORTS MANY FORMAT, ONE IS 'RGB WITH α CHANNE' / $\alpha \in (0, 1)$ -> TRANSPARENCY
  - $\forall$ COLORED PIXEL -> (r, g, b) / $r, g, b \in [0, 1]$
    - GIVEN A PIXEL $(\bar{p})$ AND A BACKGROUND $(\bar{b})$, THEY ARE PAINTED AS : $(1-\alpha)\bar{b} + \alpha\bar{p}$
  - WITH n QUANTIZATION BITS $(n = 1, 2, 4, 8, 16)$ -> $\forall$ ELEMENT : REPRESENTED AS $[0; 2^n - 1]$

- ## PERCEPTION OF COLORS :
  - COLOR : PERCEPTUAL RESULT OF LIGHT IN VISIBLE SPECTRUM IN $\lambda = [400\ nm; 700\ nm]$ (-> WAVE LENGHT)
  - BRIGHTNESS : VISUAL SENSATION OF EMITTED LIGHT
    - -> HARD TO DEFINE, IN PRACTICE IS REPRESENTED BY LUMINANCE (Y)
      - LUMINANCE : $Y(R, G, B) = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$ (-> G IS PERCEIVED AS MORE "BRIGHT")
      - CHROMINANCE : $\begin{cases} C_b = -0.09991 \cdot R - 0.33609 \cdot G + 0.436 \cdot B \\ C_r = 0.615 \cdot R - 0.55861 \cdot G - 0.05639 \cdot B \end{cases}$
        - -> YUV REPRESENTATION

- THERE ARE MANY WAYS OF COMPRESSING IMAGES (-) IMAGE TYPE

-> THE PRINCIPLE IS TO SEPARATE THE IMAGE INTO 3 GRAYSCALE IMAGES AND COMPRESS THEM SEPARATELY

- $IMG \longmapsto \left(IMG\big|_{R\ \text{GREYSCALE}}, IMG\big|_{G\ \text{GREYSCALE}}, IMG\big|_{B\ \text{GREYSCALE}} \right)$

- PROPERTIES:
  - FOR CONTINUOUS TONE IMAGES, ADJACENT PIXELS HAVE SIMILAR COLORS
  - EYE IS MORE SENSITIVE TO Y THAN YUV -> BETTER TO USE Y/YUV REPRESENTATION
    -> Y ENCODING MUST BE PRECISE, WHILE YUV CAN BE APPROXIMATED

- TRANSFORM CODING:

TECHNIQUES FOR IMAGES COMPRESSION

-> AN IMAGE CAN BE COMPRESSED TRANSFORMING ITS CORRELATED PIXELS TO A DECORRELATED REPRESENTATION, WHICH MUST BE REPRESENTED BY A LOWER AMOUNT OF BITS
- TRASFORMED VALUE ARE THEN QUANTIZED -> LOSS OF INFORMATION
- A DECODER RECONSTRUCT THE IMAGE USING AN INVERSE TRASFORM
- IF REDUNDANCY IS ELIMINATED FROM AN IMAGE -> COMPRESSION

eg.
- IMAGE PIXELS $\longmapsto$ 1-DIMENSIONAL VECTOR

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{pmatrix} \mapsto (\overrightarrow{x_{11}, x_{21}, x_{31}, x_{41}}, \overrightarrow{x_{12}, x_{22}, x_{32}, x_{42}}, x_{13}, x_{23}, x_{33}, x_{43}, x_{14}, x_{24}, x_{34}, x_{44})$$

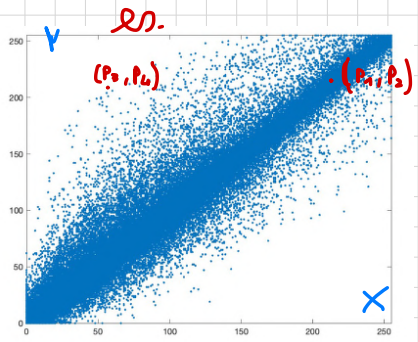
-> ADJACENT PIXELS IN MATRIX ARE ALSO CLOSE TO EACH OTHER IN VECTOR

- IF MATRIX SIZE IS EVEN -> 2-DIMENSIONAL DISCRETE OF ODD AND EVEN VALUES

- IF MATRIX SIZE IS EVEN -> 2-DIMENSIONAL DIAGRAM OF ODD AND EVEN VALUES:

-> ∀ PIXEL -> LEVEL IN $[0; 256]$

-> PIXELS SPLITTED IN EVEN AND ODD

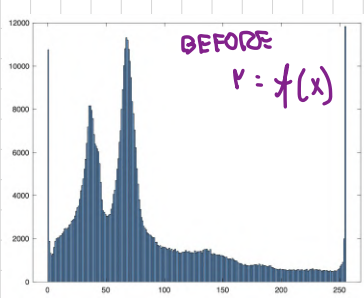


- IF 2 POINT ADJACENT WITH SIMILAR COLOR -> NEAR DIAGONAL

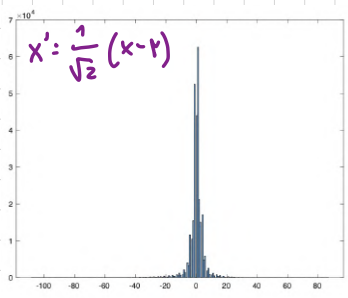
- WE CAN APPLY A 45° ROTATION TO MAP DIAGONAL POINT ↦ X AXIS:

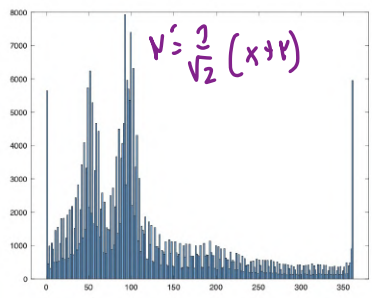
$$\begin{pmatrix} X' \\ Y' \end{pmatrix} = \begin{pmatrix} \cos\frac{\pi}{4} & -\sin\frac{\pi}{4} \\ \sin\frac{\pi}{4} & \cos\frac{\pi}{4} \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} X - Y \\ X + Y \end{pmatrix}$$

->



⟶





-> THE INFORMATION REQUIRED TO DESCRIBE X' IS LOWER THAN THE ONE FOR Y'

- IN FACT:

- $H(X) + H(Y) = 7.3765 + 7.38 = 14.7565$ bits
- $H(X-Y) + H(X+Y) = 4.3826 + 8.3576 = 12.734$ bits

-> USING A 45° ROTATION : ≈ 14% COMPRESSED BITS SAVED

- INSTEAD OF ENCODING COMPONENTS OF $\bar{p}$, IT MAY BE CONVENIENT TO ENCODE: $\bar{c} = W \cdot \bar{p}$ / W: COEFFICIENT MATRIX

- IF W IS AN ORTHOGONAL MATRIX: $W \cdot W^T = I \rightarrow \bar{p} = W^T \cdot \bar{c}$

- CHOICE OF W -> HADAMARD MATRIX:

$$W_N = \begin{pmatrix} W_{N/2} & W_{N/2} \\ W_{N/2} & -W_{N/2} \end{pmatrix}, \quad W_1 = 1 \quad / \quad N = 2^\alpha, \ \alpha \in \mathbb{N}^+$$

ex. W_2, W_4

- $W_2 = \begin{pmatrix} W_1 & W_1 \\ W_1 & -W_1 \end{pmatrix} = \left(\begin{array}{c|c} 1 & 1 \\ \hline 1 & -1 \end{array}\right)$

- $W_4 = \begin{pmatrix} W_2 & W_2 \\ W_2 & -W_2 \end{pmatrix} = \left(\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ \hline 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{array}\right)$

ex. MATLAB

H OF THE 4 ROWS OF Y, THE ORIGINAL IMAGE

H OF THE 4 ROWS OF Z, THE COMPRESSED IMAGE

- Let us use the image "peppers"
- We take all the pixel values and arrange them in a 4×147456 matrix
- Then we multiply each column by W_4
  - `X = double(imread("peppers.png"));`
  - `Y = reshape(X,4,numel(X)/4);`
  - `W = hadamard(4);` W_4
  - `Z = W*Y;`
  - `h = @(k,x)estim_entropy(x(k,:));`
  - `disp(h(1,Y)+h(2,Y)+h(3,Y)+h(4,Y))`
  - `29.5104`
  - `disp(h(1,Z)+h(2,Z)+h(3,Z)+h(4,Z))`
  - `25.1213` 15% reduction
- Now, try with size 8 and see what happens

```
function H = estim_entropy(x)
p = tabulate(x(:));
p = p(:,2);
p = p/sum(p);
H = -sum(p.*log2(p+eps));
```

- W_8:

```
>> X=double(imread("peppers.png"));
>> size(X)
ans =
   384   512     3
>> Y=reshape(X,8,numel(X)/8);
>> size(Y)
ans =
           8       73728
>> W=hadamard(8)
W =
     1     1     1     1     1     1     1     1
     1    -1     1    -1     1    -1     1    -1
     1     1    -1    -1     1     1    -1    -1
     1    -1    -1     1     1    -1    -1     1
     1     1     1     1    -1    -1    -1    -1
     1    -1     1    -1    -1     1    -1     1
     1     1    -1    -1    -1    -1     1     1
     1    -1    -1     1    -1     1     1    -1
>> Z=W*Y;
```

```
function H = estim_entropy(x)
p = tabulate(x(:));
p = p(:,2);
p = p/sum(p); p(p==0) = 1;
H = -sum(p.*log2(p));
>> tabulate([1 1 2 3])
  Value    Count   Percent
      1        2     50.00%
      2        1     25.00%
      3        1     25.00%
```

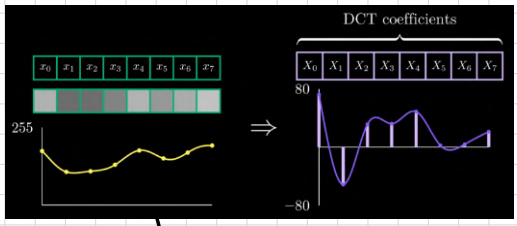
```
>> h=@(k,x)estim_entropy(x(k,:));
>> s=0;for i=1:8,s=s+h(i,Y);end;disp(s)
   59.0072
```

```
>> s=0;for i=1:8,s=s+h(i,Z);end;disp(s)
   53.0169
```

->

```
>> 53/59
ans =
    0.8983
```

- **DCT :** ⇝ DISCRETE COSINE TRANSFORM



- $f : \mathbb{R}^n \hookrightarrow \mathbb{R}^m$, SIMILAR TO DFT BUT $\mathbb{C} \not\hookrightarrow \mathbb{C}$
- GIVEN AN ARRAY OF REAL NUMBERS $\bar{p} = (p_0, \dots, p_{n-1})$

$$\text{DCT} : \quad \hat{p}_f = \sqrt{\frac{2}{n}}\, c_f \sum_{t=0}^{n-1} \cos\left(\frac{(2t+1) f \pi}{2n}\right) p_t$$

$$\hookrightarrow c_f = \begin{cases} \frac{1}{\sqrt{2}} & , f = 0 \\ 1 & , f \neq 0 \end{cases} \quad , f = 0, \dots, n-1$$

- $\text{DCT}(\bar{p}) = \sqrt{\frac{1}{2n}}\, c_f\, e^{-j \frac{(2n-1) f}{4n} 2\pi} \cdot \text{DFT}\left([p_{n-1}, \dots, p_0, p_0, \dots, p_{n-1}]\right)$
- DCT IS AN INVERTIBLE LINEAR MAPPING : $\hat{p} = W_n \cdot \bar{p}$

$$\rightarrow p_f = \sum_{t=0}^{n-1} (W_n)_{f,t} \cdot p_t$$

- PROPERTIES OF W_n :
  - $(W_n)_{f,t} = \sqrt{\frac{2}{n}}\, c_f \cos\left(\frac{(2t+1) f \pi}{2n}\right), \quad t, f = 0, \dots, n-1$
  - W_n IS AN ORTHOGONAL MATRIX : $W_n^T W_n = I_n$

- **IDCT :** ⇝ INVERSE DCT
  - GIVEN W_n ORTHOGONALITY $\rightarrow$ $\bar{p} = W_n^T \cdot \hat{p}$
  - IDCT CAN BE EXPRESSED BY :

$$p_t = \sum_{f=0}^{n-1} (W_n)_{f,t} \cdot p_f$$

- **2-DIMENSIONAL DCT:**

CONSIDER A REAL MATRIX $P = (p_{i,j})_{i=1,j=1}^{m,n}$

-> 2-DIM DCT: $\hat{P} = W_m \cdot P \cdot W_n^T$

$$\hookrightarrow (\hat{P})_{f,f'} = \sum_{t=0}^{m-1} \sum_{t'=0}^{n-1} (W_m)_{f,t} \cdot (W_n)_{f',t'} \cdot (P)_{t,t'} =$$

$$= \sqrt{\frac{4}{m \cdot n}}\, c_f c_{f'} \sum_{t=0}^{m-1} \sum_{t'=0}^{n-1} \cos\left(\frac{(2t+1) f \pi}{2m}\right) \cos\left(\frac{(2t'+1) f' \pi}{2n}\right) (P)_{t,t'}$$

~> 2-DIM

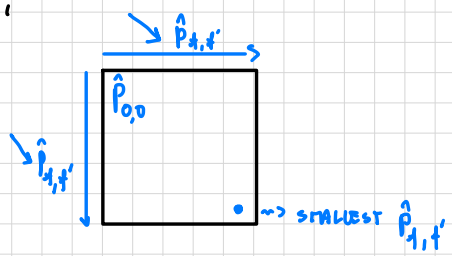
- **DCT CHARACTERISTICS:**

LET $\hat{p}_{f,f'} = (\hat{P})_{f,f'}$, $p_{t,t'} = (P)_{t,t'}$

- $\hat{p}_{0,0} = \sqrt{\frac{1}{m \cdot n}} \sum_{t=0}^{m-1} \sum_{t'=0}^{n-1} p_{t,t'}$

-> IF $p_{t,t'} > 0$ -> $\hat{p}_{0,0} > \hat{p}_{f,f'}$, $\forall f, f'$

- $\hat{p}_{00}$: DC COEFFICIENT, $\hat{p}_{f,f}$: AC COEFFICIENTS



~> 2-DIM

- **IDCT CHARACTERISTICS:**

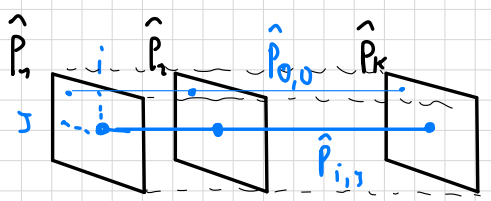
- $P = W_m^T \hat{P} W_n$

$$\hookrightarrow (\hat{P})_{t,t'} = \sum_{f=0}^{m-1} \sum_{f'=0}^{n-1} (W_m)_{f,t} \cdot (W_n)_{f',t'} \cdot (P)_{f,f'} =$$

$$= \sqrt{\frac{4}{m \cdot n}} \sum_{f=0}^{m-1} \sum_{f'=0}^{n-1} c_f c_{f'} \cos\left(\frac{(2t+1) f \pi}{2m}\right) \cos\left(\frac{(2t'+1) f' \pi}{2n}\right) (P)_{t,t'}$$

• DCT APPLICATION :

- CONSIDER AN IMAGE OF SIZE $N \times M$ PIXELS
- PARTITION IMAGE IN SQUARE BLOCK OF 8×8 (TYPICALLY): P_k ($k = 1, \dots, K$) = 8×8 SQUARE
- $\forall P_k \rightarrow \hat{P}_k = DCT(P_k)$



MATLAB

→ CONSIDERING $\hat{P}_{1,\dots,K}$ · VECTOR $\hat{P}_{i,j} = \left(\hat{P}_0\right)_{i,j}, \dots, \left(\hat{P}_K\right)_{i,j} = P(i, j, :)$

- $\hat{P}_{i,j}$ ARE QUANTIZED → LOSSY COMPRESSION
- DECODER ON QUANTIZED VALUES TO RECOVER THE ORIGINAL IMAGE

es. CORRELATED VALUES

```
B = [
12 10  8 10 12 10  8 11
11 12 10  8 10 12 10  8
 8 11 12 10  8 10 12 10
10  8 11 12 10  8 10 12
12 10  8 11 12 10  8 10
10 12 10  8 11 12 10  8
 8 10 12 10  8 11 12 10
10  8 10 12 10  8 11 12];
W = dct1(8);
C = W*B*W';
Q = round(C);
D = round(W'*Q*W);
```

```
C =
  79.8750    0.0397   -0.3382   -1.2487    1.6250   -1.7471    0.8166    1.0365
  -1.8027   -0.4426   -0.9020    1.0570   -0.7564   -1.0891   -1.3063    0.9316
   1.5491   -1.0178    0.2526   -2.3607    1.8758    0.9169    0.6831    0.7277
   1.8284   -3.1831   -0.8385    0.0335   -0.1762    0.3444   -0.2083   -1.2622
  -2.1250    1.2042    1.8922   -0.8909    2.6250   -1.0098    0.9751   -1.1933
  -0.3546    0.2193   -0.7230   -1.7466   -1.5909   -2.1245   -1.9686   -1.4760
  -0.5064   -0.1069   -1.5669   -0.4986   -0.3711    0.3514   -2.7526   -3.2423
   0.4680   -0.1594   -1.0977    1.3820    3.6474   -1.4001    0.0732    1.5336
```

```
Q =
    81     0     0     0     0     0     0     0
     0     2     1     2     0    -2     0     0
     0    -1     1     0     0     0     0     0
     0     2     0     5     1    -3     0    -1
     0     0     0    -1     8     1     0     0
     0    -2     0    -3    -1     2     1     0
     0     0     0     0     0    -1    -1     0
     0     0     0    -1     0     0     0    -1
```

1 ERROR

```
D =
    12    10     8    10    12    10     8    11
    11    12    10     8    10    12    10     8
     8    11    12    10     8    10    12    10
    10     8    11    12    10     8    10    12
    12    10     8    11    12    10     8    10
    10    12     9     8    11    12    10     8
     8    10    12    10     8    11    12    10
    10     8    10    12    10     8    11    12
```

es. UNCORRELATED VALUES

```
B = [
 8 10  9 11 11  9  9 12
11  8 12  8 11 10 11 10
 9 11  9 10 12  9  9  8
 9 12 10  8  8  9  8  9
12  8  9  9 12 10  8 11
 8 11 10 12  9 12 12 10
10 10 12 10 12 10 10 12
12  9 11 11  9  8  8 12
];
W = dct1(8);
C = W*B*W';
Q = round(C);
D = round(W'*Q*W);
```

```
C =
  79.8750    0.0397   -0.3382   -1.2487    1.6250   -1.7471    0.8166    1.0365
  -1.8027   -0.4426   -0.9020    1.0570   -0.7564   -1.0891   -1.3063    0.9316
   1.5491   -1.0178    0.2526   -2.3607    1.8758    0.9169    0.6831    0.7277
   1.8284   -3.1831   -0.8385    0.0335   -0.1762    0.3444   -0.2083   -1.2622
  -2.1250    1.2042    1.8922   -0.8909    2.6250   -1.0098    0.9751   -1.1933
  -0.3546    0.2193   -0.7230   -1.7466   -1.5909   -2.1245   -1.9686   -1.4760
  -0.5064   -0.1069   -1.5669   -0.4986   -0.3711    0.3514   -2.7526   -3.2423
   0.4680   -0.1594   -1.0977    1.3820    3.6474   -1.4001    0.0732    1.5336
```

```
Q =
    80     0     0    -1     2    -2     1     1
    -2     0    -1     1    -1    -1    -1     1
     2    -1     0    -2     2     1     1     1
     2    -3    -1     0     0     0     0    -1
    -2     1     2    -1     3    -1     1    -1
     0     0    -1    -2    -2    -2    -2    -1
    -1     0    -2     0     0     0    -3    -3
     0     0    -1     1     4    -1     0     2
```

3 ERRORS

```
D =
     8    10     9    11    12     9     9    12
    11     8    12     8    11    10    11    10
     9    11     9    10    12     9     9     8
     9    12    10     8     8     9     8     9
    12     8     9     9    12     9     8    11
     8    11    10    12     9    12    12    10
    10    10    12    10    12    10    10    12
    12     9    10    11     9     8     8    12
```

- **JPEG:**

IT'S A LOSSY / LOSSLESS IMAGE COMPRESSION ALGORITHM

- IT CAN REACH $\times 20$ COMPRESSION FACTOR

- **ALGORITHM:**

1. RGB $\longmapsto$ Y / YUV : EYE MORE SENSITIVE TO Y THAN YUV
2. YUV COMPONENTS DOWNSAMPLING:

- 2h2V: HORIZONTAL AND VERTICAL PIXELS DOWNSTAMED BY 2
  - COMPRESSION RATIO = $\frac{1}{3} \cdot 1 + \frac{2}{3} \cdot \frac{1}{2 \cdot 2} = \frac{1}{2}$ (Y, C_b e C_r)
- 2h1V: ONLY HORIZONTAL PIXELS DOWNSAMPLED BY 2
  - COMPRESSION RATIO: $\frac{1}{3} 1 + \frac{2}{3} \cdot \frac{1}{2} = \frac{2}{3}$

3. DATA UNITS: PIXELS ORGANIZED IN 8×8 BLOCKS
4. DCT APPLIED TO ALL DATA UNITS
5. QUANTIZATION BASED ON QC (QUANTIZATION COEFFICIENT):

$x \longmapsto QC \cdot \text{round}\left(\frac{x}{QC}\right)$

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Luminance

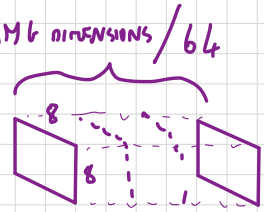
17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Chrominance

6. COMPRESSION: RUN-LENGHT ENCODING + HUFFMAN CODES

ex. (IN MATLAB) 'PEPPERS.PNG'

1.
```
Y   = rgb2ycbcr(X);
Z   = zeros(8,8,numel(Y)/64);
```
~> Z =



2. $Q_L^{\rightarrow 1}$ e $Q_C^{\rightsquigarrow 2 \cdot 3}$ DEFINED AS MATRIXES

2./3./4./5.

```
c = 0;                                  c is an index for Z
for i=1:size(Y,1)/8                     Triple loop to process JPEG data units
  for j=1:size(Y,2)/8                   Divided by image position and component
    for k=1:3 ~> Y, Cb, Cr components
      c  = c+1;                         Update c
      YY = Y(8*i+(-7:0),8*j+(-7:0),k);  ~> YY: 8x8 data units of Y
      YY = dct(double(YY));             Current data unit in the triple loop
      if k==1                           DCT
        YY = QL.*round(YY./QL);
      else                              Quantization with the
        YY = QC.*round(YY./QC);         proper coefficient
      end
      Z(:,:,c) = YY;                    Storage of quantized data unit
    end
  end
end
```

6. (NOT RECURRENT)

- ENTROPIES : ~> X, Z stored as $8 \times 8 \times (N = 9276)$

ORIGINAL IMAGE X

```
X0 = reshape(X,8,8,numel(X)/64);
H0 = zeros(8);
for i=1:8
  for j=1:8
    H0(i,j) = estim_entropy(X0(i,j,:));
  end
end
```

COMPRESSED IMAGE Z

```
H = zeros(8);
for i=1:8
  for j=1:8
    H(i,j) = estim_entropy(Z(i,j,:));
  end
end
```

→ ↓ H0

```
H0 =
   7.2306   7.1558   7.1231   7.2277   7.3336   7.3968   7.4608   7.4746
   7.2246   7.1513   7.1221   7.2388   7.3378   7.4154   7.4738   7.4849
   7.2033   7.1418   7.1277   7.2640   7.3431   7.4301   7.4867   7.4626
   7.2049   7.1254   7.1314   7.2995   7.3422   7.4516   7.4950   7.4593
   7.2080   7.1241   7.1594   7.3166   7.3554   7.4526   7.4941   7.4727
   7.2006   7.1240   7.1901   7.3131   7.3577   7.4565   7.4767   7.4755
   7.1934   7.1183   7.1993   7.3324   7.3765   7.4598   7.4742   7.4639
   7.1712   7.1172   7.2151   7.3405   7.4006   7.4545   7.4717   7.4505
```

↓ H

```
H =
   4.6692   4.8559   4.6768   4.0194   3.3056   3.0791   2.9884   2.9075
   1.7023   1.6561   1.4537   0.9817   0.7152   0.3749   0.3699   0.4009
   0.7612   0.6951   0.5023   0.3168   0.2084   0.1183   0.0859   0.1207
   0.3479   0.2806   0.2062   0.1493   0.0593   0.0157   0.0217   0.0398
   0.1772   0.1311   0.0520   0.0165   0.0119   0.0016   0.0016   0.0042
   0.0577   0.0255   0.0016   0.0016        0        0        0        0
        0        0        0        0        0        0        0        0
        0        0        0        0        0        0        0        0
```

- SIZES COMPARISON

1 B = 8 bits

- ORIGINAL IMAGE : (IMG RESOLUTION R,G,B) $(384 \times 512) \cdot 3 \cdot 8 = 4\,718\,592$ BITS
- YCbCr FORMAT : $4\,315\,267.67$ BITS
- AFTER DCT + QUANTIZATION : $392\,312.65$ BITS

→ COMPRESSION FACTOR : $\times 12.03$

* JPEG →

- FULL MATLAB CODE:

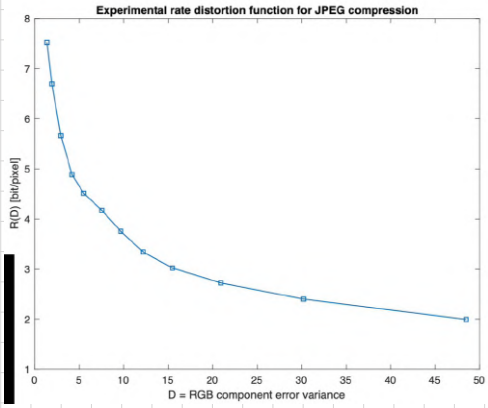
```
QL = round(QL/f);
QC = round(QC/f);
c  = 0;
for i=1:size(Y,1)/8
  for j=1:size(Y,2)/8
    for k=1:3
      c  = c+1;
      YY = Y(8*i+(-7:0),8*j+(-7:0),k);
      YY = dct(double(YY));
      if k==1
        YY = QL.*round(YY./QL);
      else
        YY = QC.*round(YY./QC);
      end
      D(8*i+(-7:0),8*j+(-7:0),k) = idct(YY);
      Z(:,:,c) = YY;
    end
  end
end
D = ycbcr2rgb(D);imshow(D)
```

*

- R(D):

```
f   = [1 1.5 2 2.5 3 4 5 6 7 10 15 20];
R   = f*0;
mse = f*0;
for i=1:numel(f)
  [X,D,R(i)] = exjpeg(f(i));
  mse(i) = var(double(X(:)-D(:)));
  fprintf('%d/%d\n',i,numel(f))
end
close all
plot(mse,R,'-s')
title('Experimental rate distortion function for JPEG compression')
xlabel('D = RGB component error variance')
ylabel('R(D) [bit/pixel]')
```

↓



- **CRYPTOGRAPHY** :

- **APPLICATIONS FOR INFORMATION THEORY** :

1. **CHANNEL CODING** :

ADD SOME PARITY BITS OBTAINED LINEARLY FROM THE MESSAGE BITS

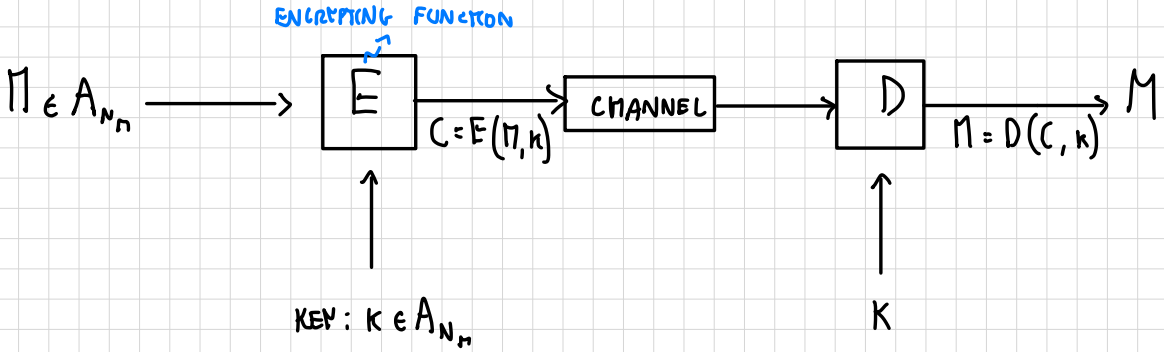
- RECIEVER EXPLOITS PARITY BITS FOR :
  - ERROR DETECTION : CHECK IF MESSAGE IS INTEGER (CRC , CHECK SUM)
  - ERROR CORRECTION : CORRECTION OF ERRORS INTRODUCED BY THE CHANNEL

2. **CRYPTOGRAPHY** :

WE WANT TO PROTECT MESSAGE AGAINST ATTACKS :

- CONFIDENTIALITY : PREVENT AN UNAUTHORIZED READING OF THE MESSAGE
- AUTHENTICATION : PREVENT IDENTITY SPOOFING
- INTEGRITY : PREVENT ATTACKER TO CHANGE THE MESSAGE

- KIRKOFF'S PRINCIPLE : " ATTACKER PERFECTLY KNOWS THE ENCRYPTING SYSTEM "
- COMPUTATIONALLY SECURE ALGORITHM : PROTECTED FROM TIME t SPENT ON ATTACK
- **ENCRYPTION SCHEME** :
  - A : ALPHABET FOR PLAINTEXT , KEY , CHYPHER TEXT
  - PLAIN TEXT MESSAGE : $M \in A_{N_M}$ / A_{N_M} : SET OF VECTORS OF N_M ELEMENTS FROM A
  - CHYPHER TEXT : $C \in A_{N_M}$



ENTROPY INTERPRETATION OF SECURITY :

ENTROPY PROPERTIES OF A COMPLETLY SECURE SYSTEM :

- $H(M|C) = H(M)$ ~> WHEN C IS REVEALED, IT DOESN'T LOWER THE AMOUNT OF INFO OF M

  $\rightarrow I(M;C) = 0$ $\quad (I(M;C) = H(M) - H(M|C))$

- $H(M|C,K) = 0$

THIS LEADS TO RESULTS ABOUT THE KEY LENGTH :

- $H(K) \overset{\uparrow}{\geq} H(K|C) = H(K,C) - H(C)$ (C SHOULDN'T GIVE MORE INFO ON K)

- $H(M,C,K) = \overset{=0}{H(M|C,K)} + H(C,K) = H(C,K)$

$$\rightarrow H(K) \geq H(M,C,K) - H(C) + \left(-H(K,C) + H(K,C)\right) =$$

$$= H(K|M,C) + H(M|C) \geq H(M|C) \overset{!}{=} H(M)$$

(COMPLETLY SECURE SYSTEM)

$\rightarrow$ COMPLETLY SECURE SYSTEM $\Leftrightarrow H(K) \geq H(M)$

- SUPPOSE M, K GENERATED BY INDEPENDENT SYMBOLS OF SAME ALPHABET A :

  ($\rightsquigarrow N_M$ SYMBOLS IN M)

  - $H(M) = N_M H(A)$
  - $H(K) = N_K H(A)$

$\rightarrow N_K \geq N_M$: LENGTH OF K SHOULD BE GREATER THEN THE LENGTH OF M

- **ONE TIME PAD:**
  - M : N_M BITS
  - K : N_K RANDOM BITS / $N_M = N_K = N$

⊕	0	1
0	0	1
1	1	0

→ XOR :

ENCRYPTION

$$\rightarrow C = E(M, K) = M \oplus K$$

DECRYPTION

$$\rightarrow M = D(C, K) = C \oplus K$$

→ THIS IS A COMPLETELY SECURE SYSTEM

- PROBLEMS:
  - IF $N_M >> 0 \rightarrow N_K >> 0$: LONG KEY NEEDS TO BE SHARED
  - K CANNOT BE RE-USED, IT MUST CHANGE ∀ NEW MESSAGE
- IN PRACTICE: N_K' BITS FOR KEY / $N_K' < N_K$

- **LONG MESSAGE SEQUENCE:**
  - **STREAM CYPHERS:** ENTIRE SEQUENCE ENCRYPTED SYMBOL BY SYMBOL.
    - ex. SNOW
    - STARTING FROM A SMALL K, A PSEUDO-RANDOM SEQUENCE IS GENERATED TO USE XOR ON M

    V: EASY E(·), D(·), HIGH BITRATE, LOW POWER CONSUME, NO ERROR PROPAGATION
  - **BLOCK CYPHERS:** LONG SEQUENCE DIVIDED IN BLOCKS.
    - ex. AES
    - K IS USED TO ENCRYPT THEM BLOCKWISE

RANDOM BINARY SEQUENCES:

- $A = \{0, 1\}$
- BITS: STAT. INDIP. AND EQUIPROBABLE $\rightsquigarrow p_0 = 0.5, \; p_1 = 0.5$
- CHARACTERISTICS:
  - FOR A LONG (STEPS $\longmapsto \infty$) RANDOM SEQ. WITH N BITS:

PROP. $N_0 \rightsquigarrow (\#\text{BITS} = 0) = N_1 = \frac{1}{2} N$

- AUTO-CORRELATION:

$$R(\tilde{i}) = \sum_{i=0}^{N-1} v'(i)\, v'(i-\tilde{i}) \Big/ N \qquad 0 \le \tilde{i} \le N-1$$

$v'(i)$: BIPOLAR SEQ. $\rightarrow$ (MAPPING) $\begin{cases} 0 \longmapsto -1 \\ 1 \longmapsto 1 \end{cases}$

PROP. $\longrightarrow \begin{cases} R(\tilde{i} = 0) = N \\ R(\tilde{i} \neq 0) = 0 \end{cases}$

ES.

SUPPOSE SEQ. PERIODIC, $N = 4$

- $R(\tilde{i} = 0)$: $-1\;1\;1\;1 \,|\, -1\;1\;1\;-1 \,|\, -1\;1\;1\;1 \quad \rightsquigarrow v'(i) \overset{\text{XOR}}{\cdot} v'(i-\tilde{i})$

$$\begin{array}{r} -1\;1\;1\;1 \\ \hline 1\;1\;1\;1 \end{array} = 4 = N$$

- $R(\tilde{i} = 1)$: (SHIFT 1 DX)

$$\begin{array}{r} -1\;1\;1\;1 \\ 1\;-1\;1\;1 \\ \hline -1\;-1\;1\;1 \end{array} = 0$$

- **RUN:**

A RUN OF LENGTH L IS A SEQUENCE OF L CONSECUTIVE EQUAL SYMBOLS

ex. $0\overline{111}0 \rightarrow L = 3$

PROP.
→ GIVEN A BINARY SEQ. OF LENGTH N. N_T = TOT N° OF RUNS $= \frac{N}{2}$ ⇝ # ROWS OF 0's = # ROWS OF 1's

PROP.
→ $N(L=1) = \frac{N_T}{2^i}$

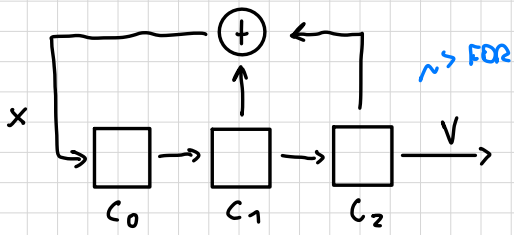
	0	1
0	1	0
1	0	1

• LINEAR FEEDBACK SHIFT REGISTER (LFSR):

- SHIFT REGISTER:

$$\ldots \rightarrow 1 \rightarrow 0 \rightarrow 1 \rightarrow \ldots$$
$$\ldots \rightarrow \%_1 \rightarrow 1 \rightarrow 0 \rightarrow \ldots$$

- LINEAR FEEDBACK SR:



~> FOR EXAMPLE

- GIVEN A SEED ~> 1 0 0 IN EXAMPLE.

$C_1 + C_2$ X	C_0	C_1	C_2	$V = C_2$
0	1	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0
0	1	1	1	1
0	0	1	1	1
1	0	0	1	1
	1	0	0	

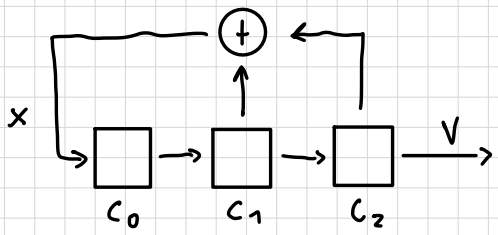
$\rightarrow N = 7 = 2^N - 1$

↳ START 0 0 0 FORBIDDEN

$\rightarrow N_0 \approx N_1 :$ ~~1~~ 0 0 0 STATE

↳ AGAIN INITIAL STATE

es. SEQ : 0 0 1 0 1 1 1



X	C_0	C_1	C_2	$V = C_2$
0	1	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0
0	1	1	1	1
0	0	1	1	1
1	0	0	1	1
	1	0	0	

$C_1 + C_2$

	0	1
0	0	1
1	1	0

	-1	1
-1	-1	1
1	1	-1

- **AUTOCORRELATION** :

- $R(\tilde{i}=0)$:

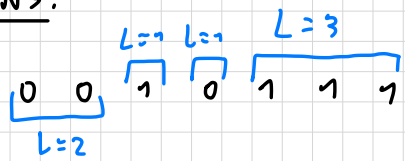
$$\begin{array}{ccccccc} -1 & -1 & 1 & -1 & 1 & 1 & 1 \\ -1 & -1 & 1 & -1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array} = 7$$

- $R(\tilde{i}=1)$:

$$\begin{array}{ccccccc} -1 & -1 & 1 & -1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 \\ \hline -1 & 1 & -1 & -1 & -1 & 1 & 1 \end{array} = -1$$

$$\rightarrow R(\tilde{i}) : \begin{cases} 7, & \tilde{i}=0 \\ -1, & \tilde{i}\neq 0 \end{cases}$$

- **RUNS**:



$\rightarrow$

- $L=1$: $N_0(1) = N_1(1) = 1$
- $L=2$: $N_0(2) = 1$
- $L=3$: $N_1(3) = 1$

$\Big\}$ $N_1 = 4$

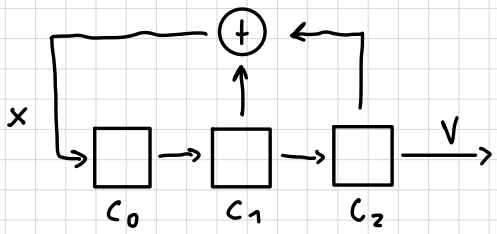
$\rightarrow N(i) = \dfrac{N_T}{2^i}$, $N_0(i) = N_1(i)$ $\rightsquigarrow$ ALMOST , $\because$ N IS ODD

- COMPARISON

	M-SEQUENCE FROM m-CELLS LFSR	RANDOM
PERIOD	$N = 2^m - 1$	
# 0/1	$N_1 = N_0 + 1$ → ONE MORE FOR 1000	$N_1 = N_0$
R(i)	$R(i \neq 0) = -1$	$R(i \neq 0) = 0$
RUNS	$N(i) = \frac{N_T}{2^i}$, $1 \leq i \leq m-2$	$N(i) = \frac{N_T}{2^i}$, $\forall i$ (!)

- **LFSR : MATRIX REPRESENTATION.**

$$\begin{cases} C_0' = C_1 + C_2 \\ C_1' = C_0 \\ C_2' = C_1 \end{cases}$$



C_1+C_2 X	C_0	C_1	C_2	$V = C_2$
0	1	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0
0	1	1	1	1
0	0	1	1	1
1	0	0	1	1
	1	0	0	

→ NEXT STATE / CURRENT STATE

$$\begin{bmatrix} C_0' \\ C_1' \\ C_2' \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ C_2 \end{bmatrix}$$

es.

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

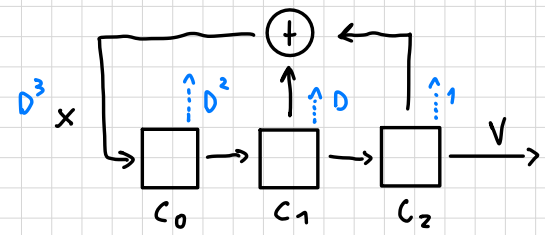
- HINT : ONLY SELECT COLUMN WHERE IS EQUAL TO 1, THE SUM BY ROWS

→ $\begin{matrix}1\\0\\0\end{matrix} \longmapsto \begin{matrix}0\\1\\0\end{matrix} \longmapsto \begin{matrix}1\\0\\1\end{matrix} \longmapsto \begin{matrix}1\\1\\0\end{matrix} \longmapsto \begin{matrix}1\\1\\1\end{matrix} \longmapsto \begin{matrix}0\\1\\1\end{matrix} \longmapsto \begin{matrix}0\\0\\1\end{matrix} \longmapsto \begin{matrix}1\\0\\0\end{matrix}$

- GIVEN N : OUTPUT LENGTH → $C^N = I$

- LFSR AS $p(D)$:

GIVEN THE INPUT VECTOR $(x_2\ x_1\ x_0)$:



$\rightarrow (x_2\ x_1\ x_2) \equiv x_2 D^2 + x_1 \cdot D + x_0 = X(D)$

↳ POLYNOMIAL REPRESENTATION OF THE VECTOR

- $p(D)$ OF LFSR : $p(D) \,/\, X(D) \cdot D^N \ \text{mod}\ p(D) = X(D)$

ex. $p(D) = D^3 + D + 1$ $\qquad D^3 \,\%\, (D^3 + D + 1) = D + 1$

$1 \xrightarrow{1\cdot D \,\%\, p(D)} D \longmapsto D^2 \longmapsto D^3 \longmapsto D+1 \longmapsto D^2 + D$

$\longmapsto D^3 + D^2 \longmapsto D^2 + D + 1 \longmapsto D^3 + D^2 + D \longmapsto D^2 + 1$

$\longmapsto D^3 + D \longmapsto 1$

$\rightarrow X(D) \cdot (D^N + 1) \ \text{mod}\ p(D) = 0$

↳ $p(D)$ DIVIDES $D^N + 1$

- PRIMITIVE POLYNOMIAL :

POLYNOMIAL SUCH THAT CAN BE DIVIDED ONLY BY 1, ITSELF

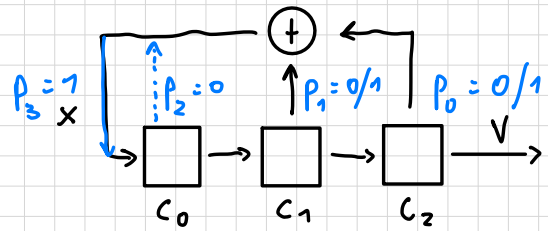
AND DIVIDES $D^N + 1 \;/\; N = 2^m + 1$, $m = \text{degree}(p(D))$

- LFSR MUST BE DESIGNED AS A PRIMITIVE POLYNOMIAL TO HAVE THE DESIRED CHARACTERISTICS

  -> THIS WAY PERIOD $N = 2^m - 1$ IS MAXIMUM

EX.
->

$$p(D) = \overset{1}{p_3} D^3 + \overset{0}{p_2} D^2 + p_1 D + p_0 = D^3 + D + 1$$

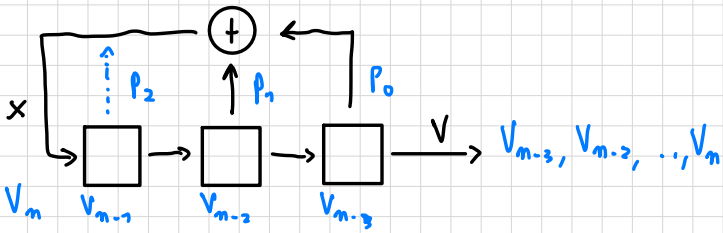


- **BREAKING A LINEAR LFSR :**

- USER KEY = STARTING SEED OF LFSR
- PSEUDO-RANDOM SEQ. S USED TO ENCRYPT M : LFSR m-SEQUENCE

->

$$C = E(M, S) = M \oplus S$$
$$M = D(C, S) = C \oplus S$$



-> $V_m = p_2 V_{m-1} \oplus p_1 V_{m-2} \oplus p_0 V_{m-3}$

->

$$\begin{bmatrix} V_5 \\ V_4 \\ V_3 \end{bmatrix} = \begin{bmatrix} V_4 & V_3 & V_2 \\ V_3 & V_2 & V_1 \\ V_2 & V_1 & V_0 \end{bmatrix} \begin{bmatrix} p_2 \\ p_1 \\ p_0 \end{bmatrix}$$

-> GOAL ; FIND $\overset{0}{p_2}, \overset{1}{p_1}, \overset{1}{p_0}$

- FOR AN LFSR WITH m-CELLS -> USE $2m$ BITS EQUATION FOR $\begin{bmatrix} p_{m-1} \\ \vdots \\ p_0 \end{bmatrix}$

EX. $V_0\ V_1\ V_2\ V_3\ V_4\ V_5$ -> NOT NEEDED

0 0 1 0 1 1 1

A = 0 0 1 0 1 1 , b = 0 1 0 1 1 1

$$b\begin{bmatrix} V_5 \\ V_4 \\ V_3 \end{bmatrix} = \begin{bmatrix} V_4 & V_3 & V_2 \\ V_3 & V_2 & V_1 \\ V_2 & V_1 & V_0 \end{bmatrix} \begin{bmatrix} p_2 \\ p_1 \\ p_0 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_2 \\ p_1 \\ p_0 \end{bmatrix}$$

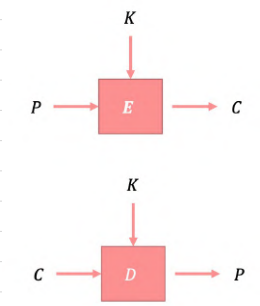
- START FOR $m = 2$ -> COMPUTE $p(D)$
- IF REMAINING BIT NOT FROM $p(D)$:
  -> $m \mathrel{+}= 1$

- **BLOCK CIPHERS AND AES:**
  - PLAINTEXT: P
  - CHIPHERTEXT: C
  - DEF. PERMUTATION: INVERTIBLE TRANSFORM / $\forall$ ITEM $\rightarrow$ $\exists!$ INVERSE

    TO BE SECURE:
    - IT'S DETERMINED BY THE KEY
    - DIFFERENT KEYS $\rightarrow$ DIFFERENT C
    - LOOK RANDOM, PATTERNS SHOULDN'T BE VISIBLE
      ~> COMPUTATIONALLY INFEASIBLE
    - HARD TO REVERSE WITHOUT THE KEY

- **BLOCK CIPHERS:**

  SET OF ENCRYPTION (E) AND DECRYPTION (D) ALGORITHM



  - ENCRYPTION: $C = E(K, P)$
  - DECRYPTION: $P = D(K, C)$

  - KEY PARAMETERS FOR SECURITY:
    - BLOCK SIZE: es. 128 bit FOR AES
    - KEY LENGTH: es. 128/256 bit FOR AES

Block Size	Memory
16-bit	128 KB
32-bit	16 GB
64-bit	128 EB

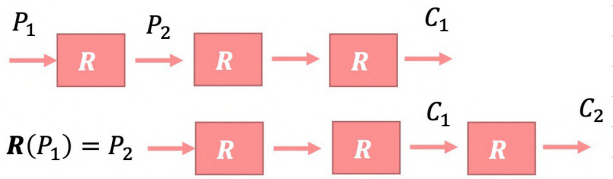
  -> IF BLOCK IS TOO SMALL (es 16 bit) -> YOU CAN BUILD A LUT OF ALL POSSIBLE P TO DECRYPT C

  (IN AES)
  OPERATION: SUBSTITUTION + PERMUTATION

  - BLOCK CIPHERS, TO BE STRONG, USE REPETITION OF MANY SIMPLE ROUNDS
  - LET $R_1, \dots, R_N$ BE A SET OF INVERTIBLE TRANSFORMS:
    - $C = R_N(R_{N-1}(R_{\dots}(R_1(P))))$
    - $D = R_N^{-1}(R_{N-1}^{-1}(R_{\dots}^{-1}(R_1^{-1}(C))))$

- K_i : ROUND KEY , K : MAIN KEY

  -> K_i ARE DERIVED FROM K, USING A KEY SCHEDULE ALGORITHM



- EVERY ROUND USES A DIFFERENT K_i TO AVOID SLIDE ATTACKS

- IF AN ATTACKER GET TO KNOW ONE OUTPUT OF A ROUND, (P_1, C_1) AND (P_2, C_2)

  -> IT CAN HELP RECOVER K

- SUBSTITUTION PERMUTATION NETWORK (SPN) :

  - 2 MAIN METHODS :

    - DIFFUSION (DEPTH) : REDUNDANCIES IN P ARE SPREAD ON WHOLE C

      -> WHEN A SUBSTITUTION IS OPERATED ON A BLOCK, SUBSTITUTION IS SUCH THAT DIFFUSE THE BITS OF PREVIOUS BLOCK

      -> IT MAKE MORE DIFFICULT TO RECOVER P WITHOUT THE KEY

    - CONFUSION (BREADTH) : RELATIONSHIP BETWEEN K AND C HAS TO BE COMPLEX

  ---> CONFUSION + DIFFUSION ---> SUBSTITUTION + PERMUTATION

- S-BOXES (~> SUBSTITUTIONAL) :

  SMALL LUTs / K INPUT BITS ⟼ K OUTPUT BITS

  - LUT MUST BE CHOSEN CAREFULLY : SHOULD BE AS NON-LINEAR AS POSSIBLE AND NO STATISTICAL BIAS

  - PROPERTIES :

    - INVERSE MAPPING OF $\bar{X}$ IS INTRACTABLE OPERATION : $\bar{X} \longmapsto \bar{X}^{-1}$ (NON-LINEAR)
    - MATRIX ROWS ARE SHIFTED FROM EACH OTHER
    - INPUT ≠ OUTPUT, COMPLEMENT OF OUTPUT

8 BIT OUTPUT ↑ 8 BIT INPUT ↑

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

- **AES** : ~> ADVANCED ENCRYPTION STANDARD

- **MATHEMATICAL PRELIMINARIES** :
  - OPERATORS ARE DEFINED AT BYTE (8 bit) AND WORD (32 bit) LEVELS
    - -> THIS IS CALLED $GF(2^8)$: GALOIS FIELD OF $2^8 = 256$ ELEMENTS (~> ON 8 bit)
  - POLYNOMIAL REPRESENTATION : $b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$
  - PROPERTIES : ASSOCIATIVE, COMMUTATIVITY, IDENTITY, INVERSE
  - ADDITION : SUM OF COEFFICIENTS mod 2, BYTE LEVEL XOR
  - MULTIPLICATION : MULTIPLICATION mod $m(x)$ / $m(x)$: IRREDUCIBLE POLYNOMIAL (INVERTIBLE POLYNOMIAL)
  - GIVEN $c(x) = a(x)\,b(x)$, $c(x)$ COULD BE LONGER THAN 4 BYTES (SO, 4 BYTES) -> $d(x) = c(x) \bmod M(x)$

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

  - $d(x)$ CAN BE WRITTEN AS A CIRCULANT MATRIX :
    - MULTIPLICATION TRICK :

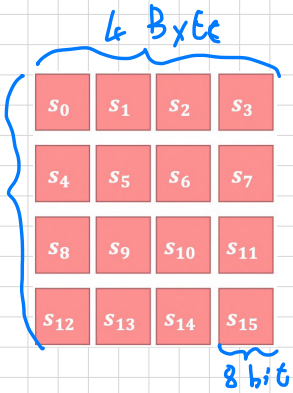
GIVEN $b(x) = b_7x^7 + \dots + b_1x + b_0$

-> $b(x) * x = b_7x^8 + b_6x^7 + b_5x^6 + \dots + b_1x^2 + b_0x$ ($b(x)\cdot x \bmod m(x)$)

  - $c(x) = b(x) \cdot x \bmod M(x)$ CAN BE WRITTEN IN A MATRIX FORM / AFTER '$\cdot x$' IS CYCLIC SHIFT OF BYTES INSIDE THE WORD

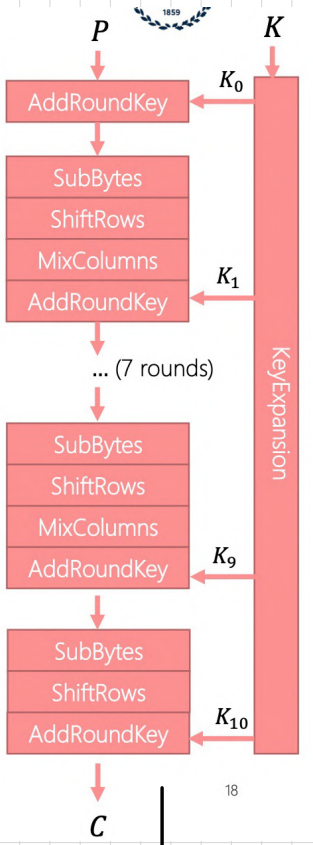
$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 00 & 00 & 00 & 01 \\ 01 & 00 & 00 & 00 \\ 00 & 01 & 00 & 00 \\ 00 & 00 & 01 & 00 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

- **AES - INTRODUCTION :**
  - GOALS : RESISTANCE AGAINST KNOWN ATTACKS, HIGH SECURITY, EFFICIENCY, SIMPLICITY
  - CORE LAYERS : LINEAR MIXING, NON-LINEAR S-BOXES APPLICATION, KEY ADDITION
  - INTERMEDIATE RESULTS ARE CALLED STATE
  - AES PROCESSES 128 bit BLOCKS $S = (S_0, S_1, \dots, S_{15})$ COMPOSED OF 4 × 4 Byte ARRAYS



  - N° OF ROUNDS : 10, 12, 14 , C PROVIDED TRANSFORMING Bytes, ROWS AND COLUMNS

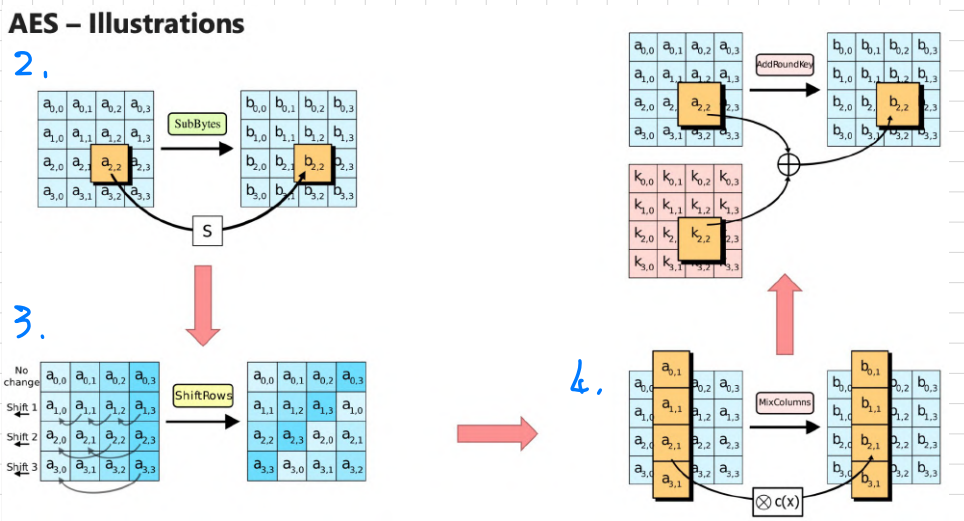
- **AES - ALGORITHM :** https://youtu.be/O4xNJsjtN6E
  - 5 CORE BLOCKS, 10 ROUND : → FOR 128 bit blocks
    1. ADD ROUND KEY: (STATE) XOR (K_i)
    2. SUB BYTES: S-BOXES BITE SUBSTITUTION → SUBSTITUTION, CONFUSION (NON-LINEAR)
    3. SHIFT ROWS : SHIFTS i-TH ROW BY i POSITIONS (CYCLICALLY) → PERMUTATION, DIFFUSION
    4. MIX COLUMNS : SHIFTS J-TH COLUMN CYCLICALLY BY J POSITIONS
    5. KEY EXPANSION : CREATES K_i -> 128 bit KEY FROM K
  - -> DECRYPTION OPERATES IN REVERSE ORDER



LAST ROUND :

NO MIX COLOMN

-> IT MAKES E,D MORE SIMILAR, WITHOUT IMPACTING SECURITY



AES BLOCK MODES:

IT HELPS TO GO FROM 128 bit MESSAGE ⟶ N LENGTH MESSAGE

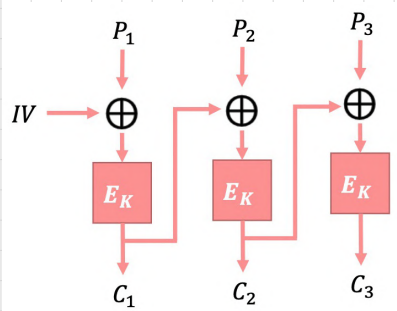
ELECTRONIC CODE BLOCK (ECB):

- SIMPLEST MODE:

1. BREAK P INTO CHUNKS OF APPROPRIATE SIZE: $P \longmapsto [P_1 | P_2 | \dots | P_N]$
2. ENCRYPT EACH CHUNK INDEPENDENTLY WITH SAME KEY: $C_i = E(P_i, k)$
3. COMBINE ALL CHUNKS FROM C_i: $C = [C_1 | C_2 | \dots | C_N]$

- COMMENTS:
  - INSECURE, NEVER USE IT
  - IF $(P_i, k) = (P_j, k) \longrightarrow C_i = C_j$ ⇝ NOT SECURE
  - IN THE LONG RUN, IF AN ATTACKER FINDS OUT SOME PAIR $(P_i ; C_i)$
    -> HE CAN BUILD UP A CODEBOOK TO DECRYPT C, WITHOUT KNOWING K

CIPHER BLOCK CHAINING (CBC)



- IT'S A MORE SECURE MODE, IT USES BLOCK CHAINING
- GIVEN AN IV (⇝ INITIALIZATION VECTOR): $C_i = E(K, P_i \oplus C_{i-1})$
- COMMENTS:
  - C_i DEPENDS ON $C_{i-1}, \dots, C_1$
  - IF IV IS RANDOM: $P_i = P_j \longmapsto C_i \neq C_j$
- DECRYPTION: IV MUST BE SENT IN THE CLEAR WITH C
- BLOCK PADDING: IF P IS NOT A MULTIPLE OF BLOCK SIZE -> PADDING ADDED TO P
  -> TO DISTINGUISH DATA / PADDING: PADDING ORACLES -> TELL US IF PADDING IS VALID / INVALID
  - BUT...

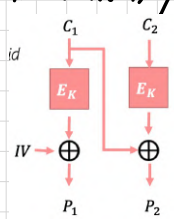
PADDING ATTACKS:

Idea: Discover P_2 by asking the oracle if our guesses are correct.

We pick a random C_1 and send $(C_1 \mid C_2)$ to the oracle.

1. *Oracle will tell us* if $C_1 \oplus P_2$ *has a valid padding, allowing us to solve for a byte of* P_2 *(i.e.,* $C_1[15] \oplus P_2[15] = \textit{Pad Byte}$*)*. If not, we change $C_1[15]$ until we obtain a positive answer.
2. We now find the value of $P_2[14]$ by setting $C_1[15] = P_2[15] + \textit{Pad Byte}$ and repeating for $C_1[14]$
3. *We iterate (2,3) until all bytes have been decrypted.*

(harder in practice)

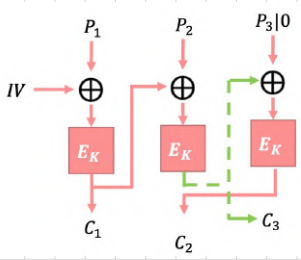


- CBC CIPHER STEALING.

  HELPS TO PROTECT AGAINST PADDING

  -> IT EXTENDS BITS OF P_i USING C_{i-1}

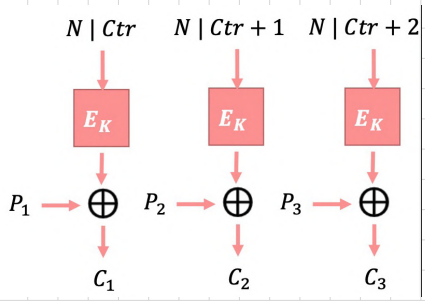
  L> COMPLEX IN PRACTICE



- COUNTER (CTR):

  ✓ ALLOW PARALLELIZATION

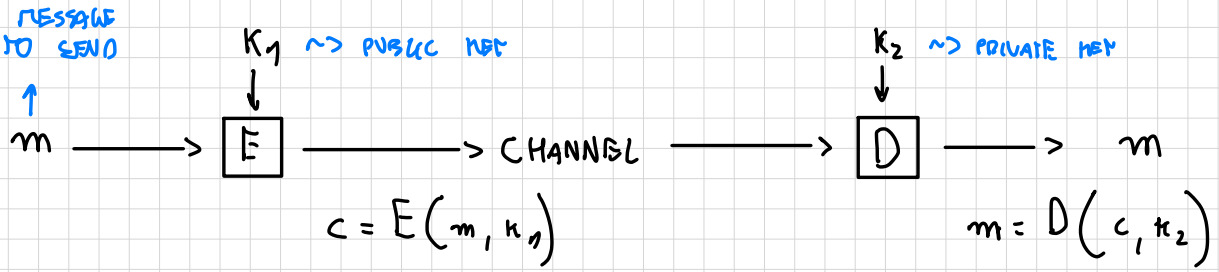
  WE CAN ENCRYPT SINGLE BIT AND OBTAIN PSEUDO-RANDOM BITS, USING A NONCE (N) AND A COUNTER (Ctr)



- N MUST NEVER BE REUSED WITH SAME K FOR DIFFERENT M

- **ASYMMETRIC ENCRYPTION:**

  - $\forall$ USER -> 1 PUBLIC KEY (K_1) + 1 PRIVATE KEY (K_2)



- TRAPDOOR 1-WAY FUNCTION:

  FUNCTION FAST TO COMPUTE BUT DIFFICULT TO BE INVERTED, UNLESS YOU HAVE SOME INFORMATION

  eg. INTEGER FACTORIZATION (RSA), DISCRETE log, ELLIPTIC CURVES, ERROR CORRECTING CODES

- **RSA:**

  FOR RSA ENCRYPTION, SOME PARAMETERS / FUNCTIONS NEED TO BE USED:

  - p, q : 2 PRIME NUMBERS
  - $N = p \cdot q$
  - **EULER FUNCTION:**

    - $\phi(p)$: n° INTEGERS i THAT ARE COPRIME WITH p / $1 \leq i \leq p$

      L> t IS COPRIME WITH p <=> $GCD(p, t) = 1$

      eg. $GCD(6,1) = 1$, $GCD(6,5) = 1$

      $\phi(6) = 2$, $\because \{1, 2, 3, 4, 5\} < 6$

- IF p IS PRIME $\rightarrow \phi(p) = p-1$
- IF p, q PRIME, $N = p \cdot q$:

  $\rightarrow \phi(N) = \phi(p)\,\phi(q) = (p-1)(q-1)$

- $e \;/\; GCD(e, \phi(N)) = 1$
- $d \;/\; e \cdot d = 1 \mod \phi(N)$

$\longrightarrow$ PUBLIC KEY : (e, N)

$\longrightarrow$ PRIVATE KEY : (d, N)

- ENCRYPTION :
  - m TO SENT : $c = E(m, e) = m^e \mod N$
- DECRYPTION :
  - c RECEIVED : $m = D(c, d) = c^d \mod N$

es.

$p = 7, \; q = 11 \rightarrow N = p \cdot q = 77 \rightarrow \phi(N) = (p-1)(q-1) = 60$

$e = 17, \; d = 53$

$\rightarrow$

- $m = 2 \longmapsto c = m^e \mod N = 2^{17} \mod 77 = 18$
- $c = 18 \longmapsto m = c^d \mod N = 18^{53} \mod 77 = 2$

- ALGORITHM SECURITY :

TO COMPUTE d IS NECESSARY TO KNOW $\phi(N)$

$\rightarrow$ IF I KNOW N ONLY $\rightarrow p, q$ DIFFICULT TO COMPUTE

- IF I KNOW p, q : $p, q \rightarrow \phi(N) = (p-1)(q-1) \rightarrow e\,d = 1 \mod \phi(N) \mapsto d$

Mc ELIECE :

MAIN CONCEPTS TO KNOW :

GALOIS FIELD OF K

CODE :

MAPPING OF A K BIT VECTOR (MESSAGE) $\bar{v} \in F^K$ TO A VECTOR $\bar{c} \in F^n$ / $n \geq K$

- REDUNDANCIES : $r = n - K$

-> MAP : $F^K \longrightarrow F^n$ / $\forall \bar{v}_i \rightarrow \exists! \bar{c}_i$

- MAPPING IS DONE USING A $n \times K$ MATRIX G

-> $\bar{c} = \bar{v} \cdot G$ / $\bar{v}$: K BITS, $\bar{c}$: n BITS

es.

$K = 2$, $n = 5$, $G = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$

->

$K = 2$ -> $\bar{v} \in F^2 = \{(0,0), (0,1), (1,0), (1,1)\}$

- $\bar{v}_1 = (0\ 0)$: $\begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix} = [0\ 0\ 0\ 0\ 0]$ (SUM ONLY ROWS = 1)
- $\bar{v}_2 = (0\ 1)$: $\begin{bmatrix} 0 \\ 1 \end{bmatrix} \cdot G = [1\ 1\ 1\ 0\ 0]$
- $\bar{v}_3 = (1\ 0)$: $\begin{bmatrix} 1 \\ 0 \end{bmatrix} \cdot G = [1\ 0\ 1\ 1\ 1]$
- $\bar{v}_4 = (1\ 1)$: $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \cdot G = [0\ 1\ 0\ 1\ 1]$

} $\bar{c}_i$

- CODEBLOCK : SET OF ALL $\bar{c}_i$ / $|C| = 2^K$ (CODEBLOCK)

- **HAMMING WEIGHT**: $w_H(\bar{c}) = n^\circ$ BITS $= 1$ OF $\bar{c}$
- **HAMMING DISTANCE**: $d_H(\bar{c}_1, \bar{c}_2) = n^\circ$ DIFFERENT BITS IN $\bar{c}_1$ AND $\bar{c}_2$
  - PROPERTY: $d_H(\bar{c}_1, \bar{c}_2) = w_H(\bar{c}_1 + \bar{c}_2)$
  - MINIMUM DISTANCE: $d_{MIN} = \min_{\substack{\bar{c}_1, \bar{c}_2 \in C \\ \bar{c}_1 \neq \bar{c}_2}} \{ d_H(\bar{c}_1, \bar{c}_2) \}$
    - PROPERTY: $d_{MIN} = \min_{\substack{\bar{c}_i \in C \\ \bar{c}_i \neq \bar{0}}} \{ w_H(\forall \bar{c}_i) \}$
- **ERROR VECTOR**:

$$\bar{e} \;/\; \begin{cases} \bar{e}_i = 0 \iff \bar{r}_i = \bar{c}_i \\ \bar{e}_i = 1 \iff \bar{r}_i \neq \bar{c}_i \end{cases}$$

  - $\bar{r} = \bar{c} + \bar{e}$
  - HOW TO RECOVER ORIGINAL MESSAGE $\bar{v}$?

$$\rightarrow \hat{\bar{c}} = \min \{ d_H(\bar{r}, \bar{c}) \}$$

IDEA UNDER M. ELIECE

$\rightarrow$ IF $k \gg 0$ $\rightarrow 2^k$ CODEWORDS $\rightarrow$ DIFFICULT TO COMPUTE $\hat{\bar{c}}$

- **ERROR CORRECTION CAPABILITY**:

GIVEN A CODE $C(n, k)$, d_{MIN}

$\rightarrow$ WE ARE ONLY ABLE TO CORRECT UP TO t BITS

$$t = \frac{1}{2} d_{MIN} - 1$$

$\rightarrow \bar{e} \;/\; w_H(\bar{e}) \leq t$ ARE CORRECT

d_{MIN}	t
3	1
4	1
5	2
6	2
7	3

PARITY CHECK MATRIX:

IT CHECKS IF A GIVEN CODE $\bar{v} \in C$ OR NOT

- $n \times r$ MATRIX H / $\bar{s} = \bar{v} H$, $C(n,k)$ / $r = n - k$
  - $\bar{s}$ · SYNDROME ⟶ $\bar{s} = \bar{0}$ ⟺ $\bar{v} \in C$
- H NEEDS 2 PROPERTIES:
  1. $GH = 0$ ⟶ $\because \bar{c} = \bar{v} G$ → $\bar{c} H = 0$
  2. $\text{rank}(H) = r$ → SINGULAR MATRIX

SYNDROME DECODING:

GIVEN: $\begin{cases} TX: & \bar{c} \\ RX: & \bar{v} = \bar{c} + \bar{e} \end{cases}$

→ COMPUTE $\bar{s} = \bar{v} H = \bar{c} \underbrace{H}_{=0} + \bar{e} H$ ⇝ $\bar{s}$!(:) $\bar{e}$

- IF WE KNOW $\bar{e}$ → WE CAN RECOVER $\bar{c} = \bar{v} + \bar{e}$, WHEN THAT $\bar{v} = \bar{c} + \bar{e}$
- WE CAN CREATE A LUT / CONTAINS PAIR $(\bar{s} / \bar{e})$, CONSIDERING $\forall \bar{e} / w_H(\bar{e}) \leq t$, AND $\bar{s} = \bar{e} H$

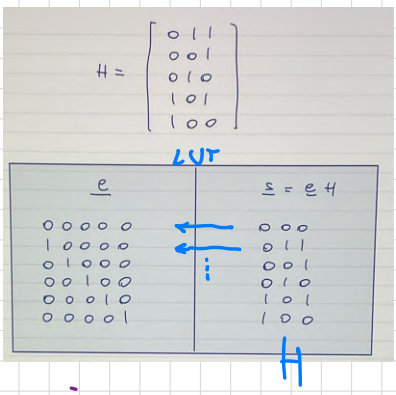
EX.

$k = 2$, $n = 5$

$d_{MIN} = 3$ → $t = 1$

→

1. $\bar{e} / w_H(\bar{e}) \leq 1$ ARE CONSIDERED
2. $\bar{e} \longmapsto \bar{s} = \bar{e} H$



→ SUPPOSE $\bar{v} = 11111$, $\bar{c} = 10111$, $\bar{e} = 01000$

- $\bar{v} H = 001 = \bar{s}$
- LUT: $\bar{s} \longmapsto \bar{e} = 01000$ → $\bar{c} = \bar{v} + \bar{e} = 10111$

- **Mc ELIECE CRYPTOSYSTEM :**

  - WE KNOW G, H, t
  - WE PUBLIC AN OBFUSCATED VERSION OF G . $G_1 = S \cdot G \cdot P$
    - S : RANDOM NON-SINGULAR MATRIX , DIM : $K \times K$
    - P : PERMUTATION MATRIX

    -> WE CANNOT COMPUTE G WITHOUT S, P
  - WE SELECT $t_1 \leq t$

  -> PUBLIC KEY = (G_1, t_1)

  - **ENCRYPTION :**

    1. m TO SENT , I RECOVER (G_1, t_1) ($n \times k$) OF RECIEVER FROM PUBLIC DIRECTORY
    2. m DIVIDED INTO VECTORS $\bar{v}$ OF K BITS
    3. $\bar{c} = \bar{v} G_1$
    4. GENERATE RANDOM $\bar{e} \;/\; w_H(\bar{e}) = t_1$
    5. CIPHERTEXT : $\bar{y} = \bar{c} + \bar{e}$

  - **DECRYPTION :**

    1. $\bar{y}$ IS RECIEVED $\longrightarrow$ $\bar{y}_1 = \bar{y} P^{-1}$
    2. GIVEN G, H KNOWN : SYNDROME DECODING -> $\bar{y}_1 \longmapsto \bar{c}_1$
    3. $\bar{v}_1 = \bar{c}_1 \cdot G^{-1}$
    4. RECOVER SENT MESSAGE : $\hat{\bar{v}} = \bar{v}_1 \cdot S^{-1}$

20.

$k = 2 \quad n = 5$

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

$$S = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \rightarrow S^{-1} = \dots \quad \text{↳ inv(GF(S))}$$

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \rightarrow P^{-1} = \dots$$

$$G_1 = SGT = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix}$$ PUBLIC k, $t_1 = 1$

MESSAGE TO SEND: $\bar{v} = 11$

- **ENCODING:**
  - $\bar{c} = \bar{v} G_1 = 10101$
  - $\bar{e} = 00100$, GENERATED RANDOMLY ($t_1 = 1$)
  - CIPHER TEXT: $\bar{r} = \bar{c} + \bar{e} = 10001$
- **DECODING:**
  - $\bar{r}_1 = \bar{r} P^{-1} = 10001 \cdot P^{-1} = 10100$
  - $\bar{s} = \bar{r}_1 H = 001$ → LUT → $\hat{\bar{e}} = 01000$ → 2ND BIT WRONG
    - ↳ CAN BE BUILT AS WISHED
  - $\hat{\bar{c}} = \bar{r} + \hat{\bar{e}} = 11100$
  - $\bar{v}_1 = \text{GFLINEQ}(G, c_1) = 01$, ∵ $\bar{c}_1 = \bar{v}_1 G \rightarrow \bar{v}_1 = \dots$
  - $\bar{v} = \bar{v}_1 S^{-1} = 11 \rightsquigarrow$ ORIGINAL MESSAGE

$$[1\,1\,1\,0\,0] = \bar{v}_1 \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$