Compression, Streaming, Interaction Vidéo

Cours n°1 : Théorie de l’information

1. Codage source

2. Débit vidéo

3. Quantité d’information

4. Entropie

5. Longueur d’un signal

6. Codage RLE / Huffman

7. Taux de compression

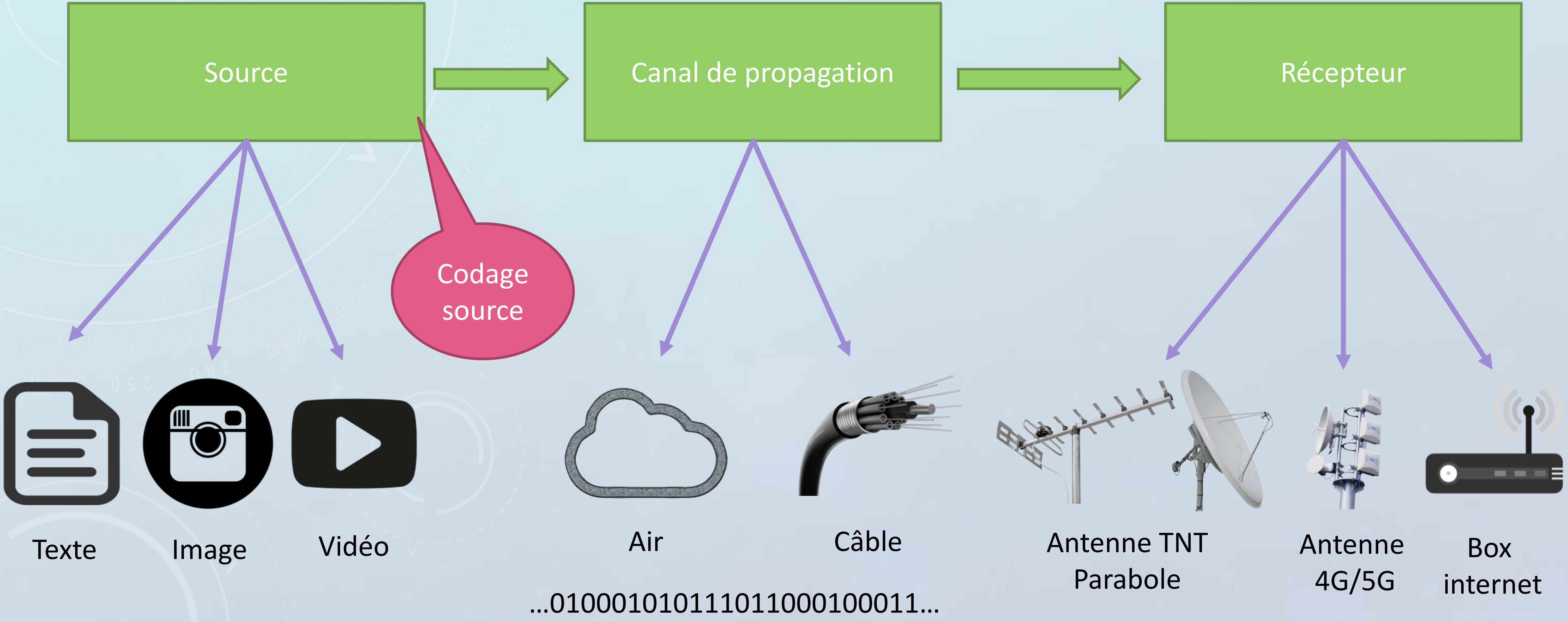
1. Redondances spatio-temporelles

2. Décorrélation

3. Quantification

4. Qualité de reconstruction

1. Codage source

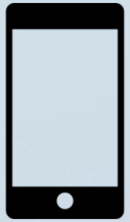
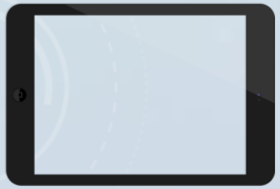


1. Codage source

Des besoins (décodeurs)
Ventes en 2021

210 $\overline{\text{M}}$ | 341 $\overline{\text{M}}$



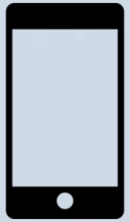
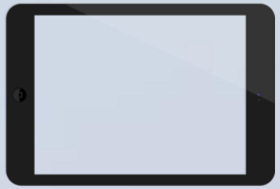



1,35 $\overline{\overline{\text{M}}}$ | 169 $\overline{\text{M}}$

Des besoins (décodeurs)
Ventes en 2022

202 $\overline{\text{M}}$ | 293 $\overline{\text{M}}$



1,21 $\overline{\overline{\text{M}}}$ | 151 $\overline{\text{M}}$

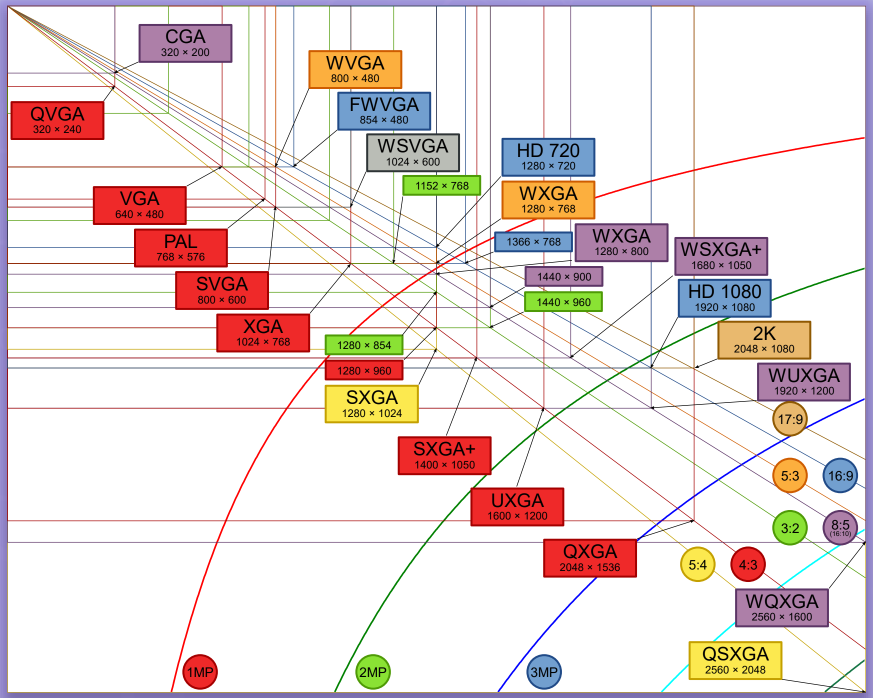
1. Codage source

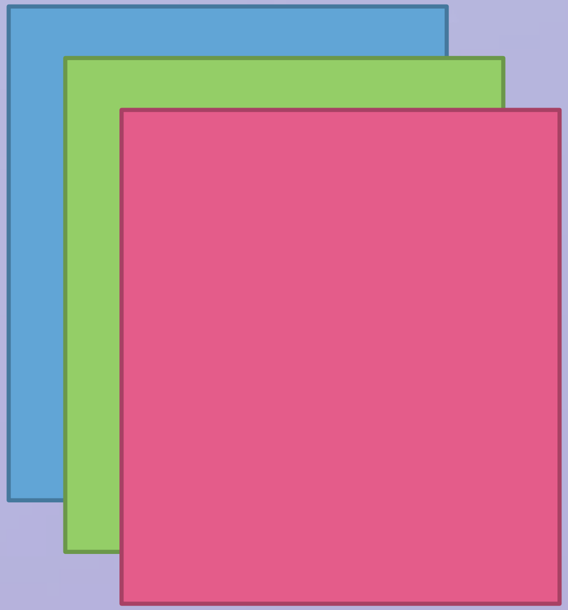
Du contenu (encodeurs) : films, musiques, sport …

2. Débit vidéo

Formats d’image et poids d’image

Formats / Ratios / Poids





Quel poids par pixel ?
- 8/24 bits (en général)

Quelle taille ?
- FHD (2K) : 1080 x 1920
- UHD (4K) : 2160 x 3840

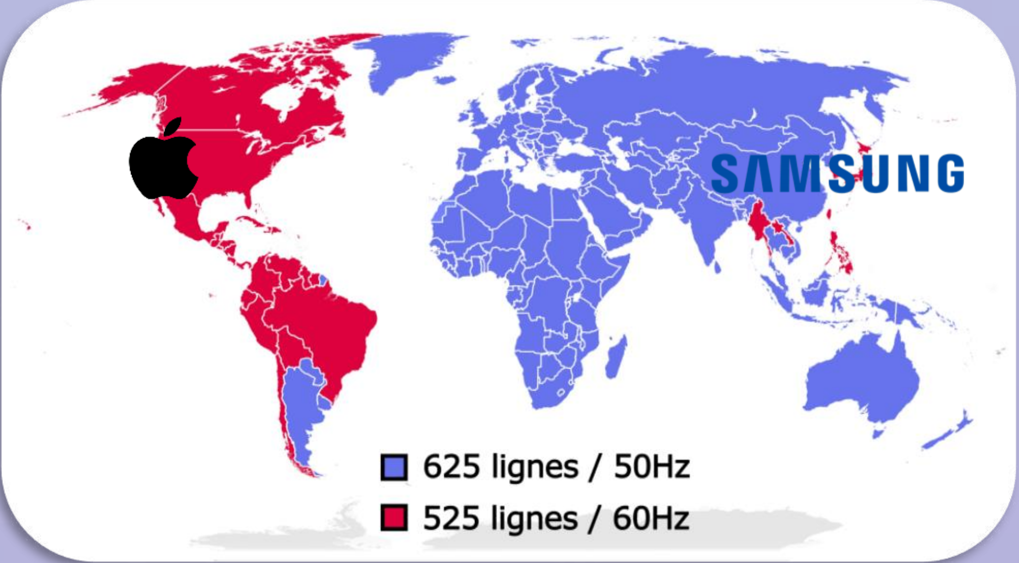
Combien de canaux ?
- 3 : Rouge, Vert, Bleu

Image UHD-RVB : 24 Mo !

2. Débit vidéo

Image → Vidéo

Fréquences des courants alternatifs dans le monde



Formats pour la télévision analogique directement liés aux fréquences :

Format	Année	Lieu	Caractéristiques
480i	1941	USA	525 lignes à 30 i/s
576i	1948	URSS	625 lignes à 25 i/s

Introduction du HFR (High Frame Rate) :
- Le Hobbit (50 i/s) et Avatar 2 (60 i/s)

2. Débit vidéo

➔ A quel débit prévisionnel (en Mo/s, ou plus…) s'attendre pour Avatar 2 en UHD-HFR 3D ?

- Environ 3 Go/s !

➔ Quel poids total (en Mo/s, ou plus…) serait pour stocker Avatar 2 en UHD-HFR 3D (durée 3h) ?

- Environ 32 To !

IMPOSSIBLE EN PRATIQUE !!!

Objectif :

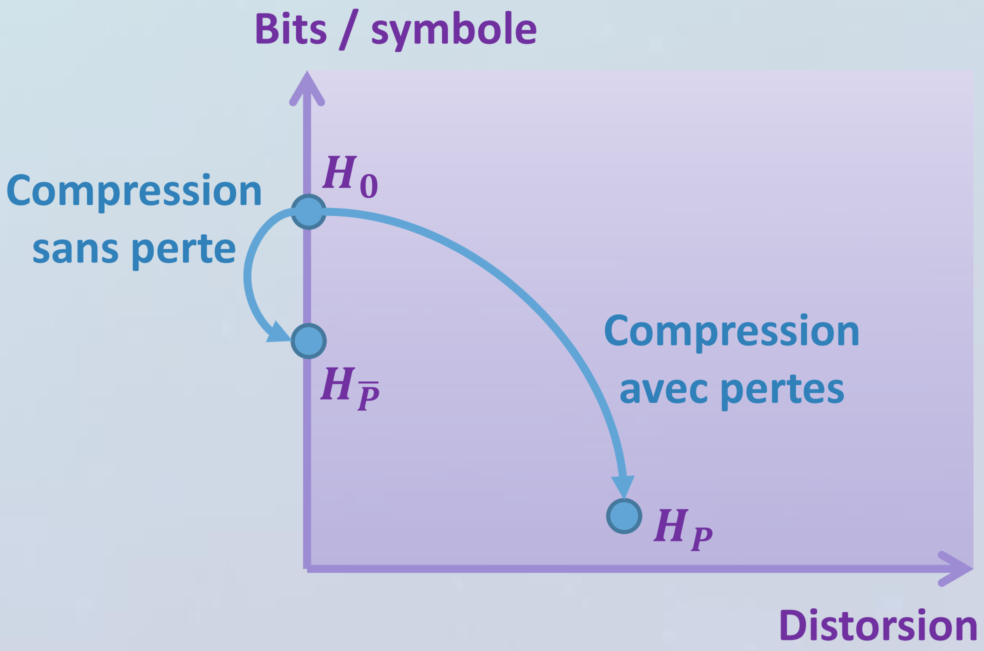
Coder la source pour diminuer la quantité de données à transmettre tout en restant certain de bien la décoder à la réception.

2. Débit vidéo

Quelle solution ? Réduire la quantité de données à envoyer

Nouveau problème : quelle qualité pour le nouveau signal ?

- ➔ Optimisation **débit-distorsion**
- ➔ Débit = nombre de bits par seconde
- ➔ Distorsion = mesure de perte de qualité
- ➔ Réduire le débit en réduisant le nombre de bits par symbole (ex : une couleur)



3. Quantité d'information

S = $\{S_k\}$ = {A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P}

A	B	E	F
C	D	G	H
I	J	M	N
K	L	O	P

Je choisis une lettre parmi les 16.

Combien de questions faut-il me poser au minimum pour être sûr de trouver la lettre choisie ?
(Mes seules réponses sont 'Oui' ou 'Non')

A	B	E	F
C	D	G	H
I	J	M	N
K	L	O	P

La réponse est $4 = \log_2(\#\boldsymbol{S})$.

Chaque question permet de différencier des groupes de symboles jusqu'à différencier chacun d'eux.

On définit ici la **quantité d'information $\boldsymbol{I}$** apportée par un symbole par :

$$\boxed{\boldsymbol{I(S_k) = \log_2(\#S)}}$$

3. Quantité d'information

$S = \{S_k\} = \{A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P\}$

$S = \{S_k\} = \{A,B,C,D,I\}$ $S = \{S_k\} = \{M,N,O,P\}$

A	B	E	F
C	D	G	H
I	J	M	N
K	L	O	P

Je choisis une lettre parmi les 16, qui est dans S.

Quel est le coût de l'information pour qu'un joueur sache que la lettre est dans S ?

En réutilisant la formule précédente, en sachant que la lettre est dans S, on a l'information I :

$$I(S_k) = \log_2(\#S)$$

Le coût de l'information I pour qu'un joueur sache que la lettre est dans S est alors :

$$I(S) = \log_2(\#S) - \log_2(\#S)$$

$$\boxed{I(S) = -\log_2\left(\frac{\#S}{\#S}\right)}$$

4. Entropie

$S = \{S_k\}$ = {A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P}

A	B	E	F
C	D	G	H
I	J	M	N
K	L	O	P

$$I(S_k) = \log_2(\#S) = -\log_2\left(\frac{1}{\#S}\right)$$

Fréquence d'apparition f_k du symbole S_k

L'entropie H de S est définie comme la moyenne pondérée des informations de chaque symbole différent :

$$\boxed{H(S) = -\sum_{S_k \in S} f_k \log_2(f_k)}$$

Son unité est le bit d'information (ou Shannon), qui correspond **au plus petit nombre de bits en moyenne pour coder un symbole que l'on pourra déchiffrer à coup sûr**. Ici les symboles sont équitablement répartis donc :

$$H(S) = -\sum_{S_k \in S} \frac{1}{\#S} \log_2\left(\frac{1}{\#S}\right) = \log_2(\#S) = 4$$

4. Entropie

$S' = \{S_k\} =$ {A,A,A,A,E,F,G,H,A,J,K,L,M,M,M,M}

A	A	E	F
A	A	G	H
A	J	M	M
K	L	M	M

Ici la répartition n'est plus équitable, mais il reste cependant 9 symboles que l'on devrait toujours coder sur 4 bits.

Mais qu'en est-il de l'entropie de S' ?

Calcul des informations I de A et M :

$$I(\mathrm{A}) = -\log_2\left(\frac{5}{16}\right) \approx 1,68$$

$$I(\mathrm{M}) = -\log_2\left(\frac{4}{16}\right) = 2$$

Calcul de l'entropie H de S' :

$$H(S') = -\sum_{S_k \in S'} f_k \log_2(f_k) = 2,77$$

$\ll 4$

4. Entropie

S'' = $\{S_k\}$ = {A,A,A,A,E,F,G,H,A,A,A,A,M,N,O,P}

A	A	E	F
A	A	G	H
A	A	M	N
A	A	O	P

Ici la répartition est encore moins équitable, mais avec toujours 9 symboles différents.

Que vaut l'entropie de S'' ?

Calcul de l'information I de A et des autres :

$$I(\mathrm{A}) = -\log_2\left(\frac{8}{16}\right) = 1$$

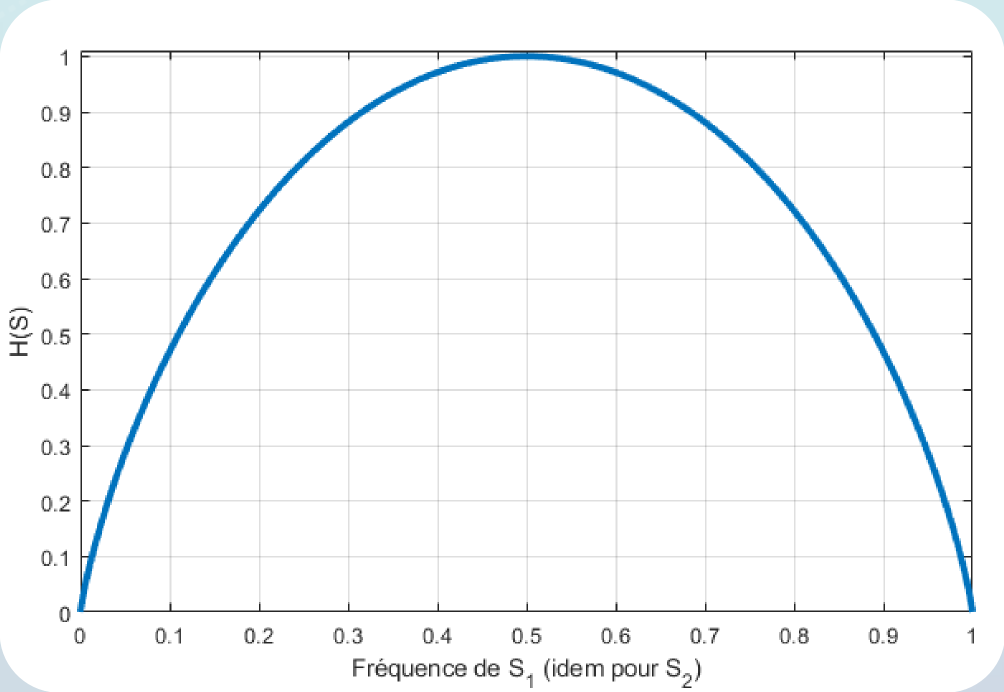
$$I(\mathrm{autres}) = -\log_2\left(\frac{1}{16}\right) = 4$$

Calcul de l'entropie H de S'' :

$$H(S'') = -\sum_{S_k \in S''} f_k \log_2(f_k) = 1,73$$

4. Entropie

Evolution de l'entropie en fonction de la fréquence d'apparition d'un symbole (cas à 2 symboles)



➔ L'entropie est maximale pour une distribution uniforme. Elle diminue d'autant plus que la distribution se concentre sur certains symboles.

<u>Code Uniquement Déchiffrable :</u>

➔ Une seule décomposition de suite de symboles

<u>Code Préfixe :</u>

➔ Aucun symbole n'est le début d'un autre
➔ Préfixe implique Uniquement Déchiffrable

5. Longueur d’un signal

<u>Longueur d’un symbole :</u>

La longueur d’un symbole $\boldsymbol{S_k}$ est notée $\ell(\boldsymbol{S_k})$ et est égale au nombre de bits nécessaires pour coder ce symbole.

<u>Longueur moyenne d’un signal $\boldsymbol{S}$:</u>

Il s’agit de la moyenne des longueurs des symboles $\boldsymbol{S_k}$ pondérée par leurs fréquences d’apparition.

$$\bar{\ell}(\boldsymbol{S}) = \sum_{\boldsymbol{S_k} \in \boldsymbol{S}} \boldsymbol{f}_k\, \ell(\boldsymbol{S_k}) = -\sum_{\boldsymbol{S_k} \in \boldsymbol{S}} \boldsymbol{f}_k \log_2\left(2^{-\ell(\boldsymbol{S_k})}\right)$$

On parle aussi de code pour désigner un signal, d’où la notion de codage par la suite.

Le codage entropique, basé sur le calcul de l’entropie d’un signal $\boldsymbol{S}$ n’est que théorique. La longueur moyenne $\bar{\ell}$ des symboles d’un code n’arrive que très rarement à atteindre l’entropie :

$$\boldsymbol{H}(\boldsymbol{S}) \leq \bar{\ell}(\boldsymbol{S})$$

Cependant, il existe toujours un code donc la longueur moyenne $\bar{\ell}$ des symboles vérifie :

$$\bar{\ell}(\boldsymbol{S}) \leq \boldsymbol{H}(\boldsymbol{S}) + \mathbf{1}$$

5. Longueur d'un signal

Comment obtient-on ses inégalités ?

Pour un code $\boldsymbol{S}$ Préfixe (respectivement UD), on a le théorème de Kraft (respectivement McMillan) qui nous dit que les différentes longueurs $\ell(\boldsymbol{S}_k)$ de ce code vérifient :

$$\boldsymbol{L} = \sum_{\boldsymbol{S_k \in S}} \boldsymbol{2}^{-\ell(\boldsymbol{S}_k)} \leq \boldsymbol{1}$$

On introduit ensuite les fréquences suivantes :

$$\boldsymbol{f} = \{\boldsymbol{f}_k\} = \frac{\boldsymbol{2}^{-\ell(\boldsymbol{S}_k)}}{L} \quad \text{et} \quad \boldsymbol{g} = \{\boldsymbol{g}_k\}$$

qui sont normalisées (i.e. qui somment à 1).

On utilise enfin la formule de la divergence de Kullback-Leibler (ou entropie relative), qui est toujours positive (admis), puis on développe :

$$D(\boldsymbol{g}\|\boldsymbol{f}) = \sum_{\boldsymbol{S_k \in S}} \boldsymbol{g}_k \, \mathbf{log_2}\left(\frac{\boldsymbol{g}_k}{\boldsymbol{f}_k}\right) \geq \mathbf{0}$$

5. Longueur d'un signal

On développe la formule de la divergence de Kullback-Leibler :

$$D(\boldsymbol{g}\|\boldsymbol{f}) = \sum_{S_k \in S} \boldsymbol{g}_k \log_2\left(\frac{\boldsymbol{g}_k}{\boldsymbol{f}_k}\right) \geq 0$$

…

$$D(\boldsymbol{g}\|\boldsymbol{f}) = \sum_{S_k \in S} \boldsymbol{g}_k \log_2(\boldsymbol{g}_k) - \sum_{S_k \in S} \boldsymbol{g}_k \log_2(\boldsymbol{f}_k)$$

…

$$D(\boldsymbol{g}\|\boldsymbol{f}) = -\boldsymbol{H}(\boldsymbol{S}) - \sum_{S_k \in S} \boldsymbol{g}_k \log_2\left(2^{-\ell(S_k)}\right) + \sum_{S_k \in S} \boldsymbol{g}_k \log_2(\boldsymbol{L})$$

…

$$D(\boldsymbol{g}\|\boldsymbol{f}) = -\boldsymbol{H}(\boldsymbol{S}) + \bar{\ell}(\boldsymbol{S}) + \log_2(\boldsymbol{L})$$

…

$$\boxed{\Rightarrow \bar{\ell}(\boldsymbol{S}) \geq \boldsymbol{H}(\boldsymbol{S}) - \log_2(\boldsymbol{L}) \geq \boldsymbol{H}(\boldsymbol{S})}$$

En gardant la formule avec les sommes pour cette première inégalité, on a :

$$\sum_{S_k \in S} \boldsymbol{g}_k\, \ell(\boldsymbol{S}_k) \geq -\sum_{S_k \in S} \boldsymbol{g}_k \log_2(\boldsymbol{g}_k)$$

Soit :

$$\ell(\boldsymbol{S}_k) \geq -\log_2(\boldsymbol{g}_k)$$

En prenant la partie entière supérieure pour chacun des $\ell(\boldsymbol{S}_k)$, on obtient alors :

$$\ell(\boldsymbol{S}_k) = \lceil -\log_2(\boldsymbol{g}_k) \rceil \leq -\log_2(\boldsymbol{g}_k) + \mathbf{1}$$

On passe ensuite à la moyenne pour avoir la seconde inégalité:

$$\boxed{\Rightarrow \bar{\ell}(\boldsymbol{S}) \leq \boldsymbol{H}(\boldsymbol{S}) + \mathbf{1}}$$

6. Codage RLE / Huffman

But de l'encodage pour la compression

➔ Diminuer le nombre de bits pour coder un signal donné

Comment ?

➔ En regroupant les mêmes symboles entre eux

➔ En faisant en sorte que les symboles les plus présents coûtent moins cher en bits

Codage RLE (Run-Length Encoding)

Principe : Remplacer plusieurs symboles identiques qui se suivent par le nombre de fois le symbole puis le symbole.

Remarque : Il faut ajouter une signalisation devant chaque couple nombre/symbole.

S = **{A,A,A,A,A,B,B,C,C,C,C,A,A,A,D,D}**

S_{RLE} = **{*5A,*2B,*4C,*3A,*2D}**

Défaut : Efficace à partir de 3 symboles ou plus identiques qui se suivent.

6. Codage RLE / Huffman

Codage de Huffman

<u>Principe :</u> Créer un arbre binaire pour regrouper les symboles les moins fréquents au plus loin de la racine et ainsi diminuer le nombre de bits de codage pour les symboles les plus fréquents.

<u>Défaut :</u> Codage en nombre entier de bits.

S = {A,O,A,O,A,B,B,N,N,N,O,N,A,A,A,O,D,O,D,O}

Symbole	**A**	**B**	**D**	**N**	**O**
Fréquence	**0,30**	**0,10**	**0,10**	**0,20**	**0,30**

6. Codage RLE / Huffman

Codage de Huffman

A = ‘11’
O = ‘10’
N = ‘01’
B = ‘001’
D = ‘000’

De base, il aurait fallu 3 bits pour chaque symbole.

➔ Le codage de Huffman est Uniquement Déchiffrable

➔ Le codage de Huffman est même Préfixe

S = {0011001000}

S = {00111010001101111}

S = {1100111010001001}

7. Taux de compression

$\boldsymbol{H_0}$ = 8 bits/pixel (ou 24 en considérant les 3 canaux)
$\boldsymbol{N_O}$ = le nombre de pixels dans l'image
$\boldsymbol{P_0}$ = poids d'origine de l'image (sans compression)

$$\boldsymbol{P_0 = H_0 \times N_0}$$

$\boldsymbol{H_C}$ = entropie de l'image compressée
$\boldsymbol{N_C}$ = le nombre de pixels de l'image compressée
$\boldsymbol{P_C}$ = poids de l'image compressée

$$\boldsymbol{P_C = H_C \times N_C}$$

On définit le taux de compression $\boldsymbol{\tau}$ par l'expression :

$$\boxed{\boldsymbol{\tau} = 1 - \frac{\boldsymbol{P_C}}{\boldsymbol{P_0}}}$$

Exemple

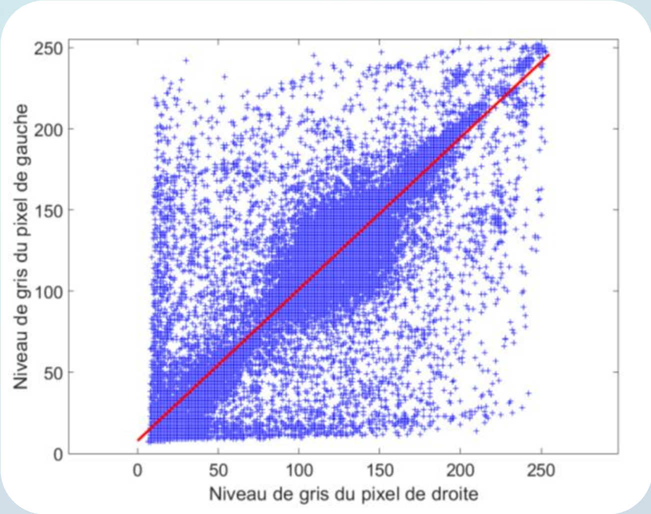
$\boldsymbol{P_0} = 64$ ko et $\boldsymbol{P_C} = 56$ ko
(compression avec codage entropique)

$\rightarrow \boldsymbol{\tau} = 13\%$ …

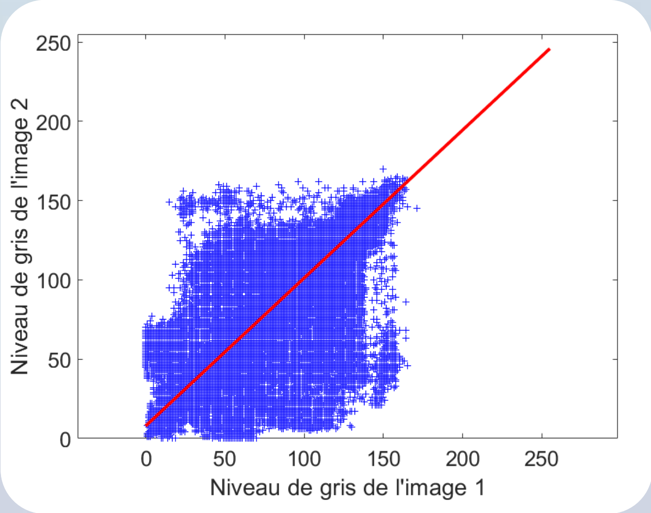
Le résultat n'est pas très impressionnant.
Peut-on faire mieux ? Et comment ?

1. Redondances spatio-temporelles

Il existe une corrélation entre pixels voisins dans une image (cf. TP de Probabilités en 1SN).

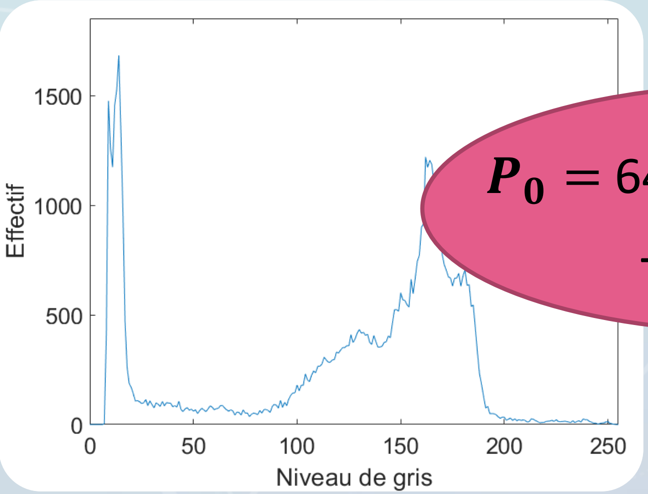


On retrouve aussi de la corrélation entre images successives.



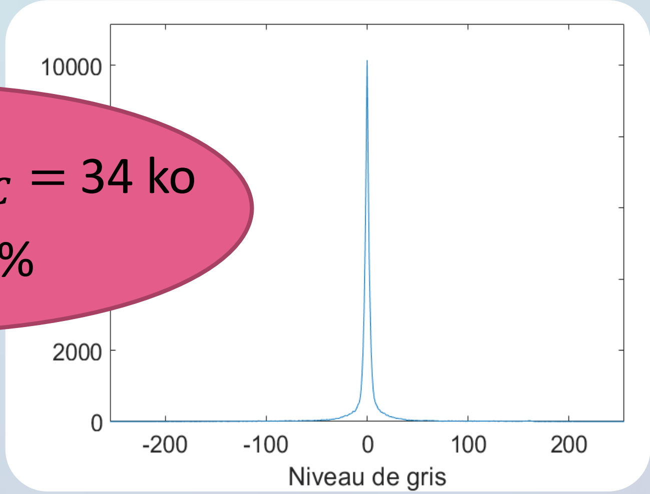
2. Décorrélation

Différence entre colonnes



P_0 = 64 ko et P_C = 34 ko

$\rightarrow \tau$ = 47%



Différence entre images

On se retrouve dans une situation plus favorable pour obtenir une entropie faible et donc une meilleure compression.

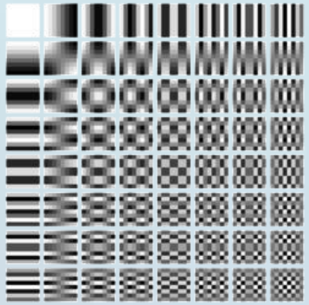
2. Décorrélation

D'autres transformations pour décorréler les pixels

Transformations spatiales :

→ Transformée de Fourier (FFT)

→ Transformée en Cosinus (DCT)



JPEG (cours n°2 + TPs)

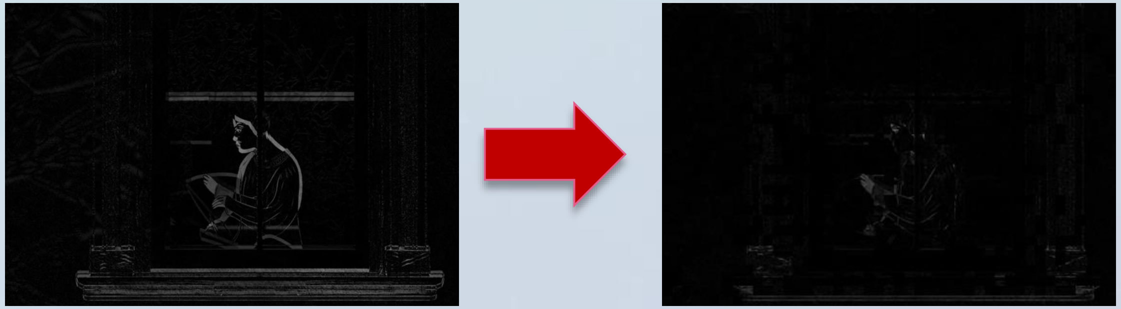
→ Transformée en Ondelettes (DWT)

JPEG2000 (cours n°4)

Compensation du mouvement + résidu de prédiction :

→ Méthodes variationnelles

→ Block Matching



MPEG-2 (cours n°3 + TPs)

→ Réseau de neurones (FlowNet, RAFT …)

3. Quantification

Les transformées vers le domaine fréquentiel concentrent l'énergie dans les basses fréquences.

Avantage :

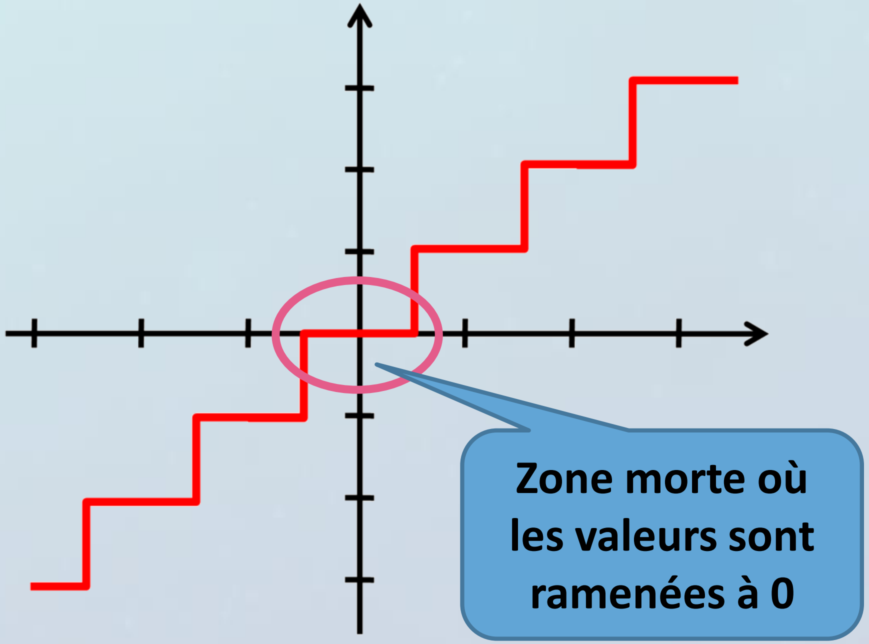
→ L'œil humain est moins sensible aux hautes fréquences donc on voudrait mettre les hautes fréquences à 0.

Inconvénient :

→ Les valeurs issues ces transformées sont décimales donc il y a concentration sur un intervalle autour de 0 mais sans être 0.

→ D'un point de vue entropique, il y a plein de valeurs différentes et longues à coder.

Quantification uniforme



$\boldsymbol{P_0} = 64$ ko et $\boldsymbol{P_C} = 7{,}3$ ko

$\rightarrow \boldsymbol{\tau} = 89\%$

4. Qualité de reconstruction

La quantification est l'étape irréversible de la compression. On perd donc en qualité lorsque l'on veut visualiser l'image après décodage.

Comment estimer la qualité de la reconstruction après décodage ?

→ Métriques de qualités relatives :

PSNR (Peak Signal to Noise Ratio)

SSIM (Structural SIMilarity)

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{d^2}{\|I_c - I_0\|^2}\right)$$

$$\mathrm{SSIM} = \underbrace{l(I_c, I_0)}_{\text{luminance}} \times \underbrace{c(I_c, I_0)}_{\text{contraste}} \times \underbrace{s(I_c, I_0)}_{\text{structure}}$$

$$\mathrm{SSIM} = \frac{(2\mu_{I_c}\mu_{I_0} + c_1)(2\sigma_{I_c}\sigma_{I_0} + c_2)(cov_{I_cI_0} + c_3)}{(\mu_{I_c}^2 + \mu_{I_0}^2 + c_1)(\sigma_{I_c}^2 + \sigma_{I_0}^2 + c_2)(\sigma_{I_c}\sigma_{I_0} + c_3)}$$

4. Qualité de reconstruction

Courbe PSNR / Poids en fonction de la quantification