الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : Electronique

PROJET DE FIN D'ETUDES

en vue de l'obtention du Diplôme d'Ingénieur d'Etat

SUJET

Quantification Vectorielle des signaux et codage par treillis

Proposé par : Mr D. Berkani

Etudié par : Mr Hallil Abdelbasset

Dirigé par : Mr D. Berkani

PROMOTION : Juin 1990

E.N.P. 10. Avenue Hacene Badi – EL-HARRACH – ALGER

الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : Electronique

PROJET DE FIN D'ETUDES

en vue de l'obtention du Diplôme d'Ingénieur d'Etat

SUJET

Quantification Vectorielle des signaux et codage par treillis

Proposé par : Mr D. Berkani

Etudié par : Mr Hallil Abdelbasset

Dirigé par : Mr D. Berkani

PROMOTION : Juin 1990

E.N.P. 10, Avenue Hassen Badi — EL HARRACH — ALGER

REMERCIEMENTS

Je remercie vivement Mr. BERKANI pour m'avoir dirigé et encadré le long de ce modeste travail.

Je remercie aussi Mr. Derras pour avoir voulu accepter d'être présent à mes côtés le jour de soutenance.

Que les enseignants qui ont contribué de façon éfficace à ma formation, que celà soit au C.E.M de Tazmalt, ou au lycée Debbih d'Akbou, ou à l'E.N.P, trouvent ici ma reconnaissance et ma gratitude.

DEDICACES

Je dédie ce présent travail à :

- mes parents
- mes frères et sœurs
- mes cousins les plus proches
- à tous mes amis

TABLE DES MATIÈRES

INTRODUCTION

Durant les 30 dernières années, des efforts considérables ont été déployés dans le codage digital des formes d'ondes telles que les signaux de parôle et d'image. La compression de données s'est avérée indispensable lors de la transmission de ces sources d'information (parôle et image) vu la largeur de débit éxigée et les contraintes en matière de capacité imposées par le canal de transmission. Ainsi des techniques de quantification éfficaces, à des débits décroissants, ont été étudiées, tout en se basant sur les comportements statistiques de ces sources d'information (stationnarité, distribution, etc...).

N'éanmoins, les quantificateurs scalaires ont demeuré depuis longtemps les systèmes de compression de données les plus usuels, à cause de leur simplicité et de leur performance quand le débit de communication allové est suffisamment large.

Ces quantificateurs ne peuvent apporter des solutions concrètes quand le débit de communication imposé est faible ou fractionnel. C'est là que la quantification vectorielle se fait prévaloir.

Les quantificateurs vectoriels, ayant fait une apparition timide dans le passé, jouissent il ya quelques années d'un intérêt particulier et plusieurs algorithmes de quantification vectorielle ont été développés.

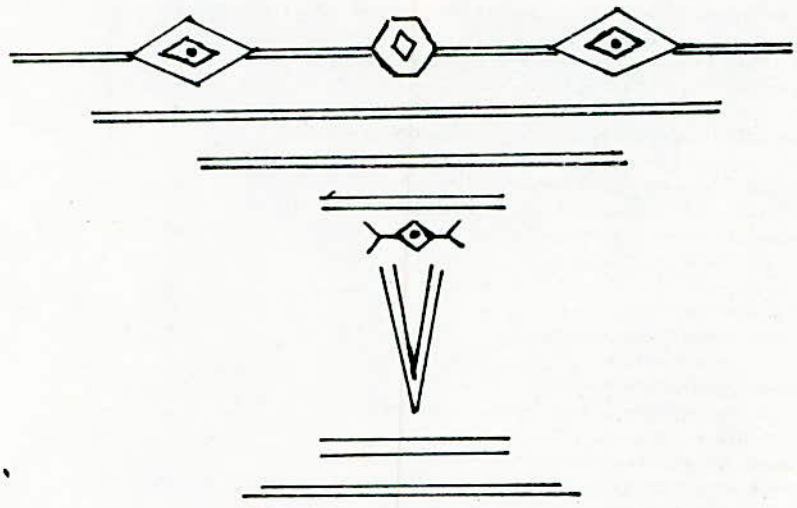
L'organisation de ce présent travail est la suivante: dans les deux premiers chapitres, nous avons jugé utile de présenter des rappels de la théorie de l'information et de la quantification scalaire, tout en mettant en evidence les insuffisances de cette dernière technique.

Dans le troisième chapitre, on définira la quantification vectorielle et on abordera les notions de distorsion et de quantificateur optimal. La quantification vectorielle algébrique ainsi que la statistique constitueront deux illustrations fondamentales de la quantification vectorielle.

Une methode récente de quantification qu'est la quantification et codage par treillis ou T.C.Q (contraction de Trellis Coded Quantization) sera introduite au quatrième chapitre. Elle sera concrétisée par un programme informatique

appliqué ici au treillis à 4 etats.

Toutes ces notions seront exposées de façon claire, tout en évitant les développements mathématiques très fastidieux et se contentant ainsi de concepts nouveaux et fondamentaux.



CHAPITRE-I
THEORIE DE L'INFORMATION

Parler de la quantification vectorielle des signaux et de sa nécessité introduit des concepts tels que le débit d'information et la capacité d'un canal.

Il serait convenable d'introduire toutes ces notions dans un ensemble plus compact qu'est la théorie de l'information.

I-1-Définitions :

I-1-a-Information:

On définit l'information comme étant la réalisation d'un évènement parmi les N possibles. Elle sera d'autant plus grande que l'événement qui vient de se réaliser aura une probabilité "p" plus petite. Elle sera souvent accompagnée de perturbations.

I-1-b-Perturbation

La perturbation est un signal qui modifie le signal aléatoire utile transmettant l'information et qui, de ce fait, diminue la quantité d'information transmise. L'information doit être générée par une source.

I-1-c-Source d'information :

Une source d'information est un mécanisme servant à choisir, parmi l'ensemble des messages possibles et d'une manière imprévisible pour l'observateur un certain message particulier destiné à être transmis à un correspondant.

I-1-d-Source discrète :

Une source discrète est une source qui débite les messages sous forme discrète (par ex : succession d'impulsions). Cependant le message, signal correspondant à une réalisation particulière, peut comporter n symboles. La totalité des symboles constitue l'alphabet.

I-1-e-Source discrète sans mémoire :

Cette source sera d'utilisation générale pour nous. Elle se caractérise par le fait que la probabilité d'apparition d'un symbole ne dépend pas des symboles précédents.

I-1-f- Canal:

On sous-entend par canal la totalité des moyens destinés à la transmission du message ; ces moyens couvrent et l'équipement et le milieu à travers lequel la transmission a lieu, y compris toutes les sources de perturbations. Toutes ces notions nous permettront de shematiser un système simple de transmission de l'information. voir fig-1.

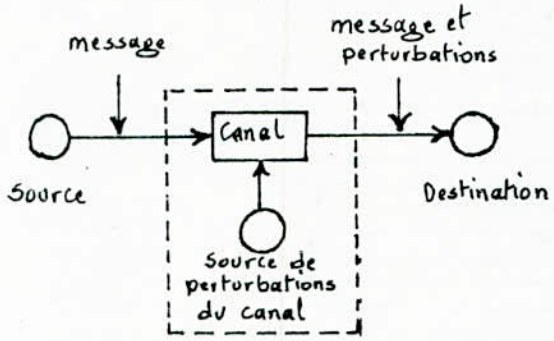


fig-1- Système simple de transmission de l'information

I-2-Mesure de l'information des signaux discrets:

La mesure de l'information consiste à mesurer objectivement l'incertitude du processus par lequel est réalisé un évènement parmi un ensemble d'événement possibles. L'ensemble discret et fini d'évènements possibles sera nommé "espace des echantillons". On le notera sous forme matricielle $[X]$.

$$[X] = [x_1, x_2 \dots x_n] \qquad \text{où } \bigcup_{i=1}^{n} x_i = E \quad \text{et } x_i \cap x_j = \phi \text{ pour } i \neq j$$

E est l'evenement sur ; ϕ est l'evenement impossible.
à chaque élément de $[X]$ est associée une probabilité, qu'on notera par

$$[p_x] = [p(x_1)\ p(x_2) \dots p(x_n)] \qquad \text{où } \sum_{i=1}^{n} p(x_i) = 1$$

La correspondance entre les évènements réalisés x_i et ceux résultant de l'observation des x_i, qu'on note y_j, n'est pas évidente. On définira ainsi l'ensemble d'évènements observés qu'on notera par $[Y]$.

$$[Y] = [y_1\ y_2\ y_3 \dots y_m] \quad \text{où } m \text{ peut être different de } n.$$

I.2.a. Specification de la fonction de mesure :

I.2.a.1. Cas de l'information propre :

La réalisation d'un evénement x_i ecarte son incertitude et l'on obtient une information $i(x_i)$ sur la réalisation de x_i. La fonction de mesure de $i(x_i)$ doit satisfaire la condition d'additivité et doit augmenter selon que la probabilité de l'évènement réalisé diminue. Par définition, l'information propre associée à l'événement x_i est :

$$i(x_i) = -\log(p(x_i))$$

I.2.a.2. Cas de l'information mutuelle :

La presence de pérturbations nous incite à parler d'information mutuelle. Elle correspondra à une mesure d'incertitude sur la réalisation de l'évènement x_i lorsque l'évènement y_j est observé. On notera par $p(x_i/y_j)$ la probabilité conditionnelle de x_i conditionnée par y_j.

Par définition, l'information mutuelle qu'on note $i(x_i;y_j)$ est donnée par

$$i(x_i;y_j) = \log \frac{p(x_i/y_j)}{p(x_i)}$$

I.2.a.3. Unité de mesure :

Comme unité de mesure, on a convenu de choisir l'information obtenue par le choix aléatoire d'un seul événement parmi deux, également probables.

Soit $[P] = [\frac{1}{2}\ \frac{1}{2}]$ associée aux événements $[X] = [x_1\ x_2]$

Si l'on travaille en logarithmes à base 2, on aura :

$$i(x_1) = i(x_2) = \log_2 2 = 1$$ unité désignée par "bit."

L'unité sera le "nit" ou le "dit" si l'on est en base "e" ou en base "10".

Pour caractériser tous les événements x_i d'une source, on doit aborder une notion plus globale qu'est l'entropie.

I.3. Entropie :

I.3.a Définition :

On définit l'entropie, notée $H(x)$, comme l'incertitude moyenne à priori que l'on a sur les évènements x_i de l'ensemble $[X]$. Elle constitue la

moyenne de l'information propre par symbole ($i(x_i)$).

$$H(X) = \sum_{i=1}^{n} i(x_i)\, p_i = -\sum_{i=1}^{n} p_i \log p_i$$

La notion d'entropie doit être associée au temps, ce qui nous amènera à parler de débit d'information.

I.3.b. Débit d'information et redondance d'une source :

— Le produit de l'entropie de la source par le nombre moyen de symboles par seconde constitue le débit d'information d'une source.

$$H_t(X) = \frac{H(X)}{\bar{\tau}} \quad \text{où } \bar{\tau} : \text{durée moyenne d'un symbole.}$$

$H_t(X)$ s'exprime en bits/seconde et sera souvent noté R.

— On définit la redondance d'une source comme la différence entre la valeur maximale possible de l'entropie d'une source (lorsque les probabilités des "n" symboles sont égales entr'elles) et sa valeur réelle. On la notera R_S

$$R_S = H_{max}(X) - H(X) \qquad \text{avec } H_{max}(X) = \log n$$

I.3.C. Entropie à l'entrée et à la sortie du canal :

Les perturbations causent souvent une différence entre l'espace observé [Y] et l'espace de départ [X]. Pour caractériser un canal, on définit un espace produit [X.Y] où chaque élément $x_i y_j$ désigne la réalisation simultannée des évènements x_i et y_j.

— L'entropie du champs d'évènements à l'entrée est exprimée par :

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i)$$

— L'entropie du champs d'évènements à la sortie est exprimée par :

$$H(Y) = -\sum_{j=1}^{m} p(y_j) \log p(y_j)$$

— L'entropie du champs réuni entrée-sortie est exprimée par :

$$H(X,Y) = -\sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i, y_j) \log p(x_i, y_j)$$

Une autre notion qu'est l'entropie conditionnelle sera définie comme la valeur moyenne de l'incertitude sur le champs à l'entrée [X], due aux perturbations, sachant le champs de sortie [Y]. Elle sera notée $H(X/Y)$ et désignera

l'entropie du champs X conditionné par le champs Y. Elle est exprimée par:

$$H(X/Y) = -\sum_{x=1}^{n}\sum_{y=1}^{m} p(x_i, y_j)\log p(x_i/y_j).$$

Cependant, il reste à déterminer l'information transmise à travers le canal ; ce qui sera fait par la transinformation.

I.4- Transinformation :

La transinformation $I(X;Y)$ est définie comme étant la valeur moyenne de l'information mutuelle $i(x_i; y_j)$

$$\text{si} \quad i(x_i; y_j) = \log \frac{p(x_i/y_j)}{p(x_i)} = \log \frac{p(x_i, y_j)}{p(x_i)\,p(y_j)}$$

$$\Rightarrow I(X;Y) = \sum_{i=1}^{n}\sum_{j=1}^{m} i(x_i; y_j)\, p(x_i, y_j) = \sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, y_j)\log \frac{p(x_i, y_j)}{p(x_i)\,p(y_j)}$$

La transinformation constitue l'information que l'on obtient sur l'alphabet à l'entrée X par la réception de l'alphabet à la sortie Y. C'est une valeur de l'information transmise à travers le canal.

I.4-a Capacité, redondance et éfficacité du canal discret :

- On définit la capacité du canal comme la valeur maximale de la transinformation. soit si on la note par "C":

$$C = \max I(X;Y).$$

- La redondance du canal sera définie comme la difference entre la capacité du canal et la transinformation. On la note R_c.

$$R_c = C - I(X;Y).$$

- L'efficacité d'utilisation du canal est définie comme étant le rapport entre la transinformation et la capacité du canal. On la note n_c.

$$n_c = \frac{I(X;Y)}{C}$$

Ici, on a fait un aperçu general sur des éléments de la théorie de l'information. On trouvera dans [1-2] les précisions nécessaires.

Les signaux discrets ont été abordés dans ce qui précède et sont d'un grand intérêt pour les systèmes de traitement numériques. Mais avant d'être discrétisés dans le temps et dans l'espace, les signaux continus subissent des opérations intermédiaires telles que l'échantillonnage et la quantification. Ces deux notions seront traitées au prochain chapitre.

CHAPITRE - II

QUANTIFICATION SCALAIRE

Pour être traités par les systèmes numériques, les échantillons successifs d'un signal doivent être transformés en signaux numériques par quantification. Mais avant de passer à la forme d'échantillons, les signaux doivent être traités par un échantillonneur. On parlera d'échantillonnage.

II-1- Echantillonnage - principe et théorème :

L'échantillonnage d'un signal $x(t)$ consiste à remplacer ce signal par un autre signal $x_1(t)$ qui est égal en valeur instantannée à $x(t)$ pendant de brefs instants de durée τ, répétés périodiquement avec une fréquence f_e appelée fréquence d'échantillonnage, et nul entre ces brefs instants.

Le théorème d'échantillonnage dû à shannon, s'énonce comme suit :

Un signal primaire $x(t)$ qui ne contient que des composantes de fréquence inférieure à f_{1max} (spectre borné) peut être entièrement déterminé par des échantillons équidistants prélevés avec une fréquence f_e telle que : $f_e \geq 2 f_{1\,max}$

Cependant, les echantillons successifs de valeurs continues, doivent être approximés par des valeurs discrètes. C'est l'objet de la quantification.

II-2- Quantification - principe et généralités :

La quantification est l'approximation de la valeur instantannée éxacte d'un signal par la plus voisine valeur tirée d'un assortiment de N valeurs discrètes.

Si on designe par X une variable aléatoire, un quantificateur est un appareil qui fait associer à une entrée X comprise dans un intervalle, une sortie y comprise dans le même intervalle.

$$\text{si } x(i-1) \leq X < x(i) \quad \Rightarrow \quad y = y(i)$$

- Les $x(i)$ sont appelés seuils de décision
- Les $y(i)$ sont appelés niveaux de reconstruction.

La différence entre deux seuils de décision consécutifs est appelé pas de quantification et est notée δ, telle que :

$$\delta(i) = x(i) - x(i-1).$$

La caractéristique du quantificateur est en marche d'escalier et elle présente deux formes différentes à l'origine selon que le nombre de niveaux de reconstruction N est pair ou impair. Voir fig-2- dans le cas non uniforme.

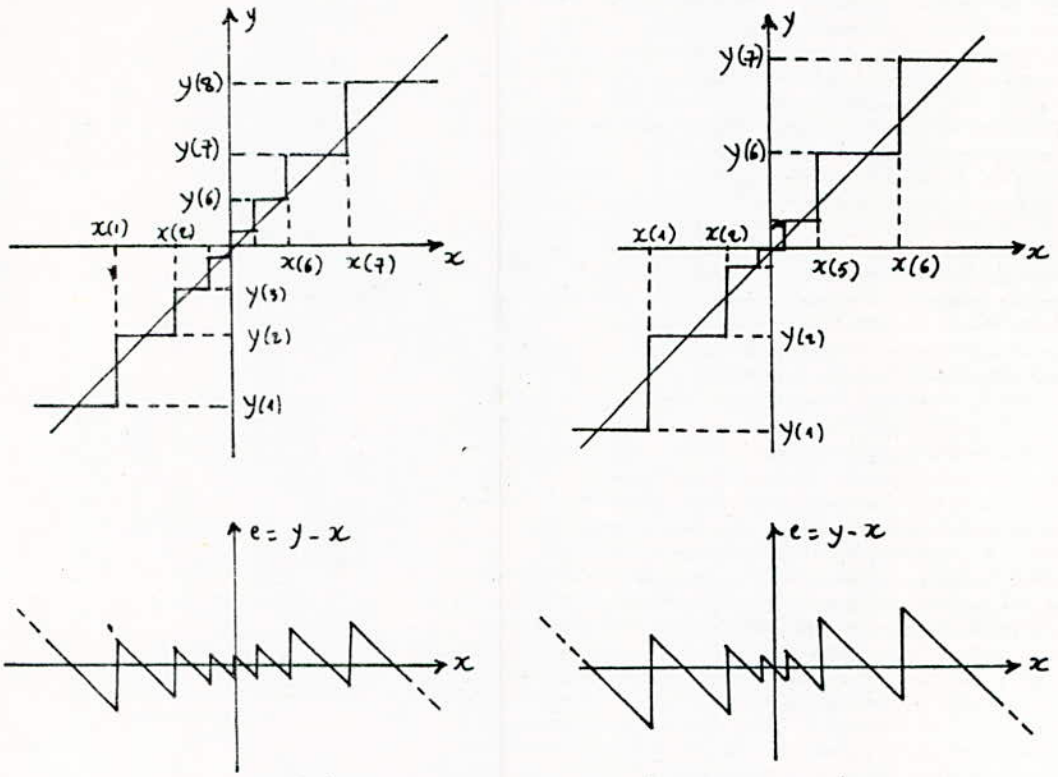


fig-2- Lois de quantification (cas non uniforme).

L'erreur de quantification "e" est donnée en fonction de la loi de quantification Q(x) par: $e = Q(x) - X$.

Le signal d'erreur "e" est de nature aléatoire du moment que les signaux sont en général de nature aléatoire. Il sera appelé ainsi 'bruit de quantification'. Cependant selon que le pas de quantification δ est constant ou variable, on parlera de quantification uniforme ou non uniforme.

II-3- Quantification uniforme :

II-3-α- Définition :

Une loi de quantification uniforme est caractérisée par des seuils de décision X(i) uniformément espacés et par "L" niveaux de reconstruction y(i) situés à mi-chemin des seuils. Voir fig-3

D'où $\delta = X(i) - X(i-1) = \text{constte}$.

$$y(i) = \frac{1}{2}[X(i-1) + X(i)]$$

Si cette loi est symétrique par rapport à l'origine, et le domaine de

variation de X est borné par $\pm X_s$ (X_s étant un niveau de saturation), on écrira :

$X(0) = -X_s$ et $X(L) = +X_s$.

Si $L = 2^b$, chaque valeur quantifiée pourra être représentée par un mot de "b" bits.

D'où $\delta = 2X_s / L = 2X_s 2^{-b}$

On définit le facteur de charge "Γ" du quantificateur par le rapport :

$\Gamma = X_s / \sigma_x$ où σ_x est l'écart type du signal.

La quantification étant toujours accompagnée d'un bruit qu'on a noté "e", on caractérise la qualité de cette opération par le rapport signal sur bruit qu'on note "RSB".

II-3-b- Rapport signal sur bruit pour un quantificateur uniforme :

Le rapport signal sur bruit, noté "RSB", est défini par le logarithme décimal du rapport de la variance du signal à celle du bruit, soit :

$$RSB = 10 \log\left(\frac{\sigma_x^2}{\sigma_e^2}\right) \quad [db].$$

Avant de calculer ce "RSB" dans le cas uniforme, on calculera la variance du bruit "σ_e^2" en admettant que "e" est uniformément distribuée dans l'intervalle $[-\frac{\delta}{2}, +\frac{\delta}{2}]$ (voir [3-6]).

La loi de probabilité de l'erreur "e" est donnée par $p(e) = \frac{1}{\delta}$ du moment que

$$\int_{-\frac{\delta}{2}}^{+\frac{\delta}{2}} p(e)\,de = 1$$

$$\text{D'où } \sigma_e^2 = \int_{-\frac{\delta}{2}}^{+\frac{\delta}{2}} e^2 p(e)\,de = \int_{-\frac{\delta}{2}}^{+\frac{\delta}{2}} e^2 \frac{1}{\delta}\,de = \frac{\delta^2}{12}$$

$$\text{or } \delta = 2X_s 2^{-b} \Rightarrow \sigma_e^2 = \left(\frac{X_s^2}{3}\right) 2^{-2b}$$

Le "RSB" sera par conséquent :

$$RSB = 10 \log\left(3\left(\frac{\sigma_x^2}{X_s^2}\right) 2^{2b}\right) = 6{,}02\,b + 4{,}77 - 20 \log \Gamma$$

La loi de variation du "RSB" en fonction de "b" et de "Γ" est donnée en fig-4.

On remarquera d'après la figure précédente que le "RSB" augmente selon que le nombre "b" de bits alloué à la quantification est grand, ce qui signifie que la qualité de la quantification augmente avec le nbre de niveaux considérés. Cependant, il subsiste le problème relatif à l'augmentation du "RSB" en fonction de "Γ" (soit $-20 \log \Gamma$).

En fait "Γ" dépend de σ_x qui lui même dépend de l'amplitude de X. Finalement donc le RSB dépend de l'amplitude de X.

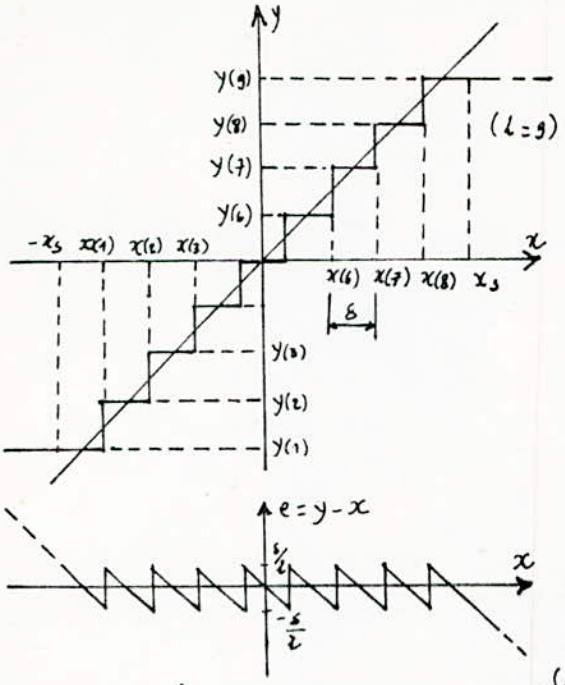


fig-3-Quantification uniforme

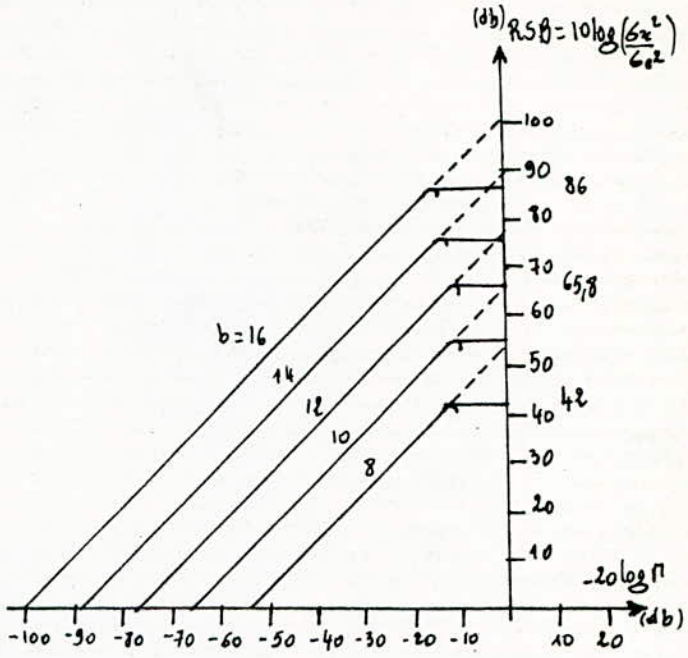


fig-4-RSB pour la quantification uniforme d'un signal gaussien.

Idéalement on souhaiterait que le "RSB" reste constant indépendamment de X. Cela nous fera penser à la quantification non uniforme.

II-4-Quantification non uniforme:

II-4-a-Définition:

On définit la quantification non uniforme comme une loi de quantification dans laquelle le pas de quantification "δ" dépend de l'amplitude du signal X. Cette quantification fait introduire des techniques de compression et extension.

II-4-b-Quantification avec compression-extension:

La fig-5 nous montre l'itinéraire suivi avant d'aboutir à la quantification non uniforme.

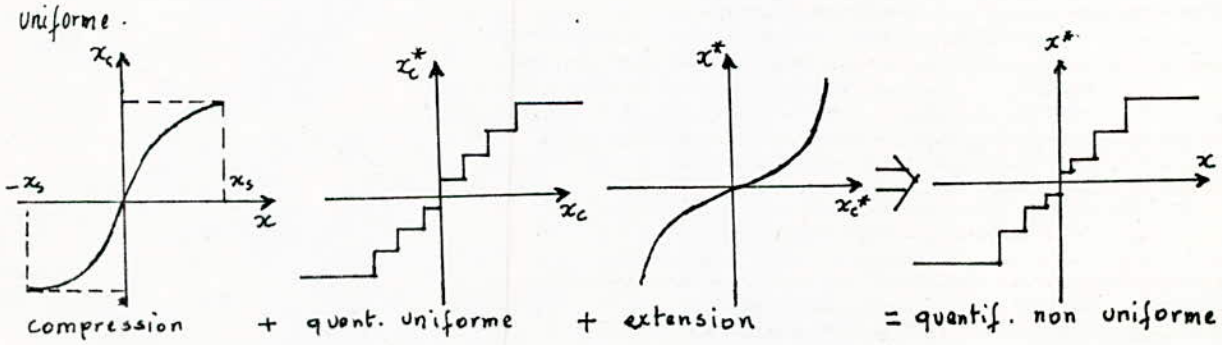


fig-5-Quantification non uniforme.

Comme on le voit sur fig-5, la quantification non uniforme peut être considérée comme une quantification uniforme précédée d'une compression des amplitudes du signal $x(t)$, dont l'effet est de favoriser les faibles amplitudes au détriment des amplitudes élevées. Après quantification uniforme, l'opération se poursuit par une extension qui agit en sens inverse de celui de la compression.

La caractéristique de la quantification non uniforme ainsi que celle de l'erreur résultante "e" sont déjà illustrées dans fig-2.

II-4-C- Caractéristiques de compression:

Il existe deux lois de compression à caractéristiques logarithmiques, l'une appelée la loi A, l'autre la loi μ. La première est répandue en Europe, la deuxième en Amérique. Si on définit "x" et "y" comme amplitudes relatives, soit : $x = |X| / X_s$ et $y = |X_c| / X_s$

On aura pour la loi A :

$$y = \frac{A x}{1 + \log A} \qquad \text{pour} \quad x \leq \frac{1}{A} \qquad (A = 87{,}6)$$

$$\text{et } y = \frac{1 + \log Ax}{1 + \log A} \qquad \text{pour} \quad \frac{1}{A} \leq x \leq 1$$

Pour la loi μ, on peut écrire :

$$y = \frac{\log(1 + \mu x)}{\log(1 + \mu)} \qquad \text{avec } (\mu = 255).$$

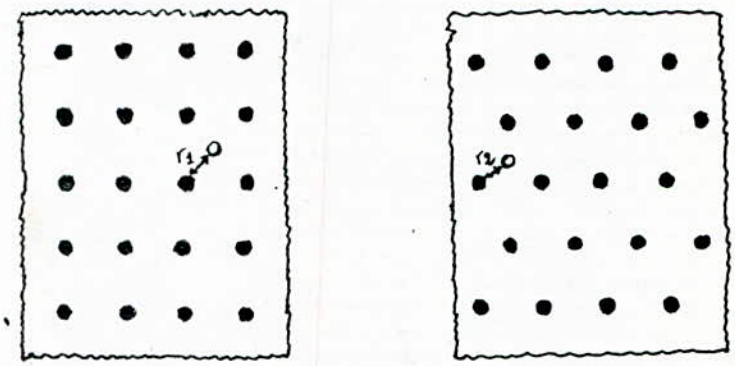
Nos aperçus sur l'échantillonnage et la quantification scalaire sont brefs. Pour plus de détails voir [3-4-5-6]. Cette quantification est dite scalaire parcequ'elle fait une approximation d'échantillons par des nombres réels correspondants aux niveaux de reconstruction y(i). Comme application typique de cette technique, on peut citer la modulation en impulsions codées (MIC) ou (PCM).

La quantification scalaire qui a tant montré de prouesses dans la discrétisation des signaux ne peut descendre au delà de 1 bit/éch qu'est le débit d'codage du signe des échantillons. C'est pour celà qu'on parlera d'insuffisances de la quantification scalaire dans ce qui suivra.

II.5. Insuffisances de la quantification scalaire :

Pour montrer une de ces insuffisances, il suffit de prendre un vecteur à deux dimensions, mais distribué de façon uniforme dans un certain domaine. On essayera de le quantifier par une paire de quantificateurs scalaires et par un autre quantificateur vectoriel qui sera défini prochainement.

Une portion des deux quantificateurs est donnée en fig-6 ; on notera que la densité moyenne des points est la même,



a- Quantif scalaire b- Quantif vectorielle.

fig-6- Comparaison des deux méthodes de quantification pour un vecteur x à 2 dimensions réparti unif. sur le domaine

On remarquera que le point le plus mal quantifié est à distance r_1 du quantificateur voisin le plus proche dans le cas scalaire. Dans le cas vectoriel, le point le plus mal quantifié est à distance r_2 du plus proche voisin. On voit bien que r_2 est inférieur à r_1 ; par conséquent le quantificateur vectoriel est le plus performant question erreur de quantification.

Une autre insuffisance, déjà citée, est qu'on ne peut quantifier au dessous de 1 bit/éch dans le cas scalaire ; tandis que dans le cas vectoriel, on peut descendre à un débit de $1/K$ (bit/ech) ; soit un bit par vecteur à K échantillons.

Toutes ces insuffisances nous incitent à parler d'une autre méthode de quantifier qu'est la quantification vectorielle.

CHAPITRE - III
QUANTIFICATION VECTORIELLE

III_1. Notions de base :

III.1.a. Objectif :

L'objectif principal de la quantification vectorielle est la compression de données. Cette operation rend l'utilisation des canaux de communication plus éfficace grâce à la réduction de la bande de fréquence occupée par le signal. Elle réduit aussi la capacité des systèmes de stockage de données.

Les systèmes de compression sont de deux sortes :

— Systèmes basés sur des transformations qui conservent l'entropie de la source intacte, mais qui réduisent la redondance lors de la transmission par le canal. C'est ainsi que le message doit être réstitué sans distorsions. A remarquer que ces transformations sont reversibles. [1].

— Systèmes basés sur des transformations qui réduisent l'entropie de la source et qui sont par conséquent irréversibles. Ces transformations introduisent des distorsions [1]; et à un taux de distorsions admis correspond un taux de compression donné.

La quantification vectorielle s'avère d'un interêt particulier pour la compression du signal de parôle ou celui d'image. Ces signaux qui sont de débits assez larges seront ramenés à des débits faibles. Avant de passer à l'aspect distorsion et définition du quantificateur optimal, qûen est-il de la signification éxacte de la quantification vectorielle.

III-1.b- Définitions et généralités :

La quantification vectorielle est une opération qui généralise la quantification scalaire. Elle concerne la représentation d'un vecteur x dontles k composantes sont à valeurs réelles continues ($x \in \mathbb{R}^k$) par un vecteur y_i appartenant à un ensemble fini $\{ y_i \in \mathbb{R}^k, i = 1, 2, \ldots, M \}$.

Les vecteurs y_i sont dits vecteurs arrondis. Ils constituent un dictionnaire de points dans $\mathbb{R}^k$. Pour la transmission ou pour l'enregistrement un code est attribué à chaque vecteur y_i. Il s'agit engénéral de mots binaires à B bits et dans ce cas.

on a $M = 2^B$ où M est le nombre de vecteurs arrondis y_i.
Le nombre moyen de bits par échantillon est noté R, il vaut $R = B/K$

La quantification vectorielle peut être algébrique ou statistique. Dans le premier cas, elle utilise comme dictionnaire des réseaux de points réguliers, ou profite des règles de transition des treillis, tandis que dans le second cas, elle opère une partition de $\mathbb{R}^K$ en classes C_i représentées par des centroïdes y_i à partir de séquences d'entrainement.

Les applications de cette quantification sont éssentiellement le codage à faible débit (moins de 8 Kbits/s) et la reconnaissance.

La quantification vectorielle doit être organisée pour minimiser la distorsion moyenne D pour un débit R donné. Qu'en est il de la distorsion et du quantificateur optimal.

III-1-C - Distorsion et quantificateur optimal :

III-1-C-1- Mesure de distorsion :

La distorsion introduite par la quantification vectorielle peut être mesurée en introduisant une "distance" $d(x,y)$ entre les vecteurs x et y. Une telle distance doit être une fonction réelle, non négative et sera utilisée aux fins de comparaisons. Elle doit être telle que la distance d'un point à lui même soit toujours la plus petite que de ce point à tout autre point.

Comme distances courantes, on peut citer la distance cepstrale, celle d'Itakura-Saito et celle basée sur le rapport de vraissemblance. Ces distances conviennent à des signaux bien particuliers [6].

Les distances d'usage général sont de la forme :

$$d(x,y) = |x-y|^r .$$

En particulier pour $r=2$, d devient le carré de la distance euclidienne, soit:

$$d(x,y) = |x-y|^2 = \sum_{j=0}^{K-1} (x_j - y_j)^2$$

Cette distance sera souvent utilisée vu la facilité de calcul procurée et vu son interprétation directe comme une puissance d'érreur de quantification.

III-1-C-2-Fonction "taux de distorsions" :

La fonction "taux de distorsions" caractérise la relation entre les distorsions, admises dans une transformation qui réduit l'entropie, et la valeur minimum du

débit d'information de la source éffective (circuit de compression). En effet, en réduisant le débit d'information (soit en augmentant la compression) les distorsions augmentent.

Cependant la réduction de l'entropie dans le processus de compression est équivalente à l'effet des perturbations dans un canal de transmission, ce qui nous fera penser à la transinformation qu'on peut éxprimer par : (voir [1])

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

La fonction "taux de distorsions" est définie par la relation

$$R(\delta) = \min I(X;Y) = \min [H(Y) - H(Y/X)]$$

Ce minimum est soumis à la condition que les distorsions $d(x,y)$ ne dépassent pas une valeur δ (à ne pas confondre avec le pas de quantification scalaire noté δ) admise.

soit $d(x,y) \leq \delta$.

La fonction $R(\delta)$ représente le débit d'information minimum à la sortie du circuit de compression (quantificateur vectoriel) qui garantit que les distorsions admises δ ne seront pas dépassées.

Ainsi donc, une quantification vectorielle (compression) transforme la source de débit d'information $H(x)$ bits/s en une nouvelle source de débit $R(\delta)$ bits/s, avec $R(\delta) < H(X)$ et de manière que les distorsions ne dépassent pas une valeur δ admise.

III-1-C-3-Quantificateur optimal:

Si l'on connait le nombre M de vecteurs arrondis, la distance "d" de travail et la distribution $p(x)$ du vecteur d'entrée, on peut reconnaître le quantificateur optimal. C'est de tous les quantificateurs $Q(x)$, ou tous les ensembles y_i, celui qui minimise l'espérance mathématique de la distance soit :

$$E(d) = \int d(x, Q(x))\, p(x)\, dx .$$

Construire un quantificateur, qui pour une situation donnée s'approche des performances du quantificateur optimal est une tâche laborieuse. On s'efforce cependant de réaliser de tels quantificateurs et celà par des algorithmes appropriés, selon que la quantification est vue de l'approche statistique ou algébrique. Ces deux approches seront passées en revue dans ce qui suivra.

III-2- Quantification vectorielle statistique:

La quantification vectorielle statistique est utilisée couramment pour le codage comme pour la reconnaissance de la parole. Les paramètres du système de quantification sont obtenus par entraînement à partir d'un grand ensemble, suffisamment représentatif des données. Divers algorithmes seront cités ici de façon brève, mettant en relief diverses méthodes de quantification statistique.

III-2-a- Methode de Lloyd généralisée (méthode de la K-moyenne):

On dispose d'un ensemble de L vecteurs x que l'on désire positionner en M classes. On designera par:

- $x_j^{(i)}$, les vecteurs appartenant à la classe i
- y_i, le centroïde de la classe i
- L_j, le nombre de vecteurs de la classe i.
- $d(x_j^{(i)}, y_i)$, la distance ou mesure de distorsion entre $x_j^{(i)}$ et y_i; dans notre cas, on a convenu d'utiliser la distance euclidienne.
- D_i, la distorsion totale de la classe i, soit:

$$D_i = \sum_j d(x_j^{(i)}, y_i)$$

- D, la distorsion pour l'ensemble des vecteurs, soit:

$$D = \sum_{i=1}^{M} D_i$$

Le problème consiste ici à trouver la partition et les centroïdes de façon à minimiser la distorsion totale D sachant qu'un nombre de classes M est imposé à priori. Une procédure itérative s'impose, tout en soulignant que:

- Pour un ensemble donné de centroïdes, la partition qui minimise D est celle pour laquelle chaque vecteur x_j est affecté à la classe dont le centroïde est le plus rapproché.
- Pour une partition donnée, il existe pour chaque classe i un vecteur y_i qui minimise la distorsion totale de la classe D_i.

L'algorithme qui en resulte sera:

1ère étape: On fait un choix de M centroïdes y_i répartis d'une façon aléatoire dans $\mathbb{R}^K$

2ème étape: On affecte l'ensemble des vecteurs x_j aux diverses classes et celà en calculant toutes les distances $d[x_j, y_i]$ $(i = 1, 2, \ldots M)$. x_j sera associé au centroïde y_i le plus proche.

3ème étape : On recalcule la position de chaque centroïde y_i, pour minimiser chaque distorsion D_i. Si la distance utilisée est la distance euclidienne, celà revient à calculer un centre de gravité ou un barycentre.

4ème étape : On calcule la distorsion totale D.

5ème étape : On itère jusqu'à ce que D varie de moins de ε% d'une itération à la suivante (par exemple $\varepsilon = 1$).

Cependant il est à remarquer que cet algorithme converge vers une situation sous-optimale (minimum local) qui est fonction du choix initial des centroïdes. Il peut y avoir intérêt à recommencer plusieurs fois la procédure avec des choix initiaux différents pour finalement retenir la solution qui fournit la distorsion la plus faible. Pour ce qui est de la distribution de la source, cet algorithme en tire profit et de façon avantageuse. [6-7]

III-2-b- Algorithme d'apprentissage à seuil :

Soit $\{x_l ; l = 1, 2, \dots, L\}$ la séquence d'entrainement. On se fixe à priori une borne supérieure (ou seuil) ε pour le rayon de chaque boule constituant une classe C_i. L'algorithme est tel que :

1ère étape : Le premier vecteur x_1 crée la première classe C_1

2ème étape : A chaque nouveau vecteur x_j une décision est prise :

- Si $d(x_j, y_i) > \varepsilon$ pour toutes les classes C_i déjà éxistantes, x_j crée une nouvelle classe.
- Sinon x_j est affecté à la classe dont le centroïde est le plus proche On recalcule ensuite la position du nouveau centroïde.

Cet algorithme a le privilège d'être simple. Il peut être interrompu et repris ensuite avec une nouvelle séquence d'entrainement. Le nombre de classes est fonction du seuil ε choisi et plusieurs experiences sont nécéssaires pour obtenir un nombre de classes voisin de celui souhaité. A remarquer que le paramètre ε est intéressant à controler du moment qu'il est directement relié à la qualité de codage souhaitée.

Les deux algorithmes précédents font une recherche exhaustive du plus proche voisin. C'est ainsi que cette recherche devient complexe dès que le nombre de centroïdes est très grand. C'est là que l'algorithme LBG intervient pour réduire le nombre de calculs.

III_2_C_ Algorithme par éclatements binaires (Procédure LBG):

Le symbole LBG est la contraction des noms de ses inventeurs (Linde, Buzo et Gray). Cet algorithme réduit le nombre de calculs de distance nécessaires à l'affectation d'un vecteur x à une classe C_i. En effet, le nombre de calculs de distance qui était auparavant M pour M classes est réduit et devient $2B$ calculs de distance si $M = 2^B$. Ces calculs de distance sont répartis sur un arbre binaire (voir fig-7), où sont disposées M classes C_i, crées par éclatements successifs comme suit :

1ère étape : On choisit au départ deux centroïdes y_1 et y_2.

2ème étape : On affecte tous les vecteurs x_j aux deux classes précédentes et on optimise par itération comme expliqué dans l'algorithme de la K-moyenne. On obtient ainsi deux classes initiales avec leurs deux centroïdes optimisés.

3ème étape : On éclate ensuite chacune des deux classes, en perturbant légèrement les composantes des centroïdes de la façon suivante :

A partir d'un centroïde y_1 par exemple on forme deux autres centroïdes y_{12} et y_{11} tels que $y_{11} = 0,99\, y_1$ et $y_{12} = 1,01\, y_1$.

Les deux centroïdes y_{12} et y_{11} vont éclater à leur tour la classe C_1 en deux classes qui chacune d'elles sera optimisée à son tour par un calcul de centroïde.

4ème étape : On poursuit les éclatements binaires successifs jusqu'à l'obtention du nombre de classes désiré soit $M = 2^B$.

On peut remarquer que l'affectation d'un vecteur x, s'effectue par affectations successives en suivant un parcours dans l'arbre des classes trouvées à chaque niveau d'éclatement. Cette méthode mémorise $2M$ centroïdes au lieu de M pour la K-moyenne mais ne fait que $2B$ calculs de distance au lieu de 2^B (voir [8]).

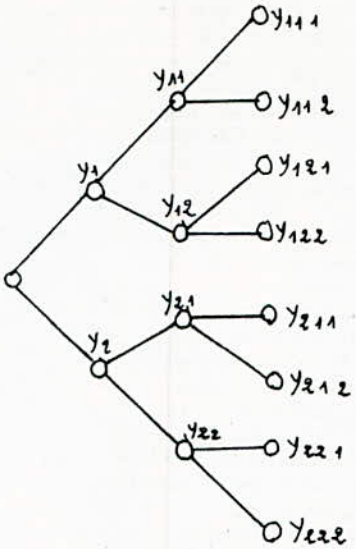


fig-7- Arbre des centroïdes ($B=3$)

III_3- Quantification vectorielle algébrique :

Dans le chapitre II, on avait évoqué l'insuffisance du quantificateur scalaire et on avait montré dans fig6-b un quantificateur vectoriel particulier. En fait, ses points présentent une structure mathématique telle que leurs coordonnées peuvent être exprimées en fonction de deux vecteurs de base. Cette quantification qui optimise l'erreur quadratique et qui utilise des structures mathématiques particulières pour former ses vecteurs arrondis sera dite algebrique. Cette méthode sera d'un intérêt particulier dans le cas où on a une distribution de points uniforme. C'est dans ce contexte que Newman [9] a montré que le réseau hexagonal A_2 à deux dimensions conduit au quantificateur optimal pour l'erreur quadratique.

Gersho a émis l'hypothèse que pour toute dimension il éxiste un réseau régulier qui atteint les performances optimales. On a pu déterminer ainsi des réseaux réguliers de dimension 8 et 24 particulierement bons.

Une méthode de quantification qui exploite les propriétés de ces réseaux poor quantifier les formes d'onde est la quantification vectorielle sphérique.

III_3_a_ Quantification vectorielle sphérique :

Afin de quantifier éfficacement les signaux échantillonnés, nous les découpons en blocs de M échantillons qui à leur tour seront les M composantes de vecteurs à M dimensions. soit: $X = [x_1\ x_2 \dots x_M]^T$

La quantification vectorielle sphérique consiste à quantifier séparément la norme G (ou gain) et l'orientation x (ou phase) du vecteur X, soit :

$G = \sqrt{\Sigma x_i^2}$ la norme

$x = X/G$ l'orientation du vecteur normalisé.

La mise en œuvre d'un tel quantificateur nécessite la spécification d'un ensemble de vecteurs normés (points sur la sphère unité) qui seront les valeurs arrondies possibles pour l'orientation du vecteur.

Commençons par la quantification de l'orientation. Elle utilise un quantificateur Q_0 tel que $\{ y_i / i = 1, 2, \ldots, N-1 \}$ et qui fournit le vecteur y_i qui minimise la quantité :

$$d(x, y_i) = \| x - y_i \|^2 = \|x\|^2 + \|y_i\|^2 - 2 x^T y_i = 2 - 2 x^T y_i .$$

Minimiser cette distance revient à maximiser le produit scalaire $x^T y_i$ qui est la projection de l'orientation x sur le vecteur y_i. Soit P la projection maximale.

$$P = \max_i \{ x^T y_i \}$$

La projection maximale sera PG dans le cas où le produit scalaire est calculé à partir du vecteur X non normalisé.

Soit : $PG = \max_i \{ X^T y_i \}$.

La fig-8 montre une situation où le vecteur normalisé x est approximé par y_i.

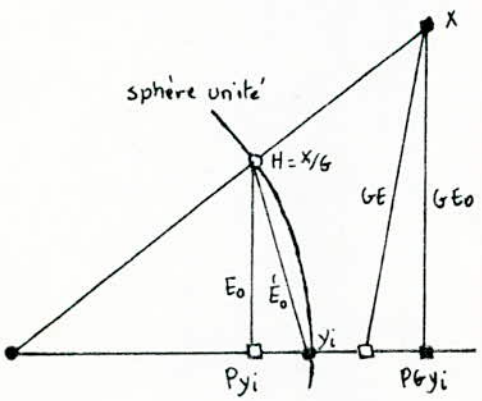


fig-8 Situation typique de quantif. vectorielle sphérique - La sphère de rayon unité est représentée par un arc de cercle - le vecteur y_i retenu est celui donnant la plus grande projection P.

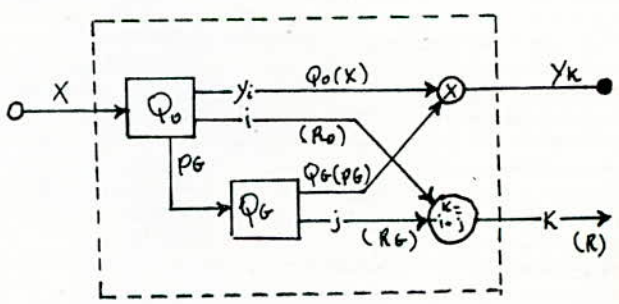


fig-9 Diagramme de la quantification vectorielle sphérique

Une meilleure approximation du vecteur x est donnée par le vecteur Py_i qui donne une erreur E_0 toujours inférieure à E_0'.

L'orientation étant quantifiée, qu'en est-il de la norme?.

On doit donc disposer d'un deuxième quantificateur Q_G qui approxime la quantité PG qui est telle que $PG = \max_i \{x^T y_i\}$.

Finalement la quantification vectorielle sphérique se résume à un quantificateur $Q(\cdot)$ qui donne :

$$Y_K = Q(X) = Q_0(x)\, Q_G(PG) \qquad \text{voir fig. 9.}$$

et $R = R_0 + R_G$

Reste maintenant à savoir comment la quantification vectorielle utilise les réseaux quand elle est sphérique ? et qu'en est il de ces réseaux?.

III-3-b- Réseaux réguliers de points:

III-3-b-1- Définition et méthode de construction :

Un réseau régulier dans $\mathbb{R}^n$ est l'ensemble des points x qui s'obtiennent par combinaison linéaire de n vecteurs de base indépendants $z_1, z_2, \ldots, z_n$ avec des coefficients de proportionnalité entiers $K_1, K_2, \ldots, K_n$.

$$L_n = \left\{ x \,/\, x = \sum_i K_i z_i \right\} \; ; \; K_i \text{ entier.}$$

Comme illustration typique, on a le réseau de la fig-6-b obtenu à partir de vecteurs de base $z_1 = [1, \sqrt{3}]$ et $z_2 = [2, 0]$.

Pour ce qui est de la construction de ces réseaux, Leech et Sloane [9] ont proposé diverses méthodes de construction à partir d'un code correcteur d'erreurs "C". La construction A consiste à prendre tous les points de $\mathbb{Z}_n$ qui sont congrus modulo 2 par composante, à un mot du code "C".

soit : $L = \left\{ x/x = [x_1, x_2, \ldots, x_n] \in \mathbb{Z}_n \text{ et } [x_1 \bmod 2, x_2 \bmod 2, \ldots, x_n \bmod 2] \in \text{"C"} \right\}$

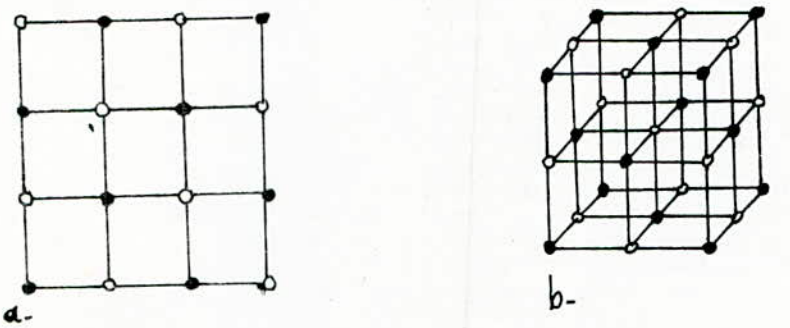
Les travaux connus, les plus interessants, pour la quantification vectorielle spheriques [9] sont : - Le réseau de Gosset en huit dimensions.
- Le réseau de Barnes-Wall en 16 dimensions.
- Le réseau de Leech Λ_{24} en 24 dimensions.

Le réseau de Gosset E_8 sera explicité ici afin de montrer un exemple de quantification vectorielle sphérique par réseau. Voyons donc sa définition et quelques unes de ses propriétés.

III_3_b_2_ Réseau de Gosset E_8 :

Le réseau de Gosset est défini comme l'union de deux réseaux obtenus par translation du réseau $2D_8$ par les deux mots du code à répétition : $[0,0,0,0,0,0,0,0]$ et $[1,1,1,1,1,1,1,1]$; Soit $E_8 = (2D_8) \cup (2D_8 + [1,1,1,1,1,1,1,1])$

Par réseau D_n, on désigne un réseau contenant les points de $\mathbb{Z}^n$ dont la somme des coordonnées est paire. On a représenté en fig10-a un réseau D_2 de dimension 2 et dont la forme est celle d'un damier. Le réseau cubique D_3 à faces centrées est représenté en fig-10-b.



a- b-

fig-10- Réseaux Dn pour les dimensions 2 et 3.

La notation $2D_8$ signifie que les composantes des points de $\mathbb{Z}^8$ sont paires. On a constaté cependant, que les points de E_8 ont une répartition particulière. En effet, ils sont répartis sur des sphères centrées à l'origine et de rayons $2\sqrt{2m}$, où m est un entier naturel. Ainsi donc, $x \in S(0, 2\sqrt{2m})$.

$S(0, 2\sqrt{2m})$ sera désignée par $E_8(m)$.

Cette propriété de répartition sur des sphères centrées à l'origine s'applique aussi pour les réseaux de dimension 16 et 24, Les propriétés particulières des points de $E_8(m)$ qui seront utilisés comme quantificateurs seront décrites.

Propriétés des points de $E_8(m)$:

L'ensemble des points d'une sphère sera désigné par $E_8(m)$ et leur nombre sera $N_8(m)$:

$E_8(m) = \{ y / y \in E_8 \cap S_m \}$ où S_m est la sphère de rayon $2\sqrt{2m}$.

Tous les points d'une sphère "m" ordonnés entre 0 et $N_8(m)-1$ seront les vecteurs y_i de notre quantificateur. Ces points seront partitionnés en "P_m" classes. Dans chaque classe l'élément maximal sera appelé "Leader" et ses composantes vont en décroissant de la position "1" à "8". En permutant les composantes d'un "leader" on obtient tous les points de la classe correspondante "j" et leur nombre sera donné par la formule :

$$K(j) = \frac{8!}{w^0!\, w^1! \dots w^{q-1}!}$$

"q" étant le nombre de valeurs a_i $(i = 0, \dots, q-1)$ distinctes que prennent les composantes du "Leader" et w^{q-1} le nombre de répétitions de la valeur distinctes a_{q-1} $(a_0 > \dots > a_{q-1})$.

Pour déterminer le rang d'un vecteur, "t", parmi K vecteurs d'une classe "j" on utilise la formule de rang de schalkwijk [9-10]

$$t = \sum_{K=1}^{8} \sum_{d=0}^{d(K)-1} \frac{(M-K)!}{(w_K^d - 1)! \prod_{\substack{i=0 \\ i \neq d}}^{q-1} (w_K^i!)}$$

- w_K^d sera le nombre de répétitions de la valeur a_d dans les composantes du vecteur x restant à partir de la position K (c.a.d de K à 8).
- $d(K)$ est l'indice de la Kième composante $a_{d(K)}$

On fait les conventions suivantes : $0! = 1$ et $(-1)! = \infty$ et $M = 8$. Un exemple de calcul de t et les détails relatifs aux classes et aux "leaders" sont donnés dans [10].

Mais qu'en est il de l'utilisation des sphères $E_8(m)$ pour la quantification vectorielle sphérique.

III-3-C- Algorithme de quantification vectorielle sphérique par $E_8(m)$:

Soit $x = (x_1, x_2, \dots, x_8)$ un vecteur quelconque de $\mathbb{R}^8$ à quantifier par $E_8(m)$ (c.a.d les vecteurs de la sphère m). L'algorithme se décompose en 5 étapes :

1ère étape : Recherche du "Leader" $\tilde{x}$ de x en réordonnant les composantes du vecteur x par ordre décroissant des valeurs (plus grande valeur en position 1).

2ème étape : Recherche du plus proche voisin de $\tilde{x}$: $\tilde{y}_j^*$, ce qui revient à rechercher parmi les $P_8(m)$ "Leaders" de $E_8(m)$, celui qui maximise le produit scalaire

$$\tilde{x}^T \tilde{y}_j^* = \max(\tilde{x}^T \tilde{y}_j) \qquad j \in (1, \ldots, f_8(m))$$

3ème étape : Détermination du plus proche voisin y de x dans $E_8(m)$.

Pour avoir le plus proche voisin y de x, il faut venir à partir de $\tilde{y}_j^*$ (le "leader" $\tilde{y}_j^*$ le plus proche de $\tilde{x}$) au vecteur y et celà par la permutation inverse éffectuée à l'étape 1 pour partir de x à $\tilde{x}$.

4ème étape : Détérmination du rang "t" de y dans sa classe j* par la formule de shalkwijk avec sa deuxième ecriture :

$$t(y) = \sum_{K=1}^{8} \frac{(8-K)!}{\prod_{i=0}^{q-1} (w_K^i!)} \left(\sum_{d=0}^{d(K)-1} w_K^d \right)$$

5ème étape : Détermination de l'indice i du vecteur y dans $E_8(m)$

$$i = t(y) + \sum_{j=1}^{j^*-1} k(j).$$

Fin.

Les deux approches de la quantification vectorielle qui sont la statistique et l'algébrique ont été abordées sans parler de leur ressemblance ou dissemblance.

III-4 - Etude comparative entre les quantifications algébrique et statistique :

Les techniques de quantification vectorielle déja citées ont été classées en deux catégories, l'une algebrique et l'autre statistique. La premiere technique est très performante quand la distribution de la source est uniforme. Dans ce cas particulier elle présente des avantages tels que :

- Le dictionnaire des vecteurs arrondis y_i n'a pas été stocké entièrement, et seuls les "leaders", en petit nombre, le sont. Les autres vecteurs y_i s'obtiennent par permutations des composantes des "leaders". Tout celà permet d'envisager des dictionnaires 100 à 10000 fois plus grands que pour la technique statistique. C'est le cas aussi pour les dimensions des vecteurs qui peuvent aller jusqu'à 24 (cas du réseau de Leech).

- Dans le cas algébrique, la recherche du plus proche voisin n'est pas exhaustive comme c'est le cas pour la K-moyenne ; elle s'opère uniquement sur les "leaders". Elle est par conséquent accélérée.

Comme inconvénient, cette méthode ne tire pas parti de la distribution de la source considérée. C'est là que la méthode statistique intervient et rend

compte de la distribution de la source. Les performances réalisées en matière de "RSB" sont bonnes (RSB: rapport signal sur bruit) quand on envisage des dictionnaires de tailles très importantes. Malheureusement, les contraintes technologiques en matière de stockage et de complexité de calcul sont là pour limiter ces dictionnaires et les performances réalisées.

L'idéal serait donc d'associer ces techniques pour constituer des quantificateurs hybrides qui d'une part profitent de la structure algébrique pour réduire les contraintes en matière de stockage de dictionnaires, d'autre part, de la structure statistique pour s'adapter à toutes les sources utilisées. Les quantificateurs hybrides sont décrits en détail dans [10].

Le RSB déja cité peut être défini comme le logarithme à base 10 du rapport de la variance du signal σ^2 à quantifier sur l'erreur moyenne D.

Soit : $RSB(R) = 10 \log_{10}\left(\frac{\sigma^2}{D}\right)$

Pour une source gaussienne sans mémoire à un débit fixe R (bit/ech), l'érreur moyenne D(R) minimum [11] est donnée par :

$$D(R) = 2^{-2R}$$

Le RSB max sera :

$$RSB_{max}(R) = 10 \log_{10}\left(\frac{\sigma^2}{D}\right) = 6{,}02 \times R \qquad \text{si } \sigma^2 = 1$$

La fig-11 montre l'évolution du RSB en fonction du nombre "n" d'échantillons (ou délai n de traitement), et ce pour diverses méthodes de quantification. La source considérée est gaussienne, et le débit est fractionnel, égal à $\frac{1}{2}$. Les différentes techniques de quantification montrées dans cette figure ne le sont qu'à titre indicatif. Cependant on voit apparaître une technique de codage avec un "RSB" nettement élevé pour des délais "n" grands. Cette technique est le "TCQ" contraction du mot original en anglais "Trellis Coded Quantization".

C'est à la base d'articles originaux reçus de l'université canadienne "sherbrooke" que l'on s'efforcera de faire une approche de cette technique, toute nouvelle, de quantification et codage par treillis.

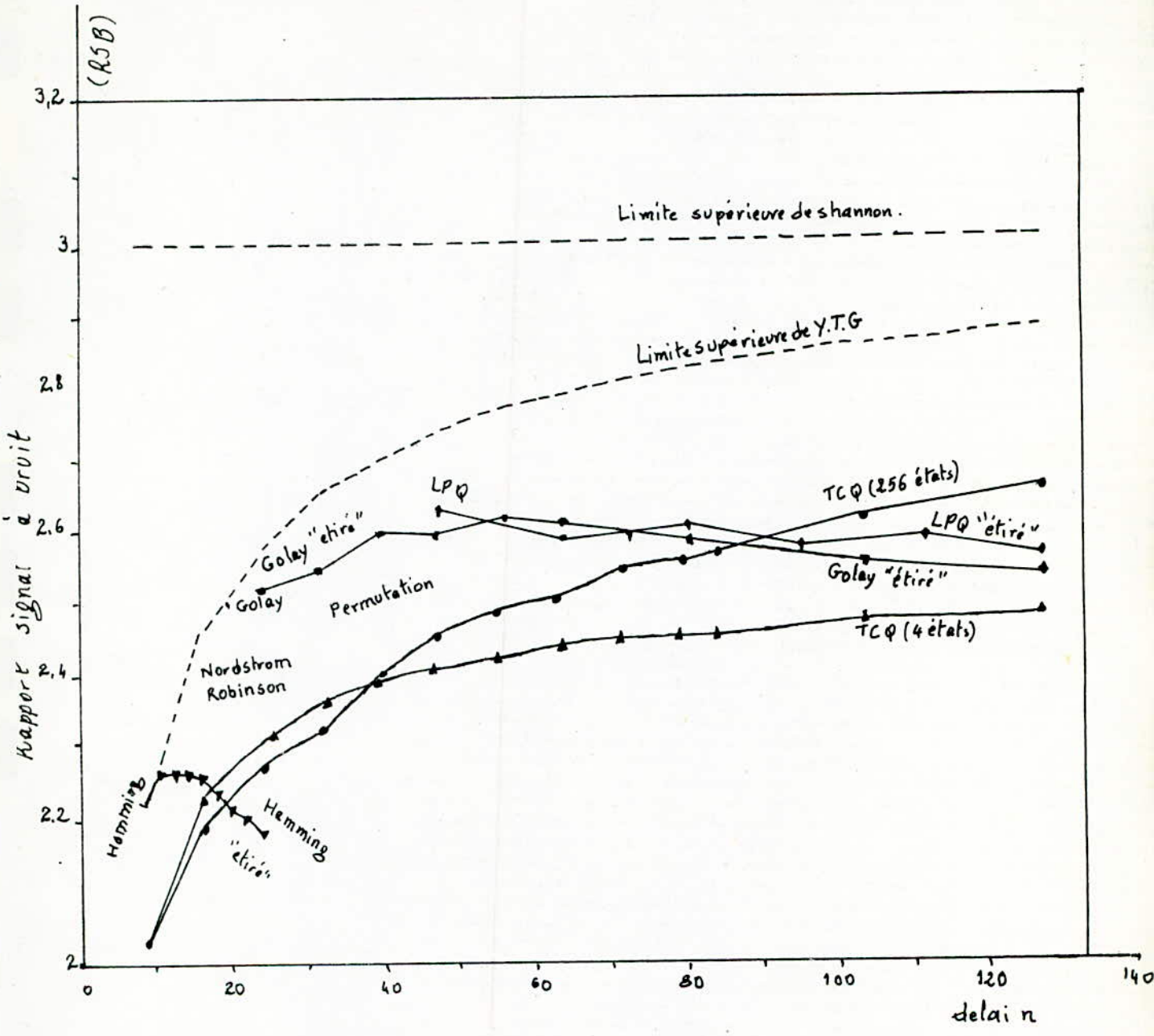


fig-11.RSB de quelques techniques de quantification d'une source Gaussienne à 1/2 bit/ech et à delai fini. [11]

CHAPITRE-IV

QUANTIFICATION ET CODAGE PAR TREILLIS

ou "TRELLIS CODED QUANTIZATION"

IV-1. Origine :

La théorie de la modulation et celle du codage de source, ont bénéficié ces dernières années des larges applications du codage par treillis. L'exploitation de treillis, dans le cas de la modulation, fût l'œuvre pour la première fois d'Ungerboeck [12-13-14], réalisant ainsi le "Treillis Coded Modulation" ou "TCM".

Profitant de la dualité entre la modulation et le codage de source, Marcellin et Fischer [15] ont généralisé la notion de codage par treillis, cette fois pour le codage de source. Celà a été nommé "TCQ" ou "Trellis Coded Quantization". Avec la quantification par treillis (TCQ), la taille de l'alphabet vectoriel va être doublée tout en conservant le débit intact. Celà est possible à condition d'imposer quelques règles d'utilisation sur le nouvel alphabet.

Avant de définir les treillis utilisés pour la quantification et le codage, il est nécessaire de passer en revue les codes convolutifs même si celà est de façon brève.

IV-2. Codes convolutifs non systématiques :

Un code convolutif est dit systématique si à chaque bloc de longueur n_0 correspondent K_0 symboles d'information et m_0 symboles de contrôle.

Il sera dit non systématique si l'on ne peut séparer les blocs en K_0 symboles d'information et m_0 de contrôle. On pourra apporter quelques précisions au sujet de ces codes non systématiques [1-2] :

- les symboles d'information et de contrôle étant inséparables, ils seront dits symboles de mots codes et seront notés u_i.
- Si à la sortie du codeur, un bloc de n_0 symboles résulte de l'application à l'entrée de K_0 symboles d'information, on dira que le taux d'émission est de $R = \frac{K_0}{n_0}$
- La contrainte "m" sera égale au nombre de blocs de n_0 symboles, lesquels sont reliés entre-eux du fait qu'un symbole d'information quelconque appliqué

à l'entrée est prise en compte dans le calcul d'aumoins un des no symboles u_j de chacun des "m" blocs.

Cependant, les codes seront pris ici dans le contexte de génération de treillis. On les utilisera plus particulièrement pour générer un treillis à 4 états lequel fera l'objet de notre travail vu la simplicité qu'il presente et l'illustration complete qu'il apporte au "TCQ".

IV.-3 - Construction du treillis à 4 états :

Considérons un codeur convolutif de taux d'emission $R = \frac{1}{2}$, de contrainte $m = 3$ et $no = 2$. Le codeur comportera un registre à décalage à $K = 2$ cellules tel que montré sur fig12. L'état du codeur sera donné par le contenu des deux cellules C_1 et C_2 du registre à décalage (2 cellules $\Rightarrow 2^2 = 4$ etats).

Pour illustrer le fonctionnement de ce codeur, supposons que l'état initial du codeur est [0 0] et qu'on applique à l'entrée à l'instant d'horloge $t = 1$, le symbole $x_1 = 1$. A la sortie du codeur apparaîtront les symboles $u_1^{(1)} = 1$ et $u_1^{(2)} = 1$ lesquels forment le bloc $u_1 = [11]$. Si maintenant, on passe à l'instant suivant $t = 2$ d'horloge, le registre passe à l'état [10]; et si encore on applique à l'entrée le symbole $x_2 = 1$, on aura à la sortie le bloc $u_2 = [01]$. Le fonctionnement total est donné au tableau indiqué par fig-13.

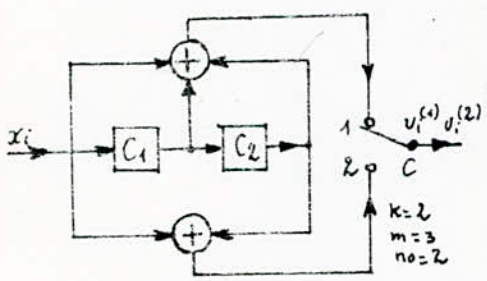


fig-11. Codeur convolutif à 2 cellules

t_i	1	2	3	4	5	6	7	8
x_i	1	1	1	0	1	0	0	0
C_1C_2	00	10	11	11	01	10	01	00
$v_i^{(1)}$ $v_i^{(2)}$	11	01	10	01	00	10	11	00

fig-13- Table de fonctionnement du codeur.

A remarquer que l'apparition d'un certain bloc (par exemple [11]) depend seulement de l'état du registre (lequel dans ce cas est [01]), ainsi que du symbole appliqué à l'entrée (lequel est 1). Cela nous fait penser au diagramme des états du registre qu'on representera en fig-14.

On détermine ainsi avec ce diagramme facilement les blocs à la sortie du codeur, en fonction des symboles appliqués à l'entrée.

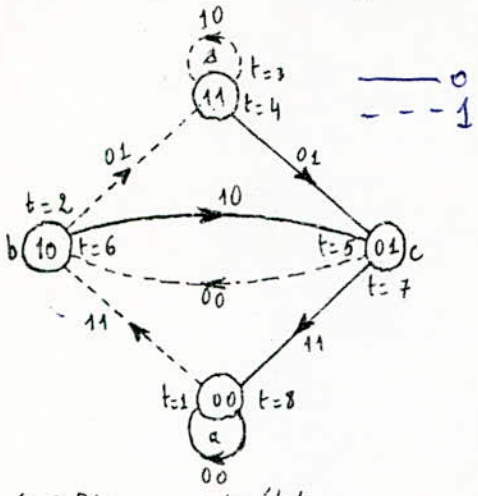


fig-14-Diagramme des états.

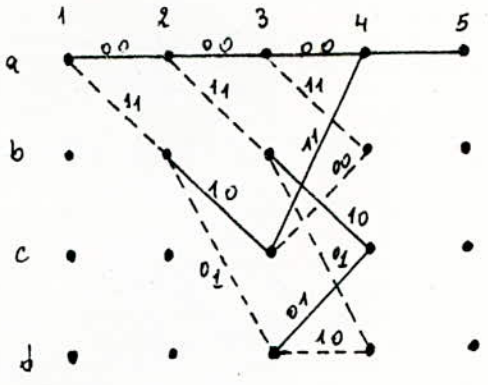


fig-15. Diagramme du treillis.

Si maintenant, on représente en fonction du temps toutes les séquences (mots codes) que le codeur peut générer durant l'intervalle de temps considéré, la structure obtenue sera appelée "treillis". Cette structure est illustrée en fig-15. Les états du codeur sont représentés par des nœuds (4 états ⇒ 4 nœuds).

La technique de quantification par le treillis (TCQ) utilise des concepts introduits par Viterbi [16] et plus particulièrement son algorithme, mais en passant du concept probabiliste à celui de la distance euclidienne. C'est ainsi que les échantillons quantifiés seront tels que leur distance euclidienne totale (erreur de quantification) doit être la plus petite possible.

Dans ce qui suivra, on donnera un algorithme général de quantification par treillis.

IV-4- Algorithme de quantification par treillis :

On se propose de quantifier "n" vecteurs à la fois par le treillis à 4 états [17]. Les indices "i" des vecteurs sont tels que : $i = 1, \ldots, n$. L'erreur (e_i^j) sera la plus petite erreur ou distance euclidienne entre un vecteur d'indice "i" à quantifier et un vecteur le plus proche dans le quantificateur D_j $(j = 0, 1, 2, 3)$ considéré.

L'algorithme se subdivise en 5 étapes.

1ère étape : Au départ, assigner une erreur cumulée nulle à l'un des états du treillis et une erreur infinie aux autres. Le codeur et le décodeur s'entendent préalablement sur le choix de l'état ayant une erreur nulle.

2ème étape : Prendre un vecteur "i" de l'échantillon et calculer les erreurs $e_i^{(0)}$, $e_i^{(1)}$, $e_i^{(2)}$, $e_i^{(3)}$. L'erreur (e_i^j) sera la plus petite erreur causée par un quantificateur D_j.

3ème étape : Pour chaque état du treillis, calculer les deux erreurs cumulées en sommant l'erreur cumulée de l'état parent avec l'erreur (e_i^j) associée à la transition considérée. Conserver le chemin donnant l'erreur cumulée la plus faible.

4ème étape : Pour chacun des états, mémoriser le chemin "survivant" (celui ayant l'erreur la plus faible des deux chemins arrivant à un état). Celà requiert un bit. Mémoriser aussi l'indice du vecteur type dans le quantificateur D_j associé au chemin. Si chaque quantificateur D_j contient $2^{(m-m')}$ vecteurs celà requiert $(m-m')$ bits.

5ème étape : Si moins de "n" vecteurs ont été considérés, reprendre à l'étape 2. Si "n" vecteurs ont été considérés, choisir parmi les 4 chemins, celui ayant la plus faible erreur cumulée, et coder les informations le concernant telles que les transitions et indices de vecteurs des quantificateurs D_j pour chaque étage.

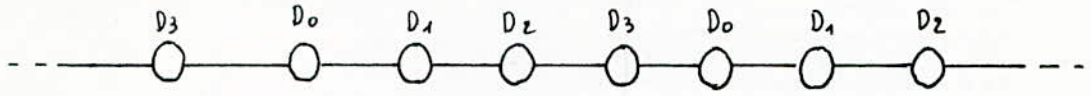
Reprendre à l'étape 1.

Cet algorithme qui est d'usage général ne spécifie pas la nature des quantificateurs D_j, et beaucoup de travail reste à faire dans la recherche. Pour un premier temps, Marcellin et Fischer ont donné une méthode de quantification par treillis, et où les D_j sont des quantificateurs scalaires contenant un jeu de mots codes désignés par "d_j". C'est cette approche particulière qu'on élucidera dans ce qui va suivre.

IV-5. Le "TCQ" selon Marcellin et Fisher :

On supposera que nous avons une source X et que nous avons droit à R bits/ech pour la quantifier. Appelons constellation l'ensemble de toute l'alphabet utilisable dans le codage. Elle sera constituée de 2^{R+1} mots. On divise cette constellation en 4 sous-ensembles (ou classes) notés par D_0, D_1, D_2 et D_3. Chaque classe contiendra 2^{R-1} points. Tous les points seront numérotés par ordre

croissant comme suit :



Ces classes seront assignées aux branches du treillis comme indiqué sur fig-16.

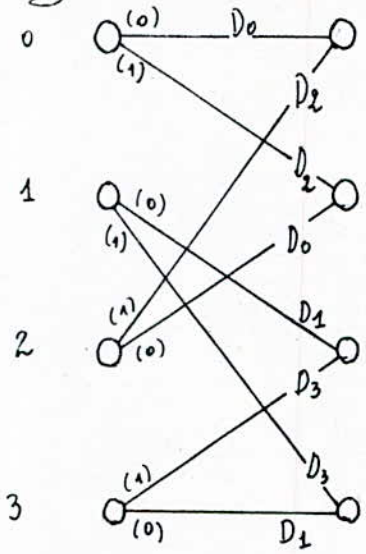


fig-16-Affectation des classes aux branches du treillis ;

Cette approche du "TCQ" pourra être décrite en trois grandes etapes.

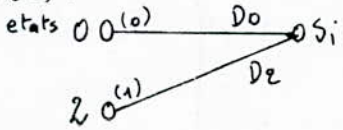
1ere etape: Soit l'echantillon X_i correspondant à l'étage i, on calcule les distorsions (ou erreurs) e_i^0, e_i^1, e_i^2, e_i^3 comme suit :

$(e_i^0)^2 = (X_i - D_0)^2$; $(e_i^1)^2 = (X_i - D_1)^2$; $(e_i^2)^2 = (X_i - D_2)^2$; $(e_i^3)^2 = (X_i - D_3)^2$.

Les éléments de chaque classe D_j seront désignés par d_K ($K = 1, \dots, 2^{R-2}$)

L'erreur e_i^j est la plus petite erreur des erreurs obtenues après 2^{R-1} comparaisons de chaque echantillon X_i avec les divers d_K. Pour pouvoir reconstituer l'echantillon X_i à la réception, on doit mémoriser le numéro de l'alphabet d_K choisi. Ceci requiert $R-1$ bits. Les points d_j ont fait l'objet d'une optimisation par Marcellin et Fisher [15] et sont tabulés pour diverses sources et divers débits. (voir annexe).

2eme etape: si l'on désigne les états d'arrivée du treillis par S_i, on constatera qu'il arrive à chaque état deux branches, l'une par la transition (0), l'autre par la transition (1).



Le calcul d'erreur total se fait au niveau de chaque état et sera l'erreur $(e_i)^2$ de l'étage "i" considéré ajoutée à l'erreur cumulée de l'étage précédent. Seule la plus petite erreur des deux erreurs résultantes sera d'un intérêt, et le chemin correspondant sera le "survivant". Les étiquettes de ces chemins (0 ou 1) sont mémorisées par 1 bit.

3ème étape : Si on laisse le treillis se développer par lui-même, cela peut continuer indéfiniment. Or en temps réel les échantillons Xi à émettre sont en nombre fini. On doit donc prévoir une décision après chaque "n" échantillons nous fixant ainsi sur la séquence à émettre.

Cela consiste à comparer les distorsions des 4 états finaux (erreurs cumulées) du treillis. L'état causant la plus petite distorsion sera pris en compte et le chemin y menant sera codé. Ainsi donc pour chaque branche optimale du treillis "survivant", on codera la transition la caractérisant ainsi que le numéro dk du quantificateur Dj.

Finalement, à chaque étage "i" du treillis, on doit disposer de deux indices qui sont : I0 : indice de l'alphabet ; I1 : chemin choisi dans le treillis. Ces indices nous feront retrouver le vecteur quantifié $y = Q(x)$ le plus proche de X.

V_5. Performances du "TCQ" dans le cas d'une source gaussienne :

Dans la précédente section (III_4), le nombre de bits transmis est "?", soit 1 bit pour désigner les transitions résultantes et (R-1) bits pour spécifier le mot code y(i) utilisé dans chaque quantificateur Dj (la classe Dj ayant 2^{R-1} mots-codes).

Cependant, on remarquera que le dictionnaire de départ utilisé est constitué de 2^{R+1} mots-code, alors que le débit nécessaire à la transmission n'est que de $((R-1)+1) = R$ bits/ech. Le treillis permet finalement une compression de l'entropie de la source et la réduit de 1 bit. Ce qui est appréciable dans le cas où la capacité du canal est de R bits/ech. En fait, la qualité de codage est améliorée pour un même débit "R".

Dans ce qui suivra, on présentera quelques performances du "TCQ" obtenues par Marcellin et Fischer. On parlera surtout du cas de la source

gaussienne ; les sources de distributions uniforme et laplacienne etant abordées dans [15].

Ces performances sont le résultat de simulations de sources obtenus à partir de générateurs de nombres pseudo-aléatoires. Le nombre de réalisations indépendantes utilisées est de 100 ; chaque réalisation contient 1000 echantillons (pseudo-aléatoires) [15]. Cette simulation faite sur 100000 échantillons nécessitera une puissance de calcul et une capacité de mémoire surprenantes. Les résultats de calcul tabulés sont à base de valeurs moyennes des distorsions obtenues dans les 100 réalisations.

Les valeurs du RSB, présentées dans la table ci-après, sont obtenue, en considérant comme variance de l'erreur de codage la difference $(V_x - V_y)$. En effet d'aprés l'optimisation de Marcellin et Fisher des valeurs $d_j(y_j)$ des quantificateur D_j (classe D_j), il en résultera une optimisation de l'erreur de quantification.

Soit : si $y = Q(x)$ et $\sigma_x^2 = E[x^2]$, $\sigma_y^2 = E(y^2]$

et si l'erreur de quantification est e^2,

soit $e^2 = (x-y)^2 = (x - Q(x))^2$

alors : $\sigma_e^2 = E[(x - Q(x))^2] = \sigma_x^2 + \sigma_y^2 - 2E[x\,Q(x)]$.

et d'après [15-18] le quantificateur est optimal si :

$$E[x\,Q(x)] = \sigma_y^2 \qquad \Rightarrow \sigma_e^2 = \sigma_x^2 - \sigma_y^2$$

Le RSB sera par conséquent :

$$RSB = 10\log_{10}\left[\frac{\sigma_x^2}{\sigma_e^2}\right] = 10\log_{10}\left[\frac{\sigma_x^2}{(\sigma_x^2 - \sigma_y^2)}\right]$$

Pour pouvoir faire une comparaison avec les résultats purements théoriques en matière de RSB_{max} dans le cas d'une source gaussienne, les deux auteurs s'étaient aidés de la formule du "RSB" donnée dans la section (III-4) ; soit :

$$RSB_{max}(R) = 10\log_{10}\left(\frac{\sigma_x^2}{\sigma_e^2}\right) = 6,02\,R. \quad \text{avec } \sigma_x^2 = 1$$

Les résultats en matière de "RSB" obtenus par la simulation précédente sont donnés dans le cas du treillis à 4 états par la table ci-dessous. [15].

débit (bits/ech)	Treillis à 4 etats	Quantificateur de Lloyd-MAX	Fonction de distorsion
1	5,00	4,40	6,02
2	10,56	9,30	12,04
3	16,19	14,62	18,06

Et selon [15] les performances du "T.C.Q" sont comparables ou supérieures dans la plupart des cas à tous les résultats déjà obtenus par les différentes techniques de codage d'une source gaussienne et sans mémoire.

Pour montrer toutes les performances, on a réalisé un programme informatique qui, non seulement réalise la quantification d'un echantillon x_i par un élément y_i, il indique aussi le chemin optimal de codage suivi dans le treillis. Le calcul d'erreur (soit de la distance euclidienne) n'a pas été négligé. Le RSB a joui aussi d'un calcul minutieux.

Toutes ces indications seront traitées en détail dans le chapitre qui suit.

CHAPITRE-V
ORGANIGRAMMES ET PROGRAMME

La quantification et codage par treillis est une technique très performante. Cependant le calcul simple (à la main) comme on le verra ci-après est très fastidieux. Pour quantifier et coder de façon raisonnable des échantillons groupés en vecteurs de dimension "n", On doit utiliser les outils offerts par l'informatique. Pour concrétiser celà, on a opté pour le langage informatique FORTRAN 77 qui est un langage bien adapté aux calculs numériques. Le materiel utilisé est le microvax de l'ENP qui offre une souplesse de travail dont on ne peut douter. Dans ce qui suit on établira un organigramme décrivant l'algorithme cité dans ce qui précéde, et enfin on donnera le programme informatique.

Mais avant d'arriver à celà, voyons un exemple illustratif de calcul du chemin optimal et de codage.

V-1. Exemple de quantification par le "TCQ":

Le treillis choisi est un treillis à 4 etats tandis que le débit alloué est de 2 bits/ech. Cependant au lieu d'utiliser un alphabet de $2^R = 2^2 = 4$ points, on s'aidera des règles d'utilisation du treillis pour augmenter le nombre de points de l'alphabet à $2^{R+1} = 2^3 = 8$ points, tout en gardant le même débit de 2 bits/ech.

Soit X un vecteur réel de dimension 5, tel que :

$X = [-1,5 \quad 1,2 \quad 0,3 \quad -0,7 \quad -2,1]$

Si X a une distribution gaussienne les valeurs quantifiées d_j seront celles données par le tableau I (voir annexe). Le débit etant ici de 2 bits/ech, les valeurs d_j sont données par :

D_0	D_1	D_2	D_3
-1,8768	-1,0728	-0,6292	-0,1775
0,1775	0,6292	1,0728	1,8768

On voit dans fig-17, le calcul de distance euclidienne pour chaque branche. Le quantificateur Do qui contient les elements $d_0 = -1,8768$ et $d_1 = 0,1775$ est utilisé dans le cas de la 1ère branche horizontale. Pour reconnaître lesquelles des valeurs d_0 et d_1, cause l'erreur la plus petite, on leur affecte respectivement deux indices (0 et 1).

On supposera ici que le 1er nœud est à erreur cumulée nulle. A chaque nœud arrivent deux branches, et seule une d'entr'elles est d'un intérêt. En effet, si l'on regarde le 1er nœud du 3eme etage du treilli, la distance obtenue est de 1,203 somme de 1,188 et 0,015 et non 1,681 somme de 1,084 et 0,597, donc seule sera gardée la branche donnant 0,015 comme erreur.

Pour décider lequel des chemins correspond au codage optimal, on laissera le treillis se développer jusqu'à la dernière composante du vecteur X, à coder. Le nœud du dernier etage du treillis donnant la distance euclidienne la plus petite sera la base de cette décision. En effet, on voit que le 1er nœud final a une distance cumulée de 0,321 contre 0,709; 2,091 et 1,580. Par conséquent le chemin y menant est optimal et sera représenté en gras.

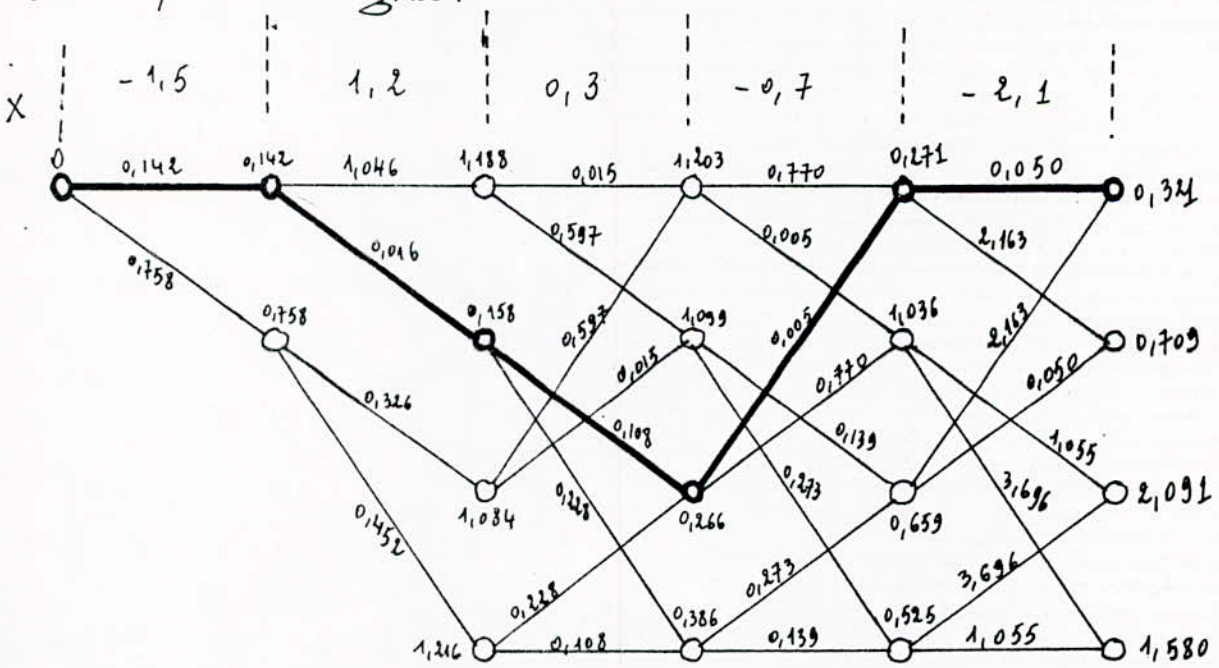


fig-17- Exemple de quantification par le "TCQ". [18]

Pour que ce chemin soit transmis, on doit, pour chaque branche, memoriser le numéro de la transition (0 ou 1). Dans notre exemple, on a les transitions optimales suivantes : I1 = [0 1 0 1 0].

Pour ce qui est des valeurs quantifiées y(i) correspondant aux dj, elles sont données ainsi que leurs indices I0 dans la table ci-dessous.

X(I)	- 1,5	1,2	0,3	- 0,7	- 2,1
y(I)	-1,8768	1,0728	0,6292	-0,6292	-1,8768
I0	0	1	1	0	0
I1	0	1	0	1	0

Finalement seuls les bits de I1 (chemin optimal) et ceux de I0 (indices de l'alphabet) sont à transmettre. A la reception le chemin optimal est reconstitué à partir de I1 = [0 1 0 1 0] et les valeurs y(i) correspondantes sont données à partir des indices contenus dans I0 = [0 1 1 0 0].

Dans le cas où on est en présence d'un vecteur de dimension "n" quelconque, un programme informatique suffit pour tous ces calculs. Mais pour le comprendre, on doit s'aider d'un organigramme.

V-2. Organigrammes :

- Comme entrée de données, on a les variables suivantes :
  - IPRO : c'est la dimension du vecteur X à coder
  - DBI : désigne le débit en bits/ech de codage.
  - X(I) : Ce sont les composantes du vecteur X, I allant de 0 à IPRO-1.
- Comme sorties du programme, on peut écrire :
  - INDEX1(I) : elle correspond aux transitions (0 ou 1) du chemin optimal.
  - Y(I) : désigne les valeurs quantifiées les plus proches des X(I).
  - ED : c'est la plus petite distance euclidienne accumulée au dernier étage du treilli.
  - B(I) : correspond au nœud d'arrivée pour chaque transition du

chemin optimal et est utile pour une représentation graphique de ce chemin optimal.

RSB : c'est le rapport signal sur bruit et détermine ainsi la qualité de codage obtenue.

Détails sur les sous-programmes :

— Le sous programme (A) nommé Subroutine code12(LI, DBI) permet l'obtention des points de tout l'alphabet correspondant à un débit DBI fixé par l'utilisateur et allant de 1 à 3.

— Dans le sous programme (B) nommé Subroutine vector-quant12(X, NP, LI, DBI, LZ2, E) l'erreur E correspond à une distance euclidienne élémentaire la plus petite pour chaque branche du Treillis. La valeur de LI donnant E la plus petite sera affectée à LZ2 pour chaque branche du treillis.

— Dans le sous programme (C) nommé Subroutine Treil4(S, A), on obtient directement les paramètres S(i,j) et A(i,j) du treillis. Pour chaque treillis ces parametres sont fixes ; et ce sont eux qui déterminent les règles de passage d'un etat à un autre état du treillis.

En effet, si on ecrit S(i,j) = K, cela veut dire qu'en partant d'un état précedant l'etat actuel (j), la transition utilisée sera designée par (i) (i = 0 ou 1). Le nombre K designe l'etat précedent l'etat actuel (j). (K = 0 en haut d'une colonne du treillis et K = 3 en bas).

Pour ce qui est de A, soit A(i', j') = K' ; K' désigne le numéro de la classe $D_{K'}$ assignée à une branche parcourue dans le treillis. (j') designe l'état d'arrivée de cette branche à partir d'une transition (i').

— Dans le sous programme (D) nommé Subroutine chemin12(. . . . .) ; les valeurs des differents paramètres sont calculées jusqu'à N = 2 ; en effet, quand n < 2 les régles de transition dans le treillis ne sont plus les mêmes ; cela doit donc jouir d'un calcul particulier. La valeur de D obtenue nous montre l'état optimal du treillis dans le 2ème étage.

- Le sousprogramme (E) nommé Subroutine chemin 2(....) continue le calcul des differents paramètres de sortie et ce pour les étage 0 et 1. Le calcul démarre à partir de l'état, désigné par la valeur de D, qui est optimal.

- Le sous-programme (F) calcule la variance. Cela est utile pour obtenir à chaque fois Vx et Vy necessaires au calcul du rapport signal sur bruit (RSB).

V-3. Résultats et interprétations :

Le programme illustrant les organigrammes déjà decrits est donné en annexe. Ce programme donne tous les paramètres utiles pour réaliser la quantification et codage par treillis à 4 états et il se nomme "PROGRAM TCQ".

En effet, il permet l'affichage des paramétres d'entrée qui sont respectivement la dimension du vecteur à coder (dimension), le débit de codage considéré en bits/ech (débit) et les echantillons X(1) (I=0, dimension - 1). Comme parametres de sortie affichés, on a la distance euclidienne résultante et la plus petite (ED), la valeur du RSB (RSB en db), les transitions optimales dans le treillis (INDEX1(I), I=0, dimension-1) et les valeurs quantifiées des divers X(1), soit le Y(1) (Y(I), I=0, dimension - 1).

Cependant, dans un but pédagogique, on a prévu une façon de dessiner le chemin optimal de codage et ce en utilisant les B(I) (B(1), I=0, dimension-1) qui ne sont que les différents états d'arrivée de chaque branche optimale du treillis.

Pour vérifier si le programme donne des résultats corrects, il suffit d'obtenir les résultats de la section (V-1) et ce pour le même vecteur X considéré, soit $X = [-1,5; 1,2; 0,3; -0,7; -2,1]$, et pour le même débit qui était 2 bits/ech.

Le programme affiche les résultats suivants et pour le même vecteur X d'entrée. Ces résultats changent selon le débit.

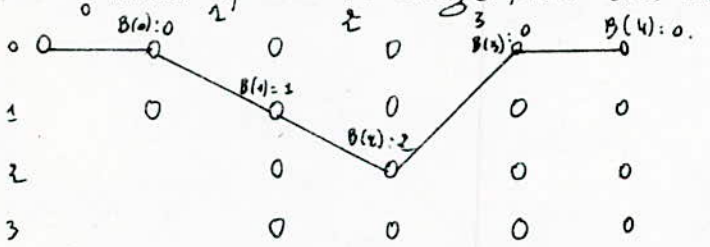
DIMENSION : 5 DEBIT : 1 RSB : 1,8789 db ED : 3.2024

INDEX1(0) : 0	Y(0) : -1,1998	B(0) : 0	X(0) : -1,5
INDEX1(1) : 1	Y(1) : 0,3804	B(1) : 1	X(1) : 1,2
INDEX1(2) : 0	Y(2) : -0,3804	B(2) : 2	X(2) : 0,3
INDEX1(3) : 1	Y(3) : 0,3804	B(3) : 0	X(3) : -0,7
INDEX1(4) : 0	Y(4) : -1,1998	B(4) : 0	X(4) : -2,1

DIMENSION : 5 DEBIT : 2 RSB : 12,1042 db ED : 0,3213

INDEX1(0) : 0	Y(0) : -1,8768	B(0) : 0	X(0) : -1,5
INDEX1(1) : 1	Y(1) : 1,0728	B(1) : 1	X(1) : 1,2
INDEX1(2) : 0	Y(2) : 0,6292	B(2) : 2	X(2) : 0,3
INDEX1(3) : 1	Y(3) : -0,6292	B(3) : 0	X(3) : -0,7
INDEX1(4) : 0	Y(4) : -1,8768	B(4) : 0	X(4) : -2,1

Le chemin optimal de codage peut être dessiné grace au B(I).



L'examen de tous les résultats obtenus en matière de distance euclidienne a montré que celle ci décroit en augmentant le débit de codage. Les valeurs de ED obtenues selon le débit pour l'exemple précédent sont :

débit = 1 => ED = 3,2024

débit = 2 => ED = 0,32136

débit = 3 => ED = 0,27854.

L'explication de la diminution de ED selon le débit, nous vient quand on regarde toutes les classes D_j ($j = 0,1,2,3$) et q celà selon le débit. (voir annexe). En effet pour chaque classe D_j, le nombre de valeurs d_j contenues est 1, 2 et 4 (soit 2^{R-1}) pour les débits respectifs 1, 2 et 3. Et comparer quatre fois un échantillon $X(1)$ à un autre d_j (si le débit = 3) donne évidemment la plus petite distorsion (distance euclidienne) que comparer le même echantillon $X(1)$ une fois avec un element d_j (si le débit = 1).

On peut remarquer aussi que la précision de calcul n'a pas été exagérée et est de 10^{-4} du moment que tous les éléments d_j sont donnés avec quatre chiffres aprés la virgule.

Pour pouvoir déterminer la qualité du codage, on a fait une simulation et ce à partir d'une source gaussienne obtenue à partir d'un sous programme nommé GWN (voir annexe)

Ce sous programme utilise comme parametres d'entrée N, VAR, ISEED qui sont respectivement le nombre d'echantillons à generer, la variance voulu et le nombre qui à partir duquel commence la generation. Les W(I) désignent les echantillons aléatoires (pseudo) et gaussiens.

Le sous programme WGN donne des valeurs W(I) correspondantes à N = 1000, V = 1, ISEED = 69069.

Le programme principal donne en conséquence :

dimension = 1000	débit = 1	RSB = 4,8908 db	ED = 319,1946
dimension = 1000	débit = 2	RSB = 11,7696 db	ED = 84,8041
dimension = 1000	débit = 3	RSB = 16,3613 db	ED = 24,5999.

On pourra remarquer que les RSB obtenus pour les dimensions 2 et 3 sont supérieurs à ceux de Marcellin et Fisher. Celà peut s'expliquer du fait que l'on a travaillé avec une seule réalisation de 1000 échantillons et qui a présenté un RSB particuliérement intéressant.

Or les RSB obtenus par les deux auteurs précédents ont été des valeurs moyennes de ceux obtenus dans les 100 réalisations indépendantes de 1000 echantillons chacune, citées dans la section (IV-5).

Un autre test du programme et ce en mettant à l'entrée 256 echantillons generés par WGN nous donne :

dimension = 256	débit = 1	RSB = 4,4768 db	ED = 82,9866
dimension = 256	débit = 2	RSB = 9,8321 db	ED = 22,5406
dimension = 256	débit = 3	RSB = 16,2718 db	ED = 5,8041.

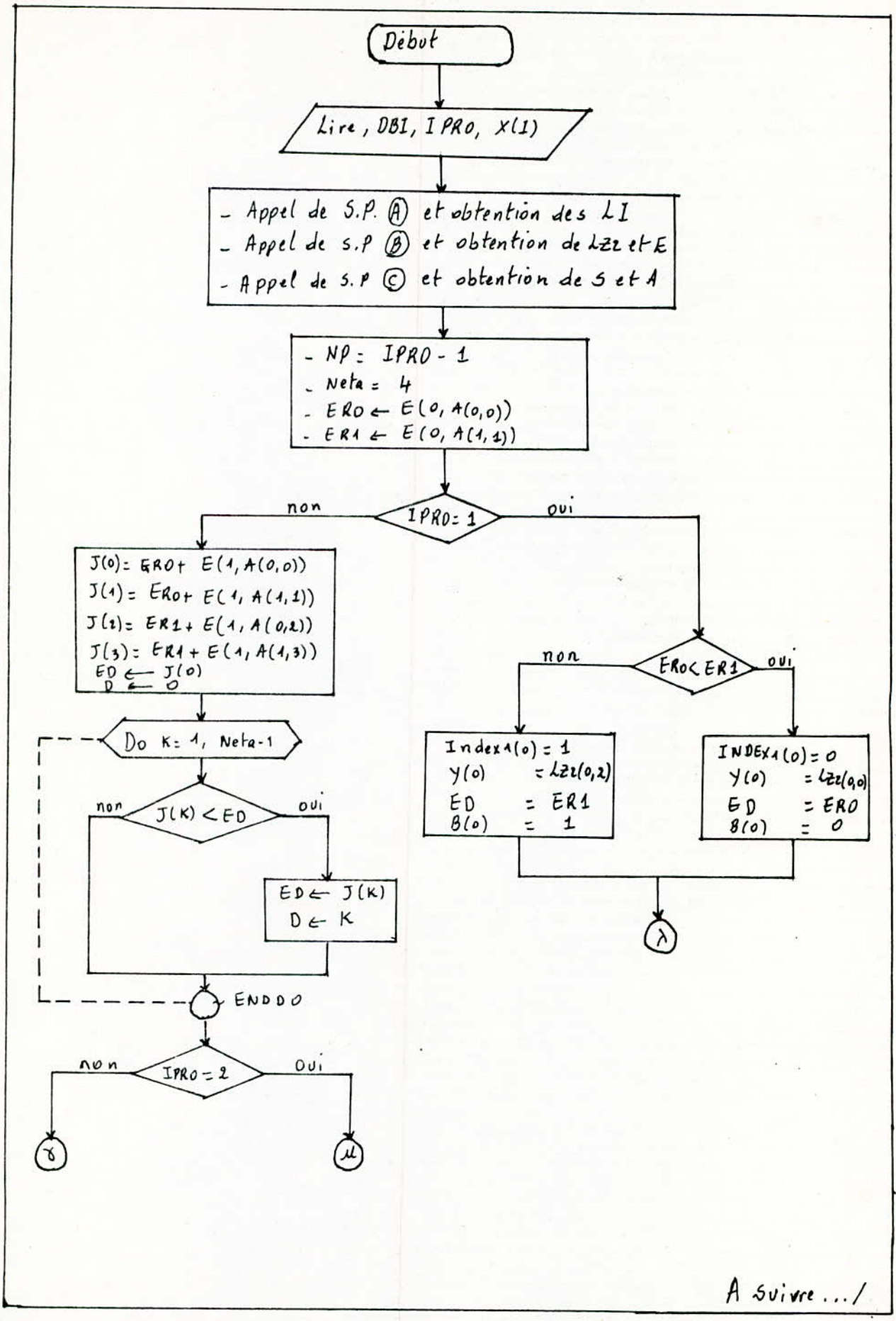
Dans le cas où on a 50 échantillons les résultats obtenus sont :

dimension : 50	débit = 1	RSB = 3,4190 db	ED = 18,9043
dimension : 50	débit = 2	RSB = 6,9480	ED = 4,6641
dimension : 50	débit = 3	RSB = 16,3654	ED = 1,1835.

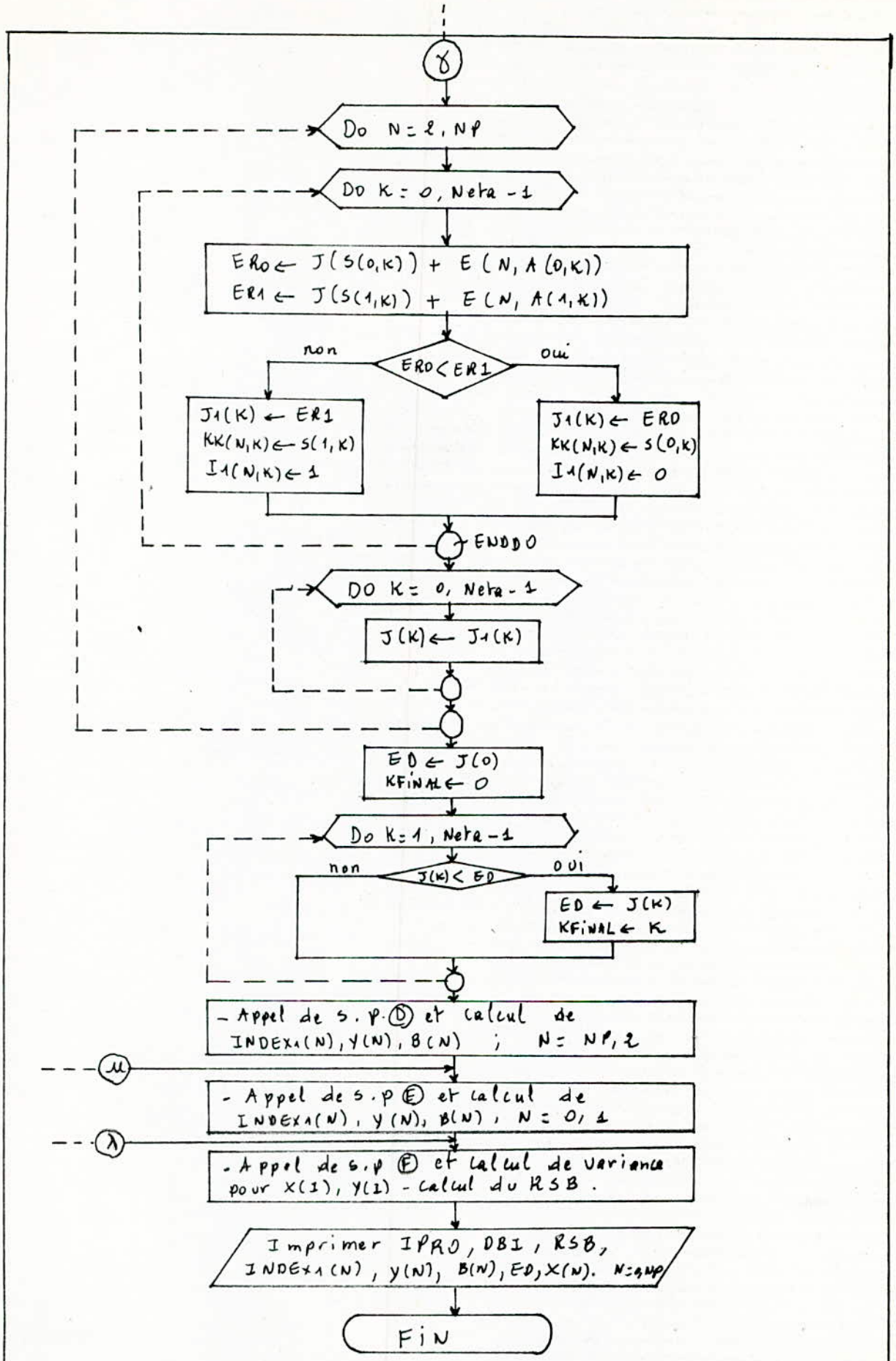
On remarquera que le RSB diminue à mesure que la dimension du vecteur à coder diminue aussi. Celà serait dû à la mauvaise représentativité des eléments dj des echantillons x(i) à coder.

Toutes ces performances montrées ici, nous ouvrent la voie vers une exploitation de cette technique de codage.

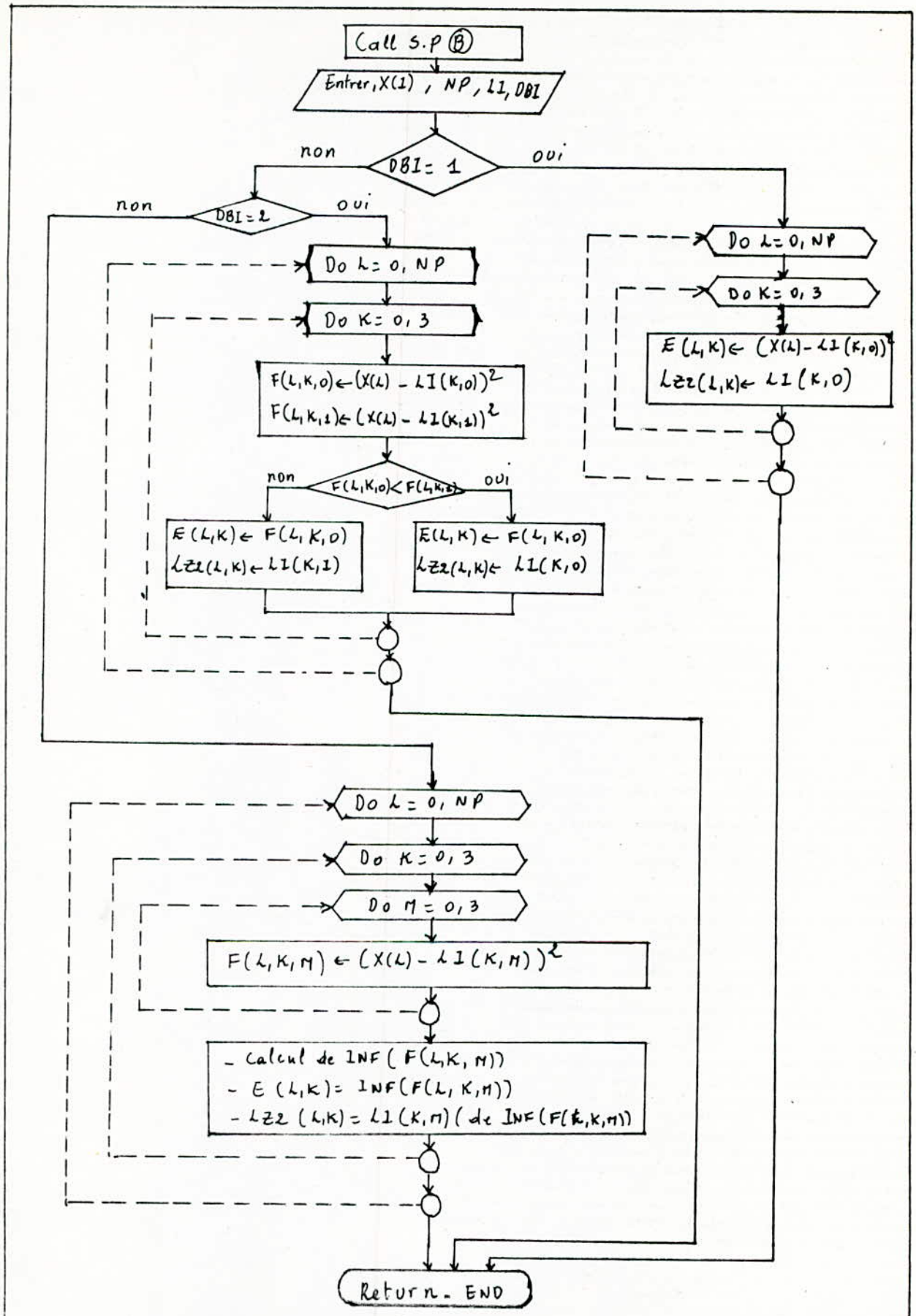
L'étude du treillis à 256 états et de ses performances sera interessante, et d'après fig-11, elle est imbattable.


Début
Lire , DBI, IPRO, X(1)
- Appel de S.P. Ⓐ et obtention des LI
- Appel de s.P Ⓑ et obtention de Lzz et E
- Appel de S.P Ⓒ et obtention de S et A
- NP = IPRO - 1
- Neta = 4
- ERO ← E(0, A(0,0))
- ER1 ← E(0, A(1,1))
IPRO = 1
non
oui
J(0) = ERO + E(1, A(0,0))
J(1) = ERO + E(1, A(1,1))
J(2) = ER1 + E(1, A(0,2))
J(3) = ER1 + E(1, A(1,3))
ED ← J(0)
D ← 0
Do K = 1, Neta-1
J(K) < ED
non
oui
ED ← J(K)
D ← K
ENDDO
IPRO = 2
non
oui
ERO < ER1
non
oui
Index1(0) = 1
y(0) = Lzz(0,2)
ED = ER1
B(0) = 1
INDEX1(0) = 0
y(0) = Lzz(0,0)
ED = ERO
B(0) = 0
λ
γ
μ

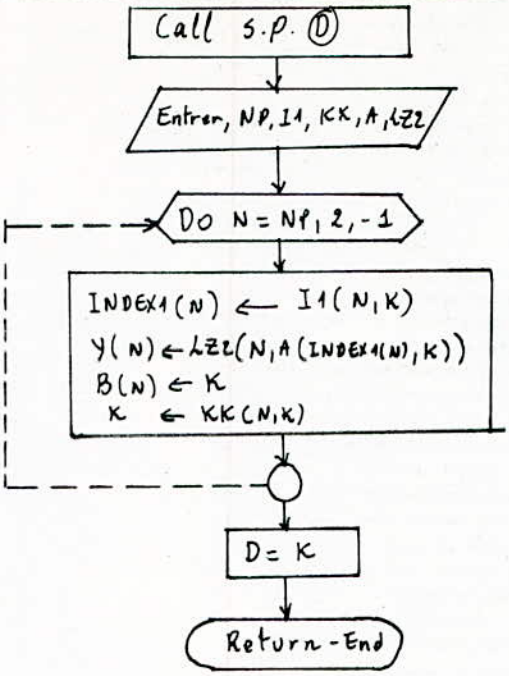
A suivre .../



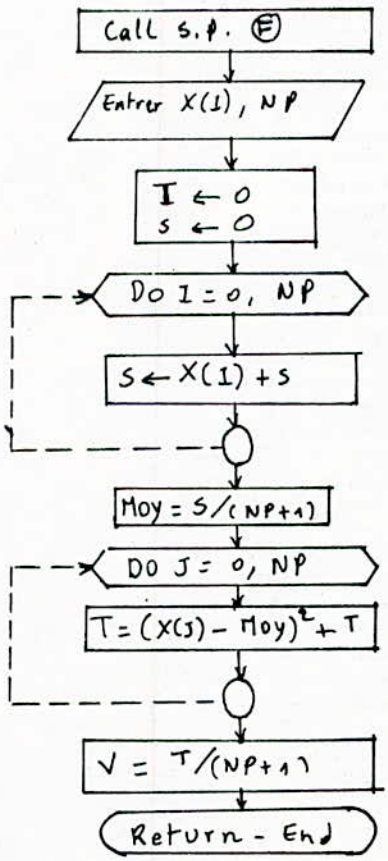
Organigramme general du programme principal:



Organigramme du S.P. (B), nommé Subroutine Vector. quant 12.



Organigramme du S.P. Ⓓ - nommé : Subroutine chemin 12



Organigramme du S.P. Ⓕ, nommé : VAR (X, NP, V).

CONCLUSION

Dans ce present travail, nous avons mis l'accent sur la définition d'une nouvelle technique de quantification des signaux qu'est la quantification vectorielle. Cette téchnique vient à temps pour pallier les insuffisances de la quantification scalaire.

Les techniques de quantification vectorielle algébrique et statistique ont été introduites dans cette étude qui pour une fois fait apparaître le contexte de quantification dans sa dimension la plus grande.

Une nouvelle téchnique de quantification qu'est la quantification et codage par treillis a fait l'objet d'une étude détaillée. Suite à cette étude, nous avons élaboré un programme informatique. Ce programme est opérationnel; tout en déterminant le chemin optimal de codage, il permet d'en juger sur l'erreur résultante qui n'est rien que la distance euclidienne totale la plus petite. Le calcul de RSB n'a pas été négligé. A remarquer que ce programme peut être simulé par diverses sources sans mémoire telles que la gaussienne ou l'uniforme ou enfin la Laplacienne et celà en changeant seulement les valeurs des LI à chaque fois (voir Annexe). Le treilli à 256 états est très performant quand il s'agit du codage pour des délais n grands. Il serait donc interessant de généraliser ce programme aux differents treillis existants.

Ce travail n'a pas eu la prétention de cerner toute la quantification vectorielle. Seule, dorénavant, une orientation dans ce domaine de recherche ferait découvrir les prouesses de cette nouvelle technique de quantification. Citons comme orientations de recherche, la quantification algébrique utilisant le réseau de Leech et celle de codage par le code de Golay etiré [11] qui toutes deux sont performantes pour des délais "n" inférieurs à 64. La quantification et codage par treillis pourra être améliorée en utilisant des classes de vecteurs et non celles de scalaires.

Toutes ces techniques sont d'un apport considerable à la quantification et codage à faible débit de l'image et de la parole améliorant ainsi la qualité de codage pour des coûts moindres.

ANNEXE

Ci-contre, les tables donnant les valeurs des quantificateurs D_0, D_1, D_2 et D_3 utilisées dans le T.C.Q. Ces valeurs, optimisées, sont données dans le cas du treillis à 4 états et pour les débits (en bits/ech) de 1, 2 et 3.

La table I correspond aux valeurs optimisées pour une source gaussienne sans mémoire. Les tables II et III correspondent respectivement aux valeurs optimisées pour une source uniforme et laplacienne, et toutes deux sans mémoire.

débit (bits/ech)	classes			
	D_0	D_1	D_2	D_3
1	-1,1998	-0,3804	0,3804	1,1998
2	-1,8768 0,1775	-1,0728 0,6292	-0,6292 1,0728	-0,1775 1,8768
3	-2,5390 -0,7732 0,1062 1,0248	-1,7992 -0,5219 0,3203 1,3484	-1,3484 -0,3203 0,5219 1,7992	-1,0248 -0,1062 0,7732 2,5390

Table I - Valeurs de quantificateurs selon le débit pour une source gaussienne et sans mémoire.

débit (bits/ech)		classes		
	D_0	D_1	D_2	D_3
1	-1,0509	-0,6680	0,6680	1,0509
2	-1,4263 0,2329	-1,1858 0,6506	-0,6506 1,1858	-0,2329 1,4263
3	-1,5690 -0,7916 0,1113 1,0215	-1,4978 -0,5480 0,3420 1,2195	-1,2195 -0,3420 0,5480 1,4978	-1,0215 -0,1113 0,7916 1,5690

Table II - Valeurs de quantificateurs pour une source uniforme, sans mémoire

débit (bits/ech)	classes			
	D_0	D_1	D_2	D_3
1	-1,4017	-0,1422	0,1422	1,4017
2	-2,5920 0,1432	-1,2178 0,5342	-0,5342 1,2178	-0,1432 2,5920
3	-3,9243 -0,8010 0,0384 1,1633	-2,5645 -0,4752 0,2769 1,7392	-1,7392 -0,2769 0,4752 2,5645	-1,1633 -0,0384 0,8010 3,9243

Table III - Valeurs de quantificateurs pour une source laplacienne et sans mémoire.

Distributions utiles.

1. Distribution normale ou de Gauss :

Une variable aléatoire continue de valeur moyenne μ_x et de variance σ_x^2 est à distribution normale (ou gaussienne) si sa densité de probabilité est du type (fig-a.)

$$p(x) = \frac{1}{\sqrt{2\pi}\,\sigma_x} \exp\left[-\frac{(x-\mu_x)^2}{2\sigma_x^2}\right] \qquad \text{avec } -\infty < x < +\infty$$

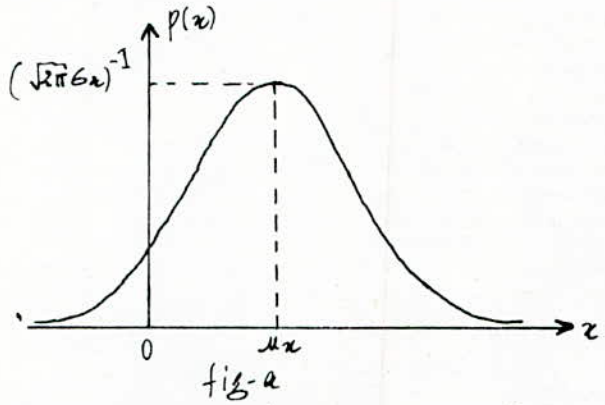


fig-a

En introduisant la variable centrée réduite, soit $z = (x-\mu_x)/\sigma_x$, on obtiendra la loi normale réduite.

$$p(z) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{z^2}{2}\right] \qquad \text{avec } \mu_z = 0 \text{ et } \sigma_z^2 = 1$$

2. Distribution uniforme :

Une variable aléatoire continue est à distribution uniforme si sa densité de probabilité est du type (fig-b.)

$$p(x) = (b-a)^{-1}\,\text{rect}\left\{\left[x - \tfrac{1}{2}(a+b)\right]/(b-a)\right\}$$

avec $\mu_x = \frac{1}{2}(a+b)$; $\sigma_x^2 = (b-a)^2/12$

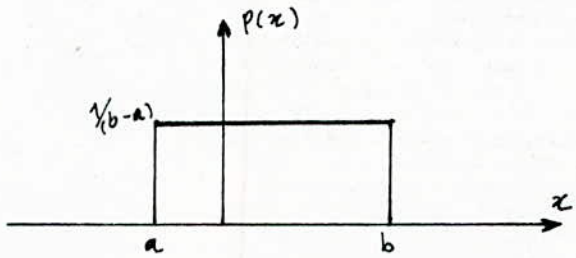


fig-b.

3- Distribution de Laplace :

C'est une distribution continue dont la densité de probabilité s'exprime par (fig-c) :

$$p(x) = \frac{\alpha}{2} \exp(-\alpha |x|)$$

Sa moyenne est : $\mu_x = 0$

Sa variance est : $\sigma_x^2 = 2/\alpha^2$

Pour $\alpha = \sqrt{2}$: $\mu_x = 0$ et $\sigma_x^2 = 1$.

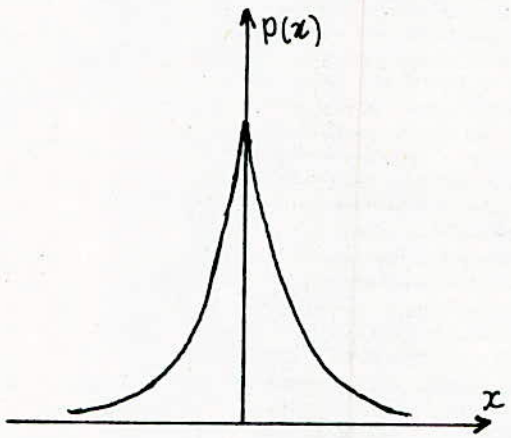


fig-c.

```
C**********************************************************************
C Programme de quantification et codage par le treilli 4 etats
C                    (T.C.Q)  realise par:
C      HALLIL  ABDELBASSET etudiant en electronique a l'E.N.P
C         sujet propose par m.BERKANI  enseignant a l'E.N.P
C----------------------------------------------------------------------
C**********************************************************************
C ----------------( Details sur le programme: )----------------------
C**********************************************************************
C Parametres utiles au programme:/
C-------------------------------/
C  IPRO: dimension du vecteur  X a coder de 1 a 256 (entree)
C  DBI : debit de quantification(1,2,3)  (entree)
C  LI  : dictionnaire de quantification(4,8,16 elements)
C  S(i,j): etat precedant l'etat j,l'atteignant par transition i
C  A(i,j): classe de branche parcourue(D0..D3);i,la transition ,
C       j etant l'etat d'arrivee
C Resultats de calcul obtenus:/
C-----------------------------/
C  INDEX1: chemin choisi dans le treilli(0 ou 1)
C  Y     : reultat de quantification de X
C  ED    : distance euclidienne resultante du chemin optimal
C  B     : Etat d'arrivee d'une branche du chemin optimal
C----------------------------------------------------------------------
C**********************************************************************
      PROGRAM TCQ
      INTEGER INDEX1(0:1000),B(0:1000),I1(0:1000,0:3),KK(0:1000,0:3
     &S(0:1,0:3),A(0:1,0:3),IPRO,DBI,D,Q,CI,C1,C2,C3,Q1
      REAL X(0:1000),E(0:1000,0:3),J(0:1000),J1(0:1000),ERO
      REAL LI(0:3,0:3),LZ2(0:1000,0:3),ER1,Y(0:1000),VX,VV,RSB
C-----------------------------------------------------------------\
C                    Entree de donnees et menu                    /
C-----------------------------------------------------------------/
 10   PRINT 3
 3    FORMAT(/,1X,'DIMENSION DU VECTEUR A CODER DE 1a999?:',$)
      READA,IPRO
      PRINT 4
 4    FORMAT(/,1X,'DEBIT DE CODAGE SOUHAITE EN bits/ech(1a3)?:',$)
      READA,DBI
         IF(IPRO.GT.1000.OR.IPRO.LE.0)GOTO 25
         IF(DBI.GT.3.OR.DBI.LE.0)GOTO 30
         GOTO 30
 25   PRINT 7
  7   FORMAT(/,1X,'ERREUR RECOMMENCEZ')
      GOTO 10
 30   NETA=4
      NP=IPRO-1
C-----------------------------------------------------------------\
C   Choix du dictionnaire LI selon DBI                            /
C-----------------------------------------------------------------/
      CALL CODE12(LI,DBI)
      PRINT 8
 8    FORMAT(/,1X,'SI DONNEES LUES PAR CLAVIER TAPER 1 SINON')
      PRINT 9
 9    FORMAT(10X,'ELLES SERONT LUES DANS FICHGAUS DAT:',$)
      READA,Q1
      IF(Q1.EQ.1)GOTO 15
      IF(Q1.NE.1)GOTO 16
```

```
 15    DO M=0,NP
         PRINT 100,M
 100     FORMAT(1X,'X(',I3,'):',$)
         READ*,X(M)
       ENDDO
       GOTO 14
 16    OPEN(UNIT=17,NAME='FICHGAUS.DAT',TYPE='OLD')
       DO I=0,999
         READ(17,11)CI,X(I)
 11      FORMAT(1X,I5,3X,F12.8)
       ENDDO
       CLOSE(UNIT=17)
C-------------------------------------------------------------------------\
C  Determination de la distance euclidienne E la p.petite entre
C  les X(i)et les LI;LZ2 affecte a LI realisant E la p.petite
C-------------------------------------------------------------------------/
    14 CALL VECTOR_QUANTI2(X,NP,LI,DBI,LZ2,E)
C-------------------------------------------------------------------------\
C  Recherche des valeurs S(i,j)et A(i,j)
C-------------------------------------------------------------------------/
       CALL TREIL4(S,A)  !S= ETAT PRECEDENT A= CLASSE DE LA BRANCHE
C=========================================================================-
       ER0=E(0,A(0,0))
       ER1=E(0,A(1,1))
         IF(IPRO.EQ.1)THEN
           IF(ER0.LT.ER1)THEN
             INDEX1(0)=0
             Y(0)=LZ2(0,0)
             ED  =ER0
             B(0)=0
            ELSE
             INDEX1(0)=1
             Y(0)=LZ2(0,2)
             ED  =ER1
             B(0)=1
            ENDIF
           GOTO 34
          ELSE
            J(0)=ER0+E(1,A(0,0))
            J(1)=ER0+E(1,A(1,1))
            J(2)=ER1+E(1,A(0,2))
            J(3)=ER1+E(1,A(1,3))
            ED  =J(0)
            D   =0
             DO K=1,NETA-1
               IF(J(K).LT.ED)THEN
                 ED=J(K)
                 D=K
               ENDIF
             ENDDO
            ENDIF
              IF(IPRO.EQ.2)THEN
               GOTO 33
```

```
                ELSE
                  DO N=2,NP
                    DO K=0,NETA-1
                      ERO=J(S(0,K))+E(N,A(0,K))
                      ER1=J(S(1,K))+E(N,A(1,K))
                    IF(ERO.LT.ER1)THEN
                      J1(K)=ERO
                      KK(N,K)=S(0,K)
                      I1(N,K)=0
                    ELSE
                      J1(K)=ER1
                      KK(N,K)=S(1,K)
                      I1(N,K)=1
                   ENDIF
                   ENDDO
                   DO K=0,NETA-1
                    J(K)=J1(K)
                   ENDDO
                  ENDDO
              ED=J(0)
              KFINAL=0
             DO K=1,NETA-1
              IF(J(K).LT.ED)THEN
                ED=J(K)
                KFINAL=K
              ENDIF
             END DO
          ENDIF
C-------------------------------------------------------------------------\
C  Calcul d'etats d'arrivee B(i);des chemins choisis INDEX1(i);           |
C   des resultats de quantification Y(i);affichage de ED                  |
C-------------------------------------------------------------------------/
      CALL CHEMIN12(B,NP,I1,KK,KFINAL,D,A,LZ2,INDEX1,Y)
   33 CALL CHEMIN2(Y,D,INDEX1,LZ2,B)
   34 CALL VAR(X,NP,VX)
      CALL VAR(Y,NP,VY)
      IF(VX-VY.EQ.0)THEN
      PRINT*,'RSB NON CALCULABLE CAR VX-VY=0'
      GOTO 19
      ENDIF
      RSB=10*LOG10(ABS(VX/(VX-VY)))
   19 PRINT 6
    6 FORMAT(/,1X,'UNITE DE SORTIE DE RESULTATS?:',$)
      READ*,Q
      WRITE(Q,400)IPRO,DBI,RSB,ED
  400 FORMAT(//,1X,'DIMENSION:',I4,3X,'DEBIT:',I2,3X,'RSB:',
     &          F8.4,' db',3X,'ED:',F8.4,/)
      DO N=0,NP
      WRITE(Q,200)N,INDEX1(N),N,Y(N),N,B(N),N,X(N)
  200 FORMAT(1X,'INDEX1(',I3,'):',I1,2X,'Y(',I3,'):',F8.4,2X,
     &       'B(',I3,'):',I1,2X,'X(',I3,'):',F8.4)
      ENDDO
      STOP
      END
C-------------------------------------------------------------------------
```

```
      SUBROUTINE CHEMIN12(B,NP,I1,KK,K,D,A,LZ2,INDEX1,Y)
      INTEGER I1(0:1000,0:3),KK(0:1000,0:3),K,NP,D,B(0:1000),
     &INDEX1(0:1000),A(0:1,0:3)
      REAL Y(0:1000),LZ2(0:1000,0:3)
      DO N=NP,2,-1
        INDEX1(N)=I1(N,K)
        Y(N)=LZ2(N,A(INDEX1(N),K))
        B(N)=K
        K=KK(N,K)
      ENDDO
       D=K
      RETURN
      END
C-----------------------------------------------------------------
      SUBROUTINE VECTOR_QUANT12(X,NP,LI,DBI,LZ2,E)
      REAL X(0:1000),E(0:1000,0:3),F(0:1000,0:3,0:3)
      REAL LI(0:3,0:3),LZ2(0:1000,0:3)
      INTEGER NP,DBI
          IF(DBI.EQ.1)THEN
           DO L=0,NP
             DO K=0,3
               E(L,K)=(X(L)-LI(K,0))**2
               LZ2(L,K)=LI(K,0)
              END DO
           END DO
          ELSE IF(DBI.EQ.2)THEN
           DO L=0,NP
            DO K=0,3
               F(L,K,0)=(X(L)-LI(K,0))**2
               F(L,K,1)=(X(L)-LI(K,1))**2
             IF(F(L,K,0).LT.F(L,K,1))THEN
               E(L,K)=F(L,K,0)
               LZ2(L,K)=LI(K,0)
             ELSE
               E(L,K)=F(L,K,1)
               LZ2(L,K)=LI(K,1)
               ENDIF
             ENDDO
            ENDDO
           ELSE
             DO L=0,NP
               DO K=0,3
                 DO M=0,3
                   F(L,K,M)=(X(L)-LI(K,M))**2
                 END DO
                   E(L,K)=F(L,K,0)
                   LZ2(L,K)=LI(K,0)
                 DO I=1,3
                 IF(F(L,K,I).LT.E(L,K))THEN
                   E(L,K)=F(L,K,I)
                   LZ2(L,K)=LI(K,I)
                 ENDIF
                 END DO
               END DO
             END DO
           ENDIF
         RETURN
         END
C-----------------------------------------------------------------
```

```
      SUBROUTINE CHEMIN2(Y,KFINAL,INDEX1,LZ2,B)
       INTEGER INDEX1(0:1000),KFINAL,B(0:1000)
       REAL LZ2(0:1000,0:3),Y(0:1000)
        IF(KFINAL.EQ.0)THEN
            INDEX1(0)=0
            INDEX1(1)=0
            B(0)=0
            B(1)=0
            Y(0)=LZ2(0,0)
            Y(1)=LZ2(1,0)
          ELSE IF(KFINAL.EQ.1)THEN
            INDEX1(0)=0
            INDEX1(1)=1
            B(0)=0
            B(1)=1
            Y(0)=LZ2(0,0)
            Y(1)=LZ2(1,2)
          ELSE IF(KFINAL.EQ.2)THEN
            INDEX1(0)=1
            INDEX1(1)=0
            B(0)=1
            B(1)=2
            Y(0)=LZ2(0,2)
            Y(1)=LZ2(1,1)
          ELSE
            INDEX1(0)=1
            INDEX1(1)=1
            B(0)=1
            B(1)=3
            Y(0)=LZ2(0,2)
            Y(1)=LZ2(1,3)
          ENDIF
        RETURN
        END
C-------------------------------------------------------------------
        SUBROUTINE CODE12(LI,DBI)
        REAL LI(0:3,0:3)
        INTEGER DBI
          IF(DBI.EQ.1)THEN
            LI(0,0)=-1.1998
            LI(1,0)=-0.3804
            LI(2,0)= 0.3804
            LI(3,0)= 1.1998
          ELSE IF(DBI.EQ.2)THEN
            LI(0,0)=-1.8768
            LI(0,1)=0.1775
            LI(1,0)=-1.0728
            LI(1,1)= 0.6292
            LI(2,0)=-0.6292
            LI(2,1)= 1.0728
            LI(3,0)=-0.1775
            LI(3,1)=1.8768
          ELSE
            LI(0,0)=-2.5390
            LI(0,1)=-0.7732
            LI(0,2)=0.1062
            LI(0,3)=1.0248
            LI(1,0)=-1.7992
            LI(1,1)=-0.5219
```

```
            LI(1,2)=0.3203
            LI(1,3)=1.3484
            LI(2,0)=-1.3484
            LI(2,1)=-0.3203
            LI(2,2)= 0.5219
            LI(2,3)=1.7992
            LI(3,0)=-1.0248
            LI(3,1)=-0.1062
            LI(3,2)=0.7732
            LI(3,3)=2.5390
          ENDIF
          RETURN
          END
C-------------------------------------------------------------------
          SUBROUTINE TREIL4(S,A)
          INTEGER S(0:1,0:3),A(0:1,0:3)
            S(0, 0)= 0
            A(0, 0)= 0
            S(1, 1)= 0
            A(1, 1)= 2
            S(0, 2)= 1
            A(0, 2)= 1
            S(1, 3)= 1
            A(1, 3)= 3
            S(0, 1)= 2
            A(0, 1)= 0
            S(1, 0)= 2
            A(1, 0)= 2
            S(0, 3)= 3
            A(0, 3)= 1
            S(1, 2)= 3
            A(1, 2)= 3
          RETURN
          END
C-------------------------------------------------------------------
          SUBROUTINE VAR(X,NP,V)
          INTEGER NP
          REAL X(0:1000),V,MOY,S,T
            T=0
            S=0
             DO I=0,NP
              S=X(I)+S
             ENDDO
            MOY=S/(NP+1)
             DO J=0,NP
              T=(X(J)-MOY)**2+T
             ENDDO
            V=T/(NP+1)
        RETURN
          END
C-------------------------------------------------------------------
C*******************************************************************
```

BIBLIOGRAPHIE

[1] - Alexandru Spătaru ; "Fondements de la théorie de la transmission de l'information". presses polytechniques romandes - 1987.

[2] - J. Oswald, "Théorie de l'information et analyse discritique des systèmes". Ed. Masson 1986.

[3] - P. G. Fontelliet, "Systèmes de télécommunications" presses polytechniques romandes - 1984.

[4] - L. Guemidi, "Codage d'une source analogique avec diverses contraintes". Thèse de Magister à L'E.N.P.

[5] - Guernaouti - Gueraichi, "Quantificateurs optimaux des signaux unidimensionnels et bidimensionnels", thèse d'ingéniorat - Juin 1989.

[6] - R. Boite et M. Kunt ; "Traitement de la parole". presses polytechniques romandes - 1987.

[7] - R.M. Gray, "Vector quantization"; IEEE. Assp. Magazine - April 1984.

[8] - Y. Linde, A - Buzo, R.M. Gray; "An algorithm for vector quantizer design". IEEE. Trans Communications - January 1980.

[9] - J.P. Adoul, "La quantification vectorielle des signaux - Approche Algébrique". Annales des télécommunications - Avril 1976.

[10] - J.P. Adoul, "Algorithmes de quantification vectorielle sphérique à partir du réseau de Gosset d'ordre 8."; Annales des telecommunications, Avr. 1988.

[11] - Z.B. Neticha, P. Nabilleau, J.P. Adoul; "Application de nouveaux codes correcteurs d'erreurs à la quantification d'une source gaussienne à ½ bit par echantillon". Congrès Canadien en génie electrique et informatique - Sept 1989.

[12] - G. Ungerboeck, "Channel coding with multilevel / phase signals". IEEE Trans Information Théory - January 1982.

[13] - G. Ungerboeck, "Trellis - coded modulation with redundant signals sets, part I: Introduction". IEEE. Comm. Mag. Febr. 1987.

[14] - G. Ungerboeck, "Trellis - coded modulation with redundant signals sets, Part II: state of the art"; IEEE. Comm. Mag. Febr. 1987.

[15] - M. W. Marcellin et T. R. Fisher. " Trellis coded quantization of memoryless and Gauss Markov Sources ". Submitted to IEEE Trans Information theory - 1988.

[16] - Viterbi A.J. et Omura. J.K. (1979); " principles of digital communications and coding ". Mc. Graw. Hill. New york.

[17] - Gilles Trurgeon, " la construction de constellations pour des quantificateurs par treillis ". presenté à J.P. Adoul en Avril 1989 - Université de sherbrooke - Canada.

[18] - Z. B. Nétticha, " Trellis coded quantizatoin (TCQ) ". Université de shorbrooke - Canada - Juil 1988.

```
_DUA1:[USER.HALL]LIWGN.FOR;1
      SUBROUTINE WGN(N,VAR,ISEED,W)
C
C   This program generates N samples of real white Gaussian
C   noise with zero mean and a specified variance.  M samples of
C   complex white Gaussian noise may be generated by calling
C   this program to generate 2XM real samples and then concatena
C   the output samples as (W(2XI-1),W(2XI)) for I=1,2,...,M.
C
C   The random number generator used in this program is an
C   intrinsic function on the DEC VAX 11/780 and requires a seed
C   number to start the generation.  The seed should be a large,
C   positive and odd integer such as 69069.  For non VAX
C   computers this program may be used if the random number
C   generator is replaced (see comments in code).
C
C   Input Parameters:
C
C     N     - Number of real white noise samples desired
C     VAR   - Variance of white noise samples desired
C     ISEED - Seed number for random number generator
C
C   Output Parameters:
C
C     W     - Real array of dimension NX1 of white noise
C             samples
C
C   External Subroutines:
C
C     RAN   - Intrinsic function on DEC VAX 11/780.  For other
C             machines replace with random number generator
C             which generates uniformly distributed random
C             numbers on the interval (0,1).
C
C   Notes:
C
C     The calling program must dimension the array W.
C
C   Verification Test Case:
C
C     If N=5, VAR=2., and ISEED=69069, the output should be
C      W(1)=-1.70822, W(2)=-2.42596, W(3)=-1.59297,
C      W(4)=[illegible], W(5)=-2.16533
C     for a DEC VAX 11/780.  For non VAX machines one should
C     replace line 10 of the subroutine RANDOM to input the
C     random numbers
C      u(1)=0.31077, u(2)=0.65205, u(3)=0.52763,
C      u(4)=[illegible], u(5)=[illegible], u(6)=[illegible]
C     to test the WGN subroutine.
C
```

Sous programme de génération de bruit blanc gaussien.

```
      DIMENSION W(1)
      PI=4.*ATAN(1.)
C     Add one to desired number of samples if N is odd
      M=N
      IF(MOD(N,2).ND.0)M=N+1
C     Generate M independent and uniformly distributed random
C     variables on [0,1]
      CALL RANDOM(M,VAR,ISEED,W)
      L=M/2
C     Convert uniformly distributed random variables to Gaussian
C     ones using Box-Muller transform
      DO 10 I=1,L
      U1=W(2*I-1)
      U2=W(2*I)
      TEMP=SQRT(-2.*ALOG(U1))
      W(2*I-1)=TEMP*COS(2.*PI*U2)*SQRT(VAR)
10    W(2*I)=TEMP*SIN(2.*PI*U2)*SQRT(VAR)
      RETURN
      END
      SUBROUTINE RANDOM(M,VAR,ISEED,W)
      DIMENSION W(1)
      DO 10 I=1,M
C     For machines other than DEC VAX 11/780 replace RAN(ISEED)
C     with random number generator.
10    W(I)=RAN(ISEED)
      RETURN
      END
```

Suite du s.p. de génération du bruit blanc gaussien.