Principes du codage sans perte

Codage d'Huffman, Lempel-Ziv, arithmétique

Marco Cagnazzo

Département Traitement du Signal et des Images
TELECOM ParisTech

11 Janvier 2013

Plan

1. Principes
2. Codage optimale
   - Huffman
   - Codage arithmétique
   - Codage adaptive et basé contexte
3. Autres Techniques
   - Lempel-Ziv
   - Run Length
   - JBIG
4. Quantification avec contrainte entropique

Plan

1 Principes

2 Codage optimale
- Huffman
- Codage arithmétique
- Codage adaptive et basé contexte

3 Autres Techniques
- Lempel-Ziv
- Run Length
- JBIG

4 Quantification avec contrainte entropique

Introduction

La compression sans perte est basée sur les statistiques des données

- Mots de code **courts** pour les symboles **probables**
- Mots de code **longs** pour les symboles **peu probables**

Définitions :

Alphabet : $\mathcal{X} = \{x_1, x_2, \ldots, x_M\}$ ensemble des symboles à coder

- $\{0, 1, \ldots, 255\}$ dans le cas de valeurs de luminance
- alphabet français dans le cas d'un texte

Code : application entre $\mathcal{X}$ et $\{0, 1\}^*$, l'ensemble des chaînes de bits de longueur finie.

- Codes à longueur fixe
- Codes à longueur variable

Choix du code

Code : $\mathcal{C} : x_i \in \mathcal{X} \to c_i \in \{0, 1\}^*$

Codes à longueur fixe (FLC)

- Tout mot de code a la même longueur
- Si on a $M = 256$ symboles, il nous faut $\lceil \log M \rceil = 8$ bits pour coder chaque symbole
- Dans le cas d'un texte, $M = 26$, il nous faut $\lceil \log M \rceil = 5$ bpS (bit par symbole)

Codes à longueur variable (VLC)

- ℓ_i : longueur du mode de code c_i
- On peut comprimer *sans pertes* si :
  - Condition de décodabilité : condition du préfixe
  - Les symboles ne sont pas équiprobables

Exemple : Compression d'un texte français

Technique	Code à longueur fixe
Nombre de symboles	26
Taux de codage ($\mathcal{L}$)	5 bpS
Rapport de compression	1

- Chaque lettre est représentée sur 5 bits
- Aucune compression est obtenue

VLC : condition de décodabilité

- On utilise pas de “séparateurs” entre les mots de code
- Codes instantanés et décodables
- Inégalité de Kraft : il existe un code instantané avec longueurs $\{\ell_1, \ldots, \ell_M\}$ si et seulement si :

$$\sum_i 2^{-\ell_i} \leq 1$$

- Les codes décodables n’ont pas des meilleures performances par rapport aux codes instantanés

Inégalité de Kraft : Principes de la demonstration

Condition du préfix $\Rightarrow \sum_i 2^{-\ell_i} \leq 1$

- Construction de l'arbre binaire de profondeur $\ell_{\max}$
- Association entre mots de code et nœuds
- Pour chaque feuille, on remonte vers la racine : combien de mots de codes peut-on rencontrer ?
  - Zéro ou un (condition du prefix)
- Numéro feuilles = $A \geq B$ = Numéro feuille avec exactement un mot de code entre les ancêtres
- $A = 2^{\ell_{\max}}$
- $B = \sum_{i=1}^{M}$ Numéro feuilles qui descendent de l'i-ème mot de code $= \sum_{i=1}^{M} 2^{\ell_{\max}-\ell_i}$

Inégalité de Kraft : Principes de la demonstration

$\sum_i 2^{-\ell_i} \leq 1 \Rightarrow$ Condition du préfix

- Construction de l'arbre binaire de profondeur $\ell_{\max}$
- Premier mot de code c_1 : prendre une feuille et remonter au niveau ℓ_1
- Couper le sous-arbre associé au premier mot de code c_1
- Tout nœud survecu n'a pas c_1 comme préfix
- Pour tout nouveau mot de code, on coupe le sous-arbre associé
- Par consequence, si il reste des feuilles, on pourra trouver un nouveau mot de code

Inégalité de Kraft : Principes de la demonstration

$\sum_i 2^{-\ell_i} \leq 1 \Rightarrow$ Condition du préfix

- Raisonnement par récurrence et par construction
- On a montré comment trouver c_1
- Récurrence : si on a trouvé $\{c_i\}_{i=1}^{n-1}$, on peut trouver c_n, avec $n \leq M$
- Combien de feuilles on a éliminé au pas $n-1$?
- Pour le mot de code c_i on a éliminé $2^{\ell_{\max}-\ell_i}$ feuilles ; en total :

$$\sum_{i=1}^{n-1} 2^{\ell_{\max}-\ell_i} = 2^{\ell_{\max}} \sum_{i=1}^{n-1} 2^{-\ell_i} \tag{1}$$

$$< 2^{\ell_{\max}} \sum_{i=1}^{M} 2^{-\ell_i} \leq 2^{\ell_{\max}} \tag{2}$$

- On peut donc ajouter c_n en remontant d'une des feuilles residuelles jusqu'au niveau ℓ_n

Inégalité de Kraft

- Un code est défini par l'ensemble des longueurs $\{\ell_1, \ldots, \ell_M\}$
- De l'ensemble des longueurs on construit un arbre
- De l'arbre on construit le code

Information et entropie

- Le symbole x_i a une probabilité p_i d'apparaître
- Longueur moyenne du code : $\mathcal{L} = \sum p_i \ell_i$
- L'information associé à x_i est $I(x_i) = -\log p_i$
  - $I(x_i) \geq 0$
  - Si $p_i = 1$, $I = 0$
  - Si deux symboles sont indépendants, $I(x_i, x_j) = I(x_i) + I(x_j)$
- Entropie de la source : information moyenne des symboles

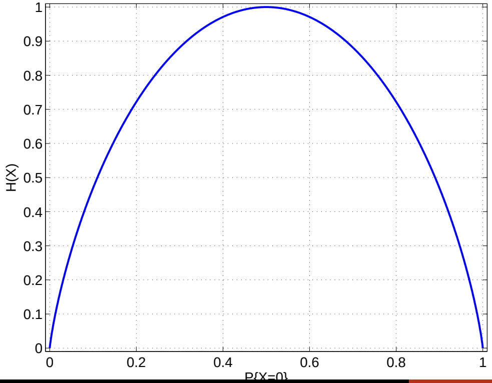
$$H(X) = -\sum_i p_i \log p_i$$

Entropie d'une variable binaire

Exemple

$$p = P\{X = 0\} \qquad q = P\{X = 1\} = 1 - p$$

$$H(X) = -p\log(p) - (1-p)\log(1-p)$$



Distribution à entropie maximum

On peut montrer que la distribution qui maximise l'entropie d'une v.a. discrete à M valeurs est le vecteur $\mathbf{p}^* = [p_1^* \, p_2^* \, \dots \, p_M^*]$ tel que
$p_i^* = \frac{1}{M} \qquad \forall i \in \{1, 2, \dots, M\}$
Problème de maximisation avec contrainte :

$$\mathbf{p}^* = \arg\max_{\mathbf{p}} \sum_{i=1}^{M} p_i \log \frac{1}{p_i} \qquad\qquad \sum_{i=1}^{M} p_i = 1$$

Distribution à entropie maximum

On peut montrer que la distribution qui maximise l'entropie d'une v.a. discrete à M valeurs est le vecteur $\mathbf{p}^* = [p_1^*\, p_2^* \, \dots \, p_M^*]$ tel que
$p_i^* = \frac{1}{M} \qquad \forall i \in \{1, 2, \dots, M\}$
Problème de maximisation avec contrainte :

$$\mathbf{p}^* = \arg\max_{\mathbf{p}} \sum_{i=1}^{M} p_i \log \frac{1}{p_i} \qquad\qquad \sum_{i=1}^{M} p_i = 1$$

$$J(\mathbf{p}) = -\sum_{i=1}^{M} p_i \log p_i + \lambda \left(\sum_{i=1}^{M} p_i - 1 \right) \qquad\qquad \frac{\partial J}{\partial p_i}(\mathbf{p}^*) = 0$$

$$\frac{\partial J}{\partial p_i} = -\left(\frac{1}{\ln 2} + \log p_i^* \right) + \lambda \qquad\qquad p_i^* = \lambda - \log e = \text{cnste}$$

Entropie conjointe

- Considerons un couple de v.a. X et Y
- Distribution de probabilité conjointe
  $p_{i,j} = \mathrm{P}\{X = x_i, Y = y_j\}$
- Entropie conjointe : information moyenne des couples

$$H(X, Y) = -\sum_{i,j} p_{i,j} \log p_{i,j}$$

- Formalement, il n'y pas de différence entre l'entropie d'un couple et l'entropie d'une variable Z avec les mêmes probabilités (independemment des valeurs)

Entropie conditionnelle

- Considerons un couple de v.a. X et Y
- Soit $p_j = \mathrm{P}\{Y = y_j\}$
- Entropie conditionnelle :

$$H(X|Y) = \sum_j p_j H(X|Y = y_j)$$

- On montre facilement que :

$$\begin{aligned} H(X,Y) =& H(Y) + H(X|Y) \\ & H(X) + H(Y|X) \end{aligned}$$

Propriétés de l'entropie

$$
\begin{aligned}
H(X) &> 0 \\
H(X,Y) &= H(Y) + H(X|Y) \\
& H(X) + H(Y|X) \\
H(X,Y) &\leq H(X) + H(Y) && \text{avec égalité} \Leftrightarrow \text{indépendence} \\
H(X|Y) &\leq H(X) && \text{avec égalité} \Leftrightarrow \text{indépendence} \\
H(X) &\leq \log_2 M && \text{avec égalité} \Leftrightarrow X \sim \mathcal{U}
\end{aligned}
$$

Code optimal

- On relâche la condition ℓ_i entier
- Minimisation avec contrainte :

$$\underline{\ell}^* = \arg\min_{\underline{\ell}} \sum_i p_i \ell_i \qquad \text{soumis à} \quad \sum_i 2^{-\ell_i} = 1$$

Code optimal

- On relâche la condition ℓ_i entier
- Minimisation avec contrainte :

$$\underline{\ell}^* = \arg\min_{\underline{\ell}} \sum_i p_i \ell_i \qquad \text{soumis à} \quad \sum_i 2^{-\ell_i} = 1$$

$$J(\underline{\ell}) = \sum_i p_i \ell_i + \lambda \left(\sum_i 2^{-\ell_i} - 1 \right) \qquad \frac{\partial J}{\partial \ell_i} = p_i - (\lambda \ln 2) 2^{-\ell_i^*} = 0$$

$$\sum_i p_i = (\lambda \ln 2) \sum_i 2^{-\ell_i^*} \qquad 1 = \lambda \ln 2$$

$$2^{-\ell_i^*} = p_i \qquad \ell_i^* = -\log_2 p_i$$

$$\mathcal{L}^* = \sum_i -p_i \log_2 p_i = H(X)$$

Thèoreme de Shannon sur le codage de source

Si on introduit à nouveau la condition $\ell_i \in \mathbb{N}$, on peut montrer que :

- **Thèoreme de Shannon**

$$\mathcal{L}^* \geq H(X)$$

avec égalité si et seulement si:

$$\forall i \in \{1, 2, \ldots M\}, \exists \ell_i \in \mathbb{N} \,|\, p_i = 2^{-\ell_i}$$

Codage *entropique*

Théorème de Shannon:

Taux de codage code optimale $\geq$ Entropie de la source

du coup le nom *Codage Entropique*.

- La relation devient une identité stricte si les probabilités sont *dyadiques* (puissances négatives de deux)
- La relation est pratiquement une identité quand il y a un nombre important de symboles dans l'alphabet.

Codage *entropique*

- En consequence du Théorème de Shannon, on peut facilement montrer que :

$$H(X) \leq \mathcal{L}^* < H(X) + 1 \qquad (3)$$

- Il suffit de prendre $\ell_k = \left\lceil log_2 \frac{1}{p_k} \right\rceil$
- Il est facile de montrer que l'inégalité de Kraft est satisfaite
- Il est aussi facile de prouver l'inégalité (3)

Codage *entropique*

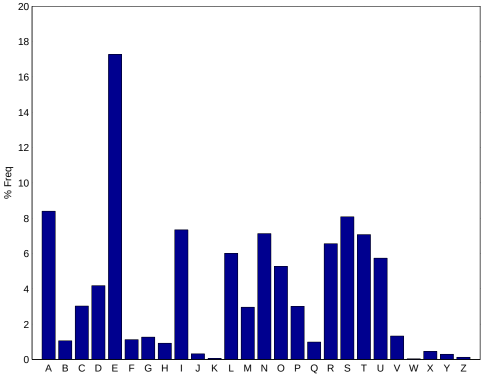
- Théorème de Shannon :

$$H(X) \leq \mathcal{L}^* < H(X) + 1$$

- L'entropie est une excellente approximation du taux de codage optimale
- Dans la suite on sera souvent amenés à considerer $\mathcal{L}^* = H(X)$

Exemple : Compression d'un texte français

Entropie de la source	3.999 bpS
Technique	Code à longueur variable
Taux de codage ($\mathcal{L}$)	$\geq$ 3.999 bpS
Rapport de compression	$\leq$ 1.25



Distribution des lettres dans un texte français

Plan

1. Principes
2. Codage optimale
   - Huffman
   - Codage arithmétique
   - Codage adaptive et basé contexte
3. Autres Techniques
   - Lempel-Ziv
   - Run Length
   - JBIG
4. Quantification avec contrainte entropique

Codage de Huffman

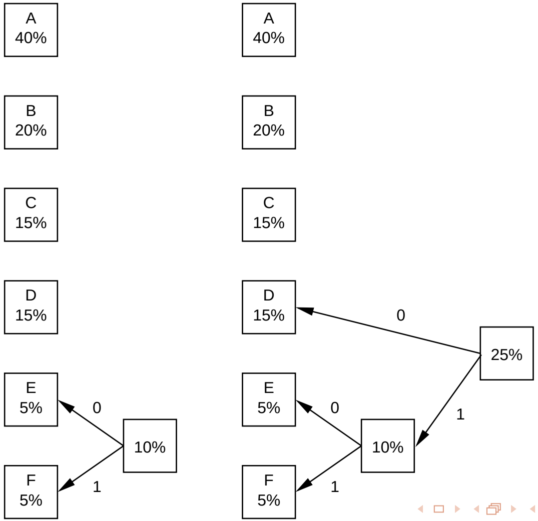
Huffman a découvert comment construire le code optimum pour n'importe quelle source.

Exemple :

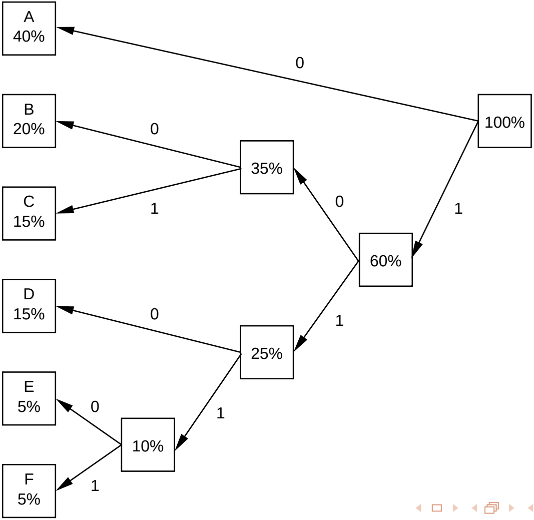
Symbole	Probabilité
A	0.4
B	0.2
C	0.15
D	0.15
E	0.05
F	0.05

6 symboles, 3 bits par symboles sans codage.

Codage de Huffman



Codage de Huffman



Codage de Huffman

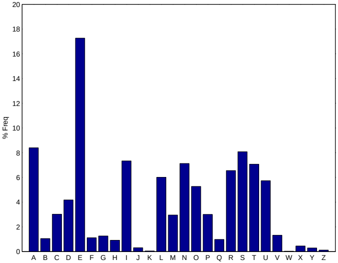
Symbole	Probabilité	Code
A	0.4	0
B	0.2	100
C	0.15	101
D	0.15	110
E	0.05	1110
F	0.05	1111

$$\begin{aligned}\mathcal{L} &= 0.4 \cdot 1 + 0.2 \cdot 3 + 0.15 \cdot 3 + 0.15 \cdot 3 + 0.05 \cdot 4 + 0.05 \cdot 4 \\ &= 2.3 \text{ bits/symbole}\end{aligned}$$

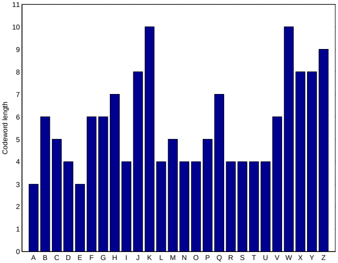
$$\begin{aligned}H &= 0.4 \cdot \log_2 \frac{1}{0.4} + 0.2 \cdot \log_2 \frac{1}{0.2} + 2 \cdot 0.15 \cdot \log_2 \frac{1}{0.15} \\ &+ 2 \cdot 0.05 \cdot \log_2 \frac{1}{0.05} \\ &\cong 2.2464 \text{ bits/symbole}\end{aligned}$$

Exemple : Compression d'un texte français

Technique	Huffman
Entropie de la source	3.999 bpS
Taux de codage ($\mathcal{L}$)	4.041 bpS
Rapport de compression	1.238



Distribution des lettres dans un texte français



Longueur des mots de code dans le code d'Huffman

Codage de Huffman

Comment améliorer les performances ?

- Le bloc des premiers K symboles du processus aléatoire X_i est appellé X^K
- $H(X^K) \leq \sum_i H(X_i)$ avec égalité si et seulement si les variables de X^K sont independentes
- Codage par blocs : on essaie de reduire la longueur du code *mesurée en bits par symbole*

Codage de Huffman
Codage par blocs

- Hypothèse : la suite $\frac{H(X^K)}{K}$ est convergente
- Cela est vrai p.e. pour un processus stationnaire
- Longueur moyenne du code optimum :

$$H(X^K) \leq \mathcal{L}^* < H(X^K) + 1$$

$$\frac{H(X^K)}{K} \leq \frac{\mathcal{L}^*}{K} = \mathcal{L}_S^* < \frac{H(X^K)}{K} + \frac{1}{K}$$

$$\lim_K \mathcal{L}_S^* = \lim_K \frac{H(X^K)}{K} = \mathcal{H}(X)$$

$$\mathcal{L}_S^* \to \mathcal{H}(X) \leq H(X)$$

- $\mathcal{H}(X)$ est appellé *taux entropique*

Codage de Huffman
Codage par blocs

- Codage par blocs :

$$\mathcal{L}_S^* \to \mathcal{H}(X) \leq H(X)$$

- Les meilleures performances sont obtenues quand on code l'entier message de K symboles comme un symbole d'un alphabet de taille N^K
- Le codage par blocs est avantageux même pour v.a.i.i.d. : cela élimine le bit supplémentaire des distributions non-dyadiques

Codage de Huffman

Codage par blocs

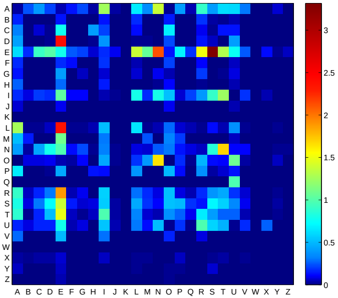
- Codage par contextes : l'entropie d'un symbole donnés ses $K-1$ voisins est typiquement largement inférieure à $H(X)$
- On peut donc imaginer de coder $X_k|X^{k-1}$ dont l'entropie est inférieure ou égale à $H(X)$
- On peut montrer que, pour un processus stationnaire,

$$\lim_k H(X_k|X^{k-1}) = \mathcal{H}(X)$$

Exemple : Compression d'un texte français

K=1	Entropie des lettres	3.999 bpB	3.999 bpS
K=2	Entropie des digrams	7.440 bpB	3.720 bpS
K=3	Entropie des trigrams	9.452 bpB	3.151 bpS

bpB : bits par bloc ; bpS : bits par symbole, bits par lettre



Distribution des digrams dans un texte français

Les trigrams les plus communs :

ait	ent	les
1.59%	1.25%	0.94%
lle	des	ant
0.78%	0.72%	0.70%
que	our	ien
0.67%	0.63%	0.60%

Limites du code de Huffman

De l'exemple précédent, on comprend qu'on voudrait aller jusqu'à la limite K=longueur du message.
C'est pratiquement impossible avec Huffman

- Difficile et coûteux de connaître les probabilités
- Complexité exponentielle du code avec la taille du bloc
- Le dictionnaire devrait comprendre tout les possibles messages de K symboles:
  - Tous les possibles textes
  - Toutes les possibles images
  - ...

Limites du code de Huffman

De l'exemple précédent, on comprend qu'on voudrait aller jusqu'à la limite K=longueur du message.

C'est pratiquement impossible avec Huffman

- Difficile et coûteux de connaître les probabilités
- Complexité exponentielle du code avec la taille du bloc
- Le dictionnaire devrait comprendre tout les possibles messages de K symboles:
  - Tous les possibles textes
  - Toutes les possibles images
  - ...

Le **codage arithmétique** résout le problème

Codage arithmétique

- Le *codage arithmétique* permets de faire un codage par blocs ou par contextes avec complexité linéaire
- Idée : ne pas chercher le code pour n'importe chaîne de n symboles, mais uniquement pour la chaîne à coder
- Le codeur arithmétique n'est pas optimal, mais *asymptotiquement* optimal

$$\mathcal{L} \leq H(X^K) + 2$$

$$\mathcal{L}_S = \mathcal{L}/K$$

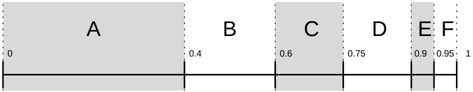
$$\lim_{K\to\infty} \mathcal{L}_S = \lim_{K\to\infty} \frac{H^K}{K} = \mathcal{H}(X)$$

- Faible complexité de codage/décodage (opérations arithmétiques, dont le nom)

Codage arithmétique: exemple

Symbole	A	B	C	D	E	F
Probabilité	0.4	0.2	0.15	0.15	0.05	0.05

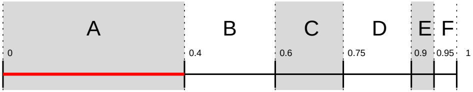
Séquence à coder : ACFD



Codage arithmétique: exemple

Symbole	A	B	C	D	E	F
Probabilité	0.4	0.2	0.15	0.15	0.05	0.05

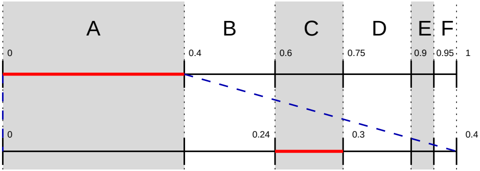
Séquence à coder : ACFD



Codage arithmétique: exemple

Symbole	A	B	C	D	E	F
Probabilité	0.4	0.2	0.15	0.15	0.05	0.05

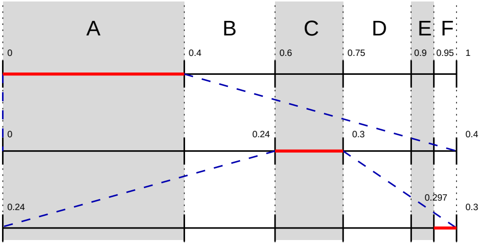
Séquence à coder : ACFD



Codage arithmétique: exemple

Symbole	A	B	C	D	E	F
Probabilité	0.4	0.2	0.15	0.15	0.05	0.05

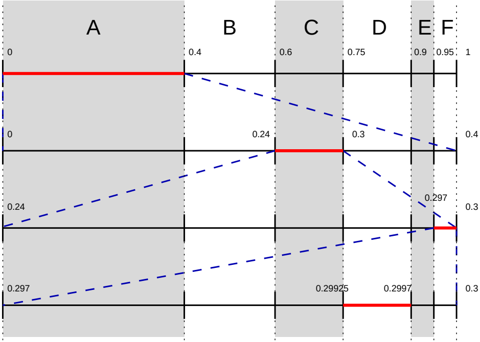
Séquence à coder : ACFD



Codage arithmétique: exemple

Symbole	A	B	C	D	E	F
Probabilité	0.4	0.2	0.15	0.15	0.05	0.05

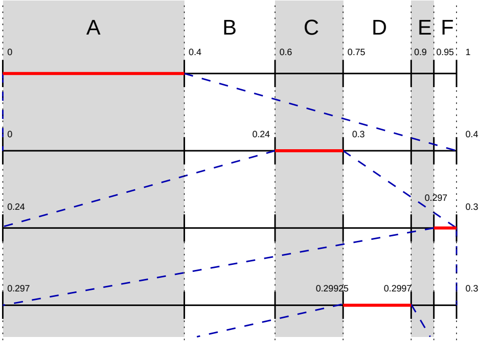
Séquence à coder : ACFD



Codage arithmétique: exemple

Symbole	A	B	C	D	E	F
Probabilité	0.4	0.2	0.15	0.15	0.05	0.05

Séquence à coder : ACFD



Codage arithmétique

- Pour chaque nouveau symbole, 2 multiplications et 2 addition
- Codage de la suite de symboles : centre de l'interval sélectionné, avec précision inférieure à la demi-taille de l'interval.
- Problème : estimation de $P(X^K)$, en principe avec K=longueur totale du message
- Exemple précédent : Symboles supposés indépendants, $P(X^K) = \prod_{i=1}^{K} p(x_i)$
- Apprentissage des statistiques au cours du codage (adaptivité)

Codage par contextes

- Estimation de $P(X^K)$: souvent le prochain symbole ne dépende que de peu de voisins.
- Idée : souvent il suffit connaître un voisinage limité du symbole courant (contexte)

$$H(X^K) = \sum_{i=1}^{K} H(X_i | X^{i-1})$$

- En théorie, le context est tout le passé : X^{i-1}
- Le contexte peut être fait par les quelques lettres précédentes ou les quelques pixels autour du pixel courant
- Si on a M possibles contextes, c'est comme si on avait M codeurs arithmétiques, et si on passe de l'un à l'autre

Codage par contextes : Image N/B

Soient X et Y deux pixels voisins.

Y	X: N	X: B
N	0.15	0.05
B	0.05	0.75

Probabilités conjointes de X et Y

Y	X: N	X: B
N	0.75	0.25
B	$\frac{1}{16}$	$\frac{15}{16}$

Probabilités conditionnelles de X donné Y

H(X)	H(X,Y)/2	H(X|Y)	HE	AE	CB AE
0.722	0.577	0.432	1	0.722	0.432

HE: Huffman Encoder, One Symbol
AE: Arithmetic Encoder
CB-AE: Context-Based AE

Codage arithmétique : conclusions

Avantages

- Permet d'implémenter le codage de longue suites de symboles avec une complexité linéaire
- Codage par blocs : statistiques d'ordre supérieure et distribution dyadiques
- Codage par contexte : simple modélisation des statistiques d'ordre supérieure
- Adaptivité : sources non-stationnaires

Codage arithmétique : conclusions

Inconvénients

- Implémentation parfois compliquée
- Choix des contextes
- Adaptivité : il faut assez de données pour une estimation robuste
- Besoin d'initialisation

Plan

1 Principes

2 Codage optimale
- Huffman
- Codage arithmétique
- Codage adaptive et basé contexte

3 Autres Techniques
- Lempel-Ziv
- Run Length
- JBIG

4 Quantification avec contrainte entropique

Codage avec dictionnaire

- Dictionnaire des suites de données communes construit au fur et à mesure
- Capable de s'adapter à des signaux non-stationnaires
- Pas besoin d'initialisation (codage universel)
- À la base des algorithmes populaires de compression sans perte (zip, gzip, bzip, *etc.*)

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1

Output

Codage avec dictionnaire: exemple

Input

0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	1	0

Present 0

Table *Output*

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

New 0 0

Table															*Output*
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	
0	1														

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Add 0 0

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1 0
0

Output

Codage avec dictionnaire: exemple

Input

$\boxed{0 \quad 0}$ 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Output 0 ~~0~~

Table *Output*

0 1 2 3 4 5 6 7 8 9 A B C D E ***0***

0 1 0
0

Codage avec dictionnaire: exemple

Input

0 $\boxed{0}$ 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 0

Table	*Output*
0 1 2 3 4 5 6 7 8 9 A B C D E	**0**

$\boxed{0}$ 1 0
0

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 0 0

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1 0
0

Output

0

Codage avec dictionnaire: exemple

Input

0 **0 0 1** 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

New 0 0 1

Table	*Output*
0 1 2 3 4 5 6 7 8 9 A B C D E	***0***

0 1 0
0

Codage avec dictionnaire: exemple

Input

0 **0 0 1** 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Add 0 0 1

Table	*Output*
0 1 2 3 4 5 6 7 8 9 A B C D E	***0***

0 1 0 0
0 0
1

Codage avec dictionnaire: exemple

Input

0 $\boxed{0 \quad 0 \quad 1}$ 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Output 0 0 ~~1~~

Table

0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0	0											
		0	0											
			1											

Output

0 2

Codage avec dictionnaire: exemple

Input

0 0 0 $\boxed{1}$ 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 1

Table	*Output*
0 1 2 3 4 5 6 7 8 9 A B C D E	***0 2***

0	$\boxed{1}$	0	0
		0	0
			1

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

New 1 0

Table																*Output*
0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***		***0 2***
0	1	0	0													
		0	0													
			1													

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Add 1 0

Table

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**
0	1	0	0	1										
		0	0	0										
			1											

Output

0 2

Codage avec dictionnaire: exemple

Input

0 0 0 **1 0** 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Output 1 ~~0~~

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1 0 0 1
0 0 0
1

Output

0 2 1

Codage avec dictionnaire: exemple

Input

0 0 0 1 $\boxed{0}$ 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 0

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

$\boxed{0}$	1	0	0	1
		0	0	0
			1	

Output

0 2 1

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 0 0

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0	1	0	0	1
		0	0	0
			1	

Output

0 2 1

Codage avec dictionnaire: exemple

Input

0 0 0 1 $\boxed{0\ 0\ 0}$ 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

New 0 0 0

Table

0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0	0	1										
		0	0	0										
			1											

Output

0 2 1

Codage avec dictionnaire: exemple

Input

0 0 0 1 **0 0 0** 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Add 0 0 0

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1	0									
		0	0	0	0									
			1		0									

Output

0 2 1

Codage avec dictionnaire: exemple

Input

0 0 0 1 $\boxed{0\ 0\ 0}$ 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Output 0 0 ~~0~~

Table

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**
0	1	0	0	1	0									
		0	0	0	0									
			1		0									

Output

0 2 1 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Prefix

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	00	001	10	000									

Output

0 2 1 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Prefix

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1										
		0	0	0										
		0	1											

Output

0 2 1 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 0

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1 0 0 1
0 0 0
0 1

Output

0 2 1 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 0 0

Table	*Output*
0 1 2 3 4 5 6 7 8 9 A B C D E	***0 2 1 2***

0 1 0 0 1
0 0 0
0 1

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 0 0 0

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0	1	2	3	4
0	1	0	0	1
		0	0	0
		0	1	

Output

0 2 1 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 $\boxed{0\ 0\ 0\ 0}$ 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

New 0 0 0 0

Table		*Output*
0 1 2 3 4 5 6 7 8 9 A B C D E		***0 2 1 2***

0 1 0 0 1
0 0 0
0 1

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 $\boxed{0\ 0\ 0\ 0}$ 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Add 0 0 0 0

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1	0									
		0	0	0	0									
		0	1		0									
					0									

Output

0 2 1 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 **0 0 0 0** 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Output 0 0 0 ~~0~~

Table

0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0	0	1	0									
		0	0	0	0									
		0	1		0									
					0									

Output

0 2 1 2 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 0

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1	0									
		0	0	0	0									
		0	1		0									
					0									

Output

0 2 1 2 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

New 0 1

Table

0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0	0	1	0									
		0	0	0	0									
		0	1		0									
					0									

Output

0 2 1 2 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 **0 1** 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Add 0 1

Table

0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0	0	1	0	0								
		0	0	0	0	1								
		0	1		0									
					0									

Output

0 2 1 2 2

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 **0 1** 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Output 0 ~~1~~

Table			*Output*
0 1 2 3 4 5 6 7 8 9 A B C D E			**0 2 1 2 2 0**

```
0 1 0   0   1 0   0
    0   0   0 0   1
    0   1     0
              0
```

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 **0 1** 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Prefix 0 1

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1	0	0								
		0	0	0	0	1								
		0	1		0									
					0									

Output

0 2 1 2 2 0

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Prefix 0 1

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0	1	0	0	1	0
1		0	0	0	0
		0	1		0
					0

Output

0 2 1 2 2 0

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 1

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0	1	0	0	1	0
1		0	0	0	0
		0	1		0
					0

Output

0 2 1 2 2 0

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Present 1 0

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1	0									
1		0	0	0	0									
		0	1		0									
					0									

Output

0 2 1 2 2 0

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 **1 0 1** 0 0 0 0 1 0 0 0 0 0 1 0

New 1 0 1

Table

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**
0	1	0	0	1	0									
1		0	0	0	0									
		0	1		0									
					0									

Output

0 2 1 2 2 0

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 **1 0 1** 0 0 0 0 1 0 0 0 0 0 1 0

Add 1 0 1

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1	0	1								
1		0	0	0	0	0								
		0	1		0	1								
					0									

Output

0 2 1 2 2 0

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 $\boxed{1\ 0\ 1}$ 0 0 0 0 1 0 0 0 0 0 1 0

Output 1 0 ~~1~~

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1	0	1								
1		0	0	0	0	0								
		0	1		0	1								
					0									

Output

0 2 1 2 2 0 4

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 **1 0 0** 0 0 1 0 0 0 0 0 1 0

New 1 0 0

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0	1	0	0	1	0	1
1		0	0	0	0	0
		0	1		0	1
					0	

Output

0 2 1 2 2 0 4

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 **1 0 0** 0 0 1 0 0 0 0 0 1 0

Output 1 0 ~~0~~

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0	1	0	0	1	0	1	1
1		0	0	0	0	0	0
		0	1		0	1	0
					0		

Output

0 2 1 2 2 0 4 4

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0

Prefix 1 0 0

Table

0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0	0	1	0	1	1							
1		0	0	0	0	0	0							
		0	1		0	1	0							
					0									

Output

0 2 1 2 2 0 4 4

Codage avec dictionnaire: exemple

Input

0 0 0 1 0 0 0 0 0 0 1 0 **1 0 0** 0 0 1 0 0 0 0 0 1 0

Prefix 1 0 0

Table

0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0	0	1	0	1								
1		0	0	0	0	0								
		0	1	0	0	1								
					0									

Output

0 2 1 2 2 0 4 4

Décodage avec dictionnaire: exemple

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1

Input

0 2 1 2 2 0

New

Output

Décodage avec dictionnaire: exemple

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1

Input

$\boxed{\mathbf{0}}$ ***2 1 2 2 0***

New

2 0x

Output

0x

Décodage avec dictionnaire: exemple

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1 0

x

Input

***0* 2 1 2 2 0**

New

2 0x

Output

0x

Décodage avec dictionnaire: exemple

Table

0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0												
		x												

Input

0 2 1 2 2 0

New

3 0xy

Output

0x
00xy

Décodage avec dictionnaire: exemple

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0 1 0 0
0 0
y

Input

0 2 1 2 2 0

New

3 00y

Output

000y

Décodage avec dictionnaire: exemple

Table														
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**
0 1	0 0													
	0 0													
	y													

Input

0 2 1 2 2 0

New

4	1z

Output

000y
0001z

Décodage avec dictionnaire: exemple

Table														
0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***
0	1	0	0	1										
		0	0	z										
			1											

Input

0 2 1 2 2 0

New

4 1z

Output

0001z

Décodage avec dictionnaire: exemple

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	00	1											
		00	z											
		1												

Input

0 2 1 $\boxed{2}$ 2 0

New

5 00w

Output

0001z
000100w

Décodage avec dictionnaire: exemple

Table

0 1 2 3 4 5 6 7 8 9 A B C D E

0	1	0	0	1	0
		0	0	0	0
			1		w

Input

0 2 1 2 2 0

New

5 00w

Output

000100w

Décodage avec dictionnaire: exemple

Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E
0	1	0	0	1	0									
		0	0	0	0									
			1		w									

Input

0 2 1 2 2 0

New

5 00w

Output

000100w

Décodage avec dictionnaire: exemple

Table

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**
0	1	0	0	1										
		0	0	0										
		0	1											

Input

0 2 1 2 2 0

New

2 000

Output

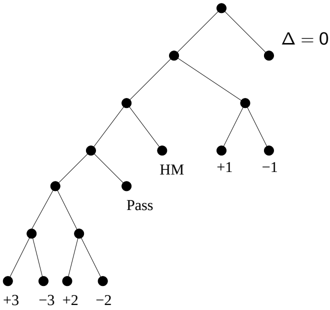
0001000

Run-Length Encoding (RLE)

- Codage d'images B/N
- Longues suites de zéros et uns
- Idée : coder la longueur des suites au lieu des valeurs des pixels

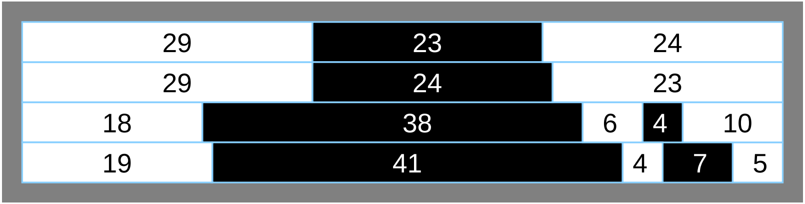
Run-Length Encoding (RLE)

- Code d'Huffmann : alphabet trop grand
- Code d'Huffmann sur les puissances de 2
- Chaque longueur est représenté comme somme de puissances de 2 (représentation binaire)
- Mode horizontale : longueur absolue
- Mode verticale : différence par rapport à la ligne supérieure (si elle est ≤ 3)
- Pass : noveau block



Mode verticale : symboles et code d'Huffmann

Run-Length Encoding (RLE) : Exemple

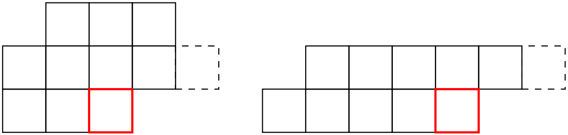


<table>
<tr><td>HM 16+8+4+1</td><td>16+4+2+1</td><td colspan="3">16+8</td></tr>
<tr><td>VM 0</td><td>+1</td><td colspan="3">-1</td></tr>
<tr><td>HM 16+2</td><td>32+4+2</td><td>4+2</td><td>PASS 4</td><td>PASS 8+2</td></tr>
<tr><td>VM +1</td><td>+3</td><td>-2</td><td>+3</td><td>HM 4+1</td></tr>
</table>

JBIG-1

Joint Bi-level Image Experts Group

- Standard ISO/IEC 11544 et recommandation ITU-T T.82
- Codeur arithmétique basé contexte
- codage progressive (scalable en résolution)
- *Template* de 10 pixels, 2 formes et pixel variable



Template à 2 ligne : meilleure vitesse d'exécution mais $\approx$ 5% de perte en compression

JBIG-2
Joint Bi-level Image Experts Group

- Standard ISO/IEC 14492 et recommandation ITU-T T.88
- Image segmentée en **texte**, **halftones**, et **autre**
- **Texte**: Un dictionnaire de symbole est créé et codé
- **Halftone**: l'image en niveau de gris originale est reconstruite et codée, avec un dictionnaire de *halftone patterns*
- **Autre**: Codage arithmétique basé contexte
- Les fichiers PDF (version 1.4 et supérieure) peuvent contenir données codées un JBIG2

Plan

1. Principes
2. Codage optimale
   - Huffman
   - Codage arithmétique
   - Codage adaptive et basé contexte
3. Autres Techniques
   - Lempel-Ziv
   - Run Length
   - JBIG
4. Quantification avec contrainte entropique

Quantification et codage

- Quantificateur optimal : doit-il être changé en vue du codage sans perte ?
- Quelles sont les performances d'un système simple comme un quantificateur uniforme suivi d'un codage entropique ?

Quantification et codage

Formulation du problème

- On represente le quantificateur avec un q.u. dont le pas est δ précédé d'une non transformation non linéaire, dont la caracteristique est $f(x)$
- Il s'agit de minimiser la puissance de l'erreur de quantification :

$$\sigma_Q^2 = \frac{\delta}{12} \int \frac{p_X(x)}{f'^2(x)} dx$$

- sous contrainte sur l'entropie : $H(\hat{X}) \leq b$
- On peut montrer que en hypothèse d'haute résolution f' doit être constant

Quantification et codage

On peut montrer que, en hypothèse d'haute résolution :

- Pour un niveau de distortion (EQM) fixé, l'entropie minimum des symboles du quantificateur est obtenue avec un quantificateur uniforme
- Pour une entropie des symboles donnée, la distortion minimum est obtenue avec un quantificateur uniforme
- Un q.u. suivi d'un codeur entropique a un gain de 2.81 dB sur un codeur de Lloyd-Max dans le cas de sources Gaussienne i.i.d.
- La courbe RD est toujours dans la forme $D \propto 2^{-2R}$