MASTER EN SCIENCES MATHÉMATIQUES

Analyse des erreurs d'arrondi dans les algorithmes de la transformée de Fourier rapide

Nys, Jacques; Parmentier, Jacques

Award date:
1978

Awarding institution:
Universite de Namur

ANALYSE DES ERREURS D'ARRONDI DANS LES ALGORITHMES DE LA TRANSFORMEE DE FOURIER RAPIDE

Jacques NYS
et
Jacques PARMENTIER

La disponibilité et les conseils de Messieurs J.P. Thiran et Ph. Dontaine nous ont été précieux.
Qu'ils trouvent ici l'expression de nos remerciements.

I N T R O D U C T I O N

Les algorithmes de la Transformée de Fourier Rapide décrits au chapitre 1 sont des méthodes efficaces pour calculer la Transformée de Fourier Discrète d'un signal de N nombres éventuellement complexes [1-4].

L'implémentation sur ordinateur de ces algorithmes entraîne inévitablement des erreurs numériques sur les résultats vu la représentation des nombres en longueur finie. Plusieurs travaux ont proposé une estimation de ces erreurs lorsque les algorithmes sont traités avec une arithmétique en virgule fixe [12,13,20] ou en virgule flottante [7,9,12,19]; le but de ce travail est de présenter les résultats les plus récents sur ce sujet [16,17].

Cette étude utilise une approche statistique dans laquelle les erreurs sont supposées aléatoires. Ces modèles statistiques sont présentés au chapitre 2. D'autre part les problèmes de l'exactitude du signal d'entrée et de la précision des coéfficients multiplicatifs dans les algorithmes ne sont pas traités; seules les erreurs causées par les opérations arithmétiques sont envisagées : le chapitre 3 analyse le cas de la virgule fixe, le chapitre 4 celui de la virgule flottante.

Le chapitre 5 est consacré à la programmation des algorithmes et à la comparaison des résultats théoriques et expérimentaux pour l'estimation des erreurs.

Pour l'étude commune, le chapitre 1 a été rédigé par J. Nys et le chapitre 2 (paragraphe 2.1) par J. Parmentier. D'autre part les chapitres 2 (paragraphe 2.2), 3 et 5 (paragraphe 5.1) ont été rédigés par J. Parmentier et les chapitres 2 (paragraphe 2.3), 4 et 5 (paragraphe 5.2) par J. Nys.

T R A N S F O R M E E DE F O U R I E R

R A P I D E

1 . 1 TRANSFORMEE DE FOURIER DISCRETE

La Transformée de Fourier Discrète d'un signal $\{ x(n),\ 0 \leq n \leq N-1 \}$ composé de N échantillons, éventuellement complexes, est définie par :

$$y(k) = \sum_{n=0}^{N-1} x(n).\exp(-j2\pi nk/N) \ , \ k=0,1,\ldots,N-1 \qquad (1.0)$$

où $j = (-1)^{1/2}$.

La transformée inverse est donnée par :

$$x(r) = \frac{1}{N} \sum_{k=0}^{N-1} y(k).\exp(j2\pi kr/N) \ , \ r=0,1,\ldots,N-1 \qquad (1.1)$$

Insérant (1.0) dans le membre de droite de (1.1), on obtient en effet :

$$\frac{1}{N} \sum_{n=0}^{N-1} x(n). \sum_{k=0}^{N-1} \exp(j2\pi kr/N).\exp(-j2\pi nk/N), \ r=0,1,\ldots,N-1,$$

soient les nombres $x(r), r=0,1,\ldots,N-1$, par la relation d'orthogonalité

$$\sum_{k=0}^{N-1} \exp(j2\pi kr/N).\exp(-j2\pi nk/N) = N\delta_{rn}$$

où δ_{rn} est le symbôle de Kronecker.

Cette relation d'orthogonalité permet également de démontrer la relation de Parseval :

$$\sum_{k=0}^{N-1} |y(k)|^2 = N \sum_{n=0}^{N-1} |x(n)|^2 \qquad (1.2)$$

Dans la suite, on utilisera une forme plus compacte des transformées directe et inverse, en posant

$$W = \exp(-j2\pi/N) \ ,$$

$$y(k) = \sum_{n=0}^{N-1} x(n).W^{nk} \ , \quad k=0,1,\ldots,N-1 \qquad (1.3)$$

$$x(n) = \sum_{k=0}^{N-1} y(k).W^{-kn} \ , \quad n=0,1,\ldots,N-1 \qquad (1.4)$$

Sous cette forme, le calcul de la Transformée de Fourier Discrète nécessite environ N^2 multiplications complexes, ce qui limite son utilisation à de faibles valeurs de N. Les algorithmes de Transformée de Fourier Rapide, les algorithmes FFT (Fast Fourier Transform), permettent d'obvier à cet inconvénient en réduisant considérablement le nombre d'opérations arithmétiques.

Remarque :

Comme les expressions (1.3) et (1.4) ont une forme semblable, on pourra utiliser les algorithmes de la Transformée de Fourier Discrète directe pour calculer la transformée inverse moyennant de légères modifications : changement du signe des exposants de W et division des résultats par N.

1 . 2 ALGORITHMES FFT DANS LE CAS $N = 2^M$

Soit M un entier positif.

Puisque $N = 2^M$ et que les indices n et k des expressions (1.3) et (1.4) varient entre 0 et N-1, ces indices peuvent se représenter chacun par un nombre binaire de M composantes : $(n_M, n_{M-1}, \ldots, n_1)$ et $(k_M, k_{M-1}, \ldots, k_1)$ tel que

$$k = \sum_{i=0}^{M-1} 2^i k_{i+1} \quad , \quad k_i = 0 \text{ ou } 1$$

$$n = \sum_{i=0}^{M-1} 2^i n_{i+1} \quad , \quad n_i = 0 \text{ ou } 1 \qquad (1.5)$$

Il est alors très utile de prendre les notations

$$x(n) = x(n_M, n_{M-1}, \ldots, n_1)$$

$$y(k) = y(k_M, k_{M-1}, \ldots, k_1) \qquad (1.6)$$

On remarque que les valeurs

$$x(n_1, \ldots, n_{M-1}, n_M) = x\left(\sum_{i=0}^{M-1} 2^i n_{M-i} \right) , \quad n=0,1,\ldots,N-1$$

$$y(k_1, \ldots, k_{M-1}, k_M) = y\left(\sum_{i=0}^{M-1} 2^i k_{M-i} \right) , \quad k=0,1,\ldots,N-1$$

sont celles des tableaux $\{x(n)\}_{n=0}^{N-1}$ et $\{y(k)\}_{k=0}^{N-1}$, mais dans l'ordre binairement inversé.

La décomposition (1.5) des indices et les notations (1.6) permettent de transformer (1.3) en

$$y(k_M,k_{M-1},\ldots,k_1) = \sum_{n_1=0}^{1}\sum_{n_2=0}^{1}\cdots\sum_{n_M=0}^{1} x(n_M,n_{M-1},\ldots,n_1).W^p \qquad (1.7)$$

où $$p = \left(\sum_{i=0}^{M-1} 2^i k_{i+1}\right).\left(\sum_{l=0}^{M-1} 2^l n_{l+1}\right) \qquad (1.8)$$

A ce stade, on peut décomposer le second membre de l'expression (1.8) de deux manières différentes, ce qui donnera lieu d'une part à l'algorithme de Cooley et Tukey [1-3] et d'autre part à l'algorithme de Sande et Tukey [1,4] .

A ALGORITHME DE COOLEY ET TUKEY

On considère d'abord la décomposition suivante :

$$p = k2^{M-1}n_M + k2^{M-2}n_{M-1} + \ldots + k2n_2 + kn_1$$

On obtient

$$W^p = W^{k2^{M-1}n_M}.W^{k2^{M-2}n_{M-1}}\ldots\ldots W^{k2n_2}.W^{kn_1}.$$

Comme $W^{2^M} = \exp(-j2\pi/N)^N = 1$, cela se réduit à

$$W^p = W^{2^{M-1}k_1n_M}.W^{2^{M-2}(2k_2+k_1)n_{M-1}}\ldots W^{(2^{M-1}k_M+2^{M-2}k_{M-1}+\ldots+k_1)n_1},$$

ce qui permet de remplacer (1.7) par

$$y(k_M,k_{M-1},\ldots,k_1) = \sum_{n_1=0}^{1}\sum_{n_2=0}^{1}\ldots\sum_{n_M=0}^{1} x(n_M,n_{M-1},\ldots,n_1)$$
$$W^{2^{M-1}k_1n_M}.W^{(2k_2+k_1)2^{M-2}n_{M-1}}\ldots.$$
$$W^{(2^{M-1}k_M + 2^{M-2}k_{M-1} + \ldots + k_1)\,n_1}.$$

En exécutant chaque sommation séparément et en numérotant les résultats intermédiaires, on obtient finalement :

$$x_0(n_M,n_{M-1},\ldots,n_1) = x(n_M,n_{M-1},\ldots,n_1)$$

$$x_1(k_1,n_{M-1},\ldots,n_1) = \sum_{n_M=0}^{1} x_0(n_M,n_{M-1},\ldots,n_1)\, W^{2^{M-1}k_1 n_M}$$

$$x_2(k_1,k_2,n_{M-2},\ldots,n_1) = \sum_{n_{M-1}=0}^{1} x_1(k_1,n_{M-1},\ldots,n_1)\, W^{(2k_2+k_1)2^{M-2}n_{M-1}}$$

$$\vdots$$

$$x_M(k_1,k_2,\ldots,k_M) = \sum_{n_1=0}^{1} x_{M-1}(k_1,k_2,\ldots,k_{M-1},n_1)\, W^{(2^{M-1}k_M+\ldots+k_1)n_1}$$

$$y(k_M,k_{M-1},\ldots,k_1) = x_M(k_1,k_2,\ldots,k_M) \qquad (1.9)$$

Cet ensemble d'équations représente la formulation originale de Cooley et Tukey de l'algorithme de la Transformée de Fourier Rapide dans le cas où $N = 2^M$ [1,3] .

On peut constater aisément que le nombre d'opérations arithmétiques à effectuer a fortement diminué par rapport à celui nécessité par l'évaluation directe. En effet, on voit que le premier groupe de N équations ne nécessite aucune multiplication. Viennent ensuite M équations de sommation représentant chacune N équations qui ne nécessitent chacune qu'une seule multiplication car le premier facteur W a toujours un exposant nul. Cet algorithme ne nécessite donc que MN multiplications complexes puisque le dernier groupe d'équations consiste uniquement en une permutation binaire de l'ordre des résultats. On constate d'autre part, en consultant la figure 1 qui illustre cet algorithme dans le cas où M=3, que le nombre de multiplications complexes peut encore se réduire de moitié.

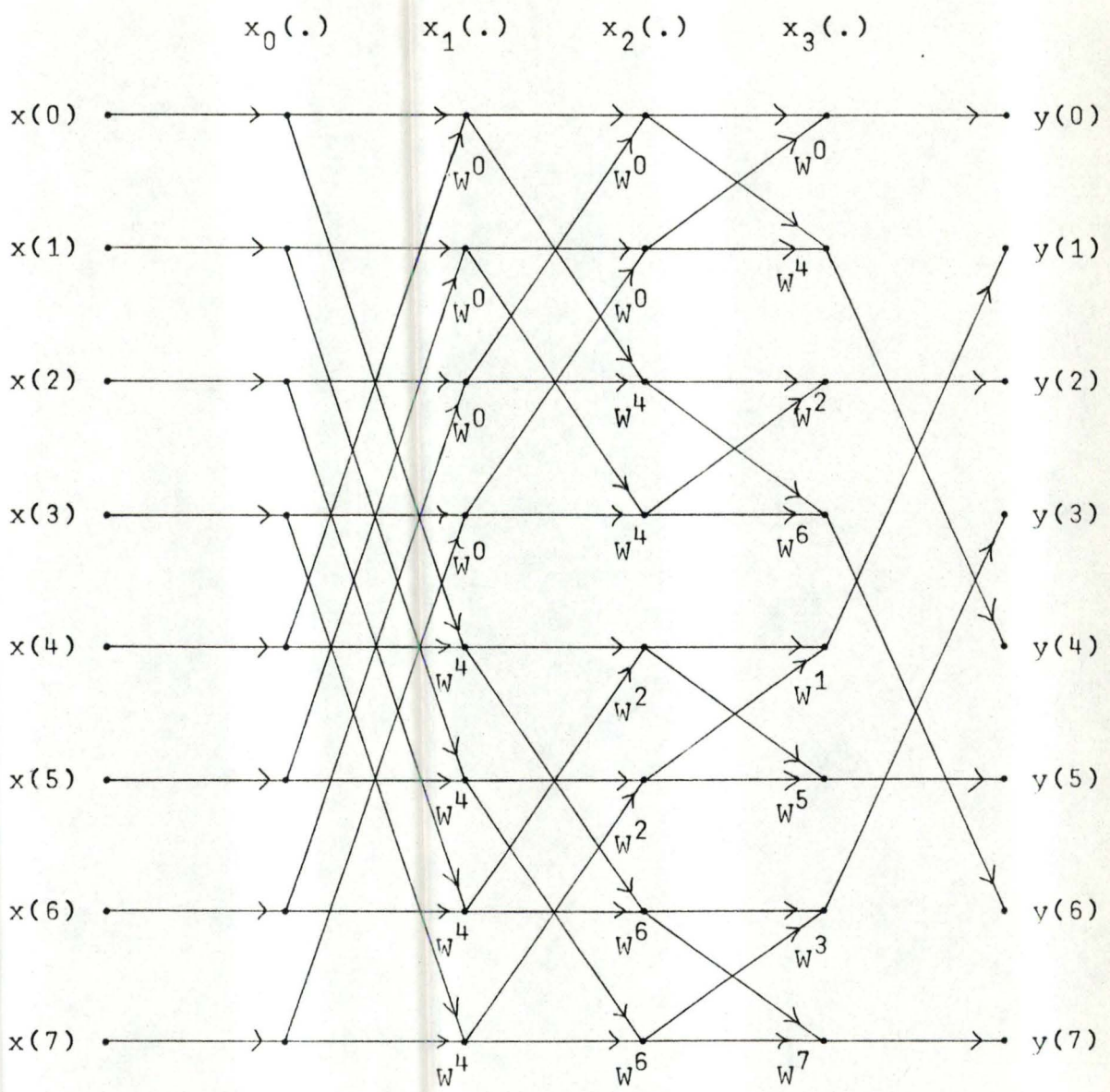

Figure 1. Algorithme de Cooley et Tukey quand $N = 2^3$.

En effet, la propriété $W^{r+N/2} = -W^r$ permet le calcul en "papillons" pour passer d'un tableau intermédiaire au suivant. On a donc :

$$i = 1,2,\ldots,M$$

$$p \text{ tel que } p_{M-i+1} = 0$$

$$q = p + N/2^i$$

$$x_i(p) = x_{i-1}(p) + x_{i-1}(q)\, W^r$$

$$x_i(q) = x_{i-1}(p) - x_{i-1}(q)\, W^r \qquad (1.10)$$

Le nombre de multiplications complexes se réduit donc à $\frac{1}{2} N \log_2 N$. On constate de plus que les données doivent être introduites dans l'ordre direct, tandis que les résultats sont inversés binairement. Les exposants des facteurs W apparaissent eux aussi dans l'ordre binaire inverse.

Une forme tout à fait équivalente, mais plus usitée de cet algorithme peut s'obtenir en inversant binairement l'ordre dans les tableaux intermédiaires. Les données, cette fois, devront être inversées, mais les résultats et les exposants des facteurs W apparaîtront dans l'ordre direct.

Les équations (1.9) et (1.10) deviennent alors :

$$x_0(n_1,n_2,\ldots,n_M) = x(n_M,n_{M-1},\ldots,n_1)$$

$$x_1(n_1,n_2,\ldots,n_{M-1},k_1) = \sum_{n_M=0}^{1} x_0(n_1,n_2,\ldots,n_M)\, W^{2^{M-1} n_M k_1}$$

$$x_2(n_1,\ldots,n_{M-2},k_2,k_1) = \sum_{n_{M-1}=0}^{1} x_1(n_1,\ldots,n_{M-1},k_1)\, W^{(2k_2+k_1)2^{M-2} n_{M-1}}$$

$$\vdots$$

$$x_M(k_M,k_{M-1},\ldots,k_1) = \sum_{n_1=0}^{1} x_{M-1}(n_1,k_{M-1},\ldots,k_1)\, W^{(2^{M-1}k_M+\ldots+k_1)n_1}$$

$$y\,(k_M,k_{M-1},\ldots,k_1) = x_M(k_M,k_{M-1},\ldots,k_1) \qquad (1.11)$$

et $\quad i = 1,2,\ldots,M$

p tel que $p_i = 0$

$$q = p + 2^{i-1}$$

$$x_i(p) = x_{i-1}(p) + x_{i-1}(q)\, W^r$$

$$x_i(q) = x_{i-1}(p) - x_{i-1}(q)\, W^r \qquad (1.12)$$

La figure 2 illustre cette forme de l'algorithme de Cooley et Tukey dans le cas où $N=2^3=8$.

Remarque :

Dans la littérature, les deux formes de cet algorithme sont souvent nommées "Decimation In Time" (DIT). Cela provient du fait que l'indice n du tableau d'entrée est appelé temps et que l'indice k du tableau de sortie est appelé fréquence; or dans la décomposition du second membre de (1.8), c'est le temps qui a été décomposé sous forme binaire.

Sande et Tukey ont également proposé un algorithme de Transformée de Fourier Rapide; ils ont, eux, décomposé la fréquence sous forme binaire dans la décomposition de (1.8). Leur algorithme est encore souvent appelé "Decimation in Frequency" (DIF).

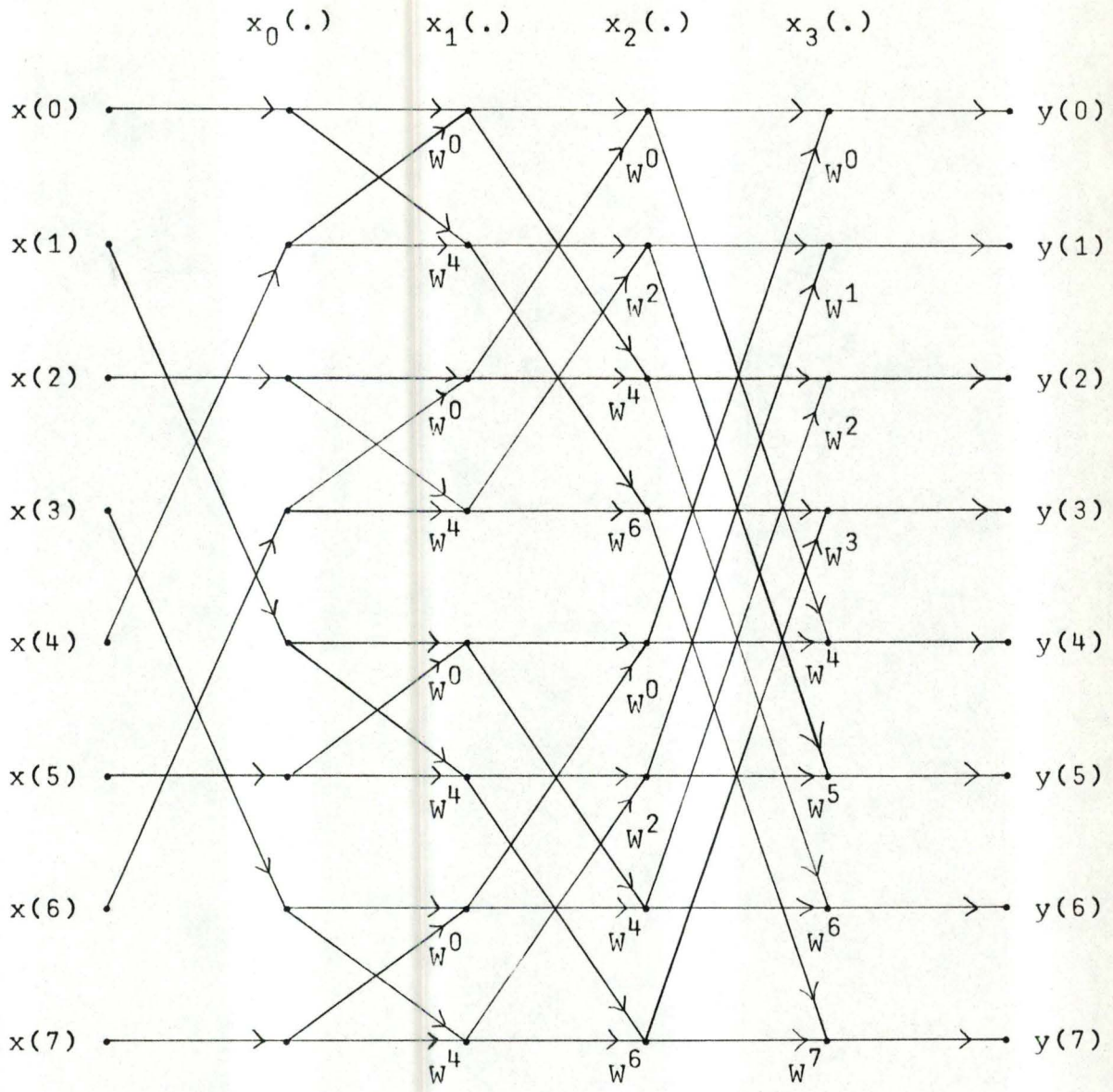

Figure 2. Algorithme de Cooley et Tukey quand $N = 2^3$.

B ALGORITHME DE SANDE ET TUKEY

On considère la décomposition suivante du deuxième membre de (1.8) :

$$p = n2^{M-1}k_M + n2^{M-2}k_{M-1} + \ldots + n2k_2 + nk_1$$

On obtient

$$W^p = W^{n2^{M-1}k_M}.W^{n2^{M-2}k_{M-1}}.....W^{n2k_2}.W^{nk_1} ,$$

qui se réduit à :

$$W^p = W^{2^{M-1}n_1k_M}.W^{(2n_2+n_1)2^{M-2}k_{M-1}}...W^{(2^{M-1}n_M+\ldots+n_1)k_1} ,$$

ce qui permet de remplacer (1.7) par :

$$y(k_M,k_{M-1},\ldots,k_1) = \sum_{n_1=0}^{1}\sum_{n_2=0}^{1}\ldots\sum_{n_M=0}^{1} x(n_M,n_{M-1},\ldots,n_1)$$
$$W^{(2^{M-1}n_M+\ldots+n_1)k_1}$$
$$W^{(2^{M-2}n_{M-1}+\ldots+n_1)2k_2}$$
$$\vdots$$
$$W^{2^{M-1}n_1k_M} .$$

En effectuant ces sommations séparément et en numérotant les résultats intermédiaires, on obtient :

$$x_0(n_M,n_{M-1},\ldots,n_1) = x(n_M,n_{M-1},\ldots,n_1)$$

$$x_1(k_1,n_{M-1},\ldots,n_1) = \left[\sum_{n_M=0}^{1} x_0(n_M,n_{M-1},\ldots,n_1)W^{2^{M-1}n_Mk_1}\right]$$

$$\times \quad W^{(2^{M-2}n_{M-1}+\ldots+n_1)k_1}$$

$$x_2(k_1,k_2,n_{M-2},\ldots,n_1) = \left[\sum_{n_{M-1}=0}^{1} x_1(k_1,n_{M-1},\ldots,n_1)\, W^{2^{M-1}n_{M-1}k_2}\right]$$

$$\times \quad W^{(2^{M-3}n_{M-2}+\ldots+n_1)2k_2}$$

$$\vdots$$

$$x_M(k_1,k_2,\ldots,k_M) = \sum_{n_1=0}^{1} x_{M-1}(k_1,\ldots,k_{M-1},n_1)\, W^{2^{M-1}n_1k_M}$$

$$y(k_M,k_{M-1},\ldots,k_1) = x_M(k_1,k_2,\ldots,k_M) \qquad (1.13)$$

Cet ensemble d'équations représente la formulation originale de Sande et Tukey de l'algorithme de la Transformée de Fourier Rapide dans le cas où $N = 2^M$ [1,4] .

On voit que, tout comme l'algorithme de Cooley et Tukey, celui-ci ne nécessite que $\frac{1}{2} N \log_2 N$ multiplications complexes et que le calcul d'un tableau intermédiaire à partir du précédent peut se faire en "papillons". En effet, on a :

$$i = 1,2,\ldots,M$$

$$p \text{ tel que } p_{M-i+1} = 0$$

$$q = p + N/2^i$$

$$x_i(p) = x_{i-1}(p) + x_{i-1}(q)$$

$$x_i(q) = \{ x_{i-1}(p) - x_{i-1}(q) \}\, W^r \qquad (1.14)$$

On constate sur la figure 3 qui illustre cet algorithme dans le cas où M=3, que le tableau de données doit être introduit

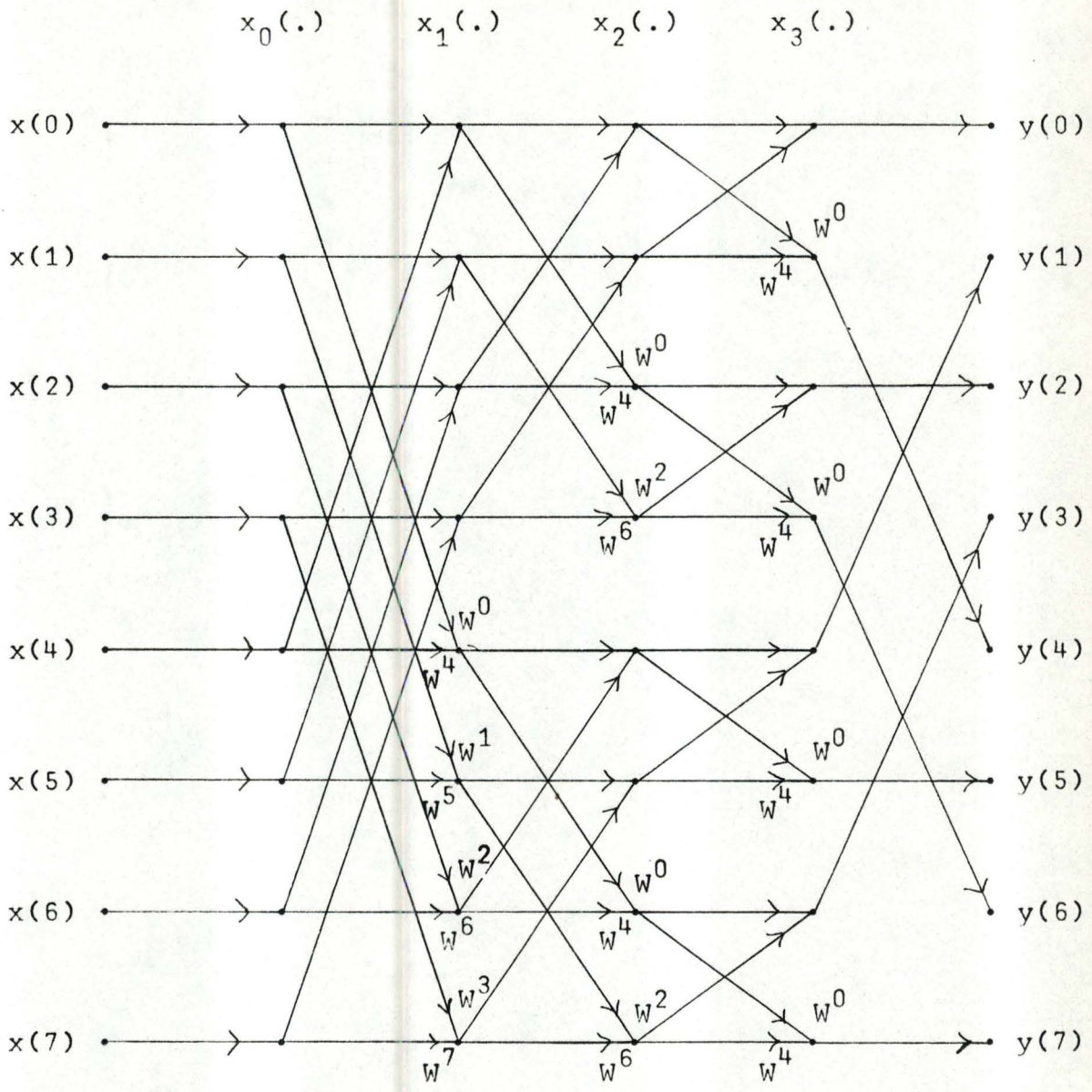

Figure 3. Algorithme de Sande et Tukey quand $N = 2^3$.

dans l'ordre direct, tandis que le tableau des résultats doit être inversé binairement. Les exposants des facteurs W apparaissent, eux, dans l'ordre direct.

Tout comme dans le cas de l'algorithme de Cooley et Tukey, on peut trouver une autre forme de l'algorithme de Sande et Tukey en inversant binairement l'ordre dans tous les tableaux intermédiaires; mais celle-ci fera apparaître les exposants des facteurs W dans l'ordre binaire inverse et en sera, dès lors, moins utilisée.

Les équations (1.13) et (1.14) deviennent alors :

$$x_0(n_1,n_2,\ldots,n_M) = x(n_M,n_{M-1},\ldots,n_1)$$

$$x_1(n_1,n_2,\ldots,n_{M-1},k_1) = \Big\{ \sum_{n_M=0}^{1} x_0(n_1,n_2,\ldots,n_M)\, W^{2^{M-1}n_M k_1} \Big\} \times W^{(2^{M-2}n_{M-1}+\ldots+n_1)k_1}$$

$$x_2(n_1,\ldots,n_{M-2},k_2,k_1) = \Big\{ \sum_{n_{M-1}=0}^{1} x_1(n_1,\ldots,n_{M-1},k_1)\, W^{2^{M-1}n_{M-1}k_2} \Big\} \times W^{(2^{M-3}n_{M-2}+\ldots+n_1)2k_2}$$

$$\vdots$$

$$x_M(k_M,k_{M-1},\ldots,k_1) = \sum_{n_1=0}^{1} x_{M-1}(n_1,k_{M-1},\ldots,k_1)\, W^{2^{M-1}k_M n_1}$$

$$v(k_M,k_{M-1},\ldots,k_1) = x_M(k_M,k_{M-1},\ldots,k_1) \qquad (1.15)$$

et $i = 1,2,\ldots,M$

p tel que $p_i = 0$

$$q = p + 2^{i}$$

$$x_i(p) = x_{i-1}(p) + x_{i-1}(q)$$

$$x_i(q) = \{ x_{i-1}(p) - x_{i-1}(q) \} W^r \qquad (1.16)$$

La figure 4 illustre cette forme de l'algorithme de Sande et Tukey dans le cas où M=3.

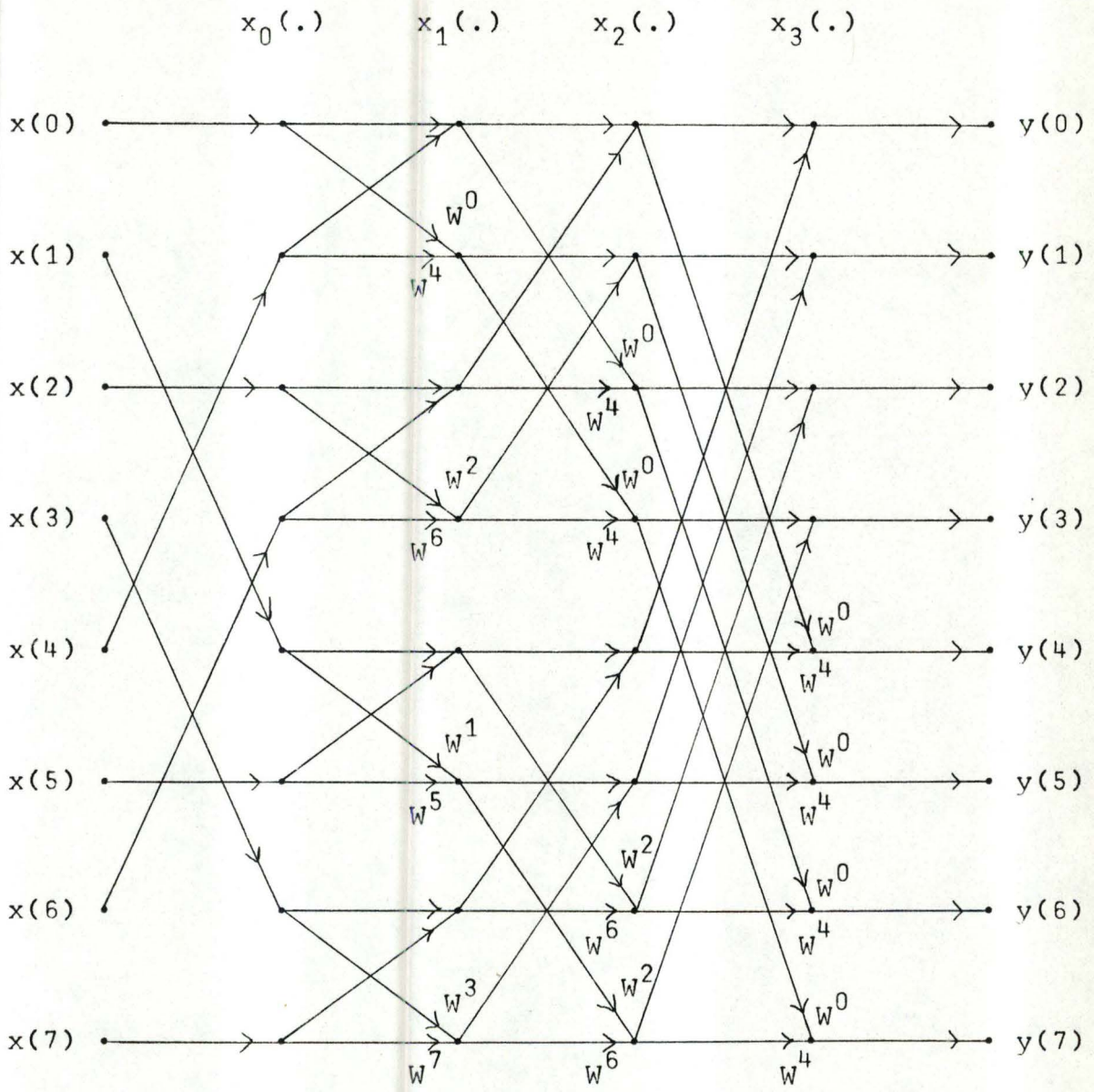

Figure 4. Algorithme de Sande et Tukey quand $N = 2^3$.

1 . 3 ALGORITHMES FFT DANS LE CAS $N = r^M$

Ce paragraphe est une généralisation du précédent : il envisage le cas où N n'est plus puissance de 2, mais une puissance entière d'un nombre entier positif r plus grand que 2 [1,16,17] .

On peut représenter les indices k et n des expressions (1.3) et (1.4) par $(k_M,k_{M-1},\ldots,k_1)$ et $(n_M,n_{M-1},\ldots,n_1)$, tels que :

$$k = \sum_{i=1}^{M} k_i r^{i-1} \quad , \quad k_i \in \{0,1,\ldots,r-1\}$$

$$n = \sum_{i=1}^{M} n_i r^{i-1} \quad , \quad n_i \in \{0,1,\ldots,r-1\} \qquad (1.17)$$

La décomposition (1.17) des indices et les notations

$$x(n) = x(n_M,n_{M-1},\ldots,n_1)$$

$$y(k) = y(k_M,k_{M-1},\ldots,k_1) \qquad (1.18)$$

permettent de transformer (1.3) en :

$$y(k_M,k_{M-1},\ldots,k_1) = \sum_{n_M} \sum_{n_{M-1}} \cdots \sum_{n_1} x(n_M,n_{M-1},\ldots,n_1)\, W^p \qquad (1.19)$$

où $W = \exp(-j2\pi/N)$

$$\sum_{n_i} \equiv \sum_{n_i=1}^{r-1} \qquad (1.20)$$

$$p = \left(\sum_{i=1}^{M} n_i r^{i-1} \right) . \left(\sum_{l=1}^{M} k_l r^{l-1} \right) \qquad (1.21)$$

La factorisation du second membre de (1.21) peut se faire de deux manières différentes, comme dans le paragraphe précédent, ce qui donne lieu à deux algorithmes différents.

A ALGORITHME DE COOLEY ET TUKEY

La décomposition du temps n dans le second membre de (1.21) et la propriété $W^N = 1$ permettent de transformer (1.19) en l'algorithme suivant :

$$x_0(n_1,n_2,\ldots,n_M) = x(n_M,n_{M-1},\ldots,n_1)$$

$$x_m(n_1,\ldots,n_{M-m},k_m,\ldots,k_1) = \sum_{n_{M-m+1}} x_{m-1}(n_1,\ldots,n_{M-m+1},k_{m-1},\ldots,k_1) \times \exp\{-j2\pi n_{M-m+1}(\sum_{i=1}^{m} k_i r^{i-1})/r^m\}$$

$$m = 1,2,\ldots,M$$

$$y(k_M,k_{M-1},\ldots,k_1) = x_M(k_M,k_{M-1},\ldots,k_1) \qquad (1.22)$$

Cet algorithme réduit le nombre de multiplications complexes au moins à $(r-1)Mr^M$. On observe que (1.22) est une généralisation de (1.11).

Il est intéressant, dans le cadre de cet algorithme, d'utiliser la notation suivante :

$$x_m(lr^m+s) = x_m(n_1,\ldots,n_{M-m},k_m,\ldots,k_1) \qquad (1.23)$$

où $l = \sum_{i=1}^{M-m} n_i r^{M-m-i}$

$s = \sum_{i=1}^{m} k_i r^{i-1}$

L'ensemble $\{x_m(lr^m+s)\}_{s=0}^{r^m-1}$ est appelé le $l^{ième}$ bloc à l'étape m. Il y en a r^{M-m} à l'étape m. Le sous-ensemble $\{x_0(lr^m+s)\}_{s=0}^{r^m-1}$ du tableau d'entrée est appelé bloc correspondant au $l^{ième}$ bloc à l'étape m. Chacun des blocs à l'étape m se déduit de son bloc correspondant par la même séquence d'opérations :

Proposition 1.1

Il existe une relation de Transformée de Fourier Discrète inverse entre chaque bloc à l'étape m et son bloc correspondant.

Démonstration :

$$x_m(n_1,\ldots,n_{M-m},k_m,\ldots,k_1) = \sum_{n_{M-m+1}} x_{m-1}(n_1,\ldots,n_{M-m+1},k_{m-1},\ldots,k_1) \times \exp\{-j2\pi n_{M-m+1}(\sum_{i=1}^{m} k_i r^{i-1})/r^m\}$$

$$= \sum_{n_{M-m+1}} \ldots \sum_{n_M} x_0(n_1,\ldots,n_M)\, W_M^p$$

où $W_M = \exp(-j2\pi/r^M)$

$$p = \sum_{i=1}^{m} n_{M-i+1} r^{M-i} (\sum_{l=1}^{i} k_l r^{l-1}) \quad .$$

Donc $$p = r^{M-m} \cdot \sum_{i=1}^{m} \sum_{l=1}^{i} n_{M-i+1}\, k_l r^{m-i+l-1} \quad .$$

On peut, à ce stade, ajouter les valeurs de $l \geq i+1$ car elles feront intervenir des facteurs 1 uniquement.

$$p = r^{M-m} \cdot \sum_{i=1}^{m} \sum_{l=1}^{m} n_{M-i+1} k_l r^{m-i+l-1}$$

$$= r^{M-m} \cdot \sum_{i=1}^{m} \sum_{l=1}^{m} n_{M-m+i} k_l r^{l+i-2}$$

$$= r^{M-m} \cdot \left(\sum_{i=1}^{m} n_{M-m+i} r^{i-1} \right) \cdot \left(\sum_{l=1}^{m} k_l r^{l-1} \right).$$

En conclusion, on obtient :

$$x_m(lr^m+s) = \sum_{n_{M-m+1}} \cdots \sum_{n_M} x_0(lr^m+\tilde{u})\, W_m^{su}$$

où $W_m = \exp(-j2\pi/r^m)$,

$$u = \sum_{i=1}^{m} n_{M-m+i} r^{i-1} ,$$

$$\tilde{u} = \sum_{i=1}^{m} n_{M-m+i} r^{m-i} ,$$

ce qui démontre la proposition.

Ces notions de blocs et de correspondance se voient aisément sur la figure 2 .

B ALGORITHME DE SANDE ET TUKEY

La décomposition de la fréquence k dans le second membre de(1.21) et la propriété $W^N = 1$ permettent de transformer (1.19) en l'algorithme suivant :

$$x_0(n_M, n_{M-1}, \ldots, n_1) = x(n_M, n_{M-1}, \ldots, n_1)$$

$$x_m(k_1, \ldots, k_m, n_{M-m}, \ldots, n_1)$$

$$= \left[\sum_{n_{M-m+1}} x_{m-1}(k_1, \ldots, k_{m-1}, n_{M-m+1}, \ldots, n_1) . \exp(-j2\pi n_{M-m+1} k_m / r) \right]$$

$$\times \exp\{-j2\pi k_m (\sum_{i=1}^{M-m} n_i r^{i-1}) / r^{M-m+1}\} \quad , \; m=1,2,\ldots,M$$

$$y(k_M, k_{M-1}, \ldots, k_1) = x_M(k_1, k_2, \ldots, k_M) \qquad (1.24)$$

Cet algorithme réduit le nombre de multiplications complexes au moins à $(r-1)Mr^M$. On observe que (1.24) est une généralisation de (1.13).

Il est intéressant, dans le cadre de cet algorithme, de prendre la notation suivante :

$$x_m(lr^{M-m}+s) = x_m(k_1, \ldots, k_m, n_{M-m}, \ldots, n_1) \qquad (1.25)$$

où $$l = \sum_{i=1}^{m} k_i r^{m-i}$$

$$s = \sum_{i=1}^{M-m} n_i r^{i-1}$$

L'ensemble $\{x_m(lr^{M-m}+s)\}_{s=0}^{r^{M-m}-1}$ est appelé $l^{\text{ième}}$ bloc à l'étape m. Il y en a r^m à l'étape m. Le sous-ensemble

$\{x_M(lr^{M-m}+s)\}_{s=0}^{r^{M-m}-1}$ du tableau de sortie est appelé bloc correspondant au $l^{ième}$ bloc à l'étape m. Chacun des blocs à l'étape m permet de calculer son bloc correspondant par la même séquence d'opérations :

Proposition 1.2

Il existe une relation de Transformée de Fourier Discrète entre chaque bloc à l'étape m et son bloc correspondant.

La démonstration de cette proposition est semblable à celle de la proposition 1.1.

Ces notions de blocs et de correspondance se voient aisément sur la figure 3.

1 . 4 ALGORITHMES FFT DANS LE CAS $N = r_1 . r_2$

Supposant que le nombre de points N satisfait la relation $N=r_1.r_2$ où r_1 et r_2 sont des entiers positifs, on peut exprimer les indices k et n des expressions (1.3) et (1.4) comme suit :

$$k = k_2 r_1 + k_1$$

$$n = n_2 r_2 + n_1$$

où k_2 et $n_1 \in \{0,1,\ldots,r_2-1\}$

k_1 et $n_2 \in \{0,1,\ldots,r_1-1\}$ (1.26)

On prend alors les notations suivantes :

$$x(n_2,n_1) = x(n)$$

$$y(k_2,k_1) = y(k) \qquad (1.27)$$

A ALGORITHME DE COOLEY ET TUKEY

Les expressions (1.26) et les notations (1.27) permettent de transformer (1.3) en :

$$y(k_2,k_1) = \sum_{n_1=0}^{r_2-1} \left[\sum_{n_2=0}^{r_1-1} x(n_2,n_1)\, W^{kn_2r_2} \right] W^{kn_1} ,$$

ce qui donne lieu à l'algorithme suivant :

$$x_0(n_2,n_1) = x(n_2,n_1)$$

$$x_1(k_1,n_1) = \sum_{n_2=0}^{r_1-1} x_0(n_2,n_1)\ W^{k_1 n_2 r_2}$$

$$x_2(k_1,k_2) = \sum_{n_1=0}^{r_2-1} x_1(k_1,n_1)\ W^{(k_2 r_1+k_1)n_1}$$

$$y(k_2,k_1) = x_2(k_1,k_2) \qquad (1.28)$$

Cet algorithme réduit le nombre de multiplications complexes au moins à $N(r_1+r_2-2)$.

B ALGORITHME DE SANDE ET TUKEY

Les expressions (1.26) et les notations (1.27) permettent de transformer (1.3) en :

$$y(k_2,k_1) = \sum_{n_1=0}^{r_2-1} \left[\sum_{n_2=0}^{r_1-1} x(n_2,n_1)\ W^{(n_2 r_2+n_1)k_1} \right] W^{n_1 k_2 r_1},$$

ce qui conne lieu à l'algorithme suivant :

$$x_0(n_2,n_1) = x(n_2,n_1)$$

$$x_1(k_1,n_1) = \left\{ \sum_{n_2=0}^{r_1-1} x_0(n_2,n_1)\ W^{n_2 k_1 r_2} \right\} W^{n_1 k_1}$$

$$x_2(k_1,k_2) = \sum_{n_1=0}^{r_2-1} x_1(k_1,n_1)\ W^{n_1 k_2 r_1}$$

$$v(k_2,k_1) = x_2(k_1,k_2) \qquad (1.29)$$

Cet algorithme réduit le nombre de multiplications complexes au moins à $N(r_1+r_2-2)$.

M O D E L E S S T A T I S T I Q U E S

POUR LES

E R R E U R S N U M E R I Q U E S

2 . 1 THEORIE STATISTIQUE ELEMENTAIRE DES ERREURS NUMERIQUES

2.1.1. INTRODUCTION

Dans l'implémentation des différents algorithmes de la FFT, les erreurs produites par la représentation de longueur finie des nombres sur l'ordinateur peuvent être interprétées comme un signal aléatoire à ajouter aux nombres traités. Ces considérations donnent lieu aux deux schémas équivalents suivants :

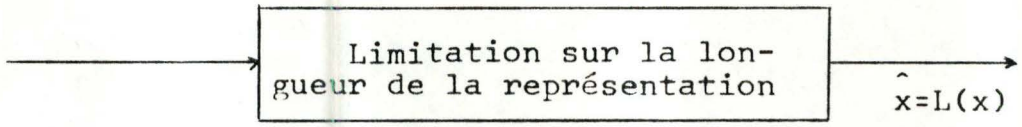

Fig. 5

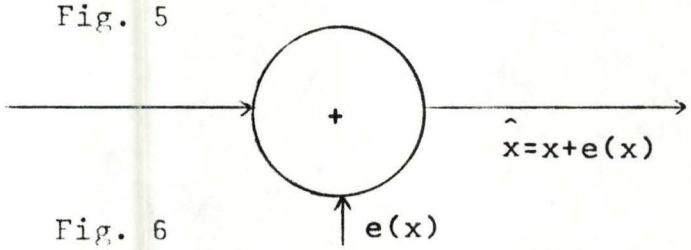

Fig. 6

Dans ces deux figures : - x désigne le nombre dont la représentation doit être réduite

- $\hat{x} = L(x)$, ce nombre après réduction de la représentation
- $e(x)$, l'erreur produite par cette opération.

Pour connaître de façon précise le nombre $\hat{x}$, il faut évaluer l'erreur $e(x)$. Il est commode de supposer ces signaux aléatoires. Ils sont décrits par des quantités faisant partie du domaine de la statistique, comme la moyenne et la variance.
Ce paragraphe a pour but d'introduire les notions qui seront utiles par la suite.

2.1.2 PROCESSUS ALEATOIRES

On appelle processus aléatoire une famille indexée de variables aléatoires notées $\{ X_n \}$; une variable aléatoire est le résultat numérique d'une expérience aléatoire qui peut être répétée plusieurs fois sous les mêmes conditions (cette définition est suffisante pour l'analyse qui va suivre). La famille $\{ X_n \}$ est caractérisée par un ensemble de fonctions de distribution qui, en général, dépendent de l'indice n, souvent associé au temps.

Un processus aléatoire est décrit par :

- la moyenne notée $E [X_n]$ ou encore m_{X_n}

où E représente l'espérance mathématique

- la variance notée $V [X_n]$ ou encore $\sigma^2_{X_n}$

où σ_{X_n} est la déviation standard de X_n

- la fonction d'autocorrélation notée et définie comme suit :

$$\phi_{xx}(n,m) = E [X_n X_m^{\dagger}]$$

où $X_m^{\dagger}$ est le nombre complexe conjugué de X_m

- la fonction de corrélation mutuelle notée $\phi_{xy}(n,m)$ et définie de la manière suivante, (X_n) et (Y_m) étant des processus aléatoires

$$\phi_{xy}(n,m) = E [X_n Y_m^{\dagger}]$$

Lorsque les variables aléatoires du processus (X_n) ne sont pas corrélées, la fonction d'autocorrélation prend la forme suivante :

$$\phi_{xx}(n,n+k) = \begin{cases} E [X_n^2] & \text{si } k = 0 \\ E [X_n] . E [X_{n+k}] & \text{si } k \neq 0 \end{cases}$$

En définissant la fonction d'autocovariance par

$$\gamma_{xx}(n,m) = \phi_{xx}(n,m) - E[X_n].E[X_m]$$

on obtient

$$\gamma_{xx}(n,n+k) = \begin{cases} \sigma^2_{X_n} & \text{si } k = 0 \\ 0 & \text{si } k \neq 0 \end{cases}$$

Un tel processus aléatoire est appelé "bruit blanc".
Dans le cas où toutes les fonctions de distribution sont indépendantes d'un déplacement de l'origine du temps, le processus aléatoire est appelé stationnaire. Les propriétés suivantes sont alors vérifiées :

- la moyenne et la variance sont constantes et notées respectivement m_X et σ^2_X

- la fonction d'autocorrélation ne dépend que de la différence entre les temps m et n. Si m = n + k on notera

$$\phi_{xx}(n, n+k) = \phi_{xx}(k)$$

Un bruit blanc stationnaire est caractérisé par les relations suivantes :

$$\phi_{xx}(n, n+k) = \phi_{xx}(k) = \begin{cases} \sigma^2_X + m^2_X & \text{si } k = 0 \\ m^2_X & \text{si } k \neq 0 \end{cases}$$

$$\gamma_{xx}(n, n+k) = \gamma_{xx}(k) = \begin{cases} \sigma^2_X & \text{si } k = 0 \\ 0 & \text{si } k \neq 0 \end{cases}$$

Remarque :

Si chacune des variables aléatoires d'un processus $\{X_n\}$ prend la valeur $X_n(x_n)$ notée $X(n)$, l'ensemble des valeurs $\{X(n)\}_{-\infty < n < \infty}$ est une réalisation du processus appelée suite d'échantillons du processus aléatoire.

2.1.3 MODELE STATISTIQUE POUR LES ERREURS NUMERIQUES

A l'aide des notions précédentes plusieurs hypothèses vont être formulées à propos des erreurs commises par la représentation de longueur finie des nombres.
Si $\hat{x}(n), -\infty < n < +\infty$, désigne la représentation de x(n), et e(n) l'erreur commise sur ce nombre, on a :

$$x(n) = \hat{x}(n) + e(n) \qquad (2.1)$$

L'hypothèse de base de la théorie qui sera présentée est que les erreurs produites par la représentation des nombres sont des variables aléatoires. En réalité, ces erreurs sont des évènements tout-à-fait déterminés : l'exécution d'un algorithme produira les mêmes erreurs pour des données identiques. Il est donc théoriquement possible de calculer au préalable chacun des termes e(n) de la relation (2.1). Toutefois, comme ces erreurs dépendent des données d'une manière très compliquée, on pourra supposer qu'elles sont des évènements aléatoires. Ce raisonnement peut être illustré par l'exemple classique du jet d'un dé : en théorie, en prédire le résultat est possible grâce aux lois de l'attraction universelle. En pratique, c'est impossible étant donné la dépendance très complexe de cette expérience par rapport aux conditions initiales.

Il est habituel de formuler les hypothèses suivantes :

1) la suite des erreurs forme un processus aléatoire stationnaire appelé "processus d'erreur";

2) la suite des erreurs ne dépend pas de la suite des valeurs exactes des nombres traités;

3) les variables aléatoires du processus d'erreur ne sont pas mises en corrélation, l'erreur est un bruit blanc;

4) la distribution de probabilité de chacune des variables aléatoires du processus d'erreur est uniforme sur son domaine de définition.

Remarques :

1. Comme les erreurs sont des variables aléatoires, la suite $\{e(n)\}$ sera envisagée comme une réalisation de la suite de ces variables aléatoires.

2. Les hypothèses adoptées conduisent à une analyse assez simple des erreurs. Il est aisé de trouver des exemples où elles sont prises en défaut; par exemple si les nombres x(n) sont construits de la façon suivante : soit $x_a(t)$ une fonction de saut

$$x_a(t) = \begin{cases} 0 & \text{si } t < a \\ 1 & \text{si } t \geq a \end{cases}$$

et $x(n) = x_a(nT)$ où T est un réel.
Il est clair qu'il est impossible de vérifier toutes les hypothèses et notamment la troisième. Par contre, quand le signal dont proviennent les nombres x(n) varie rapidement d'une manière complexe, les hypothèses deviennent plus acceptables. Ce serait le cas pour le signal de la parole ou de la musique.

2 . 2. ANALYSE DES ERREURS EN VIRGULE FIXE

2.2.1. ARITHMETIQUE EN VIRGULE FIXE

A REPRESENTATION DES NOMBRES BINAIRES

Dans l'analyse qui suit, un nombre sera représenté par une suite de t + 1 chiffres binaires appelés "bits" :

$$b_0\ b_1 \ldots\ b_t \qquad b_i = 0 \text{ ou } 1 \ ;\ i = 0, 1, \ldots, t \quad (2.2)$$

où b_0 est le bit de signe, égal à 0 pour le signe + et égal à 1 pour le signe -.

Ainsi, pour $b_0 = 0$, le nombre correspondant à la représentation (2.2) vaut

$$\sum_{i=1}^{t} b_i\ 2^{-i}$$

La virgule est située à gauche des t derniers bits.

Trois méthodes sont utilisées pour représenter les nombres binaires négatifs.

- La première est appelée "signe et grandeur".

Le nombre (2.2) avec $b_0 = 1$ est égal à

$$-\sum_{i=1}^{t} b_i\ 2^{-i}$$

Par exemple : 0.011 représente 3/8

1.011 représente - 3/8

- La seconde est appelée "complément à deux".

La grandeur du nombre, c'est-à-dire le nombre positif, est retranchée de deux qui s'écrit 10.0 en binaire.

Par exemple : 3/8 est représenté par 0.011

-3/8 est représenté par 10.000-0.011 = 1.101

- La troisième est appelée "complément à un".

La grandeur du nombre est retranchée du plus grand nombre représentable dans le registre choisi de (t + 1) bits, c'est-à-dire lorsque $b_0 = b_1 = \ldots b_t = 1$.

Par exemple : -3/8 est représenté par 1.111-0.011 = 1.100.
Dans la première représentation, changer le signe du nombre n'affecte que le bit de tête, tandis que dans les deux suivantes tous les bits sont affectés.

B OPERATIONS ELEMENTAIRES

Par la convention prise sur l'écriture des nombres binaires, ceux-ci sont tous inférieurs à l'unité en valeur absolue. Donc le produit de deux de ces nombres sera toujours inférieur à l'unité en valeur absolue et d'autre part la longueur de sa représentation sera généralement 2t + 1 bits qui seront réduits à t + 1 bits à cause de la limitation sur la longueur de l'écriture des nombres en machine. Cette réduction s'effectue soit en tronquant le nombre, c'est-à-dire en écartant les t derniers bits, soit en l'arrondissant, c'est-à-dire en l'approchant par le nombre de t + 1 bits le plus proche.
La somme de deux nombres de t + 1 bits sera un nombre de t + 1 bits si toutefois il peut être représenté dans ce registre.
En effet, il peut se produire un dépassement de la manière suivante la somme de 0.1101 et 0.1000 est 1.0101. Ce nombre ne peut être contenu dans un registre de (4 + 1) bits choisi au départ.
Il y a un report dans la partie entière du nombre. Cette limitation sur le domaine des nombres représentables peut être levée en utilisant l'arithmétique en virgule flottante. Pour continuer à utiliser l'arithmétique en virgule fixe, un artifice sera employé : il consiste à déplacer chaque nombre d'un bit vers la droite, ce qui revient à le diviser par deux. Cette opération portera le nom de déplacement.
En résumé, les sources d'erreurs dans les opérations proviendront de l'arrondi ou de la troncature pour la multiplication et du dépassement pour l'addition.

C ERREURS PRODUITES PAR LA TRONCATURE OU L'ARRONDI

En vertu des conventions prises dans le premier paragraphe, un "1" dans le dernier bit d'un nombre positif représente une valeur numérique de 2^{-t}. Cette quantité est le plus petit écart entre deux nombres représentables dans le registre choisi. Comme les nombres positifs sont décrits d'une façon unique, les erreurs seront identiques pour les trois méthodes de représentation.

1° Erreurs produites par la troncature

Soient x et T(x) le nombre avant et après la troncature et soient t_1 et t, le nombre de bits qu'ils comportent respectivement. ($t_1 > t$)

L'effet de la troncature est d'éliminer les $(t_1 - t)$ derniers bits. L'erreur de troncature est :

$$E_T = T(x) - x$$

Pour les nombres positifs :

$$E_T \leqslant 0$$

avec E_T maximale si tous les bits écartés sont égaux à "1". La valeur de E_T est alors $-(2^{-t} - 2^{-t_1})$. On obtient :

$$-(2^{-t} - 2^{-t_1}) \leqslant E_T \leqslant 0 \qquad \text{pour des nombres positifs} \quad (2.3)$$

Pour des nombres négatifs en "signe et grandeur"

$$E_T \geqslant 0$$

On obtient donc :

$$0 \leqslant E_T \leqslant (2^{-t} - 2^{-t_1}) \qquad (2.4)$$

Pour des nombres négatifs représentés en "complément - à - deux", soit $x = 1.a_1 a_2 \ldots a_{t_1}$ dont la valeur absolue est

$$A_1 = 2.0 - x_1$$

où

$$x_1 = 1 + \sum_{i=1}^{t_1} a_i \, 2^{-i}$$

Comme $T(x) = 1.a_1 a_2 \ldots a_t$, sa valeur absolue est

$$A_2 = 2.0 - x_2$$

où

$$x_2 = 1 + \sum_{i=1}^{t} a_i \, 2^{-i}$$

On peut définir la variation de valeur absolue :

$$\Delta A = A_2 - A_1 = \sum_{i=t+1}^{t_1} a_i 2^{-i}$$

où

$$0 \leqslant \Delta A \leqslant 2^{-t} - 2^{-t_1}$$

L'effet de la troncature produit dans ce cas un accroissement de la valeur absolue.

Donc $$-(2^{-t} - 2^{-t_1}) \leqslant E_T \leqslant 0 \qquad (2.5)$$

Pour les nombres négatifs représentés en "complément à un", en conservant les mêmes notations pour x, T (x), x_1, x_2, on obtient :

$$A_1 = 2.0 - 2^{-t_1} - x_1$$

$$A_2 = 2.0 - 2^{-t} - x_2$$

et la variation de valeur absolue est

$$\Delta A = A_2 - A_1 = -(2^{-t} - 2^{-t_1}) - \sum_{i=t+1}^{t_1} a_i 2^{-i}$$

où

$$-(2^{-t} - 2^{-t_1}) \leqslant \Delta A \leqslant 0$$

Donc, la valeur absolue décroît.

Par conséquent :

$$0 \leqslant E_T \leqslant (2^{-t} - 2^{-t_1}) \qquad (2.6)$$

2° Erreurs produites par l'arrondi

Par la définition de l'arrondi, l'erreur E_A est comprise entre les bornes suivantes :

$$-1/2\ (2^{-t} - 2^{-t_1}) \leqslant E_A \leqslant 1/2\ (2^{-t} - 2^{-t_1}) \qquad (2.7)$$

Si on considère que 2^{-t_1} est négligeable face à 2^{-t}, en reprenant les inégalités précédentes (2.3) à (2.7), on peut dresser le tableau suivant :

Troncature

$$-2^{-t} < E_T \leq 0$$

. nombres positifs

. nombres négatifs en "complément - à - deux" (2.8)

$$0 \leq E_T < 2^{-t}$$

. nombres négatifs

. en "signe et grandeur"

. en "complément - à - un"

Arrondi

$$-\frac{1}{2} \cdot 2^{-t} \leq E_A \leq \frac{1}{2} \cdot 2^{-t} \qquad (2.9)$$

Remarques

1. Pour la représentation en "complément - à - deux", les bornes de l'erreur de troncature ne changent pas avec le signe du nombre. C'est pourquoi elle sera utilisée par la suite car il devient justifiable de supposer que l'erreur est indépendante des nombres traités.

2. Dans le cas de l'arrondi, il faut encore définir une méthode lorsque le nombre de t_1 bits se situe exactement au milieu des deux nombres de t bits qui l'entourent. On peut soit toujours arrondir au nombre supérieur, soit toujours arrondir au nombre inférieur, soit arrondir à l'un ou à l'autre de manière aléatoire. C'est cette dernière méthode qui sera choisie car il sera légitime de supposer nulle la moyenne de l'erreur d'arrondi, quels que soient les nombres traités.

D ERREURS PRODUITES PAR LE DEPASSEMENT

Dans le cas d'un dépassement du registre, les nombres seront déplacés d'un bit vers la droite avant d'effectuer les calculs.
Deux éventualités peuvent se présenter; le dernier bit est perdu, ou il est arrondi de manière aléatoire.
Dans le premier cas, il suffit de modifier les résultats établis pour la troncature. Si l'on considère égales les probabilités d'avoir le dernier bit soit "0", soit "1", on obtient :

$$E_D = \begin{cases} 0 & \text{avec probabilité } 1/2 \\ -2^{-t} & \text{avec probabilité } 1/2 \end{cases} \qquad (2.10a)$$

pour les nombres positifs et les nombres négatifs de "complément - à - deux".

$$E_D = \begin{cases} 0 & \text{avec probabilité } 1/2 \\ 2^{-t} & \text{avec probabilité } 1/2 \end{cases}$$

pour les nombres négatifs représentés en "signe et grandeur" ou en "complément - à - un".

Dans le second cas, il suffit également de modifier les résultats établis pour l'arrondi. On obtient :

$$E_D = \begin{cases} 0 & \text{avec probabilité } 1/2 \\ -2^{-t}/2 & \text{avec probabilité } 1/4 \\ +2^{-t}/2 & \text{avec probabilité } 1/4 \end{cases} \qquad (2.10b)$$

2.2.2. APPLICATION DES HYPOTHESES DU MODELE STATISTIQUE

Comme les erreurs sont distribuées uniformément, la moyenne et la variance peuvent être calculées.
Pour l'arrondi, par (2.9) la fonction de densité de E_A devient

$$f_A(x) = \begin{cases} 1/2^{-t} & \text{si } -2^{-t}/2 \leq x \leq 2^{-t}/2 \\ 0 & \text{ailleurs} \end{cases}$$

La moyenne et la variance sont respectivement

$$m_A = 0 \qquad (2.11)$$

$$\sigma_A^2 = 2^{-2t}/12 \qquad (2.12)$$

De la même façon, pour la troncature et la représentation en "complément - à - deux", par (2.8) on obtient :

$$f_T(x) = \begin{cases} 1/2^{-t} & \text{si } -2^{-t} < x < 0 \\ 0 & \text{ailleurs} \end{cases}$$

$$m_T = -2^{-t}/2 \qquad (2.13)$$

$$\sigma_T^2 = 2^{-2t}/12 \qquad (2.14)$$

De plus, pour le dépassement, pour la représentation en "complément-à - deux", par (2.10a) devient :

$$m_D = -2^{-t}/2 \qquad (2.15)$$

$$\sigma_D^2 = 2^{-2t}/4 \qquad (2.16)$$

2 . 3 ANALYSE DES ERREURS EN VIRGULE FLOTTANTE

2.3.1 ARITHMETIQUE ET BORNES D'ERREUR

A REPRESENTATION NUMERIQUE EN VIRGULE FLOTTANTE

Arithmétique

Sur ordinateur, un nombre réel non nul se représente, en virgule flottante, par

$$\text{sign } b^{a}.\text{mant}$$

où sign est le signe du nombre x,

b est la base de l'arithmétique de l'ordinateur,

a est la valeur de l'exposant du nombre x, relativement à la base b (nombre entier),

mant est la mantisse du nombre x, relativement à la base b; mant est un nombre réel positif tel que $b^{-1} \leq \text{mant} \leq 1 - b^{-t}$ où t est le nombre de chiffres, relatifs à la base b, qui composent la mantisse.

Ce modèle de représentation est appelé "format de signe et de grandeur".

Remarques :

1- Les chiffres qui composent la mantisse sont des chiffres binaires en cas de base 2, des chiffres décimaux en cas de base 10, ou encore des chiffres hexadécimaux en cas de base 16. Par la suite, on utilisera les mots "exposant", "mantisse" et "chiffre"

sans plus jamais spécifier qu'ils sont relatifs à une base b.

2- Les ordinateurs Siemens 4004 et IBM 360 travaillent tous deux en base 16 avec une mantisse de 6 chiffres en simple précision et une mantisse de 14 chiffres en double précision. De plus, le format utilisé est celui de signe et de grandeur.

Bornes d'erreur

La représentation en virgule flottante entraîne inévitablement des erreurs, vu que la mantisse est réduite à un nombre fini t de chiffres.

Contrairement au cas de la virgule fixe, le cas de la virgule flottante est basé sur l'analyse de l'erreur relative ε, définie par :

$$fl(x) = x(1+\varepsilon)$$

où x est un nombre réel non nul, et

$fl(x) = \text{sign } b^a.\text{mant}$ est sa représentation numérique.

Les erreurs de représentation numérique sont différentes pour les deux modes d'arithmétique : l'arrondi

la troncature.

Erreur d'arrondi

L'erreur absolue εx est bornée de la manière suivante :

$$-b^a.\frac{b^{-t}}{2} \leq \varepsilon x \leq b^a.\frac{b^{-t}}{2}$$

et comme $b^{a-1} \leq |x| < b^a$, on a :

$$-\frac{b^{-t+1}}{2} \leq \varepsilon \leq \frac{b^{-t+1}}{2} .$$

Erreur de troncature

L'erreur absolue εx est bornée de la manière suivante :

$$-b^{a}.b^{-t} < \varepsilon x \leq 0 \qquad \text{si } x > 0,$$

$$b^{a}.b^{-t} > \varepsilon x \geq 0 \qquad \text{si } x < 0$$

et comme $b^{a-1} \leq |x| < b^{a}$, on a :

$$-b^{-t+1} < \varepsilon \leq 0.$$

B MULTIPLICATION A L'AIDE D'UN REGISTRE DE LONGUEUR DOUBLE

Arithmétique

Le processus de multiplication à l'aide d'un registre de longueur double est le suivant :

- Le produit de deux nombres réels non nuls x_1 et x_2 qui sont supposés être représentés exactement en machine respectivement par $sign_1\ b^{a_1}.mant_1$ et $sign_2\ b^{a_2}.mant_2$, est représenté par $sign\ b^{a}.mant$, où

$sign = (sign_1)(sign_2)$,

$a = a_1 + a_2 - L$

le produit des deux mantisses $mant_1$ et $mant_2$ se fait de manière exacte dans un registre de longueur $2t$ (longueur double); puis le résultat est normalisé (on déplace la mantisse de L places vers la gauche) et, ensuite, est arrondi ou tronqué,

suivant l'arithmétique de l'ordinateur, à t chiffres et placé dans mant.

- Le produit de deux nombres réels, dont l'un au moins est nul, est nul.

Remarque :

L'ordinateur Siemens 4004 utilise cette méthode de multiplication, avec le mode de troncature, dans le cas de la simple précision.

Bornes d'erreur

Ce processus de multiplication calcule la valeur exacte du produit des mantisses (puisque le registre est de longueur double), puis l'arrondit ou la tronque à t chiffres. Donc, l'erreur est semblable à celle produite par la représentation numérique, et, si on définit

$$fl(x_1 x_2) = x_1 x_2 (1+\beta)$$

où $x_1 x_2$ est le produit des nombres x_1 et x_2, supposés représentables exactement en machine,

$fl(x_1 x_2)$ est la représentation de ce produit,

on obtient :

en cas d'arrondi $$-\frac{b^{-t+1}}{2} \leq \beta \leq \frac{b^{-t+1}}{2},$$

en cas de troncature $$-b^{-t+1} < \beta \leq 0.$$

C ADDITION A L'AIDE DE CHIFFRES DE GARDE

Arithmétique

Le processus d'addition à l'aide de g chiffres de garde est le suivant :

Si l'un des deux nombres est nul, la somme égale directement et exactement l'autre nombre.

Dans le cas contraire, on peut toujours supposer que les nombres réels non nuls, représentés exactement en machine respectivement par $sign_1\ b^{a_1}.mant_1$ et $sign_2\ b^{a_2}.mant_2$ sont tels que $a_2 > a_1$

ou $a_2 = a_1$ et $|mant_1| \geq |mant_2|$.

La mantisse $mant_1$ est d'abord déplacée de $a_2 - a_1$ places vers la droite, puis est arrondie ou tronquée à $t + g$ chiffres (g chiffres de garde), ce qui entraîne une erreur absolue ε_1. Puis, la valeur ainsi obtenue, affectée du signe $sign_1$, est additionnée dans un registre de $t + g$ chiffres à la mantisse $mant_2$, affectée du signe $sign_2$. Le résultat est alors normalisé en déplaçant la mantisse de L places vers la gauche, puis arrondi ou tronqué à t chiffres et assigné à mant , ce qui entraîne une erreur absolue ε_2. Le résultat de l'addition est : $sign\ b^{a_2-L}.mant$.

Remarque :

L'ordinateur Siemens 4004 utilise cette méthode d'addition, avec le mode de troncature, dans le cas de la simple précision ($g = 1$).

Bornes d'erreur

J.P. Thiran calcule des bornes exactes de l'erreur engendrée par cette méthode d'addition [15]. Il définit l'erreur relative α par :

$$fl(x_1+x_2) = (x_1+x_2)(1+\alpha)$$

où x_1+x_2 est la somme des nombres x_1 et x_2, supposés représentables exactement en machine,

$fl(x_1+x_2)$ est la valeur calculée de cette somme,

et il obtient :

en cas d'arrondi : $$-\frac{b^{-t+1}}{2} < \alpha < \frac{b^{-t+1}}{2}(1 + b^{-g}) \quad (2.17)$$

en cas de troncature : $$-l \leq \alpha \leq u \quad (2.18)$$

où $$l = b^{-t+1}\,\frac{1 - b^{-t}}{1 + b^{-t+1}(1 - b^{-t})}$$

$$u = b^{-t-g+1}\,\frac{1 - b^{-t}}{1 - b^{-t-g+1}(1 - b^{-t})}$$

Cependant, à toutes fins pratiques, il conseille de remplacer (2.17) par :

$$-\frac{b^{-t+1}}{2} < \alpha < \frac{b^{-t+1}}{2}$$

et (2.18) par : $$-b^{-t+1} < \alpha < b^{-t-g+1},$$

ce qui constitue une approximation d'autant meilleure que la base b de l'arithmétique est grande.

2.3.2 MODELES STATISTIQUES POUR LES ERREURS

Comme on l'a montré au paragraphe 2.3.1, l'erreur relative ne dépend ni du signe du nombre exact, ni de la valeur de l'exposant dans la décomposition en format de signe et de grandeur. On pourra donc se limiter ici à étudier l'erreur relative

$$\varepsilon = \frac{fl(x) - x}{x} = \frac{\delta}{x}$$

où $b^{-1} \leq x < 1$, et

δ est l'erreur absolue.

D'un point de vue non déterministe, on peut considérer l'erreur relative comme la variable aléatoire quotient des variables aléatoires indépendantes δ et x (voir paragraphe 2.1).

Dans ce cas, pour pouvoir déterminer la fonction de densité de ε, on devra se servir du théorème suivant :

Théorème ([6], paragraphe 6.2)

Etant données deux variables aléatoires indépendantes X et Y, si leurs fonctions de densité sont respectivement $f_X(x)$ et $f_Y(y)$, la fonction de densité de la variable aléatoire $W = \frac{X}{Y}$ est donnée par :

$$f_W(w) = \int_{-\infty}^{\infty} |y| f_X(wy) f_Y(y)\, dy.$$

Les résultats présentés dans les paragraphes suivants à propos des erreurs relatives sont obtenus par Kaneko et Liu [8].

A DISTRIBUTION DE LA MANTISSE

Hamming a proposé un modèle propre à la représentation des nombres en virgule flottante [5]. Il considère que la mantisse d'un nombre flottant suit une distribution réciproque. En fait, la fonction de densité est :

$$r(x) = \frac{1}{x.\ln b} \quad , \quad b^{-1} \leq x \leq 1.$$

où b est la base de l'arithmétique.

Ce modèle est appuyé par des résultats expérimentaux et par la persistance de cette distribution à travers des opérations arithmétiques simples.

B ERREUR DE REPPESENTATION NUMERIQUE

Comme on considère que l'erreur absolue δ admet une distribution uniforme sur l'intervalle

$\left[-\frac{b^{-t}}{2} , \frac{b^{-t}}{2} \right]$ dans le cas d'arrondi et

$[- b^{-t}, 0]$ dans le cas de troncature,

on peut énoncer les propositions suivantes :

Proposition 2.1

En cas d'arrondi, l'erreur relative de représentation numérique admet la fonction de densité suivante :

$$f_A(u) = \begin{cases} 0 & \text{si } |u| > \dfrac{b^{-t+1}}{2} \\ (b-1)/b^{-t+1}\ln b & \text{si } |u| < \dfrac{b^{-t}}{2} \\ (b^{-t+1}-2u)/2ub^{-t+1}\ln b & \text{si } \dfrac{b^{-t}}{2} \leq u \leq \dfrac{b^{-t+1}}{2} \\ (-b^{-t+1}-2u)/2ub^{-t+1}\ln b & \text{si } -\dfrac{b^{-t+1}}{2} \leq u \leq -\dfrac{b^{-t}}{2} \end{cases}$$

Démonstration :

En appliquant le théorème avec $A = \frac{X}{Y}$, X = δ, Y = mantisse, on obtient :

$$f_A(u) = \int_{-\infty}^{\infty} |v| f_X(uv) f_Y(y)\, dy$$

$$= \int_{S_u} \frac{1}{b^{-t}.v\ln b} |v|\, dv$$

où $S_u = \{ v \text{ tel que } v \in [b^{-1},1] \text{ et } uv \in [-\frac{b^{-t}}{2}, \frac{b^{-t}}{2}] \}$.

Si $|u| > \frac{b^{-t+1}}{2}$ alors $S_u = \phi$ et $f_A(u) = 0$.

Si $|u| < \frac{b^{-t}}{2}$ alors $\begin{cases} S_u = [b^{-1}, 1] & \text{et} \\ f_A(u) = (b-1)/b^{-t+1}\ln b. \end{cases}$

Si $\frac{b^{-t}}{2} \leq u \leq \frac{b^{-t+1}}{2}$ alors $\begin{cases} S_u = [b^{-1}, \frac{b^{-t}}{2u}] & \text{et} \\ f_A(u) = (b^{-t+1}-2u)/2ub^{-t+1}\ln b. \end{cases}$

Si $-\frac{b^{-t}}{2} \geq u \geq -\frac{b^{-t+1}}{2}$ alors $\left[\begin{array}{l} S_u = \left[b^{-1}, -\frac{b^{-t}}{2u} \right] \quad \text{et} \\ f_A(u) = (-b^{-t+1}-2u)/2ub^{-t+1}\ln b. \end{array} \right.$

Proposition 2.2

En cas de troncature, l'erreur relative de représentation numérique admet la fonction de densité suivante :

$$f_T(u) = \begin{cases} 0 & \text{si } u > 0 \text{ ou } u \leq -b^{-t+1} \\ (b-1)/b^{-t+1}\ln b & \text{si } -b^{-t} < u \leq 0 \\ (-b^{-t+1}-u)/ub^{-t+1}\ln b & \text{si } -b^{-t+1} < u \leq -b^{-t} \end{cases}$$

Démonstration :

En appliquant le théorème avec $T = \frac{X}{Y}$, X = δ, Y = mantisse, on obtient :

$$f_T(u) = \int_{-\infty}^{\infty} |v| f_X(uy) f_Y(v)\, dy$$

$$= \int_{S_u} \frac{b^t}{v.\ln b} |y|\, dy$$

où $S_u = \{ y \text{ tel que } y \in [b^{-1},1] \text{ et } uy \in [-b^{-t},0] \}$.

Si $u > 0$ alors $S_u = \phi$ et $f_T(u) = 0$.

Si $u < -b^{-t+1}$ alors $S_u = \phi$ et $f_T(u) = 0$.

Si $-b^{-t} < u \leq 0$ alors $\left[\begin{array}{l} S_u = [b^{-1}, 1] \quad \text{et} \\ f_T(u) = (b-1)/b^{-t+1}\ln b. \end{array} \right.$

Si $-b^{-t+1} \leq u \leq -b^{-t}$ alors $\begin{cases} S_u = [b^{-1}, -\frac{b^{-t}}{u}] & \text{et} \\ f_T(u) = (-b^{-t+1}-u)/ub^{-t+1}\ln b. \end{cases}$

Remarque :

Comme dans le cas où $b = 2$ et t est suffisamment grand, $(b-1)/\ln b \simeq 1.44$ et

$$b^{-t} \simeq b^{-t+1},$$

on peut considérer que l'erreur relative est distribuée uniformément sur l'intervalle

$[-2^{-t}, 2^{-t}]$ dans le cas d'arrondi et

$[-2^{-t+1}, 0]$ dans le cas de troncature.

C ERREUR DE MULTIPLICATION

Etant donné que l'erreur produite par une multiplication utilisant un registre de longueur double est tout à fait semblable à celle produite par la représentation numérique, on obtient ici exactement les mêmes résultats qu'au paragraphe précédent.

Si on définit $\delta_x = E(\varepsilon)$

$$\Delta_x^2 = \text{var}(\varepsilon) \qquad (2.19)$$

on a :

si $b = 2$ $\quad \delta_x = \begin{cases} 0 & \text{en cas d'arrondi} \\ -2^{-t} & \text{en cas de troncature} \end{cases}$

$$\Delta_x^2 = \frac{1}{3} 2^{-2t}.$$

si $b > 2$ $\quad \delta_x = \begin{cases} 0 & \text{en cas d'arrondi} \\ -b^{-t}(b-1)/2\ln b & \text{en cas de troncature} \end{cases}$

$$\Delta_x^2 = \begin{cases} b^{-2t}(b^2-1)/24\ln b & \text{en cas d'arrondi} \\ b^{-2t}(b^2-1)/6\ln b - b^{-2t}(b-1)^2/(2\ln b)^2 & \text{en cas de troncature.} \end{cases}$$

D ERREUR D'ADDITION

Le cas de l'erreur d'addition est un peu plus compliqué que les précédents. En effet, l'erreur absolue n'est plus considérée comme uniformément distribuée.

Tout d'abord, Kaneko et Liu [8] se basent sur un article de Sweeney [14] pour négliger l'erreur absolue ε_1. Ils affirment d'autre part qu'il y a une grande probabilité que $\varepsilon_2 = 0$. Ceci arrive en fait quand :

(1) $a_2 = a_1$ et $sign_1 = -sign_2$

(2) $a_2 = a_1$ et $sign_1 = sign_2$ mais $L = 0$

(3) $a_2 = a_1 + 1$ et $L = 1$

Soit p_o la probabilité que $\varepsilon_2 = 0$. On obtient donc pour ε_2 une densité de probabilité constituée de deux composantes : une fonction delta d'amplitude p_o à l'origine et une distribution uniforme sur

$$\left[-\frac{b^{-t}}{2}, \frac{b^{-t}}{2}\right] \text{ en cas d'arrondi}$$

$$\left[-b^{-t+1}, 0\right] \text{ en cas de troncature.}$$

D'une manière tout à fait similaire à celle du paragraphe précédent, on obtient les propositions suivantes :

Proposition 2.3

En cas d'arrondi, l'erreur relative ε admet la fonction de densité suivante :

$$f_A(u) = \begin{cases} 0 & \text{si } |u| \geq \frac{b^{-t+1}}{2} \\ p_o\delta(u)+(1-p_o)(b-1)/b^{-t+1}\ln b & \text{si } |u| \leq \frac{b^{-t}}{2} \\ (1-p_o)(b^{-t+1}-2u)/2ub^{-t+1}\ln b & \text{si } \frac{b^{-t}}{2} < u < \frac{b^{-t+1}}{2} \\ (1-p_o)(-b^{-t+1}-2u)/2ub^{-t+1}\ln b & \text{si } -\frac{b^{-t+1}}{2} < u < -\frac{b^{-t}}{2} \end{cases}$$

Proposition 2.4

En cas de troncature, l'erreur relative ε admet la fonction de densité suivante :

$$f_T(u) = \begin{cases} 0 & \text{si } u > 0 \text{ ou } u < -b^{-t+1} \\ p_o\delta(u)+(1-p_o)(b-1)/b^{-t+1}\ln b & \text{si } -b^{-t} \leq u \leq 0 \\ (1-p_o)(-b^{-t+1}-u)/ub^{-t+1}\ln b & \text{si } -b^{-t+1} \leq u \leq -b^{-t} \end{cases}$$

Remarques :

1- La valeur p_o dépend de la base de l'arithmétique et des données des additions. Elle varie de 0.3 à 0.8 [8] .

2- Dans le cas où $b = 2$ et t est suffisamment grand, on peut considérer que l'erreur relative admet la fonction de densité suivante :

$$f_A(u) = \begin{cases} 0 & \text{si } |u| > 2^{-t} \\ p_o\delta(u)+(1-p_o)/2^{-t+1} & \text{si } |u| < 2^{-t} \end{cases}$$

en cas d'arrondi, et

$$f_T(u) = \begin{cases} 0 & \text{si } u > 0 \text{ ou } u < -2^{-t+1} \\ p_o\delta(u)+(1-p_o)/2^{-t+1} & \text{si } -2^{-t+1} \leq u \leq 0 \end{cases}$$

en cas de troncature.

Si on définit $\delta_+ = E(\varepsilon)$

$$\Delta_+^2 = \text{var}(\varepsilon) \qquad (2.20)$$

on a :

si $b = 2$ $\quad \delta_+ = \begin{cases} 0 & \text{en cas d'arrondi} \\ -(1-p_o)2^{-t} & \text{en cas de troncature} \end{cases}$

$$\Delta_+^2 = \begin{cases} (1-p_o)2^{-2t}/3 & \text{en cas d'arrondi} \\ 4(1-p_o)2^{-2t}/3-(1-p_o)^2 2^{-2t} & \text{en cas de troncature} \end{cases}$$

si $b > 2$ $\quad \delta_+ = \begin{cases} 0 & \text{en cas d'arrondi} \\ -b^{-t}(1-p_o)(b-1)/2\ln b & \text{en cas de troncature} \end{cases}$

$$\Delta_+^2 = \begin{cases} (1-p_o)b^{-2t}(b^2-1)/24\ln b & \text{en cas d'arrondi} \\ (1-p_o)b^{-2t}(b^2-1)/6\ln b - (1-p_o)^2 b^{-2t}(b-1)^2/(2\ln b)^2 & \text{en cas de troncature.} \end{cases}$$

A N A L Y S E DES E R R E U R S N U M E R I Q U E S

DANS LES

A L G O R I T H M E S DE LA T R A N S F O R M E E

DE F O U R I E R R A P I D E

EN

V I R G U L E F I X E

INTRODUCTION

Les erreurs numériques des algorithmes de Cooley et Tukey et de Sande et Tukey sont présentées dans ce chapitre avec l'hypothèse de l'arithmétique en virgule fixe et de la représentation binaire en "complément - à - deux". La plupart des estimations sont données pour les algorithmes en base 2 aux paragraphes 3.1 et 3.2. Quelques résultats sont indiqués pour les algorithmes dans une base composite au paragraphe 3.3.

Les nombres sont représentés par une suite de t+1 bits, dont le premier est le signe; leur valeur absolue est inférieure à l'unité. Le résultat d'un produit est soit arrondi, soit tronqué, sauf dans le cas des multiplications réelles par $\pm$ 1, pour lesquelles aucune opération n'est effectuée. Deux procédés permettent d'éviter un dépassement après une addition. Le premier consiste à diviser le signal d'entrée par le nombre de ses composantes, le second à diviser par 2 les résultats avant d'effectuer les calculs de chaque étape.

La moyenne et le total des carrés des erreurs sont prédits au moyen du modèle statistique décrit dans la sous-division 3.1.1. Les paragraphes 3.1 et 3.2 traitent l'arrondi et la troncature avec ou sans déplacement à chaque étape; le paragraphe 3.3 envisage seulement l'arrondi sans déplacement. Cette analyse est complétée au chapitre 5 (paragraphe 5.1) par les résultats expérimentaux.

Comme les nombres y sont représentés en "signe et grandeur", le modèle statistique est modifié de manière à annuler les moyennes des erreurs. La bonne correspondance entre les résultats expérimentaux et les valeurs prédites par la théorie confirme le choix du modèle modifié et du modèle initial décrit en 3.1.1.

Une question demeure ouverte. Un déplacement peut n'être effectué que lors d'un dépassement. Cette méthode produit des erreurs moindres que le déplacement à chaque étape qui en est le pire cas. Il est nécessaire de tenir compte du signal d'entrée dont dépend le nombre de déplacement. Ce problème a été abordé par Weinstein [18].

3 . 1 RESULTATS DE L'ANALYSE DE L'ALGORITHME DE COOLEY ET TUKEY

3.1.1 MODELE STATISTIQUE

Le calcul élémentaire de l'algorithme de Cooley et Tukey est un "papillon" de la forme indiquée en (1.12) . Le modèle statistique de l'erreur de discrétisation numérique s'établit en associant à chaque multiplication un signal d'erreur de type aléatoire.

Le schéma habituel d'un "papillon" sera remplacé par celui indiqué à la figure (8a) dans laquelle $\varepsilon(i,q)$ représente l'erreur complexe introduite dans le calcul du (i + 1)-ème tableau et plus précisément dans la multiplication du q-ème élément par un coefficient complexe.

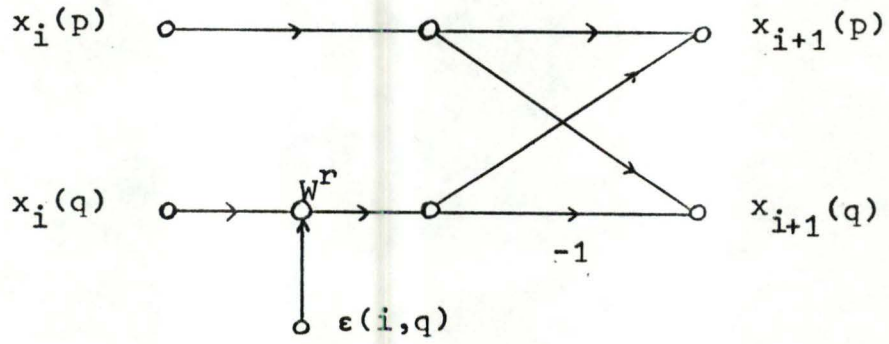

Fig. 8a

Le produit des deux nombres complexes $z_1 = x_1 + jy_1$ et $z_2 = x_2 + jy_2$ est généralement effectué de la manière suivante :

$$L\,[z_1 z_2] = L\,[x_1\ x_2] + L\,[-y_1\ y_2] + j\,(L\,[y_1\ x_2] + L\,[x_1\ y_2])$$

où le symbôle L [.] désigne soit l'arrondi, soit la troncature.

Une autre méthode a été proposée par Liu et Peled [10] :

$$L\,[z_1\ z_2] = L\,[x_1\ x_2 - y_1\ y_2] + j\ L\,[y_1\ x_2 + x_1\ y_2] \qquad (2)$$

Dans le premier cas, quatre erreurs de discrétisation numérique sont introduites, dans le second cas deux seulement.
Rappelons et précisons les hypothèses du modèle statistique décrit au paragraphe 2.1.3.

1) L'erreur de discrétisation numérique provenant d'une multiplication entre nombre réels est distribuée uniformément.
Elle est caractérisée par les relations (2.11) à (2.14).

2) Toutes les erreurs provenant des multiplications entre nombres réels ne sont pas corrélées. Par conséquent, la variance de l'erreur introduite par une multiplication entre nombres complexes est :

$$\alpha\ (2^{-2t}/12) \qquad (3.1)$$

où α vaut 4 pour le schéma de multiplication (1) et 2 pour le schéma (2).
En outre, en désignant par e(i,p) l'erreur numérique introduite dans le calcul de $x_{i+1}(p)$, $0 \leqslant i \leqslant M-1$, les processus aléatoires

$$\{\ e(i,p),\ i \text{ fixé},\ p = 0,1,\ldots,\ N-1\ \}$$

constituent des bruits blancs. De plus, les variables aléatoires de deux de ces processus ne sont pas corrélées.

3) Les erreurs dues aux multiplications ne sont pas corrélées avec le signal d'entrée. Donc :

$$E\,[e(i,p).\ X_n] = E\,[e(i,p)]\ .\ E\,[X_n]$$

où i = 0, 1, ... M-1 ; n,p = 0, 1, ... N-1 et où l'élément x(n) du signal d'entrée est une réalisation particulière de la variable aléatoire X_n.
Il faut encore empêcher un dépassement éventuel du registre.
Des relations (1.12) on déduit les inégalités

$$\max\{[x_i(p)],[x_i(1)]\} \leqslant \max\{[x_{i+1}(p)],[x_{i+1}(1)]\}$$

$$\leqslant 2\ \max\{[x_i(p)],[x_i(1)]\} \qquad (3.2)$$

Par conséquent, il est clair que la condition

$$|y(p)| < 1 \qquad p = 0,1,\ldots, N-1$$

suffit pour éviter tout dépassement. On obtient

$$|y(p)| = \left| \sum_{n=0}^{N-1} x(n)\, W^{np} \right| < N \max_n \left(|x(n)| \right)$$

Il suffit donc d'exiger que

$$|x(n)| < 1/N \qquad n = 0,1,\ldots,N-1 \tag{3.3}$$

Une seconde méthode est basée sur l'inégalité

$$|x_i(p)| < 1/2 \qquad p = 0,1,\ldots,N-1$$

qui suffit pour éviter tout dépassement à l'étape suivante.
Le signal d'entrée est soumis à la contrainte

$$|x(n)| < 1 \qquad n = 0,1,\ldots,N-1 \tag{3.4}$$

De plus, avant toute opération il faut diviser par deux les éléments de chaque tableau. La fig. (8a) devient :

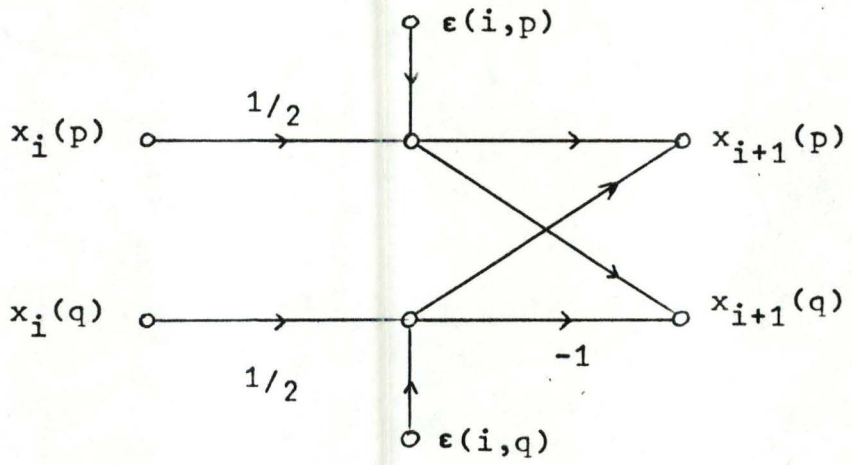

Fig. 8b

Le modèle statistique est complété par une quatrième hypothèse

4) Les erreurs dues aux déplacements sont distribuées uniformément et caractérisées par les relations (23.6) et (23.7).
Elles ne sont corrélées ni entre elles, ni avec le signal d'entrée, dans le sens défini pour les hypothèses 2 et 4. Elles ne sont également pas corrélées avec les erreurs de discrétisation numérique :

$$E[\, d(i,p).\ e(j,q)] = E[d(i,p)]\ \ E[\, e(j,q)]$$

où $i,j=0,1,\ldots,M-1$; $p,q=0,1,\ldots,N-1$ et où $d(i,p)$ désigne

l'erreur aléatoire introduite par le déplacement pendant le calcul de $x_{i+1}(p)$

Remarques

1) Une troisième méthode pour éviter le dépassement du registre est de n'effectuer une division par 2 que lorsqu'un dépassement est détecté. Le calcul est alors repris après avoir divisé l'entièreté du tableau par 2.

2) La seconde hypothèse du modèle statistique est contredit par le fait que les erreurs commises sur deux éléments d'un même "papillon" sont de signes opposés : de la fig. 8a on déduit immédiatement que $e(i,p) = e(i,q) = \varepsilon(i,q)$. Comme ces erreurs se propagent ensuite suivant les chemins qui ne se coupent pas, cette corrélation ne perturbe pas le calcul des variances.

3.1.2 PROPRIETES DE L'ALGORITHME DE COOLEY ET TUKEY

Les relations (1.11) qui définissent l'algorithme peuvent être écrites sous la forme suivante :

$$x_m(n_1,\ldots,n_{M-m},\ p_m,\ldots,p_1)$$

$$= \sum_{n_{M-m+1}=0}^{1} x_{m-1}(n_1,\ldots,n_{M-m+1},p_{m-1},\ldots,p_1)$$

$$\cdot \exp\left[-j\,2\pi\, n_{M-m+1}(p_m 2^{m-1} + \ldots + p_1)/2^m\right]$$

$$x_0(n_1,\ldots,n_M) = x(n)$$

$$\text{où } n = \sum_{i=1}^{M} n_i\, 2^{i-1} \quad \text{et} \quad p = \sum_{i=1}^{M} p_i\, 2^{i-1} \qquad (3.5)$$

Les relations (1.23) définissent le s-ème élément du l-ème bloc de l'étape m. Il suffit de poser r=2.

Proposition 3.1

Une erreur de variance Δ^2 introduite dans le calcul du tableau x_m d'une TFR de $N = 2^M$ échantillons donne lieu à une erreur de variance Δ^2 dans 2^{M-m} élément du tableau des résultats de la TFR.

Démonstration

L'algorithme de Cooley et Tukey est représenté par un graphique en structure d'arbre dont les branches se dédoublent successivement. En particulier, un élément du m-ème étage des calculs sera relié à deux éléments du tableau x_{m+1}, quatre du tableau x_{m+2},....,2^{M-m} du tableau x_M (fig. 9a).
L'erreur qui affecte ces 2^{M-m} éléments est toujours de variance Δ^2, puisqu'elle n'est multipliée que par des nombres complexes de grandeur unité.

Proposition 3.2

La variance de l'erreur de discrétion sur chaque résultat de l'algorithme est la somme de N-1 variances d'erreur dont 2^{M-m} d'entre elles proviennent des erreurs introduites dans le calcul du tableau x_m.

Démonstration

Un élément du tableau final x_M est relié à deux éléments du tableau x_{M-1},....2^{M-m} du tableau x_m,...,2^{M-1} du tableau x_1 par une structure

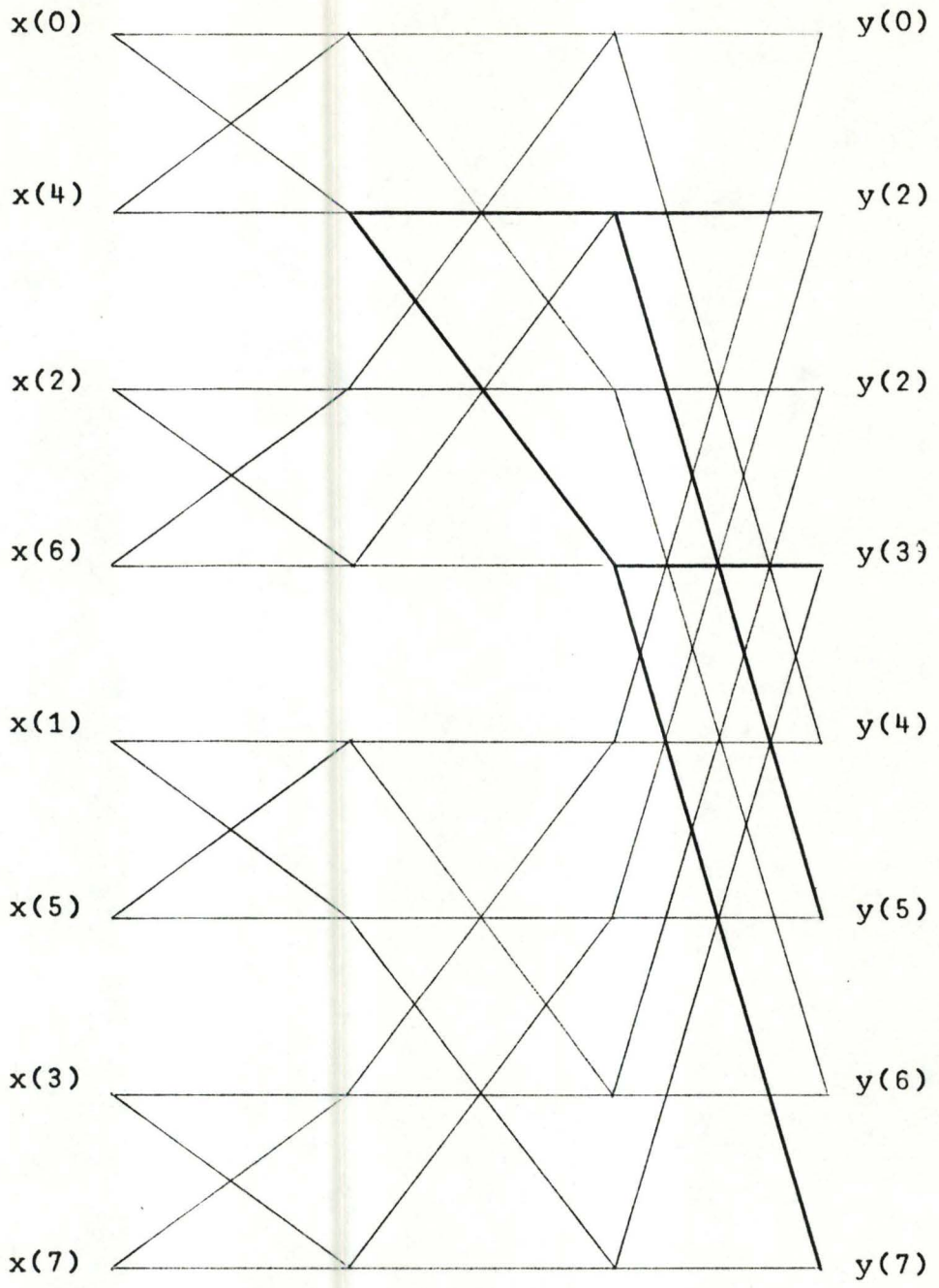

Fig. 9a

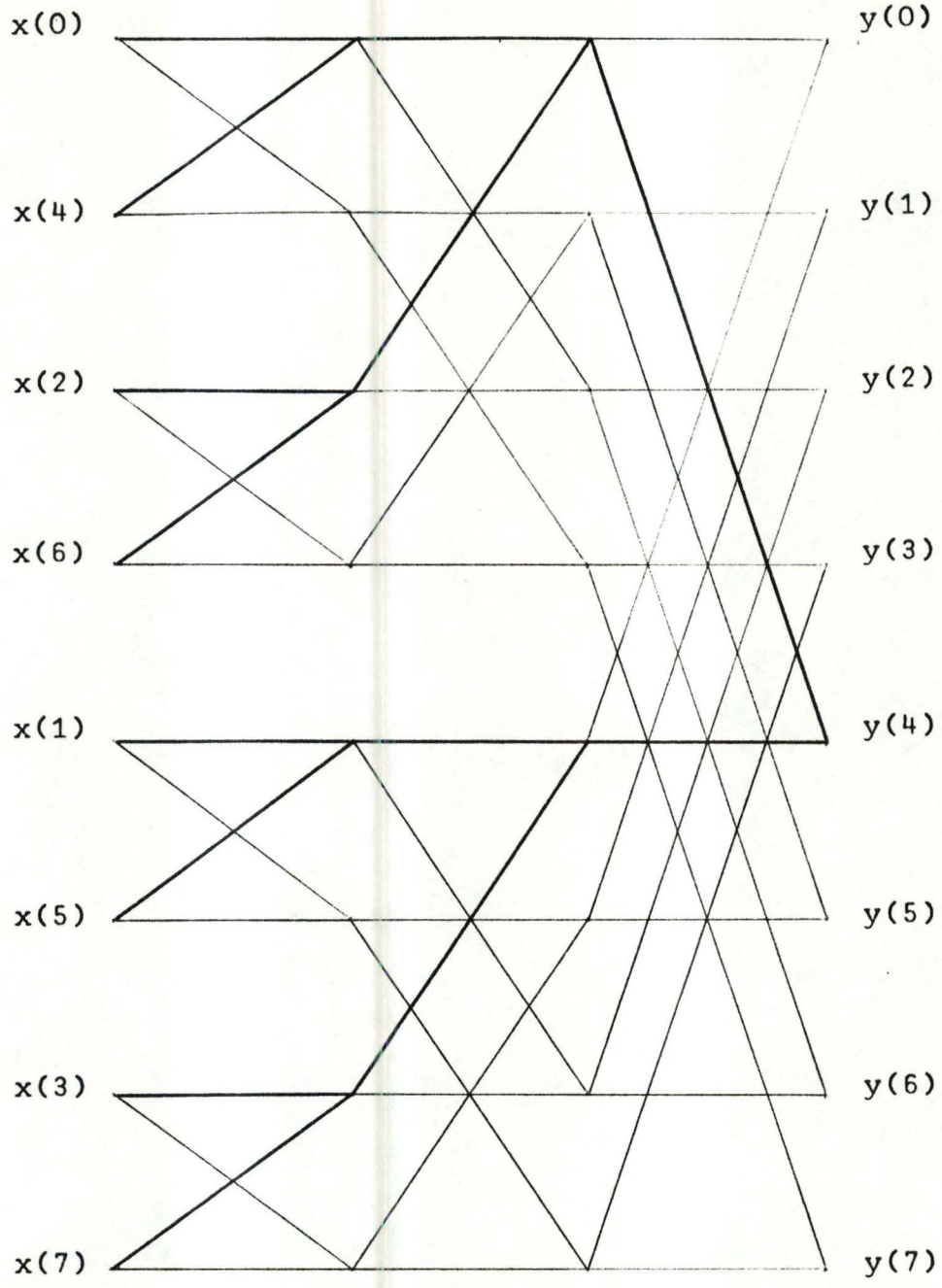

Fig. 9b

d'arbre (fig.9b).
Aucune erreur d'arrondi ou de troncature n'est commise sur le tableau initial. Le nombre de contributions est donc égal à

$$\sum_{i=1}^{M} 2^{M-i} = N-1$$

Il suffit de les additionner puisque les erreurs sont non corrélées

Remarque

L'erreur due aux déplacements n'a pas encore été envisagée.
Par les hypothèses de non corrélation, il suffit de cumuler les contributions des erreurs numériques provenant des déplacements et de la discrétisation pour obtenir l'erreur totale.

Proposition 3.3

Dans l'hypothèse d'un déplacement à chaque étape, la contribution de l'erreur ainsi créée est $N(N-1)\ 2^{-2t}/2$ pour chaque élément du tableau final.

Démonstration

A chaque étape de l'algorithme, le déplacement est introduit avant toute opération. Une erreur de variance $2.(2^{-2t}/4).2^{2m}$ affecte chaque élément du tableau x_m. Le facteur 2 tient compte des erreurs sur les parties réelles et imaginaires, le facteur $2^{-2t}/4$ est la variance de l'erreur introduite et le facteur 2^{2m} tient compte des déplacements précédents.
Par un raisonnement analogue à celui de la proposition 3.2, chaque élément du tableau final reçoit une erreur de variance égale à

$$2^{-2t}/2 \sum_{m=0}^{M} 2^{M-m} \cdot 2^{2m} = (2^{-2t}/2)\ N(N-1)$$

La somme débute à l'étape 0 et se termine à l'étape M-1 car le tableau initial subit un déplacement mais pas le tableau final.

Proposition 3.4

Il existe une relation de transformée de Fourier discrète inverse entre les éléments d'un bloc de l'étape m et les éléments correspondants du tableau initial.

La démonstration a été donnée dans le premier chapitre par la proposition 1.1 dans le cas où $N = r^M$, $r \geq 2$. Les éléments du tableau initial correspondants au bloc $\left\{ x_m(12^m + s) \right\}_{s=0}^{2^m-1}$ sont $\left\{ x_0 (1\ 2^m + s) \right\}_{s=0}^{2^m-1}$

3.1.3 ETUDE DES ERREURS NUMERIQUES

3.1.3.1 ARRONDI - SANS DEPLACEMENT

Tout d'abord, par (2.11), il est évident que $E[e(p)] = 0$; $e(p)$ désigne l'erreur commise sur $y(p)$.
Ensuite pour évaluer le total du carré des erreurs, il faut remarquer que les deux premières étapes qui ne comportent que des multiplications par ± 1 ou $\pm j$ sont effectuées sans erreur.
De plus, à chaque étape suivante, N multiplications complexes sont calculées dans les 2^{M-m} blocs. Toutefois dans chacun d'eux quatre multiplications par ± 1 ou $\pm j$ n'entraînent pas d'erreur. Par la proposition 3.1 et par (3.1), on obtient :

$$\sum_{p=0}^{N-1} (E[|e(p)|^2])_A = \alpha (2^{-2t}/12) \cdot \sum_{m=3}^{M} (N-4.2^{M-m})\ 2^{M-m}$$

$$= \alpha (2^{-2t}/12) (N^2/6 - N + 4/3) \qquad (3.6)$$

Tous les résultats ne reçoivent pas la même contribution d'erreur. Pour évaluer chacun des termes de (3.6) il est nécessaire de décomposer l'indice p en :

$$p = \sum_{i=1}^{M} p_i \, 2^{i-1}$$

Dans ce développement si m est le premier indice pour lequel apparaisse un "1", on a $m = \min\{ i : p_i = 1\}$. Par (3.5), à l'étape m les éléments reliés à y (p) sont multipliés par -1, à l'étape m + 1, ils sont multipliés par $\pm j$. Dans toutes les étapes ultérieures les multiplications entraînent des erreurs, c'est-à-dire à partir de l'étape s (p) définie par $s(p) = 2 + \min\{ i : p_i = 1\}$ Par la proposition 53.2) on obtient :

$$E[|e(p)|^2] = \sum_{m=s(p)}^{M} \alpha \, (2^{-2t}/2) \, 2^{M-m}$$

$$= \alpha \, (2^{-2t}/2) \, [N/2^{s(p)-1} - 1]$$

D'autre part, il est immédiat que y (p) est calculé sans erreur pour les indices p = 0, N/4, N/2, 3N/4. Le résultat final est donc

$$(E[|e(p)|^2])_A = \begin{cases} 0 & p = 0, N/4, N/2, 3N/4 \\ \alpha \, (2^{-2t}/2)[N/2^{s(p)-1}] - 1 & \text{autrement.} \end{cases} \qquad (3.7)$$

Le résultat établi en (3.6) peut être retrouvé à partir de (3.7). En effet :

$$\sum_{p=0}^{N-1} (E[|e(p)|^2])_A = \sum_{p=1}^{N-1} \alpha(2^{-2t}/2)(N/2^{s(p)-1} - 1) \qquad (3.8)$$

où $p \neq N/4, N/2, 3N/4$

Seule la somme suivante pose des problèmes :

$$\sum_{p=1}^{N-1} 1/2^{\,s(p)-1} = 1/2 . \sum_{p=1}^{N-1} 1/2^{\min\{i, p_i = i\}} \qquad p \neq N/4, N/2, 3N/4 \tag{3.9}$$

Elle peut être décomposée suivant les indices

$$p = 1.2^0 \;;\; p = 1.2^1 \;;\; \ldots \;;\; p = 1.2^{M-2} \;;\; p = 1.2^{M-1}$$

où l est impair et $\min \{ i : p_i = 1 \}$ vaut respectivement 1, 2,...M.
En remarquant que les indices à rejeter sont :

$p = 1.2^{M-1}$, l impair, qui équivaut à $p = 2^{M-1}$

$p = 1.2^{M-2}$, l impair, qui équivaut à $p = 2^{M-2}$ ou $p = 3.2^{M-2}$

la somme (3.9) devient

$$1/2 \sum_{k=0}^{M-3} \sum_{\{p=1.2^k ; l=2r+1\}} 2^{-k-1}$$

Comme l'ensemble $\{p=1.2^k ;\ l=2r+1\}$, où $1 \leq p \leq N-1$, compte $N/2^{k+1}$ éléments

on obtient

$$N/2 \sum_{k=0}^{M-3} 2^{-2k-2} = N/6 \,.\, (1-16/N^2)$$

L'expression (3.8) devient

$$\sum_{p=0}^{N-1} \left(E\left[|e(p)|^2\right]\right)_A = \alpha \, (2^{-2t}/2)\left[(N^2/6 - 8/3) - (N-4)\right]$$

qui est le résultat établi en (3.6).

3.1.3.2 TRONCATURE - SANS DEPLACEMENT

La relation (2.1.3)indique que la moyenne de l'erreur n'est pas nulle. Pour l'évaluer, il est nécessaire de détailler le calcul d'un "papillon";

$$x_m(r) = x_{m-1}(r) + x_{m-1}(s).\ U$$

$$x_m(s) = x_{m-1}(r) - x_{m-1}(s).\ U$$

On déduit de (3.5) que

$$r = \sum_{i=1}^{M-m} n_i\ 2^{M-i} + \sum_{i=1}^{m} p_i\ 2^{i-1} \quad \text{avec } p_m = 0$$

$$s = r + 2^{m-1}$$

$$U = \exp\left[-j2\pi \cdot \sum_{i=1}^{m-1} p_i\ 2^{i-1}/2^m \right]$$

où m varie de 1 à M. Habituellement $x_{m-1}(s).U$ est d'abord calculé et tronqué à t bits avant d'être utilisé dans les calculs

La moyenne de l'erreur est donc :

$-2^{-t-1} \cdot \alpha/2\ (1+j)$ sur $x_m(r)$ (1)

$+2^{-t-1} \cdot \alpha/2\ (1+j)$ sur $x_m(s)$ (2)

où les coefficients $\alpha/2$ et $(1 + j)$ tiennent compte des $\alpha/2$ multiplications réelles nécessaires au calcul des parties réelle et imaginaire. A l'étape m, chaque bloc est composé de 2^m éléments dont la première moitié reçoit l'erreur indiquée en (1) et la seconde reçoit celle indiquée en (2).

Par la proposition (3.4) ces erreurs peuvent être ramenées à des erreurs équivalentes sur le signal d'entrée. Pour un bloc du tableau x_m, elles s'obtiennent en prenant la transformée de Fourier inverse des 2^m échantillons suivants :

$$E_i = \begin{cases} 0 & i = 0,\ P/4,\ P/2,\ 3P/4 \\ A & 0 < i \leq P/2 - 1 \\ -A & P/2 \leq i \leq P-1 \end{cases} \qquad (3.10)$$

où $A = -2^{-t-1}\ (\alpha/2)\ (1 + j)$ et $P = 2^m$

Les indices $i = 0, P/4, P/2, 3P/4$ correspondent à des éléments calculés sans erreur. Les erreurs équivalentes des éléments de ce bloc sont donc :

$$e_{k^\dagger} = 1/P. \sum_{i=0}^{P-1} E_i\ W_p^{ik^\dagger}$$

où $W_p = \exp(j2\pi/P)$ et $0 \leq k^\dagger \leq P-1$

En tenant compte de (3.10) on a

$$e_{k^\dagger} = 1/P \left[\sum_{i=0}^{P/2-1} E_i\ W_p^{ik^\dagger} \sum_{i=0}^{P/2-1} E_i\ W_p^{(P/2+i)k^\dagger} \right]$$

$$= 1/P \sum_{i=0}^{P/2-1} E_i W_p^{ik^\dagger} \left[1 - W_p^{P/2k^\dagger} \right]$$

où $W_p^{P/2.k^\dagger} = (-1)^{k^\dagger}$

Dans le cas où $k^\dagger$ est pair on obtient $e_{k^\dagger} = 0$

Dans le cas où $k^\dagger$ est impair, en tenant compte des termes nuls, on obtient :

$$e_{k^\dagger} = (2A/P) \left[\sum_{i=0}^{P/2-1} W_p^{ik^\dagger} - W_p^{iO} - W_p^{P/4.k^\dagger} \right]$$

où $\sum_{i=0}^{P/2-1} W_p^{ik^\dagger} = 2/(1-W_p^{k^\dagger})$ et $W_p^{P/4 \cdot k^\dagger} = j^{k^\dagger}$

Si $k^\dagger$ est le nombre binaire inversé de k, l'erreur équivalente sur le k-ème élément du bloc correspondant du tableau initial noté $x_0(12^m + k)$ est donc :

$$\begin{cases} -2/2^m \cdot 2^{-t-1} \cdot (1+j)\ \alpha/2 \cdot \left[(1+W_m^{k^\dagger})/(1-W_m^{k^\dagger}) - j^{k^\dagger}\right] & \text{si } k^\dagger \text{ est impair} \\ 0 & \text{si } k^\dagger \text{ est pair} \end{cases} \qquad (3.11)$$

Comme $x_o(12^m+k) = x_0(n_1, \ldots, n_M)$ les indices k et $k^\dagger$ sont représentés respectivement par :

$$\sum_{i=1}^{m} n_{M-i+1}\ 2^{i-1} \quad \text{et} \quad \sum_{i=1}^{m} n_{M-m+i}\ 2^{i-1}$$

Puisque $k^\dagger$ dépend de n et m on peut le désigner par f(n,m) qui sera pair ou impair suivant que n_{m-m+1} vaut 0 ou 1.
Si f (n,m) est impair, on constate que $j^{f(n,m)}$ vaut +j ou-j suivant que n_{M-m+2} vaut 0 ou 1.
L'expression (3.11) devient :

$$n_{M-m+1} \left[-2/2^m \cdot 2^{-t-1} \cdot (1+j)\ \alpha/2\right] \cdot \left[(1+W_m^{f(n,m)})/(1-W_m^{f(n,m)}) - j(-1)^{n_{M-m+2}}\right] \qquad (3.12)$$

L'erreur qui affecte y (p) est le p-ème résultat de la TFD des N échantillons déterminés par la somme des contributions (3.12) pour m = 3,...M puisque les deux premières étapes ne comportent pas d'erreur.

En tenant compte des résultats y (p), p = 0, N/4, N/2, 3N/4, qui sont calculés sans erreur on obtient finalement :

$$(\mathbb{E}\,|e(p)|)_T = \begin{cases} 0 & p = 0,\ N/4,\ N/2,\ 3N/4 \\ -\alpha.\ 2^{-t}/2.\ (1+j) \\ \quad .\ \Big(\text{TFD}\ \Big\{ \sum_{m=3}^{M} n_{M-m+1}.\ (1/2^m) \\ \quad .\ \Big[(1+W_m^{f(n,m)})/(1-W_m^{f(n,m)}) \\ \quad -\ j(-1)^{n_{M-m+2}} \Big] \Big)_{n=0,\ldots,N-1} \Big)_p & \text{autrement} \end{cases} \qquad (3.13)$$

où -TFD $\{\,.\,\}_n$ désigne la transformée de Fourier discrète des N échantillons de la suite $\{\,.\,\}_n$

-(TFD $\{\,.\,\}$)$_p$ en désigne le p-ème résultat.

La variance des erreurs numériques ne change pas, que la troncature (2.14) ou l'arrondi (2.12) soit utilisé. Donc :

$$\sum_{p=0}^{N-1} (E\left[|e(p)|^2\right])_T = \sum_{p=0}^{N-1} (E\left[|e(p)|^2\right]) + \sum_{p=0}^{N-1} |\ E\left[e(p)\right]|^2$$

où $E\left[\,e\,(p)\,\right]$ est détaillé par (3.13).

Par la relation de Parseval (1.2) on obtient :

$$\sum_{p=0}^{N-1} (E\left[|e(p)|^2\right])_T = \sum_{p=0}^{N-1} (E\left[|e(p)|^2\right])_A$$

$$+\ \alpha^2.\ 2^{-2t/2}.\ N \Big(\sum_{n=0}^{N-1} |\ldots \sum_{m=3}^{M} n_{M-m+1}$$

$$\cdot \left[(1+W_m^{f(n,m)})/(1-W_m^{f(n,m)}) - j\ (-1)^{n_{M-m+2}} \right] .1/2^m \Big|^2$$

(3.14)

3.1.3.3 ARRONDI - DEPLACEMENT A CHAQUE ETAPE

la variance des erreurs dues aux déplacements est donnée par la proposition (3.3). Celle des erreurs d'arrondi est

$$\begin{cases} 0 & p=0,\ N/4,\ N/2,\ 3N/4 \\ \alpha.2^{-2t}/12.\ \sum_{m=s(p)}^{M} 2^{M-m}.\ 2^{2m} & \text{autrement} \end{cases}$$

où le facteur 2^{2m} est dû aux déplacements.

En effet l'arrondi s'effectue sur des nombres qui ont été précédemment divisés m fois par 2. Il faut donc multiplier les erreurs par 2^m pour qu'elles correspondent aux véritables résultats d'une transformée de Fourier appliquée au signal d'entrée. Les erreurs d'arrondi et de déplacement étant non corrélées par la quatrième hypothèse du modèle statistique, la variance de l'erreur commise sur $y(p)$ est :

$$V\left[e(p)\right] = 2^{-2t}/12.\ N(N-1) + \begin{cases} 0 & p = 0,\ N/4,\ N/2, 3N/ \\ \alpha\ .\ 2^{-2t}/12.\ N\left[\ 2N - 2^{s(p)}\right] \end{cases}$$

(3.15)

où $s(p) = 2 + \min \left\{ i : p_i = 1 \right\}$

La moyenne des erreurs dues aux déplacements reste à déterminer. Comme le déplacement du tableau x_m a lieu avant de calculer x_{m+1}, chaque élément de x_m, m = 0, ... M-1 est affecté d'une erreur de moyenne $-(1+j)\ 2^{-b-1+m}$. Par la proposition (3.4), l'erreur équivalente sur les termes $x_o\ (l_2^m)$, $l = 0, \ldots\ 2^{M-m}-1$ possède une

moyenne égale à $-(1+j)\ 2^{-b-1+m}$.
Il faut effectuer la somme des contributions provenant des étapes $m = 0, \ldots, M-1$. Désignant l'erreur équivalente reportée sur l'élément $x_0\ (l2^m)$ par $\mu_e\ [x_0(l2^m)]$ on obtient :

$$\mu_e\ [\ x_0(l2^m)] \quad = \quad -\ (1+j)\ 2^{-b-1+m}$$

où $0 \leqslant l \leqslant 2^{M-m} - 1$ et $0 \leqslant m \leqslant M-1$ Par conséquent :

$$\mu_e[\ x_0(i)] \quad = \sum_{(\ m\ :l2^m\ =i)} [-(1+j)] \quad 2^{-b-1+m} \qquad (3.16)$$

Si i est nul, la somme débute, à m = 0 et se termine à m = M-1. Donc

$$\mu_e\ [\ x_0\ (0)\ ] = -(1+j)\ 2^{-b-1}\ (N-1)$$

Si i n'est pas nul, on peut le décomposer en :

$$i = \sum_{k=1}^{M} i_k\ 2^{k-1} \quad = \quad 2^{k-1} \sum_{k=K}^{M} i_k\ 2^{k-K}$$

où $K = \min(\ k : i_k = 1, \quad 1 \leqslant k \leqslant M\ )$
L'ensemble des indices de sommation de (3.16) est

$$\{m : \ l2^m = i;\ 0 \leqslant l \leqslant 2^{M-m} - 1\ \}$$

qui est identique à

$$\{m : 0 \leqslant m \leqslant \ K - 1\ \}$$

Donc, dans le cas où i n'est pas nul, (3.16) devient

$$\mu_e\ \ [\ x_0\ (i)\ \ ] = \sum_{m=0}^{K-1} \ -\ (1+j)\ \ 2^{-t-1+m}$$

$$= -2^{-t-1}\ (1+j)\ (2^K - 1)$$

Comme $x_0\ (i) = x(n)$ avec

$$n = \sum_{k=1}^{M} n_k\ 2^{k-1} \quad = \quad \sum_{k=1}^{M} i_{M-k+1}\ 2^{k-1}$$

On obtient $K = \min\{ k : i_k = 1\} = M + 1 - \max\{ k : n_k = 1\}$

La relation (3.16) s'écrit finalement :

$$\mu_e [x(n)] = -2^{-t-1} (1+j) (2^{g(n)} - 1)$$

où $g(n) = \begin{cases} M \\ M+1 - \max\{ k : n_k = 1\} & \text{autrement} \end{cases}$

On en conclut que :

$$E [e(p)] = 2^{-t}/2 \cdot (1+j) (TFD \{2^{g(n)}-1\}_{n=0,\ldots,N-1})_p$$

On établit par (3.15) :

$$(E \left[|e(p)|^2 \right])_{AD} = |E [e(p)]|^2 + 2^{-2t}/2 . N(N-1) + \begin{cases} 0 & p = 0, N/4, N/2, 3N/4 \\ \alpha . 2^{-2t}/12 . N \left[2N - 2^{s(p)} \right] & \text{autrement} \end{cases}$$

Par la relation de Parseval (1.2) appliquée à (3.17) on obtient :

$$\sum_{p=0}^{N-1} \left| E [e(p)] \right|^2 = 2^{-2t}/2 \; N. \sum_{n=0}^{N-1} |2^{g(n)} - 1|^2$$

Puisque $\max \{m : n_m = 1\} = k$ équivaut à $2^{k-1} \leq n \leq 2^k - 1$
et que $1 \leq n \leq N-1$ équivaut à $1 \leq k \leq M$ on a

$$\sum_{n=0}^{N-1} | 2^{g(n)} - 1|^2 = \sum_{k=1}^{M} (2^k - 2^{k-1}) (2^{M+1-k} - 1)^2 + (2^M - 1)^2$$

$$= (N-1) 3N - 2MN$$

Par conséquent :

$$\sum_{P=0}^{N-1} | E [e(p)] |^2 = 2^{-2t}/2 . \quad N \quad [3(N-1)N - 2MN]$$

(3.18)

D'autre part, il est évident que

$$\sum_{P=0}^{N-1} (2^{-2t}/2)\, N(N-1) = N^2\,(N-1)\,2^{-2t}/2 \qquad (3.19)$$

Par un raisonnement semblable à celui qui a servi à développer l'expression (3.9) on obtient :

$$\sum_{p=0}^{N-1} 2^{s(p)} = 4N\,(M-2)$$

avec $p \neq 0, N/4, N/2, 3N/4$

Pour $p \neq 0, N/4, N/2, 3N/4$ on trouve

$$\sum_{p=0}^{N-1} \alpha.(2^{-2t}/12)\; N \mid 2N - 2^{s(p)} \mid$$

$$= \alpha\; (2^{-2t}/12)\, N^2\,(2N-4M) \qquad (3.20)$$

Par (3.18), (3.19) et (3.20)

$$\sum_{p=0}^{N-1} (E\left[|e(p)|^2\right])_{AD} = 2^{-2t}\; \left[(2+\alpha/6)N^3 - 2 + M + (\alpha/3)M N^2\right]$$

(3.21)

3.1.3.4 TRONCATURE - DEPLACEMENT A CHAQUE ETAPE

Les résultats sont dérivés des paragraphes 3132 et 3133. Pour tenir compte des effets des déplacements il faut multiplier la relation (3.12) par 2^m. En y ajoutant la moyenne de l'erreur due aux déplacements indiquée en (3.17), on obtient :

$$E\,[\,e(p)\,] = -2^{-t}/2\,(1+j).\,(TFD\{\,2^{g(n)}-1+\alpha\sum_{m=3}^{M} n_{M-m+1} \cdot \left[\,1+W_m^{f(n,m)}\,\right)/(1-W_m^{f(n,m)}) - j(-1)^{n_{M-m+2}}\,|\,\}_{n=0,\ldots,N-1}\}_p \qquad (3.22)$$

La variance des erreurs pour la troncature est donnée par (3.15). Par conséquent :

$$(E[|e(p)|^2])_{TD} = |E[\,e(p)]|^2 + 2^{-2t}\,N.(N-1) \cdot \begin{cases} 0 & p = 0,\ N/4,\ N/2,\ 3N/4 \\ \alpha\,(2^{-2t}/12)N(2N-2^{s(p)} & \text{autrement} \end{cases}$$

où $E\,[e\,(p)\,]$ est détaillé par (3.22)

En utilisant (3.19), (3.20) et la relation de Parseval on trouve

$$\sum_{p=0}^{N-1} (E[|\,e(p)|^2])_{TD} = 2^{-2t}.[\,N^3(\alpha/6+1/2) - N^2(M\alpha/3+1/2 + (2^{-2t}/2)N \sum |2^{g(n)} - 1 + \alpha \sum_{m=3}^{M} n_{M-m+1} \cdot \left[(1+W_m^{f(n,m)})/(1-W_m^{f(n,m)}) - j\,(-1)^{n_{M-m+2}}\right]|^2 \qquad (3.23)$$

3.1.3.5 REMARQUES

Comme on pouvait s'y attendre, les relations (3.6), (3.(3.21),(3.14), (3.23) montrent que le total du carré des erreurs est supérieur dans l'hypothèse d'un déplacement à chaque étape. Toutefois, cette méthode est plus avantageuse car elle garde le signal plus élevé pendant les opérations et diminue de cette façon le rapport :

$$\sum_{p=0}^{N-1} E\left[|e(p)|^2\right] / \sum_{p=0}^{N-1} |y(p)|^2$$

appelé rapport du bruit au signal. En effet, sans déplacement, la condition imposée est $|x(n)| < 1/N \quad n = 0,1,\ldots,N-1$.
Le rapport minimal du bruit au signal est alors de l'ordre de N^2.
Dans l'hypothèse d'un déplacement à chaque étape la condition imposée est $|x(n)| < 1$ et le rapport minimal du bruit au signal est de l'ordre de N.
Dans les analyses précédentes de l'algorithme de Cooley et Tukey, il n'est pas tenu compte de toutes les multiplications effectuées sans erreur. Weinstein [18] et Welch [20] se sont limités au cas de l'arrondi. Les estimations du total des carrés des erreurs qu'ils ont établies sont évidemment supérieures au résultat établi en (3.6) : en supposant que toutes les multiplications produisent des erreurs, Weinstein obtient $\alpha.2^{-2t} N^2/12$.
D'autre part, en remarquant que les trois premières étapes sont effectuées sans erreur, Welch aboutit à $\alpha .2^{-2t} N^2/64$
en ne retenant que les termes de l'ordre de N^2.

3 . 2 RESULTATS DE L'ANALYSE DE L'ALGORITHME DE SANDE ET TUKEY

3.2.1 MODELE STATISTIQUE

Le calcul élémentaire de l'algorithme de Sande et Tukey est un "papillon" de la forme (1.14). Malgré les différences avec l'algorithme de Cooley et Tukey, les hypothèses 1) à 4) des pages 59 et 60 seront adoptées. En conservant les notations du paragraphe 3.1.1, les schémas modifiés des "papillons" sont représentés à la figure 10a s'il n'y a aucun déplacement et à la figure 10b dans l'hypothèse d'un déplacement à chaque étape.

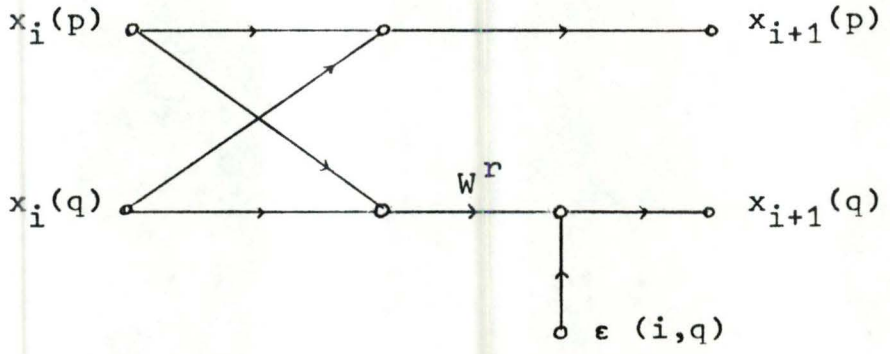

Fig. 10a

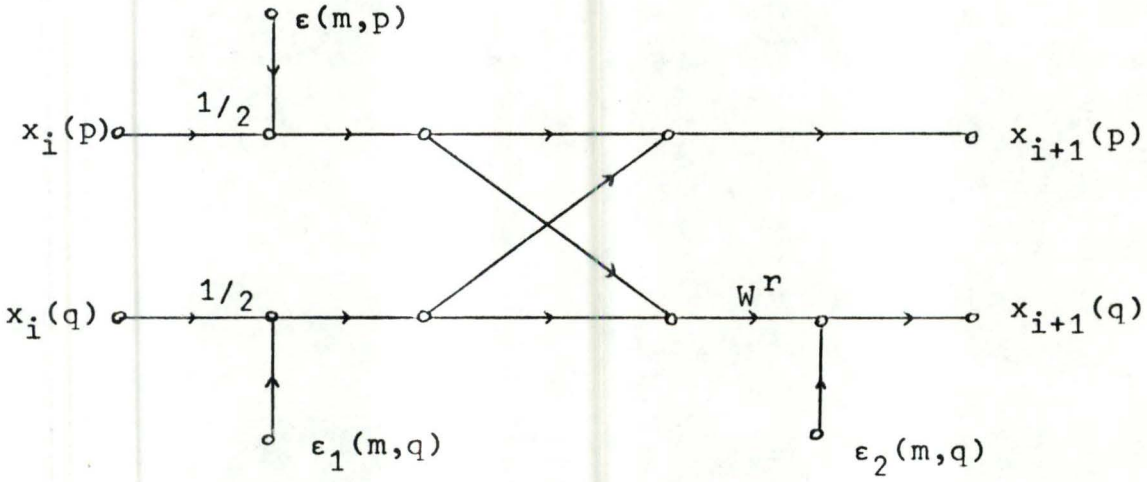

Fig. 10b

3.2.2 PROPRIETES DE L'ALGORITHME DE SANDE ET TUKEY

Les relations (1.13) qui définissent l'algorithme peuvent être mises sous la forme suivante :

$$x_m\,(p_1,\ldots,p_m,\ n_{M-m},\ldots,n_1)$$

$$= \left[\sum_{n_{M-m+1}=0}^{1} x_{m-1}\,(p_1,\ldots,p_{m-1},\ n_{M-m+1},\ldots,n_1) \right.$$

$$\left. .\exp\,(-j2\pi\ n_{M-m+1}\,p_m/2)\right]$$

$$.\ \exp\left[-j2\pi p_m\ 2^{m-1}\ (n_{M-m}.\ 2^{M-m-1} + \ldots + n_1)/N\right]$$

$$x_0(n_M,\ldots,n_1) = x(n) \qquad (3.24)$$

où n et p sont définis comme pour (3.5)
Les relations (1.25) définissent le s-ème élément du l-ème bloc de l'étape m. Il suffit de poser r = 2.

Les propositions 3.1, 3.2 et 3.3 demeurent valables.
Seule la proposition 3.4 est modifiée et devient la proposition suivante.

Proposition 3.5

Il existe une relation de transformée de Fourier discrète entre les éléments d'un bloc à l'étape m et les éléments correspondants du tableau final.

Cette proposition est un cas particulier de la proposition 1.2.
Les éléments du bloc $(x_m\ (12^{M-m} + s))$ correspondent aux éléments $(x_M\ (12^{M-m} + s))$ où $0 \leq s \leq 2^{M-m} - 1$.

Corollaire

Toutes les erreurs qui affectent les éléments d'un bloc de l'étape m se propagent à tous les résultats qui leur correspondent.
La démonstration découle des propositions 3.2 et 3.5.

3.2.3 ETUDE DES ERREURS NUMERIQUES

Dans l'algorithme de Sande et Tukey, le passage de l'étape m-1 à l'étape m décompose un bloc de longueur 2^{M-m+1} en deux blocs de longueur 2^{M-m}. Le calcul est donné par :

$$x_m(r) = x_{m-1}(r) + x_{m-1}(s)$$

$$x_m(s) = \left(x_{m-1}(r) - x_{m-1}(s)\right) U$$

avec, par (3.24),

$$r = \sum_{i=1}^{m} p_i \, 2^{M-i} + \sum_{i=1}^{M-m} n_i \, 2^{i-1} \qquad \text{avec } p_m = 0$$

$$s = r + 2^{M-m}$$

$$U = \exp\left[- j2\pi \, 2^{m-1} \sum_{i=1}^{M-m} n_i \, 2^{i-1} / N\right] \qquad (3.25)$$

où m varie de 1 à M.
On remarque que la première relation d'un "papillon" est calculée sans erreur numérique. Elle est caractérisée par $p_m = 0$.

3.2.3.1 ARRONDI-SANS DEPLACEMENT

On déduit de (2.11) que la moyenne des erreurs sur les résultats est nulle.
A l'étape m, 2^m blocs sont calculés, dont la moitié avec des erreurs d'arrondi. Pour chacun d'eux, on déduit de l'expression (3.25) des coefficients U que deux éléments sont obtenus sans erreur. Le nombre de multiplications qui produisent des erreurs à l'étape m est donc égal à $N/2 - 2.\,2^{m-1}$. Par conséquent :

$$\sum_{p=0}^{N-1} \left(E\left[|e(p)|^2\right]\right)_A = \alpha . 2^{-2t}/12 . \sum_{m=1}^{M-2} (N/2 - 2.2^{m-1}) 2^{M-m}$$

car les étapes M-1 et M ne produisent pas d'erreur.
On trouve finalement

$$\sum_{p=0}^{N-1} (E[|\, e(p)\,|^{2}])_{A} = \alpha.2^{-2t}/12.\ N(N/2-M) \qquad (3.26)$$

En examinant la relation (3.24) on constate que chaque bloc caractérisé par $p_m = 0$, est calculé sans erreur. En tenant compte des deux multiplications sans erreur de chaque bloc on a par le corollaire :

$$(E[|e(p)|^{2}])_{A} = \alpha .2^{-2t}/12 \sum_{n=1}^{M-2} p_m\ (2^{M-m}-2) \qquad (3.27)$$

La relation (3.26) peut être retrouvée facilement à partir de(3.27). Il suffit de constater que pour m fixé on a :

$$\sum_{p=0}^{N-1} p_m = N/2 \qquad (3.28)$$

Par conséquent :

$$\sum_{m=1}^{M-2}\sum_{p=0}^{N-1} p_m = (M-2)\ N/2 \text{ et } \sum_{m-1}^{M-2}\sum_{p=0}^{N-1} p_m\ 2^{-m} = N/2 - 2$$

La relation (3.26) en découle immédiatement.

3.2.3.2. TRONCATURE - SANS DEPLACEMENT

Pour évaluer la moyenne de l'erreur de troncature, on remarque que les blocs caractérisés par $p_m = 1$ sont calculés avec une erreur de moyenne égale à $-\alpha/2\ (1+j)\ 2^{-b-1}$.
Les éléments zéro et 2^{M-m-1} de chacun de ces blocs sont obtenus sans erreur.

Par la proposition 3.5, les erreurs commises sur un bloc peuvent être reportées en erreurs équivalentes sur le tableau final.
Il suffit de prendre la transformée de Fourier des 2^{M-m} échantillons suivants :

$$E_i \begin{cases} 0 & i = 0, P/2 \\ A & \text{autrement} \end{cases}$$

où $A = -\alpha/2 \cdot (1 + j)\ 2^{-t-1}$ et $P = 2^{M-m}$

Les erreurs équivalentes pour ce bloc sont :

$$e_{k^\dagger} = \sum_{i=0}^{p-1} E_i\ W^{ik^\dagger}$$

où $W = \exp(-j2\pi/P)$ et $0 \leqslant k^\dagger \leqslant P-1$

Si $k^\dagger$ est nul il est évident que $e_0 = (P-2)A$

Si $k^\dagger$ n'est pas nul on obtient :

$$e_{k^\dagger} = A \sum_{i=0}^{P-1} W^{ik^\dagger} - A\ (W^0 + W^{k^\dagger P/2})$$

$$= -A\ (1 + (-1)^{k^\dagger})$$

L'erreur équivalente $e_{k^\dagger}$ est située à la k-ème place du tableau final, $k^\dagger$ étant le nombre binaire inversé de k. L'erreur équivalente sur $x_M\ (l2^{M-m} + k)$ est donc

$$-\alpha/2.(1+j)\ 2^{-t-1}\ (2^{M-m} - 2) \qquad \text{si } k^\dagger = 0$$

$$\alpha/2.(1+j)\ 2^{-t-1}\ (1 + (-1)^{k^\dagger}) \qquad \text{si } k^\dagger \neq 0$$

Comme $x_m\ (l2^{M-m} + k) = y\ (p_1, \ldots, p_M)$, les indices k et $k^\dagger$ sont représentés respectivement par :

$$\sum_{i=1}^{M-m} p_{M-i+1}\ 2^{i-1} \qquad \text{et} \qquad \sum_{i=1}^{M-m} p_{m+i}\ 2^{i-1}$$

On remarque que $(-1)^{k^\dagger} = (-1)^{P_{m+1}}$. D'autre part, en désignant $x_m\,(12^{M-m} + k)$ par $x_m\,(q)$, $k^\dagger = 0 = k$ équivaut à $\min\{i : q_i = 1\} = M - m + 1$. Comme q est le nombre binaire inversé de p, $k^\dagger = 0$ équivant à $\max\{ i : p_i = 1 \} = m$.

L'erreur commise sur y (p) est la somme des contributions provenant des étapes m = 1, ..., M-2. Sa moyenne est égale à :

$$(E[\,e(p)])_T = \begin{cases} 0 & \text{si } p = 0 \\ -\alpha/2.2^{-t}/2(1+j)\sum_{m=1}^{M-2} p_m. & \\ \left[N2^{-m}\,\delta(\,m,t(p)) - 1 - (-1)^{P_{M+1}}\right| & \text{autrement} \end{cases} \tag{3.29}$$

où $\delta(.,.)$ est le symbole de Kronecker et t(p) est donné par

$$t(p) = \begin{cases} 0 & \text{si } p = 0 \\ \max\{ i : p_i = 1 \} & \text{autrement} \end{cases}$$

On en tire immédiatement :

$$(E[|e(p)|^2])_T = (E[|e(p)|^2])_A + |(E[\,e(p)])_T|^2$$

Il suffit de faire la somme de ces relations pour p = 0,..., N-1 et on obtient le total du carré des erreurs.

3.2.3.3 ARRONDI - DEPLACEMENT A CHAQUE ETAPE

Puisque le tableau x_m subit un déplacement avant que le tableau x_{m+1} soit calculé, chaque élément $x_m(i)$, $i = 0,\ldots,N-1$ reçoit une erreur de moyenne égale à $-(1+j)2^{-b-1+m}$. En appliquant la proposition 3.5, chaque élément $A_M(l2^{M-m})$, $l = 0,\ldots,2^m-1$ reçoit une erreur équivalente égale à $-(1+j)\ 2^{-t-1+m}.\ 2^{M-m}$.
En désignant par $\mu_e\ [x_M(r)\ ]$ le total des erreurs équivalentes qui atteignent l'élément $x_M(r)$, on obtient :

$$\mu_e\ [\ x_M(r)\ ] = \sum_{\{m:l2^{M-m}=r\}} -(1+j)\ 2^{-t-1}.\ 2^M$$

Pour que r soit égal à $l2^{M-m}$, il faut qu'il soit divisible par 2^m.
Si $\min\{\ i : r_i = 1\ \} = K$, r est au plus divisible par 2^{K-1} et $x_m(r)$ recevra K-1 contributions d'erreurs équivalentes.
Comme r est le nombre binaire inversé de p on trouve :

$$K = \min\{i : r_i = 1\} = M - \max\{i : p_i = 1\} + 1$$

Par conséquent

$$(E[\ e(p)]\ )_{AD} = -(1+j)\ 2^{-t}/2\ .\ N\ (M-t(p)) \qquad (3.30)$$

où t(p) a été défini pour établir la relation (3.29)
De (3.27), et de la proposition 3.3, on déduit :

$$(E[|e(p)|^2])_{AD} = |\ E\ [e(p)]\ |^2 + 2^{-2t}/2.\ N(N-1) + \alpha 2^{-2t}/2 \sum_{m=1}^{M-2} p_m.\ (2^{M-m}-2)\ 2^{2m} \qquad (3.31)$$

où le facteur 2^{2m} est dû aux déplacements.

. E [e(p)] est donné par la relation (3.30)

Il reste à effectuer la somme des relations (3.31) pour les indices $p = 0,\ldots, N-1$.

D'abord, pour évaluer la somme des erreurs $|E[e(p)]|^2$ il est nécessaire de connaître

$$\sum_{p=0}^{N-1} t(p) = \sum_{l=1}^{M} \sum_{p \varepsilon s} 1$$

où $p \varepsilon s$ est équivalent à $\max \{ i : p_i = 1 \} = l$

Donc
$$\sum_{p=0}^{N-1} t(p) = \sum_{l=1}^{M} 2^{l-1} \cdot l$$

$$= NM - N+1$$

ainsi que

$$\sum_{p=0}^{N-1} (t(p))^2 = \sum_{l=1}^{M} 2^{l-1} \cdot l^2$$

$$= NM^2 - 2MN + 3N - 3$$

Par conséquent :

$$\sum_{p=0}^{N-1} |E[e(p)]|^2 = 2^{-2t}/2 \cdot N^2 (3N-2M-3) \qquad (3.32)$$

D'autre part, en utilisant (3.28) on obtient

$$\sum_{p=0}^{N-1} \sum_{m=1}^{M-2} p_m (2^{M-m} - 2) 2^{2m} = N^3/6 - N^2 - 4N/3 \qquad (3.33)$$

Par (3.19), (3.32) et (3.33) on a que :

$$\sum_{p=0}^{N-1} (E[|e(p)|^2])_{AD} = 2^{-2t} [N^3 (2 + \alpha/72) - N^2(2+M+ \alpha/12) + N. \alpha/9]$$

(3.34)

3.2.3.4 TRONCATURE - DEPLACEMENT A CHAQUE ETAPE

Les résultats s'obtiennent à partir des paragraphes 3.2.3.2.et 3.2.3.3. La moyenne de l'erreur est la somme de l'erreur de déplacement (3.30) et de l'erreur de troncature, donnée par la relation (3.29) multipliée par un facteur 2^m. Donc on a que :

$$E[\, e(p)\,] = -2^{-t}/2\,[(1+j)\,N\,(M-t(p))$$

$$+ \alpha/2 \sum_{m=1}^{M-2} p_m [\, N\,\delta(m,t(p)) - (1+(-1)^{P_{m+1}})2^m])$$

La variance de l'erreur s'obtient par la somme de (3.27) et de la variance de l'erreur de déplacement. Par conséquent :

$$(E[|e(p)|^2])_{TD} = |E\,[e(p)]|^2 + 2^{-2t}/2.\,N\,(N-1)$$

$$+\alpha.2^{-2t}/12. \sum_{m=1}^{M-2} p_m\,(2^{M-m} - 2)\,2^{2m}$$

A l'aide de (3.19) et (3.33) on déduit :

$$\sum_{p=0}^{N-1} (E[|e(p)|^2])_{TD} = \sum_{p=0}^{N-1} |\,E\,[e(p)]|^2$$

$$+\; 2^{-2t}/2[\, N^3(1+\alpha/36) - N^2(1+\alpha/6)$$

$$+\; 2\alpha/9N] \quad (3.35)$$

3 . 3 ALGORITHMES POUR DES BASES COMPOSITES

3.3.1 ALGORITHME DE COOLEY ET TUKEY

Dans l'hypothèse où $N = c^{M_1} \cdot d^{M-M_1}$, l'algorithme de Cooley et Tykey s'établit en combinant les démarches développées dans les paragraphes 1.3 et 1.4. On pose

$$p = \left[\sum_{i=1}^{M-M_1} p_{M_1+i} \; d^{i-1} \right] c^{M_1} + \sum_{i=1}^{M_1} p_i \; c^{i-1} \qquad (3.40)$$

$$n = \left[\sum_{i=1}^{M_1} n_{M-M_1} + i \cdot c^{i-1} \right] d^{M-M_1} + \sum_{i=1}^{M-M_1} n_i \; d^{i-1} \qquad (3.41)$$

où . $n_M, \ldots, n_{M-M_1+1}$, $= 0,1,\ldots,c-1$

. $n_{M-M_1}, \ldots, n_1$ $= 0,1,\ldots,d-1$

. $p_M, \ldots, p_{M_1+1}$ $= 0,1,\ldots,d-1$

. $p_{M_1}, \ldots, p_1$ $= 0,1,\ldots,c-1$

On obtient l'algorithme suivant :

$$x_m \; (lc^m + p_m \; c^{m-1} + q) = \sum_{i=0}^{c-1} x_{m-1} \; (lc^m + ic^{m-1} + q) \cdot \exp \left[- j2\pi \, i(p_m c^{m-1} + q)/ c^m \right]$$

$$m = 1, \ldots, M_1$$

où $l = 0, \ldots, (N/c^m) - 1$, $p_m = 0, \ldots, c-1$ et $q = 0, \ldots c^{m-1} - 1$

$$x_m\,(lc^{M_1}\,d^{M-M_1} + p_m\,c^{M_1}\,d^{M-1-M_1} + q)$$

$$= \sum_{i=0}^{d-1} x_{m-1}\,(lc^{M_1}\,d^{m-M_1} + ic^{M_1}\,d^{m-1-M_1} + q)$$

$$\cdot \exp\left[-j2\pi i\,(p_m\,c^{M_1}\,d^{m-1-M_1} + q)/(c^{M_1}\,d^{M-M_1})\right]$$

$$m = M_1 + 1,\ldots,M \qquad (3.43)$$

où $l = 0,\ldots, d^{M-m} - 1$, $p_m = 0,\ldots,d-1$ et $q=0,\ldots,c^{M_1}\,d^{m-1-M_1} - 1$

Le calcul de chaque élément $x_m(.)$ nécessite (c-1) ou (d-1) multiplications complexes. Comme c et d sont arbitraires, on suppose que toutes les multiplications produisent des erreurs.
Toutefois, dans le cas où p_m et q sont nuls aucune multiplication n'est requise. Pour la relation (3.42), N/c^m éléments sont calculés sans erreur; pour la relation (3.43) il y en a d^{M-m}.
En utilisant l'argument qui a servi à établir (3.6) on a dans le cas de l'arrondi :

$$\sum_{p=0}^{N-1} E\left[|e(p)|^2\right] = \alpha \cdot 2^{-2t}/12\left[\left[\sum_{m-1}^{M_1} c^{M_1-M}\,(N-N/c^m)\right]d^{M-M_1}\right.$$

$$\left.\cdot(c-1)+\left[\sum_{m=M_1+1}^{M} d^{M-m}\,(N-d^{M-m})\right](d-1)\right] \qquad (3.44)$$

où les facteurs (c-1) et (d-1) tiennent compte du nombre de multiplications, les facteurs c^{M_1-m}, d^{M-m} sont des facteurs de propagation et d^{M-M_1} est un facteur de propagation des erreurs de (3.42) à travers (3.43). En effectuant les différentes sommes de (3.44), on trouve :

$$\sum_{p=0}^{N-1} E\left[\,|e(p)|^2\,\right] = \alpha \cdot 2^{-2t}/12\left[N(N-1) - (N^2/c^{2M_1})\right.$$

$$\left.\cdot\,(c^{2M_1} - 1)/(c+1) - (d^{2(M-M_1)}-1)(d+1)\right]$$

Par le raisonnement qui a conduit à la relation (3.7) on obtient

$$E\left[|e(p)|^2\right] = \alpha \cdot 2^{-2t}/12\left[(c-1)\sum_{m=q(p)}^{M_1} N/c^m\right.$$

$$\left.+(d-1)\sum_{m=r(p)}^{M} d^M/d^m\right]$$

où $-q(p) = \begin{cases} M+1 & \text{si } p = 0 \\ \min \{i : p_i \neq 0\} & \text{autrement} \end{cases}$

- $r(p) = \max \{M_1 + 1, q(p)\}$
- $\sum_{m=i}^{j}$ est nulle si $j < i$

Les termes des deux sommes sont les facteurs de propagation des erreurs.

3.3.2 ALGORITHME DE SANDE ET TUKEY

Les indices p et n étant décomposés suivant (3.40) et (3.41), l'algorithme de Sande et Tukey est défini par :

$$x_m (l.N/c^{m-1} + p_m . N/c^m + q) = \left[\sum_{i=0}^{c-1} x_{m-1} (l. N/c^{m-1} + i. N/c^m + q) . \exp(-j2\pi i p_m/c) \right] . \exp(-j2\pi p_m q c^{m-1}/N)$$

$$m = 1, \ldots, M_1 \qquad (3.45)$$

où $l = 0, \ldots, c^{m-1}-1$, $p_m = 0,\ldots,c-1$ et $q = 0,\ldots,N/c^m - 1$

$$x_m (ld^{M-m+1} + p_m d^{M-m} + q) = \left[\sum_{i=0}^{d-1} x_{m-1} (ld^{M-m+1} + id^{M-m} + q) . \exp(-j2\pi i p_m/d) \right] \exp(-j2\pi p_m q/d^{M-m+1})$$

$$m = M_1 + 1, \ldots, M$$

où $l = 0,\ldots,(N/d^{M-m+1})-1$, $p_m=0,\ldots,d-1$ et $q = 0,\ldots, d^{M-m} - 1$

En reprenant les arguments utilisés dans le paragraphe 3.2.3.1, on obtient dans le cas de l'arrondi :

$$\sum_{p=0}^{N-1} E\left[\,|e(p)|^{2}\right] = \alpha\ 2^{-2t}/12 \sum_{m=1}^{M_1} c^{M_1-m}\, d^{M-M_1} \cdot \left[\frac{c-1}{c}\,(N-c^{m}) + (c-1)\,N\,\frac{c-1}{c}\right] + \sum_{m=M_1+1}^{M} d^{M-m} \cdot \left[\frac{d-1}{d}\,(N-d^{-M_1}\,.\,d^{m}) + (d-1)\,N\,\frac{d-1}{d}\right] \qquad (3.47)$$

où le premier terme de chaque somme est dû à la seconde exponentielle des expressions (3.45) et (3.46) et où le second terme est dû à la première exponentielle.
La relation (3.47) peut être mise sous la forme :

$$\sum_{p=0}^{N-1} E\left[\,|e(p)|^{2}\right] = \alpha\ (2^{-2t}/12).\ N\left[(N-1)-M_1(c-1)/c - (M-M_1)\,(d-1)/d\right]$$

Toujours en suivant la démarche du paragraphe 3.2.3.1, on trouve

$$E\left[|e(p)|^{2}\right] = \alpha\ 2^{-2t}/12 \left(\sum_{m=1}^{M_1} u(p_m) \cdot \left[(N/c^{m}-1) + N/c^{m}\,.\,(c-1)\right] + \sum_{m=M_1+1}^{M} u(p_m)\left[(d^{M-m}-1) + d^{M-m}(d-1)\right]\right)$$

où $u(p_m) = \begin{cases} 0 & \text{si } p_m = 0 \\ 1 & \text{si non} \end{cases}$

Par conséquent :

$$E\left[\,|e(p)|^{2}\right] = \alpha \cdot 2^{-2t/12}\left[\sum_{m=1}^{M_1} u(p_m)\,(cN/c^{m} - 1) + \sum_{m=M_1+1}^{M} u(p_m)\,(d^{M-m+1} - 1)\right]$$

ANALYSE DES ERREURS NUMERIQUES

DANS LES

ALGORITHMES DE LA TRANSFORMEE

DE FOURIER RAPIDE

EN

VIRGULE FLOTTANTE

4 . 1 GENERALITES

L'implémentation sur ordinateur des algorithmes de la Transformée de Fourier Rapide engendre inévitablement plusieurs types d'erreurs :

- des erreurs de représentation numérique des valeurs du tableau d'entrée,
- des erreurs d'estimation des facteurs W^p,
- des erreurs d'addition et de multiplication au cours de l'algorithme lui-même.

On s'intéressera dans ce chapitre uniquement à ces dernières erreurs. On supposera donc pour cette étude que les valeurs d'entrée et les facteurs W^p sont représentés exactement sur l'ordinateur. On suppose également vérifiées les hypothèses générales pour un modèle de l'erreur (voir au paragraphe 2.1).

Tran-Thong et Bede Liu [17] ont établi des expressions pour le rapport bruit à signal, défini par

$$\left(\frac{B}{S}\right) = \frac{\sum_{k=0}^{N-1} E\{ |y'(k) - y(k)|^2 \}}{\sum_{k=0}^{N-1} |y(k)|^2} \qquad (4.1)$$

où $E\{.\}$ est l'espérance mathématique,

$y(k)$ sont les valeurs exactes de sortie de l'algorithme,

$y'(k)$ sont les valeurs calculées de sortie de l'algorithme.

On suppose de plus que l'arithmétique choisie correspond au format de signe et de grandeur.

Les résultats de ce chapitre sont exprimés en termes d'espérances et de variances des erreurs locales, à savoir δ_x, δ_+, Δ_x^2, Δ_+^2 (voir paragraphe 2.3.2).

Ces valeurs dépendent de la base et de l'arithmétique.

Si $b=2$

$$\delta_x = \begin{cases} 0 & \text{en cas d'arrondi} \\ -2^{-t} & \text{en cas de troncature} \end{cases}$$

$$\delta_+ = (1-p_o)\delta_x$$

$$\Delta_x^2 = 2^{-2t}/3$$

$$\Delta_+^2 = \begin{cases} (1-p_o)2^{-2t}/3 & \text{en cas d'arrondi} \\ 4(1-p_o)2^{-2t}/3 - (1-p_o)^2 2^{-2t} & \text{en cas de troncature} \end{cases}$$

(4.2)

Si $b>2$

$$\delta_x = \begin{cases} 0 & \text{en cas d'arrondi} \\ -b^{-t}(b-1)/2\ln b & \text{en cas de troncature} \end{cases}$$

$$\delta_+ = (1-p_o)\delta_x$$

$$\Delta_x^2 = \begin{cases} b^{-2t}(b^2-1)/24\ln b & \text{en cas d'arrondi} \\ b^{-2t}(b^2-1)/6\ln b - b^{-2t}(b-1)^2/(2\ln b)^2 & \text{en cas de troncature} \end{cases}$$

$$\Delta_+^2 = \begin{cases} (1-p_o)b^{-2t}(b^2-1)/24\ln b & \text{en cas d'arrondi} \\ (1-p_o)b^{-2t}(b^2-1)/6\ln b - (1-p_o)^2 b^{-2t}(b-1)^2/(2\ln b)^2 & \text{en cas de troncature} \end{cases}$$

(4.3)

Remarque :

La valeur de p_o dépend de la base de l'arithmétique et des données du problème [8]. Dans ce cas-ci, Tran-Thong et Bede Liu [17] l'estiment à 0.7 pour la famille d'ordinateurs IBM 360/370.

A ce stade, il est intéressant de développer un modèle d'erreurs pour la multiplication et l'addition de nombres complexes.

Soient A , B , C des nombres complexes.

Dans la mesure où seules les espérances mathématiques et les variances sont envisagées, on peut montrer qu'en négligeant les termes de second ordre, on obtient :

$$\begin{aligned} fl(A + B) &= (A + B)(1 + \alpha) \\ fl(AC) &= (AC)(1 + \beta) \end{aligned} \qquad (4.4)$$

avec
$$\begin{aligned} E(\alpha) &= \delta_+ \\ var(\alpha) &= \Delta_+^2 \\ E(\beta) &= \delta_\times + \delta_+ \\ var(\beta) &= \Delta_\times^2 + \Delta_+^2 \end{aligned} \qquad (4.5)$$

où les erreurs relatives α et β sont des nombres complexes et les nombres réels $\delta_\times$, δ_+, $\Delta_\times^2$, Δ_+^2 sont donnés par (4.2) ou (4.3).

4 . 2 ALGORITHME DE COOLEY ET TUKEY

En utilisant la notion de bloc définie au paragraphe 1.3.A, on établit des expressions et des bornes pour le rapport bruit à signal, défini par (4.1), dans le cas de l'algorithme de Cooley et Tukey quand $N = 2^M$ (voir paragraphe 1.2.A).

On suppose, tout d'abord, introduire une erreur en calculant $x_m(p)$ et que, de plus, la valeur réellement calculée est de la forme

$$x'_m(p) = x_m(p) + a(p).\gamma \quad .$$

Cette erreur affecte le résultat final selon les trois propositions suivantes :

Proposition 4.1

Une erreur introduite à l'étape m se propage à 2^{M-m} éléments du résultat. La variance de l'erreur du résultat est égale, en chacun des 2^{M-m} éléments affectés, à la variance de l'erreur à l'étape m.

Démonstration :

La première partie de cette proposition est évidente (voir figure 2 au paragraphe 1.2)

En négligeant les effets de second ordre et en notant que l'erreur n'est multipliée que par des facteurs de module unité, la variance de l'erreur en chacun des 2^{M-m} éléments vaut :

$$\text{var}\{|y'(k)-y(k)|\} = |\{\prod_{i=m+1}^{M} W_i\}\, a(p)|^2 . \text{var}(\gamma)$$

$$= |a(p)|^2 . \text{var}(\gamma)$$

$$= \text{var}\{|x'_m(p)-x_m(p)|\} ,$$

où les W_i sont des puissances entières de $W = \exp(-j2\pi/N)$.

Proposition 4.2

Si à l'étape m, tous les éléments du $l^{ième}$ bloc ont une erreur relative γ_q :

$$x'_m(l2^m+q) = x_m(l2^m+q).(1+\gamma_q)$$

où $q = 0,1,\ldots,2^m-1$,

et que la variance de γ_q est indépendante de q, c'est-à-dire

$$\text{var}(\gamma_q) = d^2,$$

alors la contribution de toutes les erreurs dans le $l^{ième}$ bloc à l'étape m à la variance normalisée de l'erreur à la sortie vaut :

$$\frac{\sum_{k=0}^{N-1} \text{var}\{|y'(k)-y(k)|\}}{\sum_{k=0}^{N-1} |y(k)|^2} = \frac{d^2}{\sum_{n=0}^{N-1} |x(n)|^2} \sum_{q=0}^{2^m-1} |x_0(l2^m+q)|^2$$

Démonstration :

Par la proposition 4.1, la contribution vaut :

$$\frac{d^2}{\sum_{k=0}^{N-1} |v(k)|^2} \; 2^{M-m} \sum_{q=0}^{2^m-1} |x_m(12^m+q)|^2 \; .$$

Par la proposition 1.1 et la relation de Parseval (1.2), on obtient :

$$\frac{\sum_{k=0}^{N-1} \mathrm{var}\{|v'(k)-v(k)|\}}{\sum_{k=0}^{N-1} |v(k)|^2} = \frac{d^2}{N \sum_{n=0}^{N-1} |x(n)|^2} \; 2^{M-m}$$

$$\times \left[2^m \sum_{q=0}^{2^m-1} |x_0(12^m+q)|^2 \right]$$

$$= \frac{d^2}{\sum_{n=0}^{N-1} |x(n)|^2} \sum_{q=0}^{2^m-1} |x_0(12^m+q)|^2 .$$

Proposition 4.3

Si à l'étape m, les éléments du $l^{\text{ième}}$ bloc sont tels que

$$x'_m(12^m+q) = x_m(12^m+q) + x_{m-1}(12^m+2^{m-1}+q).\gamma_q$$

$$x'_m(12^m+2^{m-1}+q) = x_m(12^m+2^{m-1}+q) + x_{m-1}(12^m+2^{m-1}+q).\gamma_q$$

où $q = 0,1,\ldots,2^{m-1}-1$,

et que la variance de γ_q est indépendante de q, c'est-à-dire

$$\mathrm{var}(\gamma_q) = d^2,$$

alors la contribution de toutes les erreurs dans le $l^{\text{ième}}$ bloc

à l'étape m à la variance normalisée de l'erreur à la sortie vaut :

$$\frac{\sum_{k=0}^{N-1} \mathrm{var}\{|v'(k)-v(k)|\}}{\sum_{k=0}^{N-1} |v(k)|^2} = \frac{n_{M-m+1}\, d^2}{\sum_{n=0}^{N-1} |x(n)|^2} \sum_{q=0}^{2^{m-1}-1} |x_0(l'2^{m-1}+q)|^2$$

où $\quad l' = \sum_{i=1}^{M-m+1} n_i 2^{M-i-m+1}$

Démonstration :

On a : $\quad \mathrm{var}\{|x'_m(l2^m+q)-x_m(l2^m+q)|\}$

$$= |x_{m-1}(l2^m+2^{m-1}+q)|^2.d^2$$

$$= n_{M-m+1}\cdot|x_{m-1}(l'2^{m-1}+q)|^2.d^2$$

et $\quad \mathrm{var}\{|x'_m(l2^m+2^{m-1}+q)-x_m(l2^m+2^{m-1}+q)|\}$

$$= |x_{m-1}(l2^m+2^{m-1}+q)|^2.d^2$$

$$= n_{M-m+1}\cdot|x_{m-1}(l'2^{m-1}+q)|^2.d^2.$$

Donc,

$$\frac{\sum_{k=0}^{N-1} \mathrm{var}\{|v'(k)-v(k)|\}}{\sum_{k=0}^{N-1} |v(k)|^2}$$

$$= \frac{n_{M-m+1}\, d^2}{N \sum_{n=0}^{N-1} |x(n)|^2} 2^{M-m}.2. \sum_{q=0}^{2^{m-1}-1} |x_{m-1}(l'2^{m-1}+q)|^2$$

$$= \frac{n_{M-m+1}\, d^2}{\sum_{n=0}^{N-1} |x(n)|^2} \sum_{q=0}^{2^{m-1}-1} |x_0(l'2^{m-1}+q)|^2 \ .$$

ANALYSE DES OPERATIONS ARITHMETIQUES DANS L'ALGORITHME

Le calcul du $l^{ième}$ bloc à l'étape m par les "papillons" définis en (1.12) sont :

$$x_m(l2^m+q') = x_{m-1}(l2^m+q') + x_{m-1}(l2^m+2^{m-1}+q').\exp(-j2\pi q'/2^m)$$

$$x_m(l2^m+2^{m-1}+q') = x_{m-1}(l2^m+q') - x_{m-1}(l2^m+2^{m-1}+q') \times \exp(-j2\pi q'/2^m)$$

$$q' = \sum_{i=1}^{m-1} k_i 2^{i-1} \ , \quad k_i = 0 \text{ ou } 1 \qquad (4.6)$$

Les expressions (4.6) sont illustrées à la figure 11. Dès que l'on introduit les erreurs selon (4.4), il faut considérer la figure 12. Ici, β correspond à une erreur complexe de multiplication et α_1 , α_2 à des erreurs complexes d'addition.

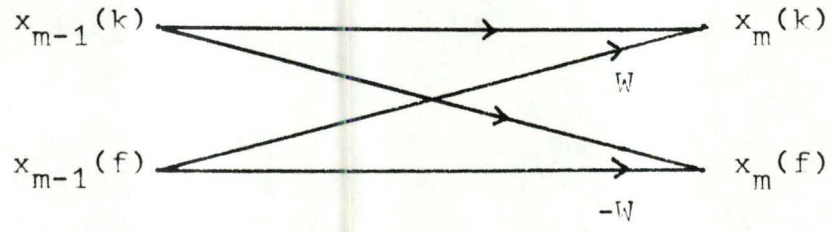

Figure 11. "Papillon" de la Transformée de Fourier Rapide (DIT) où $k = l2^m+q'$

$f = k+2^{m-1}$

$W = \exp(-j2\pi q'/2^m)$

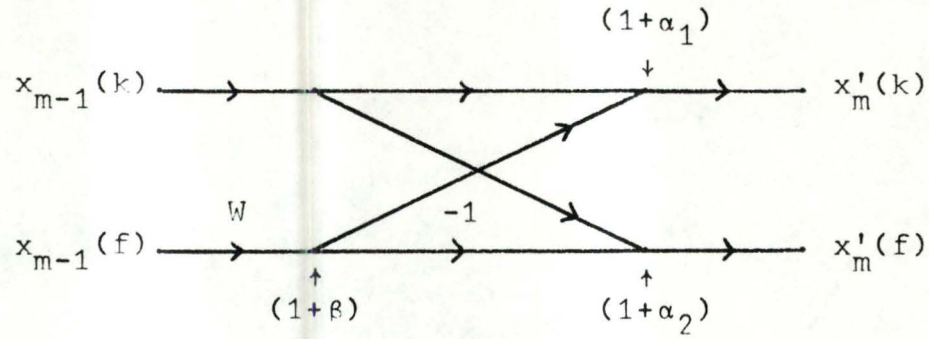

Figure 12. "Papillon de la DIT avec modèle d'erreur.

En fait, si on inclut l'erreur aux expressions (4.6) et que l'on prend les notations de la figure 11, on obtient :

$$x'_m(k) - x_m(k) = x_m(k).\alpha_1 + x_{m-1}(f).W.(\beta+\beta\alpha_1)$$

$$x'_m(f) - x_m(f) = x_m(k).\alpha_2 - x_{m-1}(f).W.(\beta+\beta\alpha_2)$$

$$q' = \sum_{i=1}^{m-1} k_i 2^{i-1} \quad , \; k_i = 0 \text{ ou } 1 \qquad (4.7)$$

Pour calculer l'erreur totale à la sortie de l'algorithme, on somme toutes les différentes contributions. D'autre part, on traite les contributions des erreurs d'addition et de multiplication séparément vu que leur interaction ne se marque qu'au second ordre qui est négligé ici.

CAS DE L'ARRONDI

La contribution des erreurs d'addition à l'étape m au rapport bruit à signal vaut, en accord avec la proposition 4.2 et les expressions (4.5) et (4.7) :

$$\frac{\Delta_+^2}{\sum_{n=0}^{N-1} |x(n)|^2} \sum_{n=0}^{N-1} |x(n)|^2 = \Delta_+^2 .$$

La contribution des erreurs de multiplication à l'étape m au rapport bruit à signal vaut, en accord avec la proposition (4.3) et les expressions (4.5) et (4.7) :

$$\frac{\Delta_+^2 + \Delta_\times^2}{\sum_{n=0}^{N-1} |x(n)|^2} \sum_{n=0}^{N-1} n_{M-m+1} \cdot |x(n)|^2 \qquad \text{si } m \neq 1,2$$

$$0 \qquad \text{si } m = 1,2 .$$

En effet, pour $m = 1$ ou 2, les facteurs multiplicatifs valent tous ± 1 ou $\pm i$, ce qui ne cause aucune erreur.

Toutes ces contributions sont sommées sur m pour donner le rapport bruit à signal :

$$\left(\frac{B}{S}\right) = \frac{1}{\sum_{n=0}^{N-1} |x(n)|^2} \sum_{n=0}^{N-1} |x(n)|^2 \left[\sum_{m=1}^{M} \Delta_+^2 + \sum_{m=3}^{M} n_{M-m+1} \cdot \{\Delta_\times^2 + \Delta_+^2\} \right],$$

ce qui peut encore s'écrire, en réarrangeant les termes :

$$\left(\frac{B}{S}\right)_A = M\Delta_+^2 + \frac{\Delta_\times^2 + \Delta_+^2}{\sum_n |x(n)|^2} \sum_{n=1}^{N-1} f(n) . |x(n)|^2 \qquad (4.8)$$

où l'indice A signifie arrondi,

$$\sum_n = \sum_{n=0}^{N-1} ,$$

$$f(n) = \sum_{i=1}^{M-2} n_i . \qquad (4.9)$$

CAS DE LA TRONCATURE

La somme des variances normalisées est bien sûr donnée par (4.8), mais pour obtenir le rapport bruit à signal à la sortie de l'algorithme de la Transformée de Fourier Rapide, il faut encore y ajouter la somme normalisée des carrés des espérances de l'erreur :

$$\frac{\sum_{k=0}^{N-1} \{E\{|y'(k)-y(k)|\}\}^2}{\sum_{k=0}^{N-1} |y(k)|^2} \quad .$$

Pour déterminer quelles entrées produiraient les mêmes valeurs à la sortie si toutes les opérations arithmétiques se faisaient sans erreur, on remplace β , α_1 et α_2 par leurs espérances dans la figure 12. Pour obtenir le même résultat à la sortie, il faut donc multiplier le $n^{ième}$ élément, x(n), du tableau d'entrée par $\{1 + (\delta_x + \delta_+).n_{M-m+1} + \delta_+\}$ pour $m=3,..,M$, et par $\{1 + \delta_+\}$ pour $m=1,2$, puisque les facteurs multiplicatifs sont tous ± 1 ou $\pm j$.

Négligeant les effets de second ordre, l'espérance de l'erreur sur l'élément de sortie y(k) peut être causé par l'entrée $\{x'(n)\}_{n=0}^{N-1}$ définie par :

$$x'(n) = x(n).\{1 + f(n).(\delta_x + \delta_+) + M.\delta_+\}$$

où f(n) est donné par (4.9). L'erreur ainsi engendrée vaut :

$$x'(n) - x(n) = x(n).\{f(n).(\delta_x + \delta_+) + M.\delta_+\} \qquad (4.10)$$

Comme l'espérance des erreurs à la sortie est juste la Transformée de Fourier Discrète des erreurs données en (4.10), on obtient, par la relation de Parseval (1.2) :

$$\frac{\sum_{k=0}^{N-1} \{E\{|y'(k)-y(k)|\}\}^2}{\sum_{k=0}^{N-1} |y(k)|^2} = \frac{N \sum_{n=0}^{N-1} |x(n)|^2.\{f(n).(\delta_x + \delta_+) + M.\delta_+\}^2}{\sum_{k=0}^{N-1} |y(k)|^2}$$

Ajoutant ceci à (4.8), on obtient le rapport bruit à signal :

$$\left(\frac{B}{S}\right)_T = \left(\frac{B}{S}\right)_A + \frac{1}{\sum_{n=0}^{N-1} |x(n)|^2} \sum_{n=0}^{N-1} \{f(n).(\delta_x + \delta_+) + M.\delta_+\}^2.|x(n)|^2 \qquad (4.11)$$

où l'indice T signifie troncature,

$\left(\frac{B}{S}\right)_A$ est donné par (4.8).

CALCUL DES BORNES

Comme $0 \leq f(n) \leq M-2$ pour tout n appartenant à l'ensemble $\{0,1,\ldots,N-1\}$, on peut déterminer des bornes inférieures et supérieures indépendantes du tableau des données :

$$M.\Delta_+^2 \leq \left(\frac{B}{S}\right)_A \leq (M-2)\Delta_x^2 + (2M-2)\Delta_+^2 \qquad (4.12)$$

$$M.\Delta_+^2 + M^2.\delta_+^2 \leq \left(\frac{B}{S}\right)_T \leq (M-2)\Delta_x^2 + (2M-2)\Delta_+^2$$

$$+ \{(M-2)\delta_x + (2M-2)\delta_+\}^2 \qquad (4.13)$$

Ces bornes sont des généralisations de celles proposées par Kaneko et Liu ([9], expressions (11) et (12)).

4 . 3 ALGORITHME DE SANDE ET TUKEY

En utilisant la notion de bloc définie au paragraphe 1.3.B, on établit des expressions et des bornes pour le rapport bruit à signal dans le cas de l'algorithme de Sande et Tukey quand $N = 2^M$ (voir paragraphe 1.2.B).

Dans ce cas, la proposition 4.1 reste valable (voir la figure 3 au paragraphe 1.2), mais les autres doivent être remplacées par :

Proposition 4.4

Si à l'étape m, tous les éléments du $l^{ième}$ bloc ont une erreur relative γ_q :

$$x'_m(l2^{M-m}+q) = x_m(l2^{M-m}+q).(1+\gamma_q)$$

où $q = 0,1,\ldots,2^{M-m}-1$,

et que la variance γ_q est indépendante de q, c'est-à-dire

$$var(\gamma_q) = d^2,$$

alors la contribution de toutes les erreurs dans le $l^{ième}$ bloc à l'étape m à la variance normalisée de l'erreur à la sortie vaut :

$$\frac{\sum_{k=0}^{N-1} var\{|v'(k)-v(k)|\}}{\sum_{k=0}^{N-1} |v(k)|^2} = \frac{d^2}{\sum_{k=0}^{N-1} |v(k)|^2} \sum_{q=0}^{2^{M-m}-1} |x_M(l2^{M-m}+q)|^2$$

Démonstration :

Par la proposition 4.1, on a :

$$\frac{\sum_{k=0}^{N-1} \mathrm{var}\{|v'(k)-v(k)|\}}{\sum_{k=0}^{N-1} |v(k)|^2}$$

$$= \frac{d^2}{\sum_{k=0}^{N-1} |v(k)|^2} \, 2^{M-m} \sum_{q=0}^{2^{M-m}-1} |x_m(l2^{M-m}+q)|^2 \ ,$$

ce qui donne, par la proposition 1.2 et la relation de Parseval (1.2) :

$$= \frac{d^2}{\sum_{k=0}^{N-1} |v(k)|^2} \sum_{q=0}^{2^{M-m}-1} |x_M(l2^{M-m}+q)|^2 .$$

ANALYSE DES OPERATIONS ARITHMETIQUES DANS L'ALGORITHME

Le calcul d'un même bloc à l'étape m par les "papillons" définis en (1.14) peut s'écrire de la manière suivante :

$$x_m(l'2^{M-m+1}+q) = x_{m-1}(l'2^{M-m+1}+q) + x_{m-1}(l'2^{M-m+1}+2^{M-m}+q)$$

$$x_m(l'2^{M-m+1}+2^{M-m}+q) = \{x_{m-1}(l'2^{M-m+1}+q) - x_{m-1}(l'2^{M-m+1}+2^{M-m}+q)\}$$

$$\times \exp(-j2\pi q/2^{M-m+1})$$

$$l' = \sum_{i=1}^{m-1} k_i 2^{m-i-1}$$

$$q = \sum_{i=1}^{M-m} n_i 2^{i-1} \qquad (4.14)$$

Les expressions (4.14) sont illustrées à la figure 13. Dès que l'on introduit les erreurs selon (4.4), il faut considérer la figure 14. Ici aussi, l'erreur complexe β est dûe à la multiplication et les erreurs complexes α_1 et α_2 sont dûes aux additions. Négligeant les effets de second ordre, on peut combiner les erreurs comme sur la figure 15.

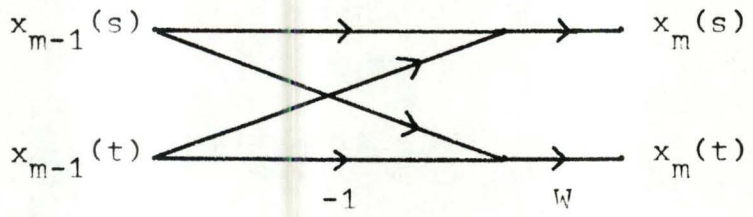

Figure 13. "Papillon" de la Transformée de Fourier Rapide (DIF) où $s = l'2^{M-m+1}+q$

$t = s+2^{M-m}$

$W = \exp(-j2\pi q/2^{M-m+1})$.

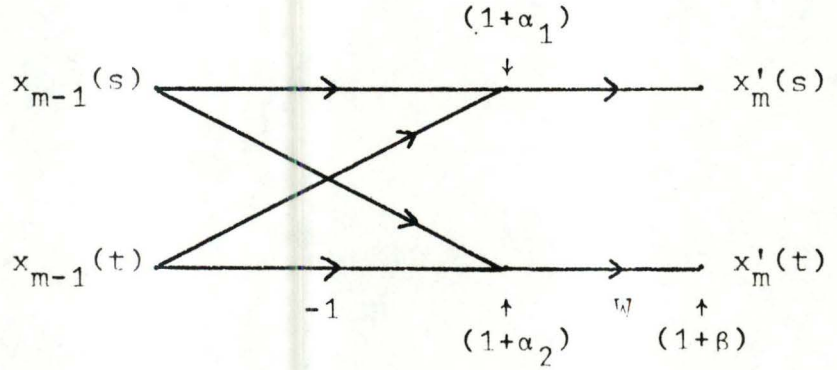

Figure 14. "Papillon" de la DIF avec modèle d'erreur.

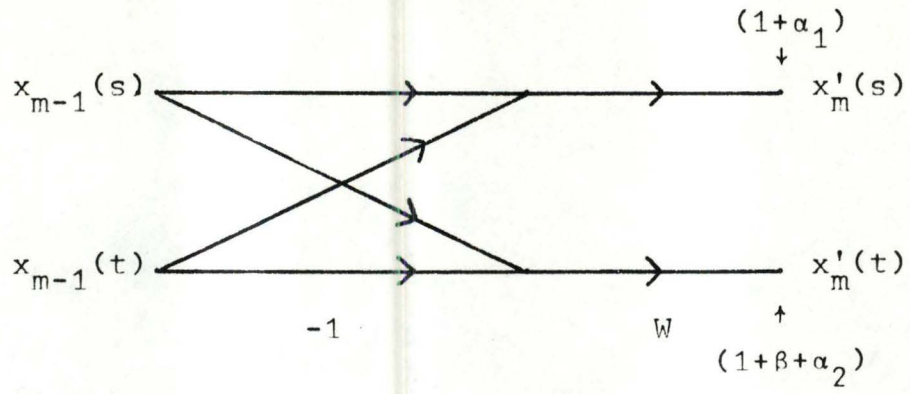

Figure 15."Papillon" de la DIF avec erreurs combinées.

En fait, si on inclut l'erreur aux expressions (4.14) et que l'on prend les notations de la figure 13, on obtient :

$$x'_m(s) = x_m(s).(1+\alpha_1)$$

$$x'_m(t) = x_m(t).(1+\beta+\alpha_2+\beta\alpha_2)$$

$$l' = \sum_{i=1}^{m-1} k_i 2^{m-i-1}$$

$$q = \sum_{i=1}^{M-m} n_i 2^{i-1} \qquad (4.15)$$

Comme dans le cas précédent, on va sommer toutes les contributions en séparant les erreurs de multiplication et d'addition et en négligeant le second ordre.

CAS DE L'ARRONDI

Comme $x_m(s)$ et $x_m(t)$ sont dans des blocs différents à l'étape m, on peut appliquer la proposition 4.4 pour obtenir la contribution des erreurs à l'étape m au rapport bruit à signal.

Elle vaut :

$$\frac{1}{\sum_{k=0}^{N-1} |y(k)|^2} \sum_{k=0}^{N-1} \{k_m(\Delta_x^2 + \Delta_+^2) + \Delta_+^2\} \cdot |y(k)|^2$$

En sommant sut tous les m et en notant que les étapes M-1 et M n'engendrent pas d'erreurs de multiplication, on obtient :

$$\left(\frac{B}{S}\right)_A = M.\Delta_+^2 + \frac{\Delta_+^2 + \Delta_x^2}{\sum_{k=0}^{N-1} |y(k)|^2} \sum_{k=1}^{N-1} f(k).|y(k)|^2 \qquad (4.16)$$

où l'indice A signifie arrondi,

f(k) est défini en (4.9).

CAS DE LA TRONCATURE

La contribution des variances des erreurs au rapport bruit à signal est donnée par (4.16). Pour calculer la contribution dûe aux espérances, il suffit de se référer à la figure 15.

Pour un bloc de type s, $k_m = 0$ et la contribution des erreurs à l'étape m à chaque élément du bloc correspondant vaut $\delta_+.y(k)$; pour un bloc de type t, $k_m = 1$ et la contribution vaut $(2\delta_+ + \delta_x).y(k)$. A partir de ces remarques, on obtient aisément que :

$$\left(\frac{B}{S}\right)_T = \left(\frac{B}{S}\right)_A + \frac{1}{\sum_{k=0}^{N-1} |y(k)|^2} \sum_{k=0}^{N-1} \{f(k).(\delta_x + \delta_+) + M.\delta_+\}^2.|y(k)|^2 \qquad (4.17)$$

où l'indice T signifie troncature,

$\left(\frac{B}{S}\right)_A$ est donné par (4.16).

CALCUL DES BORNES

On trouve les mêmes bornes que celles pour l'algorithme de Cooley et Tukey. Elles sont des généralisations de celles proposées par Kaneko et Liu ([7], expressions (18) et (19)).

4 . 4 CAS PARTICULIERS

Dans ce paragraphe, on détermine des expressions pour le rapport bruit à signal lorsque le signal d'entrée de la Transformée de Fourier Rapide est un bruit blanc tel que :

$$E\{x(n)\} = 0$$

$$E\{x(n).x^*(m)\} = \begin{cases} N_0 & \text{si } n = m \\ 0 & \text{sinon.} \end{cases} \qquad (4.18)$$

où $x^*(m)$ est le nombre complexe conjugué de $x(m)$.

Dans les expressions du rapport bruit à signal, les valeurs $|x(n)|^2$ et $|y(k)|^2$ sont à remplacer par leurs espérances, à savoir N_0 et NN_0 .

On analyse les algorithmes de Cooley et Tukey et ceux de Sande et Tukey quand $N = r^M$ [17] et quand $N = r_1 . r_2$.

4.4.1 $N = 2^M$

A CAS DE L'APRONDI

On peut interpréter les expressions (4.8) et (4.16) en tenant compte de (4.18). On obtient dans les deux cas :

$$\left(\frac{B}{S}\right)_A = {}^M\Delta_+^2 + \frac{\Delta_+^2 + \Delta_\times^2}{N} \sum_{n=1}^{N-1} f(n)$$

$$= {}^M\Delta_+^2 + \frac{\Delta_+^2 + \Delta_\times^2}{N} \sum_{n=1}^{N-1} \sum_{i=1}^{M-2} n_i$$

$$= {}^M\Delta_+^2 + \frac{\Delta_+^2 + \Delta_\times^2}{N} (M-2)2^{M-1}$$

$$= \frac{3M-2}{2} \Delta_+^2 + \frac{M-2}{2} \Delta_\times^2 .$$

Cependant, cette expression ne tient compte des multiplications sans erreur qu'à deux étapes de l'algorithme, alors qu'il y en a à chaque étape. Ce fait n'a pas pu s'utiliser dans l'étude précédente vu que les valeurs intermédiaires $x_m(.)$ étaient inconnues; mais dans le cas particulier (4.18), on sait que

$$E\{|x_m(.)|^2\} = 2^m N_0 .$$

En appliquant directement la proposition 4.1 à l'algorithme de Cooley et Tukey, on obtient

$$\left(\frac{B}{S}\right)_A = {}^M\Delta_+^2 + \frac{\Delta_+^2 + \Delta_\times^2}{N^2 . N_0} \sum_{m=2}^{M} 2^{M-m} . (N - 4 . 2^{M-m}) . 2^{m-1} N_0$$

où l'indice A signifie arrondi,

$M\Delta_+^2$ est la contribution des erreurs d'addition,

$\Delta_+^2 + \Delta_\times^2$ est la variance d'une erreur de multiplication,

2^{M-m} est le facteur de propagation de la proposition 4.1,

4.2^{M-m} est le nombre de multiplications par ±1 et ±j à l'étape m,

$2^{m-1}N_0$ est la variance de $x_{m-1}(.)$.

Tenant compte du fait que

$$\sum_{i=0}^{D} 2^i = 2^{D+1}-1 \qquad (4.19)$$

l'expression ci-dessus devient :

$$\left(\frac{B}{S}\right)_A = \left\{\frac{3M-3}{2} + \frac{2}{N}\right\} \Delta_+^2 + \left\{\frac{M-3}{2} + \frac{2}{N}\right\} \Delta_\times^2 \qquad (4.20)$$

En appliquant directement la proposition 4.1 à l'algorithme de Sande et Tukey, on obtient :

$$\left(\frac{B}{S}\right)_A = M\Delta_+^2 + \frac{\Delta_+^2 + \Delta_\times^2}{N^2.N_0} \sum_{m=1}^{M-2} 2^{M-m}.\left\{\frac{N}{2} - 2.2^{m-1}\right\}.2^m N_0$$

où l'indice A signifie arrondi,

$M\Delta_+^2$ est la contribution des erreurs d'addition,

$\Delta_+^2 + \Delta_\times^2$ est la variance d'une erreue de multiplication,

2^{M-m} est le facteur de propagation de la proposition 4.1,

2.2^{m-1} est le nombre de multiplications par ±1 et ±j à l'étape m,

$2^m N_0$ est la variance de $x_m(.)$.

Tenant compte de (4.19), cette expression se réduit à (4.20).

Les deux algorithmes de la Transformée de Fourier Rapide admettent donc le même rapport bruit à signal (4.20) dans le cas d'arrondi et pour le problème particulier (4.18)

B CAS DE LA TRONCATURE

Dans ce cas, il faut encore ajouter les effets des espérances des erreurs. Pour ce faire, on reprend les expressions (4.11) et (4.17) où cette fois-ci, $\left(\frac{B}{S}\right)_A$ est la valeur donnée en (4.20). En remplaçant $|x(.)|^2$ et $|y(.)|^2$ par N_0 et NN_0, on obtient pour chacune des deux expressions :

$$\left(\frac{B}{S}\right)_T = \left(\frac{B}{S}\right)_A + \frac{1}{N} \sum_{n=0}^{N-1} \{f(n).(\delta_+ + \delta_x) + M\delta_+\}^2$$

où l'indice T signifie troncature,

f(n) est donné en (4.9).

En développant cette dernière expression, on a :

$$\left(\frac{B}{S}\right)_T = \left(\frac{B}{S}\right)_A + \frac{1}{N} \left[\sum_{n=0}^{N-1} \{f(n)\}^2\right].(\delta_+ + \delta_x)^2 + M^2\delta_+^2 + (M-2)M\delta_+(\delta_+ + \delta_x) \quad .$$

Comme $$\sum_{n=0}^{N-1} \{f(n)\}^2 = \sum_{n=0}^{N-1} \left[\sum_{m=1}^{M-2} n_m\right]^2$$

$$= 4. \sum_{n=0}^{M-2} C_{M-2}^{i} . i^2$$

$$= 4.(M-1).(M-2).2^{M-4} \quad ,$$

l'expression devient :

$$\left(\frac{B}{S}\right)_T = \left(\frac{B}{S}\right)_A + \frac{M^2}{4}\left\{9\delta_+^2 + \delta_\times^2 + 6\delta_\times\delta_+\right\} - \frac{M}{4}\left\{3\delta_\times^2 + 11\delta_+^2 + 14\delta_\times\delta_+\right\} + \frac{1}{2}\left\{\delta_\times + \delta_+\right\}^2 \qquad (4.21)$$

où l'indice T signifie troncature,

$\left(\frac{B}{S}\right)_A$ est donné en (4.20).

En cas de troncature, les deux algorithmes de la Transformée de Fourier Rapide admettent donc aussi la même expression (4.21) du rapport bruit à signal pour le problème particulier (4.18).

Remarque :

Les expressions (4.20) et (4.21) sont des généralisations d'expressions proposées pour le rapport bruit à signal par Kaneko et Liu [7], expressions (22) et (24),

[9], expressions (14) et (15).

4.4.2 $N = r^M$, r quelconque

On détermine ici des expressions pour le rapport bruit à signal dans le cas d'arrondi pour le problème particulier (4.18) Le cas de la troncature est nettement plus compliqué et ne sera pas traité.

Tout d'abord, on peut énoncer une propriété commune aux deux algorithmes définis respectivement en (1.22) et en (1.24).

Proposition 4.5

Une erreur introduite à l'étape m se propage à r^{M-m} éléments du résultat. La variance de l'erreur du résultat est égale, en chacun des r^{M-m} éléments affectés, à la variance de l'erreur à l'étape m.

La démonstration est semblable à celle de la proposition 4.1.

On constate d'autre part que $E\{|x_m(.)|^2\} = r^m N_0$.

A ALGORITHME DE COOLEY ET TUKEY

Le calcul du $l^{ième}$ bloc à l'étape m peut se faire en "papillons". Par exemple, dans le cas où r=3, on a :

$$x_m(l3^m+q')$$

$$= x_{m-1}(l3^m+q') + x_{m-1}(l3^m+3^{m-1}+q').\exp(-j2\pi q'/3^m)$$

$$+ x_{m-1}(l3^m+2.3^{m-1}+q').\exp(-j2\pi.2q'/3^m)$$

$$x_m(13^m+3^{m-1}+q')$$

$$= x_{m-1}(13^m+q')$$

$$+ x_{m-1}(13^m+3^{m-1}+q').\exp(-j2\pi q'/3^m).\exp(-j2\pi/3)$$

$$+ x_{m-1}(13^m+2.3^{m-1}+q').\exp(-j2\pi.2q'/3^m).\exp(-j2\pi.2/3)$$

$$x_m(13^m+2.3^{m-1}+q')$$

$$= x_{m-1}(13^m+q')$$

$$+ x_{m-1}(13^m+3^{m-1}+q').\exp(-j2\pi q'/3^m).\exp(-j2\pi.2/3)$$

$$+ x_{m-1}(13^m+2.3^{m-1}+q').\exp(-j2\pi.2q'/3^m).\exp(-j2\pi.4/3)$$

$$q' = \sum_{i=1}^{m-1} k_i 3^{i-1} \qquad (4.22)$$

Les expressions (4.22) sont illustrées à la figure 16.

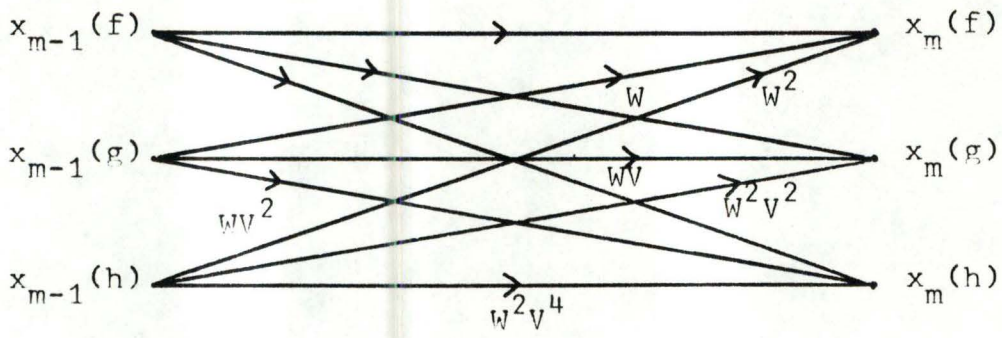

Figure 16. "Papillon" de la Transformée de Fourier Rapide pour $N=3^M$ (DIT).

$W = \exp(-j2\pi q'/3^m)$ $\quad f = 13^m+q'$

$V = \exp(-j2\pi/3)$ $\quad g = 13^m+3^{m-1}+q'$

$h = 13^m+2.3^{m-1}+q'$

En prenant les notations de la figure 16, en introduisant les erreurs selon (4.4) et en négligeant les effets de second ordre, les expressions (4.22) deviennent :

$$x'_m(f) - x_m(f) = x_{m-1}(f).(\alpha_1 + \alpha_4) + x_{m-1}(g).W.(\beta_1 + \alpha_1 + \alpha_4) + x_{m-1}(h).W^2.(\beta_4 + \alpha_4)$$

$$x'_m(g) - x_m(g) = x_{m-1}(f).(\alpha_2 + \alpha_5) + x_{m-1}(g).WV.(\beta_2 + \alpha_2 + \alpha_5) + x_{m-1}(h).W^2V^2.(\beta_5 + \alpha_5)$$

$$x'_m(h) - x_m(h) = x_{m-1}(f).(\alpha_3 + \alpha_6) + x_{m-1}(g).WV^2.(\beta_3 + \alpha_3 + \alpha_6) + x_{m-1}(h).W^2V^4.(\beta_6 + \alpha_6),$$

où les α_i sont les erreurs complexes d'addition,

les β_i sont les erreurs complexes de multiplication.

La contribution des erreurs d'addition au rapport bruit à signal vaut :

$$\frac{1}{N^2.N_0} \sum_{n=0}^{N-1} \sum_{m=1}^{M} r^{M-m}.r.r^{m-1}N_0.s(n_{M-m+1}).\Delta_+^2$$

où r^{M-m} est le facteur de propagation de l'erreur (prop. 4.5),

r est le nombre de termes influençant l'erreur sur un élément à l'étape m,

$r^{m-1}N_0$ est la variance de $x_{m-1}(.)$,

$$s(n_i) = \begin{cases} r-1 & \text{si } n_i = 0 \\ r-n_i & \text{si } n_i \neq 0 \end{cases}$$

Δ_+^2 est la variance d'une erreur d'addition.

On a donc : $\frac{1}{N} \sum_{m=1}^{M} \frac{N}{r} (r-1) \frac{(r+2)}{2} \Delta_+^2$.

La contribution des erreurs de multiplication au rapport bruit à signal vaut :

$$\frac{1}{N^2 . N_0} \sum_{m=1}^{M} r^{M-m} . r^{m-1} N_0 . (r-1) . (N - r^{M-m}) . (\Delta_+^2 + \Delta_\times^2)$$

où r^{M-m} est le facteur de propagation de l'erreur de la proposition 4.5,

$r^{m-1} N_0$ est la variance de $x_{m-1}(.)$,

$(r-1)$ est le nombre de termes influençant l'erreur sur un élément à l'étape m,

$(N - r^{M-m})$ est le nombre de facteurs multiplicatifs non égaux à 1 à l'étape m,

$(\Delta_+^2 + \Delta_\times^2)$ est la variance d'une erreur de multiplication.

On a donc : $\frac{1}{N} \sum_{m=1}^{M} \frac{r-1}{r} (N - r^{M-m}) . (\Delta_+^2 + \Delta_\times^2)$.

En sommant ces contributions, on obtient finalement, dans le cas de l'algorithme de Cooley et Tukey, le rapport bruit à signal :

$$\left(\frac{B}{S}\right)_A = \frac{\Delta_\times^2}{r} \left\{ (r-1)^M - 1 + \frac{1}{N} \right\} + \frac{\Delta_+^2}{r} \left\{ \frac{r^2+3r-4}{r} M - 1 + \frac{1}{N} \right\} \qquad (4.23)$$

Remarque :

Cette expression n'est pas une généralisation de (4.20). On n'a en effet pas tenu compte des multiplications par -1 et $\pm j$.

B ALGORITHME DE SANDE ET TUKEY

Le calcul d'un bloc à l'étape m peut également se faire en "papillons". Par exemple, dans le cas où r=3, on a :

$$x_m(l'3^{M-m+1}+q) = x_{m-1}(l'3^{M-m+1}+q) + x_{m-1}(l'3^{M-m+1}+3^{M-m}+q) + x_{m-1}(l'3^{M-m+1}+2.3^{M-m}+q)$$

$$x_m(l'3^{M-m+1}+3^{M-m}+q) = \left[x_{m-1}(l'3^{M-m+1}+q) + x_{m-1}(l'3^{M-m+1}+3^{M-m}+q).\exp(-j2\pi/3) + x_{m-1}(l'3^{M-m+1}+2.3^{M-m}+q).\exp(-j2\pi.2/3)\right] \times \exp(-j2\pi q/3^{M-m+1})$$

$$x_m(l'3^{M-m+1}+2.3^{M-m}+q) = \left[x_{m-1}(l'3^{M-m+1}+q) + x_{m-1}(l'3^{M-m+1}+3^{M-m}+q).\exp(-j2\pi.2/3) + x_{m-1}(l'3^{M-m+1}+2.3^{M-m}+q).\exp(-j2\pi.4/3)\right] \times \exp(-j2\pi q.2/3^{M-m+1})$$

$$l' = \sum_{i=1}^{m-1} k_i 3^{m-i-1}$$

$$q = \sum_{i=1}^{M-m} n_i 3^{i-1} \qquad (4.24)$$

Les expressions (4.24) sont illustrées à la figure 17.

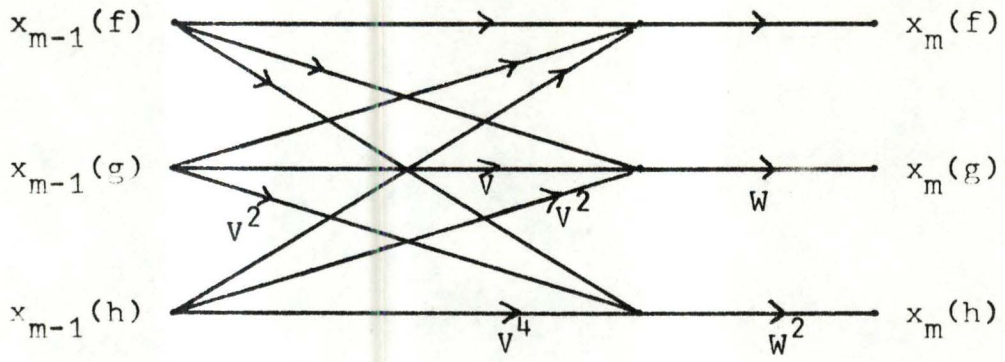

Figure 17. "Papillon" de la Transformée de Fourier Rapide pour $N=3^M$ (DIF).

$W = \exp(-j2\pi q/3^{M-m+1})$ $\quad f = l'3^{M-m+1}+q$

$V = \exp(-j2\pi/3)$ $\quad g = l'3^{M-m+1}+3^{M-m}+q$

$h = l'3^{M-m+1}+2.3^{M-m}+q$

En prenant les notations de la figure 17, en introduisant les erreurs selon (4.4) et en négligeant les effets de second ordre, les expressions (4.24) deviennent :

$$x'_m(f) - x_m(f) = x_{m-1}(f).(\alpha_1 + \alpha_2) + x_{m-1}(g).(\alpha_1 + \alpha_2) + x_{m-1}(h).\alpha_2$$

$$x'_m(g) - x_m(g) = \left[x_{m-1}(f).(\beta_1 + \alpha_3 + \alpha_4) + x_{m-1}(g).V.(\beta_1 + \beta_2 + \alpha_3 + \alpha_4) + x_{m-1}(h).V^2.(\beta_1 + \beta_3 + \alpha_4)\right] \times W$$

$$x'_m(h) - x_m(h) = \left[x_{m-1}(f).(\beta_4 + \alpha_5 + \alpha_6) + x_{m-1}(g).V^2.(\beta_4 + \beta_5 + \alpha_5 + \alpha_6) + x_{m-1}(h).V^4.(\beta_4 + \beta_6 + \alpha_6)\right] \times W^2$$

où les α_i correspondent à des erreurs d'addition,

les β_i correspondent à des erreurs de multiplication.

On constate que la contribution des erreurs d'addition est la même que pour l'algorithme de Cooley et Tukey.

Par contre, la contribution des erreurs de multiplication par les facteurs de type V au rapport bruit à signal vaut :

$$\frac{1}{N^2.N_0} \sum_{m=1}^{M} r^{M-m}.r^{m-1}N_0.(\Delta_x^2 + \Delta_+^2).(r-1)^2 r^{M-1}$$

où r^{M-m} est le facteur de propagation de l'erreur,

$r^{m-1}N_0$ est la variance de $x_{m-1}(.)$,

$\Delta_+^2 + \Delta_x^2$ est la variance d'une erreur de multiplication,

$(r-1)^2 r^{M-1}$ est le nombre de multiplications par un facteur de type V, à chaque étape.

La contribution des erreurs de multiplication par les facteurs de type W au rapport bruit à signal vaut :

$$\frac{1}{N^2.N_0} \sum_{m=1}^{M} r^{M-m}.r^{m}N_0.(\Delta_x^2 + \Delta_+^2).(r-1)r^{-1}(N-r^m)$$

où r^{M-m} est le facteur de propagation de l'erreur,

$r^m N_0$ est la variance des éléments du tableau intermédiaire, avant qu'ils ne soient multipliés par un facteur de type W à l'étape m,

$(r-1)r^{-1}(N-r^m)$ est le nombre de multiplications par un facteur de type W, non égal à 1, à l'étape m,

$(\Delta_x^2 + \Delta_+^2)$ est la variance d'une erreur de multiplication.

En sommant toutes ces différentes contributions, on obtient finalement, dans le cas de l'algorithme de Sande et Tukey, le rapport bruit à signal :

$$\left(\frac{B}{S}\right)_A = \left[\frac{2r^2-3r+1}{r^2} M -1 + \frac{1}{N} \right] \Delta_x^2 + \left[\frac{r^3+5r^2-8r+2}{2r^2} M -1 + \frac{1}{N} \right] \Delta_+^2 \qquad (4.25)$$

Remarque :

Cette expression n'est pas une généralisation de (4.20) car, comme dans le cas de l'algorithme de Cooley et Tukey, on n'a pas tenu compte des multiplications par -1 et ±j.

4.4.3 $N = r_1 \cdot r_2$, r_1 et r_2 quelconques

On détermine ici des expressions pour le rapport bruit à signal dans le cas d'arrondi pour le problème particulier (4.18). L'étude est semblable à celle des paragraphes précédents : elle est basée sur l'idée de Tran-Thong et Bede Liu [17], mais n'a encore jamais été publiée.

On peut tout d'abord énoncer une propriété commune aux deux algorithmes définis respectivement en (1.28) et en (1.29).

Proposition 4.6

Une erreur introduite à l'étape 1 se propage à r_2 éléments du résultat. La variance de l'erreur du résultat est égale, en chacun des r_2 éléments affectés, à la variance de l'erreur à l'étape 1.

La démonstration est semblable à celle de la proposition 4.1.

On constate d'autre part que $E\{|x_1(.)|^2\} = r_1 N_0$.

A ALGORITHME DE COOLEY ET TUKEY

On peut interpréter l'algorithme (1.28) comme suit :

- l'étape 1 consiste à calculer r_2 transformées de Fourier discrètes de r_1 éléments chacune,
- l'étape 2 consiste à calculer r_1 "quasi-transformées de Fourier discrètes" de r_2 éléments chacune, c'est-à-dire à

effectuer des sommations pondérées de r_2 termes, comme pour une Transformée de Fourier Discrète, mais où les facteurs de poids ont été modifiés.

En introduisant les erreurs d'addition à la manière de (4.4) et en négligeant les effets de second ordre, la contribution des erreurs d'addition introduites à l'étape 1 au rapport bruit à signal vaut :

$$\frac{1}{N^2 N_0} \; r_2 \cdot \Delta_+^2 \cdot r_1 \cdot \left[\sum_{n_2=0}^{r_1-1} s_1(n_2) \right] \cdot r_2 \cdot N_0$$

où r_2 est le nombre de transformées effectuées à l'étape 1,

Δ_+^2 est la variance d'une erreur d'addition,

$$s_1(n_2) = \begin{cases} r_1 - 1 & \text{si } n_2 = 0 \\ r_1 - n_2 & \text{sinon,} \end{cases}$$

r_1 est le nombre de termes influençant l'erreur sur un élément à l'étape 1,

r_2 est le facteur de propagation de l'erreur de la proposition 4.6,

N_0 est la variance de $x_0(.)$.

De manière semblable, la contribution des erreurs d'addition introduites à l'étape 2 vaut :

$$\frac{1}{N^2 N_0} \; r_1 \cdot \Delta_+^2 \cdot r_2 \cdot \left[\sum_{n_1=0}^{r_2-1} s_2(n_1) \right] \cdot r_1 N_0$$

où r_1 est le nombre de "quasi-transformées" effectuées à l'étape 2,

Δ_+^2 est la variance d'une erreur d'addition,

$$s_2(n_1) = \begin{cases} r_2 - 1 & \text{si } n_1 = 0 \\ r_2 - n_1 & \text{sinon,} \end{cases}$$

r_2 est le nombre de termes influençant l'erreur sur un élément à l'étape 2,

$r_1 N_0$ est la variance de $x_1(.)$.

D'autre part, en introduisant les erreurs de multiplication à la manière de (4.4) et en négligeant les effets de second ordre, la contribution des erreurs de multiplication introduites à l'étape 1 vaut :

$$\frac{1}{N^2 N_0} (\Delta_+^2+\Delta_\times^2).N_0.r_2.r_2(r_1-1)^2$$

où $\Delta_+^2+\Delta_\times^2$ est la variance d'une erreur de multiplication,

N_0 est la variance de $x_0(.)$,

r_2 est le facteur de propagation de l'erreur à l'étape 1,

$r_2(r_1-1)^2$ est le nombre de multiplications par des facteurs non égaux à 1, à l'étape 1.

La contribution des erreurs de multiplication introduites à l'étape 2 vaut :

$$\frac{1}{N^2 N_0} r_1 N_0.(\Delta_\times^2+\Delta_+^2).(r_2-1)(N-1)$$

où $r_1 N_0$ est la variance de $x_1(.)$,

$\Delta_x^2+\Delta_+^2$ est la variance d'une erreur de multiplication,

$(r_2-1)(N-1)$ est le nombre de multiplications par des facteurs non égaux à 1, à l'étape 2.

En sommant ces différentes contributions, on obtient finalement, dans le cas de l'algorithme de Cooley et Tukey, le rapport bruit à signal pour le problème particulier (4.18) :

$$\left(\frac{B}{S}\right)_A = \Delta_+^2 \cdot \left[\frac{r_1+r_2+6}{2} - \frac{1}{N}\cdot(3r_2+2r_1+1) + \frac{1}{N^2}\cdot(r_2^2+r_1) \right]$$

$$+ \Delta_x^2 \cdot \left[2 - \frac{1}{N}\cdot(2r_2+r_1+1) + \frac{1}{N^2}\cdot(r_2^2+r_1) \right] \qquad (4.26)$$

Remarque :

Cette expression est une généralisation de (4.23), c'est-à-dire que l'expression (4.26), prise avec $r_1 = r_2 = r$, se réduit à (4.23) où $M = 2$.

B ALGORITHME DE SANDE ET TUKEY

L'algorithme (1.29) peut s'interpréter comme suit :

- l'étape 1 consiste à calculer r_2 transformées de Fourier discrètes de r_1 éléments chacune, puis à multiplier ces valeurs par des coéfficients complexes de module unité,
- l'étape 2 consiste à calculer r_1 transformées de Fourier discrètes de r_2 éléments chacune.

La contribution des erreurs d'addition au rapport bruit à signal est la même que dans le cas de l'algorithme de Cooley et Tukey.

Par contre, en négligeant les effets de second ordre, la contribution des erreurs de multiplication à l'étape 1 au rapport bruit à signal vaut :

$$\frac{1}{N^2 N_0}(\Delta_x^2+\Delta_+^2).r_2.\left[N_0.r_2(r_1-1)^2+N_0 r_1.(N-r_1)\frac{(r_1-1)}{r_1}\right]$$

où $\Delta_x^2+\Delta_+^2$ est la variance d'une erreur de multiplication,

r_2 est le facteur de propagation de l'erreur à l'étape 1,

N_0 est la variance de $x_0(.)$,

$r_2(r_1-1)^2$ est le nombre de facteurs multiplicatifs non égaux à 1, dans les transformées à l'étape 1,

$N_0 r_1$ est la variance des éléments après les transformées de l'étape 1,

$(N-r_1)\frac{(r_1-1)}{r_1}$ est le nombre de facteurs multiplicatifs non égaux à 1, intervenant après les transformées de l'étape 1.

Si on néglige les effets de second ordre, la contribution des erreurs de multiplication introduites à l'étape 2 vaut :

$$\frac{1}{N^2 N_0}(\Delta_x^2+\Delta_+^2).r_1(r_2-1)^2.N_0 r_1$$

où $\Delta_x^2+\Delta_+^2$ est la variance d'une erreur de multiplication,

$r_1(r_2-1)^2$ est le nombre de facteurs multiplicatifs non égaux à 1, à l'étape 2,

$N_0 r_1$ est la variance de $x_1(.)$.

En sommant ces différentes contributions, on obtient finalement, dans le cas de l'algorithme de Sande et Tukey, le rapport bruit à signal pour le problème particulier (4.18) :

$$\left(\frac{B}{S}\right)_A = \Delta_+^2 . \left[\frac{r_1+r_2+8}{2} - \frac{1}{N}.(4r_1+4r_2-1) + \frac{1}{N^2}.(r_1^2+r_2^2) \right]$$

$$+ \Delta_x^2 . \left[3 - \frac{1}{N}.(3r_1+3r_2-1) + \frac{1}{N^2}.(r_1^2+r_2^2) \right] \qquad (4.27)$$

Remarque :

Cette expression est une généralisation de (4.25), c'est-à-dire que l'expression (4.27), évaluée avec $r_1 = r_2 = r$, se réduit à (4.25) où $M = 2$.

R E S U L T A T S

N U M E R I Q U E S

5 . 1 RESULTATS NUMERIQUES POUR LA VIRGULE FIXE

Les sous-programmes qui permettent d'effectuer les différentes opérations en utilisant une arithmétique en virgule fixe ont été écrits en ASSEMBLER.
L'idée de base est d'utiliser l'arithmétique en virgule flottante de l'ordinateur Siemens 4004. La longueur double a été choisie. Chaque nombre est représenté en "signe et grandeur" par une suite de 16 chiffres hexadécimaux : les deux premiers sont réservés au signe et à la caractéristique, les 14 suivants à la mantisse. Comme chaque chiffre hexadécimal est composé de 4 bits, il est aisé d'adapter cette représentation; en annulant la caractéristique; la mantisse coupée au nombre de bits désiré, traduit un nombre binaire avec virgule fixe. Par exemple, le nombre 0.1010111 est traduit par 40AD000000000000. Le passage de la représentation en virgule flottante à la représentation en virgule fixe n'est possible que si le nombre est compris entre -1 et +1. Par exemple, pour une longueur de (1+7) bits, C0AFBE643F124C devient C0AD000000000000 qui traduit -0.1010111.
La multiplication introduit une contrainte sur la taille maximale des nombres. Chacun d'eux pourrait être traduit par une suite de 56 bits. Toutefois, on se limite à 36 bits pour des raisons techniques. Le produit de deux nombres de 36 bits, x et y, est effectué en deux étapes.
En effet, le produit de x par y est formé de 72 bits qui ne peuvent être contenus dans une seule mantisse. On décompose y en y_1, formé des 20 premiers bits de y et en y_2, formé des 16 derniers.
Le produit de x par y_1 compte 56 bits et peut être contenu dans une première mantisse. Le produit de x par y_2 compte 42 bits et peut être contenu dans une seconde mantisse dont la caractéristique retient un facteur 2^{-20}. Tous les bits du produit de x par y peuvent ainsi être retrouvés et le résultat est évalué et arrondi correctement : s'il est situé exactement entre deux nombres de la longueur choisie, t, il est de la forme $0.b_1....b_t10....0$; le résultat arron-

di est $0.b_1 \ldots b_t + 2^{-t}$ si $b_t = 1$ et $0.b_1 \ldots b_t$ si $b_t = 0$.
A cause de la représentation en "signe et grandeur" le modèle statistique doit être légèrement modifié. Après un déplacement, l'arrondi est utilisé au lieu de la troncature. Par (2.10b) on obtient une erreur de moyenne nulle et de variance égale à $2^{-2t}/8$. Pour la troncature, l'erreur dépend du signe des nombres, comme l'indiquent les relations (2.3) et (2.4). Si le signal de départ est distribué de manière uniforme, on suppose nulle la moyenne de l'erreur de troncature. De cette manière, les relations (3.6) et (3.26) des algorithmes de Cooley et Tukey et de Sande et Tukey, sont valables pour l'arrondi et la troncature sans déplacement. Dans l'hypothèse d'un déplacement à chaque étape, les relations (3.21) et (3.34) de ces deux algorithmes deviennent respectivement

$$2^{-2t} \left[(1/4 + \alpha/6).\ N^3 - (1/4 + \alpha/3)M) \ .\ N^2 \right] \qquad (5.1)$$

et

$$2^{-2t} \left[(1/4 + \alpha/72).N^3 - (1/4 + \alpha/12)\ N^2 + (\alpha/9).N \right] \qquad (5.2)$$

Elles sont également valables pour l'arrondi et la troncature.
Chacun des résultats présentés sont obtenus à partir de 50 signaux complexes dont les parties réelles et imaginaires sont des nombres aléatoires distribués uniformément dans l'intervalle $-1/\sqrt{2}$, $+1/\sqrt{2}$
La plupart de ces 50 estimations est proche des valeurs théoriques, quelques unes s'en écartent assez fort. Le total des carrés des erreurs ainsi que les valeurs prédites par la théorie sont indiqués à la fig.18 pour l'algorithme de Cooley et Tukey et à la fig.19 pour l'algorithme de Sande et Tukey. La longueur des nombres a été fixée à 15 bits, non compris le bit de signe; des résultats semblables ont été obtenus avec des longueurs différentes. Le coefficient α est égal à 4.
Le tableau 1 présente les différents résultats obtenus pour l'algorithme de Cooley et Tukey avec N = 16.

ALGORITHME DE COOLEY ET TUKEY		
	$\sum E[\|e(p)\|^2]$	
	théorique	expérimentale
Arrondi sans déplacement	$0.869.10^{-8}$	$0.934.10^{-8}$
Troncature sans déplacement	$0.869.10^{-8}$	$0.257.10^{-8}$
Arrondi avec déplacement	$0.184.10^{-5}$	$0.456.10^{-5}$
Troncature sans déplacement	$0.184.10^{-5}$	$0.831.10^{-5}$

Tableau 1.

Remarques

Les relations (3.6) et (3.26), (5.1) et (5.2), ainsi que les fig.18 et 19 montrent que les erreurs numériques produites par l'algorithme de Cooley et Tukey sont inférieures à celles produites par l'algorithme de Sande et Tukey s'il n'y a pas de déplacement. Par contre, s'il y en a un à chaque étape la situation est renversée. Cela s'explique par le fait que les multiplications sans erreur sont concentrées au début de l'algorithme de Cooley et Tykey et à la fin de l'algorithme de Sande et Tukey. Sans déplacement le facteur de propagation de la proposition 3.1 est plus important pour les premières étapes que pour les dernières.

Dans l'hypothèse d'un déplacement à chaque étape, ce facteur 2^{M-m} est multiplié par 2^{2m} et devient plus important dans les dernières étapes.

Le tableau 1 indique que pour la troncature, les estimations sont moins bonnes. Cela s'explique par l'hypothèse supplémentaire de la page 133. Si les nombres ne sont pas distribués de manière tout-à-fait uniforme, la moyenne de l'erreur n'est pas nulle, ce qui altère les résultats expérimentaux.

ALGORITHME DE COOLEY ET TUKEY

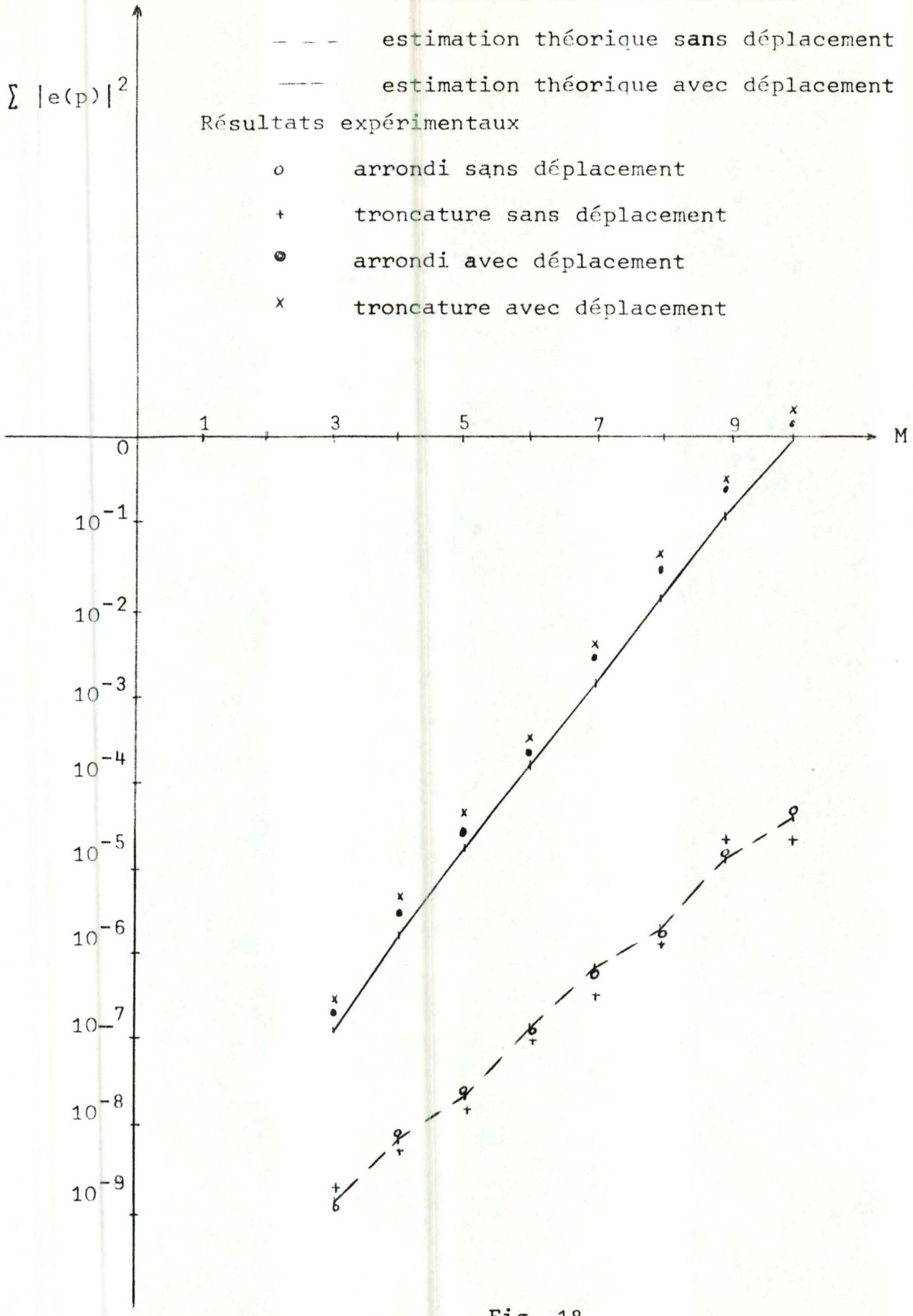

Fig. 18

ALGORITHME DE SANDE ET TUKEY

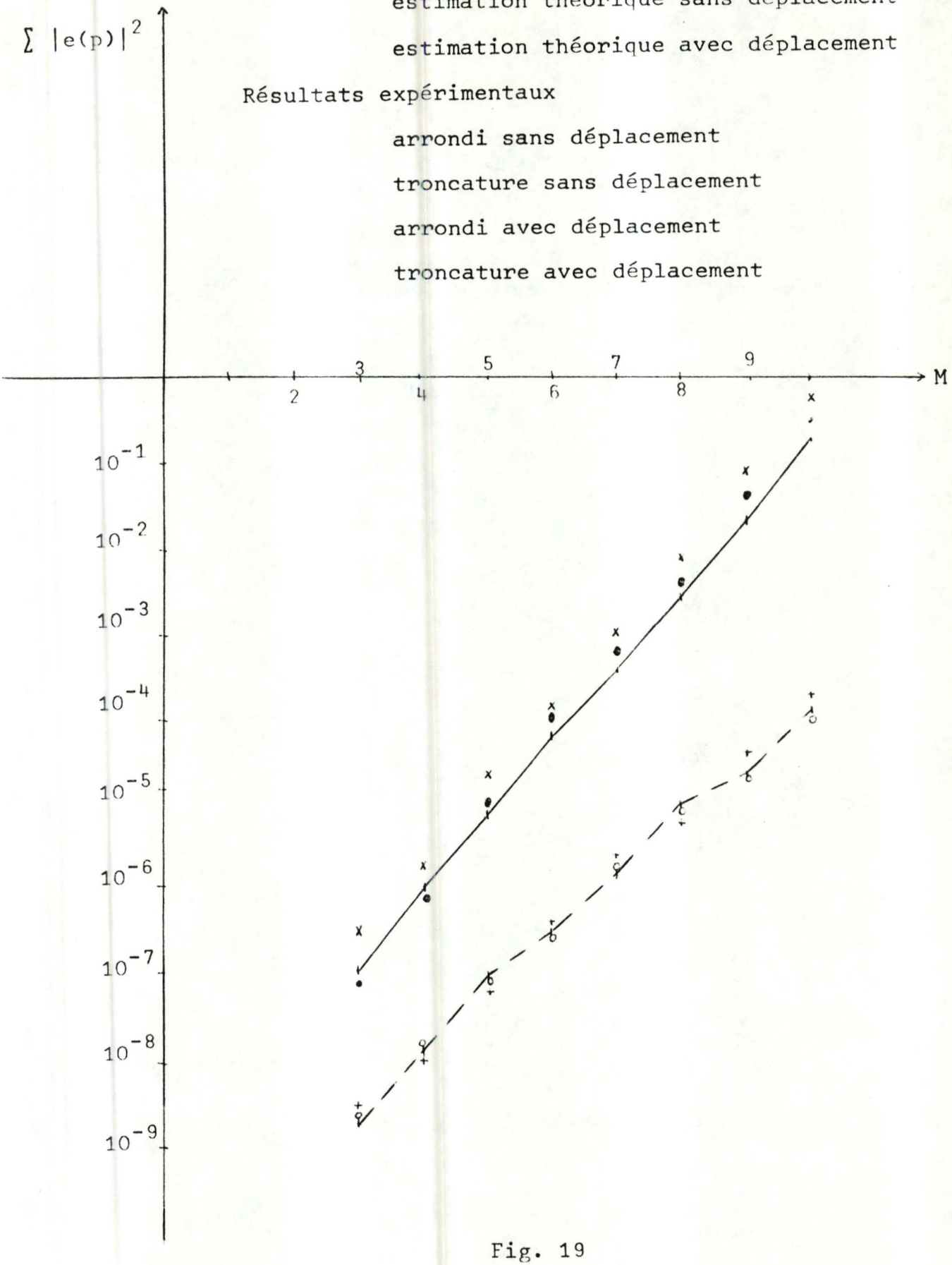

Fig. 19

ANNEXE

Programmes et sous-programmes FORTRAN

```
      PROGRAM FFTBIN
C
C     CE PROGRAMME EST DESTINE A COMPARER LES RESULTATS DE
C     TRAN-THONG ET LIU AVEC LE TOTAL DU CARRE DES ERREURS
C     OBTENUES EXPERIMENTALEMENT.
C
C     TABLE DES VARIABLES
C     *******************
C     NOM       TYPE                    ROLE
C     X1        (C16)           SIGNAL D'ENTREE COMPLEXE QUI EST
C                               TRANSFORME PAR LE SOUS-PROGRAMME
C                               FFTDIT POUR OBTENIR LES RESULTATS
C                               SUPPOSES EXACTS.
C     X2        (C16)           SIGNAL D'ENTREE COMPLEXE QUI EST
C                               TRANSFORME PAR LE SOUS-PROGRAMME
C                               DITBIN ET CONTIENT ENSUITE
C                               LES ERREURS NUMERIQUES
C     Y1        R8              SOMME DES CARRES DES RESULTATS
C     Y2        R8              SOMME DES CARRES DES ERREURS
C     NSR       R8              RAPPORT DU BRUIT AU SIGNAL
C     LIU       R8              SOMME THEORIQUE DES CARRES DES ERREURS
C     RAP       R8              RAPPORT THEORIQUE DU BRUIT AU SIGNAL
C     N         I4              NOMBRE D'ECHANTILLONS DE LA TFR
C     BIT       I4              LONGUEUR DES NOMBRES TRAITES PAR
C                               L'ARITHMETIQUE EN VIRGULE FIXE
C     MODE      I4              MODE D'ARRONDI OU DE TRONCATURE
C     ISHIF     I4              VAUT 1 S'IL Y A UN DEPLACEMENT
C                               A CHAQUE ETAPE ET O SINON
C     M         I4              NOMBRE D'ETAGES DE LA TFR
C     JERR      I4              DEVIENT DIFFERENT DE O SI
C                               UNE ERREUR EST DETECTEE
C
      COMPLEX*16 X1(1024),X2(1024),T
      REAL*8 Y1,Y2,NSR,S
      REAL*8 DELT,LIU,V
      REAL*8 RAP
      INTEGER BIT,BIT1,BIT2,OVF1,OVF2
C     LECTURE ET IMPRESSION DES DONNEES
      READ 70,IX1,IX2
      READ 72,MODE
      READ 72,BIT
      READ 70,K,J,IP
      PRINT 80
      IF(MODE.EQ.10) PRINT 81
      IF(MODE.EQ.16) PRINT 82
      PRINT 86,BIT
17    READ 72,ISHIF
      IF(ISHIF) 13,22,24
22    PRINT 83
      IDIST=10
```

```
      GOTO 18
24    PRINT 84
      IDIST=0
      CALL INIT(BIT,MODE)
18    DO 23 M=K,J,IP
      NUMIT=0
      N=2**M
      PRINT 85,N
C     CALCUL DE L'EVALUATION THEORIQUE
      V=2.D00*DFLOAT(BIT)
      DELT=2.D00**V
      IF(ISHIF) 15,25,26
C           SANS DEPLACEMENT
25    LIU=(N*N/6.-N+4/3)/(3*DELT)
      GOTO 27
C           AVEC DEPLACEMENT
26    LIU=(11*N**3-(3+10*M)*N*N)/(12*DELT)
27    PRINT 87,LIU
C     25 SIGNAUX DIFFERENTS SONT TRAITES DANS CHAQUE CAS
14    NUMIT=NUMIT+1
      IF(NUMIT.GT.25) GOTO 23
      PRINT 89,NUMIT
C     INITIALISATION DU SIGNAL D'ENTREE
      DO 10 I=1,N
      CALL RANDOM(IX1,Y1,N,IDIST)
      CALL RANDOM(IX2,Y2,N,IDIST)
      V=1.D00
      IF(ISHIF.EQ.0) V=V/DFLOAT(N)
      S=Y1**2+Y2**2
      IF(DSQRT(S).GE.V) GOTO 15
      T=DCMPLX(Y1,Y2)
      CALL ENTERC(T,X1(I),BIT1,BIT2,OVF1,OVF2)
       IF((OVF1.NE.0).OR.(OVF2.NE.0)) GOTO 11
      X2(I)=X1(I)
10    CONTINUE
C     CALCUL DES RESULTATS SUPPOSES CORRECTS
      CALL FFTDIT(X1,M)
C     CALCUL DES RESULTATS PAR L'ARITHMETIQUE EN VIRGULE FIXE
      CALL DITBIN(X2,M,IERR,ISHIF,BIT,MODE)
      IF(IERR.NE.0) GOTO 14
C     CORRECTION DES RESULTATS QUAND IL Y A EU DES DEPLACEMENTS
      IF(ISHIF.EQ.0) GOTO 19
      DO 16 I=1,N
      Y1=DREAL(X2(I))*N
      Y2=DIMAG(X2(I))*N
      X2(I)=DCMPLX(Y1,Y2)
16    CONTINUE
C     CALCUL ET IMPRESSION DES RESULTATS EXPERIMENTAUX
19    Y1=0.D00
      Y2=0.D00
```

```
      DO 12 I=1,N
      X2(I)=X2(I)-X1(I)
      Y1=Y1+CDABS(X1(I))**2
      Y2=Y2+CDABS(X2(I))**2
12    CONTINUE
      NSR=Y2/Y1
      V=Y2-LIU
      RAP=LIU/Y1
      S=NSR-RAP
      PRINT 88,Y1,Y2,V,RAP,NSR,S
      GOTO 14
C     IMPRESSION DES MESSAGES D'ERREUR
11    PRINT 90,I
      GOTO 14
15    PRINT 91,I
      GOTO 14
23    CONTINUE
      GOTO 17
70    FORMAT(3I5)
72    FORMAT(2I3)
80    FORMAT(1H1,49X,'ERREURS NUMERIQUES DANS LA T.F.R.',
     C2(/,50X,33('*')),5(/,1X),'*',/,1X,'* ',
     C'MODE DE DISCRETISATION :')
81    FORMAT(1X,'*',10X,'TRONCATURE',/,1X,'*')
82    FORMAT(1X,'*',10X,'ARRONDI ALEATOIRE',/,1X,'*')
83    FORMAT(/1X,2(/1X,'*'),'TRAITEMENT SANS DEPLACEMENT',/1X,'*')
84    FORMAT(/1X,2(/1X,'*'),'TRAITEMENT AVEC DEPLACEMENT',/1X,'*')
85    FORMAT(/1X,2(/1X,'*'),'NOMBRE D''ECHANTILLONS',I4,/1X,'*')
86    FORMAT(/1X,2(/1X,'*'),'LONGUEUR DES NOMBRES :',I3,
     C' BITS',/1X,'*')
87    FORMAT(//1X,2(/1X,'*'),'PUISSANCE THEORIQUE DU BRUIT',
     CD30.16,/1X,'*')
88    FORMAT(//1X,'PUISSANCE DU SIGNAL :',D32.16,/1X,
     C'PUISSANCE DU BRUIT :',D34.16,10X,'DIFFERENCE :',
     CD30.16,/1X,'RAPPORT BRUIT/SIGNAL :',/5X,
     C'THEORIQUE :',D33.16,/5X,'REEL :',D30.16,39X,
     C'DIFFERENCE',D30.16)
89    FORMAT(5(/,1X),'DONNEES NUMERO',I4,/,1X,14('*'))
90    FORMAT(5(/,1X),'OVERFLOW DE ENTERC POUR L''INDICE',I5)
91    FORMAT(5(/,1X),'LE MODULE DU ',I4,'-EME ELEMENT EST > OU = 1')
13    STOP
      END
```

```
      SUBROUTINE DATBIN(X,M,JERR,ISHIF,BIT,MODE)
C
C     CE SOUS-PROGRAMME EST DESTINE A CALCULER LA TFR
C     PAR L'ARITHMETIQUE EN VIRGULE FIXE
C
C     TABLE DES VARIABLES
C     *******************
C     NOM       TYPE                        ROLE
C
C     X       (C16)           SIGNAL D'ENTREE ET RESULTATS DE LA TFR
C     U,W       C16           COEFFICIENTS MULTIPLICATIFS
C     N         I4            NOMBRE D'ECHANTILLONS DE LA TFR
C     M         I4            NOMBRE D'ETAGES DE LA TFR
C     ISHIF     I4            VAUT 1 S'IL Y A UN DEPLACEMENT
C                             A CHAQUE ETAPE ET 0 SINON
C     MODE      I4            MODE D'ARRONDI OU DE TRONCATURE
C     BIT       I4            LONGUEUR DES NOMBRES TRAITES PAR
C                             L'ARITHMETIQUE EN VIRGULE FIXE
C
      COMPLEX*16 X(1),U,W,T,S
      REAL*8 X1,X2,V,R1,R2,EPS
      REAL*8 PI,ARO1,ARO2
      INTEGER BIT,BIT1,BIT2,OVF1,OVF2
      EPS=1.D-13
      N=2**M
      PI=4.D0*DATAN(1.D0)
C     INVERSION DU SIGNAL D'ENTREE
      CALL INVBIN(X,N)
C     CALCUL DES ETAGES DE LA TFR
      DO 30 L=1,M
C     EXECUTION DES DEPLACEMENTS
      IF(ISHIF.EQ.0) GOTO 13
      IERR=-1
      V=0.5D00
      MODESH=16
      CALL INIT(BIT,MODESH)
      DO 16 K=1,N
      X1=DREAL(X(K))
      X2=DIMAG(X(K))
      CALL PRODUI(X1,V,R1,ARO1,OVF1)
      CALL PRODUI(X2,V,R2,ARO2,OVF2)
      IF((OVF1.NE.0).OR.(OVF2.NE.0)) GOTO 11
      X(K)=DCMPLX(R1,R2)
16    CONTINUE
13    LE=2**L
      LE1=LE/2
      U=(1.D0,0.D0)
      W=DCMPLX(DCOS(PI/DFLOAT(LE1)),-DSIN(PI/DFLOAT(LE1)))
C     CALCUL DES "PAPILLONS"
      DO 20 J=1,LE1
```

```
      DO 10 I=J,N,LE
      IP=I+LE1
      IERR=0
      CALL PROCPL(X(IP),S,T,AR01,AR02,OVF1,OVF2,EPS)
      IF((OVF1.NE.0).OR.(OVF2.NE.0)) GOTO 12
      T=-T
      IERR=1
      CALL ADDCPL(X(I),T,X(IP),OVF1,OVF2)
      IF((OVF1.NE.0).OR.(OVF2.NE.0)) GOTO 12
      T=-T
      IERR=2
      CALL ADDCPL(X(I),T,S,OVF1,OVF2)
      IF((OVF1.NE.0).OR.(OVF2.NE.0)) GOTO 12
      X(I)=S
10    CONTINUE
      U=U*W
C     REINITIALISATION DES COEFFICIENTS MULTIPLICATIFS
      R1=DREAL(U)
      R2=DIMAG(U)
      IF(R2.GT.EPS) GOTO 17
      IF(R1.LT.0.D00) R1=-R1
      R1=1.D00
      R2=0.D00
17    IF(R1.GT.EPS) GOTO 20
      R1=0.D00
      R2=1.D00
      IF(R2.LT.0.D00) R2=-R2
20    CONTINUE
30    CONTINUE
15    RETURN
11    JERR=1
      PRINT 85
      PRINT 87,K,IERR
      RETURN
12    JERR=2
      PRINT 85
      PRINT 86,L,IP,I,IERR
      RETURN
85    FORMAT(5(/,1X),6('$'),'  OVERFLOW DANS DITBIN  ',6('$')//)
86    FORMAT(1X,'ETAPE NUMERO ',I4,10X,
     C'ELEMENT NUMERO ',I4,4X,I4,10X,'IERR = ',I4/)
87    FORMAT(1X,'ELEMENT NUMERO ',I4,10X,'IERR = ',I4/)
      END
```

```
      SUBROUTINE FFTDIT(X,M)
C
C     CE SOUS-PROGRAMME EST DESTINE A CALCULER LES RESULTATS
C     SUPPOSES EXACTS
C
C     TABLE DES VARIABLES
C     *******************
C     NOM       TYPE                        ROLE
C
C     X        (C16)          SIGNAL D'ENTREE ET RESULTATS DE LA TFR
C     U,W       C16           COEFFICIENTS MULTIPLICATIFS
C     N         I4            NOMBRES D'ECHANTILLONS DE LA TFR
C     M         I4            NOMBRE D'ETAGES DE LA TFR
C
      COMPLEX*16 X(1024),U,W,T
      REAL*8 R1,R2,EPS
      REAL*8 PI
      EPS=1.D-13
      N=2**M
C     INVERSION DU SIGNAL D'ENTREE
      CALL INVBIN(X,N)
      PI=4.D0*DATAN(1.D0)
C     CALCUL DES ETAGES DE LA TFR
      DO 30 L=1,M
      LE=2**L
      LE1=LE/2
      U=(1.D0,0.D0)
      W=DCMPLX(DCOS(PI/DFLOAT(LE1)),-DSIN(PI/DFLOAT(LE1)))
C     CALCUL DES "PAPILLONS"
      DO 20 J=1,LE1
      DO 10 I=J,N,LE
      IP=I+LE1
      T=X(IP)*U
      X(IP)=X(I)-T
   10 X(I)=X(I)+T
      U=U*W
C     REINITIALISATION DES COEFFICIENTS MULTIPLICATIFS
      R1=DREAL(U)
      R2=DIMAG(U)
      IF(R2.GT.EPS) GOTO 17
      IF(R1.LT.0.D00) R1=-R1
      R1=1.D00
      R2=0.D00
   17 IF(R1.GT.EPS) GOTO 20
      R1=0.D00
      R2=1.D00
      IF(R2.LT.0.D00) R2=-R2
   20 CONTINUE
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE PROCPL(X,Y,Z,ARQZ1,ARQZ2,OVFZ1,OVFZ2,EPS)
C
C     CE SOUS-PROGRAMME EST DESTINE A EFFECTUER LES PRODUITS PAR
C     LES COEFFICIENTS MULTIPLICATIFS DE LA TFR
C     IL SE BASE SUR LE SOUS-PROGRAMME PRODUI(X,Y,Z,ARQ,OVF)
C     AVEC:    Z    RESULTAT DU PRODUIT DE X PAR Y
C              ARQ  ERREUR D'ARRONDI OU DE TRONCATURE
C              OVF  INDIQUE LES DEPASSEMENTS
C
      COMPLEX*16 X,Y,Z
      REAL*8 ARQZ1,ARQZ2,X1,X2,Y1,Y2,Z1,Z2,T1,T2,ARQ1,ARQ2
      REAL*8 YD,EPS
      INTEGER OVFZ1,OVFZ2,OVF1,OVF2
      INTEGER BIT,OVF
      OVFZ1=0
      OVFZ2=0
      ARQZ1=0
      ARQZ2=0
      X1=DREAL(X)
      X2=DIMAG(X)
      Y1=DREAL(Y)
      Y2=DIMAG(Y)
C     TESTS POUR EVITER D'EFFECTUER DES MULTIPLICATIONS PAR
C     + OU - 1,+ OU - J
      IF(DABS(DABS(Y1)-1.D0).LT.EPS) GOTO 10
      IF(DABS(DABS(Y2)-1.D0).LT.EPS) GOTO 11
C     CALCUL DE LA PARTIE REELLE
      CALL PRODUI(X1,Y1,T1,ARQ1,OVF1)
      CALL PRODUI(X2,Y2,T2,ARQ2,OVF2)
      IF((OVF1.NE.0).OR.(OVF2.NE.0)) PRINT 83
      T2=-T2
      ARQZ1=ARQ1+ARQ2
      CALL ADD(T1,T2,Z1,OVFZ1)
      IF(OVFZ1.NE.0) PRINT 80
C     CALCUL DE LA PARTIE IMAGINAIRE
      CALL PRODUI(X1,Y2,T1,ARQ1,OVF1)
      CALL PRODUI(X2,Y1,T2,ARQ2,OVF2)
      ARQZ2=ARQ1+ARQ2
      CALL ADD(T1,T2,Z2,OVFZ2)
      IF(OVFZ2.NE.0) PRINT 81
      Z=DCMPLX(Z1,Z2)
      RETURN
C     MULTIPLICATIONS PAR + OU - 1 , + OU - J
10    IF(Y1) 12,13,14
12    T1=-X1
      T2=-X2
      Z=DCMPLX(T1,T2)
      RETURN
13    PRINT 84,Y1,Y2
      RETURN
```

```
      Z=DCMPLX(X1,X2)
      RETURN
   11 IF(Y2) 15,13,17
   15 T1=X2
      T2=-X1
      Z=DCMPLX(T1,T2)
      RETURN
   17 T1=-X2
      T2=X1
      Z=DCMPLX(T1,T2)
      RETURN
   80 FORMAT(/1X,5('$'),'OVERFLOW DU PRODUIT DS DREAL',5('$')/)
   81 FORMAT(/1X,5('$'),'OVERFLOW DU PRODUIT DS DIMAG',5('$')/)
   83 FORMAT(/1X,5('$'),'OVERFLOW DU PRODUIT DS CALCUL',5('$')/)
   84 FORMAT(/1X,5('$'),'INCOMPATIBILITE DU PTODUIT',5('$'),2D30.16/)
      END
```

```
      SUBROUTINE ADDCPL(X,Y,Z,OVF1,OVF2)
C
C     CE SOUS-PROGRAMME EST DESTINE A CALCULER LA SOMME
C     DE DEUX NOMBRES COMPLEXES A PARTIR DU SOUS-PROGRAMME
C     ADD(X,Y,Z,OVF)
C     AVEC     Z     SOMME DE X ET Y
C              OVF   INDIQUE LES DEPASSEMENTS
C
      COMPLEX*16 X,Y,Z
      REAL*8 X1,X2,Y1,Y2,Z1,Z2
      X1=DREAL(X)
      X2=DIMAG(X)
      Y1=DREAL(Y)
      Y2=DIMAG(Y)
C     CALCUL DE LA PARTIE REELLE
      CALL ADD(X1,Y1,Z1,OVF1)
      IF(OVF1.NE.0) PRINT 80
C     CALCUL DE LA PARTIE IMAGINAIRE
      CALL ADD(X2,Y2,Z2,OVF2)
      IF(OVF2.NE.0) PRINT 81
      Z=DCMPLX(Z1,Z2)
80    FORMAT(/1X,5('$'),'OVERFLOW D''ADD DS DREAL'5('$')/)
81    FORMAT(/1X,5('$'),'OVERFLOW D''ADD DS DIMAG'5('$')/)
      RETURN
      END
```

```
      SUBROUTINE ENTERC(X,XD,BIT1,BIT2,OVF1,OVF2)
C
C     CE SOUS-PROGRAMME PERMET DE REPRESENTER UN NOMBRE COMPLEXE
C     EN VIRGULE FIXE A L'AIDE DU SOUS-PROGRAMME ENTER(X,XD,BIT,OVF)
C     AVEC     X     NOMBRE A TRAITER
C              XD    RESULTAT EN VIRGULE FIXE
C              OVF   INDIQUE LES DEPASSEMENTS
C
      COMPLEX*16 X,XD
      REAL*8 X1,X2,X1D,X2D
      INTEGER BIT1,BIT2,OVF1,OVF2
      X1=DREAL(X)
      X2=DIMAG(X)
C     TRANSFORMATION DE LA PARTIE REELLE
      CALL ENTER(X1,X1D,BIT1,OVF1)
      IF(OVF1.NE.0) PRINT 80
C     TRANSFORMATION DE LA PARTIE IMAGINAIRE
      CALL ENTER(X2,X2D,BIT2,OVF2)
      IF(OVF2.NE.0) PRINT 81
      XD=DCMPLX(X1D,X2D)
80    FORMAT(/1X,5('$'),'OVERFLOW DS ENTER DE DREAL'5('$')/)
81    FORMAT(/1X,5('$'),'OVERFLOW DS ENTER DE DIMAG'5('$')/)
      RETURN
      END
```

```
      SUBROUTINE INVBIN(X,N)
C
C     CE SOUS-PROGRAMME PERMET D'INVERSER UN TABLEAU COMPLEXE X
C     DE N COMPOSANTES
C
      COMPLEX*16 X(1),T
      NV2=N/2
      NM1=N-1
      J=1
      DO 7 I=1,NM1
      IF(I.GE.J) GOTO 5
      T=X(J)
      X(J)=X(I)
      X(I)=T
5     K=NV2
6     IF(K.GE.J) GOTO 7
      J=J-K
      K=K/2
      GOTO 6
7     J=J+K
      RETURN
      END
```

```
      SUBROUTINE RANDOM(IX,YFL,N,IDIST)
C
C     CE SOUS-PROGRAMME PERMET D'OBTENIR LES NOMBRES ALEATOIRES
C
      REAL*8 YFL
      IX=IX*65539
      IF(IX.GE.0) GOTO 6
      IX=IX+2147483647+1
6     YFL=DFLOAT(IX)
      YFL=YFL*0.4656613E-09
      IF(IDIST.EQ.0) YFL=(YFL*2.D00-1.D00)/DSQRT(2.D00)
      IF(IDIST.EQ.10) YFL=(YFL*2.D00-1.D00)/(DSQRT(2.D00)*N)
      RETURN
      END
```

5 . 2 RESULTATS NUMERIQUES POUR LA VIRGULE FLOTTANTE

Ce chapitre présente la manière dont on a implémenté certains algorithmes de la Transformée de Fourier Rapide sur ordinateur et commente les résultats numériques obtenus pour les valeurs théoriques et expérimentales des rapports bruit à signal pour ces algorithmes. On propose en fin de chapitre une annexe dans laquelle sont présentées les différentes listes des programmes et sous-programmes FORTRAN utilisés.

Les résultats numériques de ce chapitre ont été obtenus sur l'ordinateur Siemens 4004, avec les hypothèses expérimentales suivantes :

H : | - seules les opérations arithmétiques exécutées en simple précision sont des sources d'erreur,
| - les valeurs obtenues et exprimées en double précision sont supposées exactes.

De plus, comme cet ordinateur utilise le mode de troncature pour les opérations arithmétiques en simple précision, seules les valeurs théoriques proposées en (4.11), (4.16) et (4.21) ont pu être comparées à des valeurs expérimentales. C'est pour cette raison que l'on a programmé uniquement les algorithmes de la Transformée de Fourier Rapide dans le cas où $N = 2^M$.

Ces algorithmes ont été programmés en langage FORTRAN :

SDIT Algorithme de Cooley et Tukey
avec exécution des opérations arithmétiques en simple précision,

DDIT Algorithme de Cooley et Tukey
avec exécution des opérations arithmétiques en double précision,

SDIF Algorithme de Sande et Tukey
avec exécution des opérations arithmétiques en simple précision,

DDIF Algorithme de Sande et Tukey
avec exécution des opérations arithmétiques en double précision.

Remarques :

1- Ces algorithmes n'ont pas été implémentés de manière à minimiser le temps de calcul ou la place utilisée en mémoire, mais plutôt de manière à s'identifier le mieux possible aux figures présentées au chapitre 1 :

SDIT et DDIT s'identifient à la figure 2,

SDIF et DDIF s'identifient à la figure 3.

2- Ces algorithmes ont été programmés de manière à pouvoir exécuter aussi bien la transformée inverse que la transformée directe.

3- A l'entrée de ces algorithmes, on suppose que le tableau des résultats a été initialisé au tableau des données.

Les hypothèses H, énoncées ci-dessus, conduisent à la démarche suivante pour comparer les valeurs théoriques et expérimentales du rapport bruit à signal :
soient X,Y,W,Z quatre tableaux de nombres complexes en double précision :

- on évalue les espérances et les variances des erreurs locales en double précision,
- on évalue les bornes théoriques en double précision,
- le tableau Y est la transformée du tableau X, effectuée en simple précision (algorithme SDIT ou SDIF),
- le tableau Z est la transformée du tableau W = X, effectuée en double précision (algorithme DDIT ou DDIF),
- on calcule la valeur théorique du rapport bruit à signal en utilisant les valeurs des tableaux W et Z et en exécutant les opérations arithmétiques en double précision,
- on calcule la valeur expérimentale du rapport bruit à signal, à savoir

$$\frac{\sum_{k=0}^{N-1} |Z(k) - Y(k)|^2}{\sum_{k=0}^{N-1} |Z(k)|^2} ,$$

en exécutant les opérations arithmétiques en double précision.

A ce stade, il faut encore remarquer que le problème, tel qu'il a été posé au chapitre 4, suppose que les données de l'algorithme sont représentées exactement sur l'ordinateur, de même que

les facteurs multiplicatifs intervenant dans l'algorithme. On doit donc :

- considérer le tableau X en simple précision et le tableau W en double précision,

- évaluer les facteurs multiplicatifs en double précision.

Toutes ces considérations ont permis d'établir trois programmes FORTRAN :

1- Le programme DITDIF traite le cas général :
pour M=2,3,...,9, on évalue

- les bornes données en (4.13),

- les valeurs théoriques des rapports bruit à signal donnés en (4.11) et (4.16),

- les valeurs expérimentales des rapports bruit à signal pour un algorithme de Cooley et Tukey et pour un algorithme de Sande et Tukey quand $N = 2^M$.

Les résultats de ce programme sont présentés dans les tableaux 2 et 3.

2- Le programme FFT traite le problème particulier (4.18) :
on considère des données complexes dont les parties réelles et imaginaires forment des suites pseudo-aléatoires distribuées uniformément entre -1 et +1.
Pour M=2,3,...,9, on évalue

- la valeur théorique du rapport bruit à signal donné en (4.21),

- la moyenne de 50 valeurs expérimentales du rapport bruit à signal pour l'algorithme de Cooley et Tukey et pour l'algorithme

de Sande et Tukey quand $N = 2^M$.

Les résultats de ce programme sont présentés dans le tableau 4.

Remarque :

Pour chaque valeur de M, les 50 valeurs expérimentales pour chaque algorithme sont peu dispersées. Aussi présente-t-on ici uniquement la moyenne de ces 50 valeurs, ce qui permet d'alléger fortement le texte, sans toutefois perdre trop d'information.

3- Le programme ARROND traite le cas d'arrondi :
pour M=2,3,...,9, on évalue

- les bornes données en (4.12),
- la valeur théorique du rapport bruit à signal donné en (4.20).

Les résultats de ce programme sont comparés à ceux des deux autres dans les tableaux 5 et 6.

L'analyse des tableaux 2,3 et 4 conduit aux considérations suivantes :

- les valeurs expérimentales sont comprises entre les bornes théoriques, sauf dans le cas où N=4,
- les valeurs expérimentales sont du même ordre de grandeur que les valeurs théoriques, mais elles sont affectées d'un coéfficient généralement supérieur,
- les valeurs expérimentales pour l'algorithme de Sande et Tukey sont généralement plus élevées que celles pour l'algorithme

de Cooley et Tukey, quoique la théorie prévoie le contraire,

- toutes ces valeurs croissent avec le nombre de points N; en parcourant les tableaux de M=2 à M=9, on passe d'un ordre de grandeur au suivant,

- pour N=4, les deux algorithmes sont les mêmes, ce qui explique le fait que les valeurs expérimentales soient rigoureusement les mêmes,

- la théorie développée pour le problème particulier (4.18) semble mieux épouser la réalité que la théorie développée dans le cas général.

A l'analyse des tableaux 5 et 6, on constate que la théorie prévoit des valeurs pour le rapport bruit à signal nettement moins élevées dans le cas d'arrondi que dans le cas de troncature.

Parmi les explications possibles de l'écart entre les valeurs expérimentales et les valeurs théoriques, on note que :

- la valeur de p_o a peut-être été mal évaluée,

- les effets de second ordre ont été négligés dans la théorie développée au chapitre 4,

- les hypothèses générales d'indépendance à propos des erreurs locales ne sont pas vérifiées,

- les valeurs considérées comme exactes ne le sont pas en réalité puisque la double précision est également source d'erreurs,

- on n'a pas tenu compte de toutes les multiplications sans erreur pour établir les expressions (4.11) et (4.16).

N	BORNES		VALEURS	VALEURS
	INFERIEURES	SUPERIEURES	THEORIQUES	EXPERIMENTALES
4	.3735428746D-13	.3735428746D-13	.3735428746D-13	.4008281573D-12
8	.6305043688D-13	.2102753356D-12	.2090486123D-12	.1180268259D-12
16	.9342592342D-13	.5162975287D-12	.2439830681D-12	.1186920972D-12
32	.1284807470D-12	.9554208667D-12	.3142197481D-12	.3304829053D-12
64	.1682149078D-12	.1527645349D-11	.3746213838D-12	.2948374578D-12
128	.2126284058D-12	.2232970977D-11	.4391836430D-12	.6029845392D-12
256	.2617212408D-12	.3071397749D-11	.5087739677D-12	.9598387510D-12
512	.3154934130D-12	.4042925667D-11	.5839905678D-12	.1654666552D-11

Tableau 2. Tableau comparatif des rapports bruit à signal dans le cas général pour l'algorithme de Cooley et Tukey.

N	BORNES		VALEURS	VALEURS
	INFERIEURES	SUPERIEURES	THEORIQUES	EXPERIMENTALES
4	.3735428746D-13	.3735428746D-13	.3735428746D-13	.4008281573D-12
8	.6305043688D-13	.2102753356D-12	.1686414842D-12	.1314349838D-12
16	.9342592342D-13	.5162975287D-12	.1913735459D-12	.1395671287D-12
32	.1284807470D-12	.9554208667D-12	.2283043369D-12	.3796501186D-12
64	.1682149078D-12	.1527645349D-11	.3009494837D-12	.4259324767D-12
128	.2126284058D-12	.2232970977D-11	.3408688551D-12	.6678435809D-12
256	.2617212408D-12	.3071397749D-11	.4128168867D-12	.9680440401D-12
512	.3154934130D-12	.4042925667D-11	.4674608783D-12	.1718619533D-11

Tableau 3. Tableau comparatif des rapports bruit à signal dans le cas général pour l'algorithme de Sande et Tukey.

N	VALEURS THEORIQUES	VALEURS EXPERIMENTALES DIT	DIF
4	.1298114004D-12	.1459438982D-12	.1459438982D-12
8	.2516537193D-12	.3173777318D-12	.3474280863D-12
16	.2404350981D-12	.5285683278D-12	.6219354588D-12
32	.4845860558D-12	.6888954767D-12	.8901062311D-12
64	.7801239795D-12	.8266575789D-12	.1061714493D-11
128	.1118121642D-11	.9672078757D-12	.1345874405D-11
256	.1294596433D-11	.1342969910D-11	.1872969349D-11
512	.1726440802D-11	.2115981261D-11	.2622420141D-11

Tableau 4. Tableau comparatif des rapports bruit à signal pour le problème particulier (4.18).

N	BORNES INFERIEURES		BORNES SUPERIEURES	
	ARRONDI	TRONCATURE	ARRONDI	TRONCATURE
4	.8168737584D-14	.3735428746D-13	.8168737584D-14	.3735428746D-13
8	.1225310637D-13	.6305043688D-13	.2995203781D-13	.2102753356D-12
16	.1633747516D-13	.9342592342D-13	.5173533803D-13	.5162975287D-12
32	.2042184396D-13	.1284807470D-12	.7351863826D-13	.9554208667D-12
64	.2450621275D-13	.1682149078D-12	.9530193848D-13	.1527645349D-11
128	.2859058154D-13	.2126284058D-12	.1170852387D-12	.2232970977D-11
256	.3267495033D-13	.2617212408D-12	.1388685389D-12	.3071397749D-11
512	.3675931913D-13	.3154934130D-12	.1606518391D-12	.4042925667D-11

Tableau 5. Tableau comparatif des bornes pour le rapport bruit à signal en cas d'arrondi et de troncature.

N	VALEURS THEORIQUES	
	ARRONDI	TRONCATURE
4	.4084368792D-14	.1298114004D-12
8	.1225310637D-13	.2516537193D-12
16	.1633747516D-13	.2404350981D-12
32	.3812077539D-13	.4845860558D-12
64	.4220514418D-13	.7801239795D-12
128	.6398844441D-13	.1118121642D-11
256	.6807281320D-13	.1294596433D-11
512	.8985611343D-13	.1726440802D-11

Tableau 6. Tableau comparatif des valeurs théoriques du rapport bruit à signal pour le problème (4.18) en cas d'arrondi et de troncature.

En conclusion de ce chapitre, on peut estimer cette théorie assez satisfaisante puisqu'elle permet de trouver l'ordre de grandeur du rapport bruit à signal, qui est en quelque sorte une estimation moyenne du carré du module de l'erreur relative sur un élément du résultat à la sortie de l'algorithme. On a en effet pour le problème particulier (4.18) :

$$\frac{\sum_{k=0}^{N-1} E\{|y'(k)-y(k)|^2\}}{\sum_{k=0}^{N-1} E\{|y(k)|^2\}} = \frac{\frac{1}{N}.\sum_{k=0}^{N-1} E\{|y'(k)-y(k)|^2\}}{N.N_0}$$

$$\simeq \frac{|y'(.) - y(.)|^2}{|y(.)|^2} \quad .$$

ANNEXE

Programmes et sous-programmes FORTRAN

```
      PROGRAM DITDIF
C
C
C
C     CE PROGRAMME COMPARE LES VALEURS THEORIQUES ET EXPERIMENTALES
C     DU RAPPORT BRUIT A SIGNAL POUR LES ALGORITHMES FFT DANS LE CAS OU
C     N = 2**NU.
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE  TYPE          SIGNIFICATION
C
C     W         COMPLEX*16    TABLEAU D'ENTREE DE LA TRANSFORMEE
C     Z         COMPLEX*16    TABLEAU DE SORTIE DE LA TRANSFORMEE
C     X         COMPLEX*8     TABLEAU D'ENTREE DE LA TRANSFORMEE
C     Y         COMPLEX*8     TABLEAU DE SORTIE DE LA TRANSFORMEE
C     N         INTEGER       NOMBRE DE POINTS DE LA TRANSFORMEE
C     T         INTEGER       LE NOMBRE DE CHIFFRES DE LA MANTISSE
C     INVERS    LOGICAL       SI VRAI, ALORS TRANSFORMEE INVERSE
C                             SI FAUX, ALORS TRANSFORMEE DIRECTE
C     PI        REAL*8        LE NOMBRE PI
C     XI        REAL*8        LA BASE DE L'ORDINATEUR
C     DELTAF    REAL*8        ESPERANCE D'UNE ERREUR DE MULTIPLICATION
C     DELTAP    REAL*8        ESPERANCE D'UNE ERREUR D'ADDITION
C     DELT2F    REAL*8        VARIANCE D'UNE ERREUR DE MULTIPLICATION
C     DELT2P    REAL*8        VARIANCE D'UNE ERREUR D'ADDITION
C     NSRDTI    REAL*8        VALEUR THEORIQUE DU RAPPORT BRUIT A SIGNAL
C                             POUR LA DIT
C     NSRDFI    REAL*8        VALEUR THEORIQUE DU RAPPORT BRUIT A SIGNAL
C                             POUR LA DIF
C     NSRDIT    REAL*8        VALEUR EXPERIMENTALE DU RAPPORT BRUIT A
C                             SIGNAL POUR LA DIT
C     NSRDIF    REAL*8        VALEUR EXPERIMENTALE DU RAPPORT BRUIT A
C                             SIGNAL POUR LA DIF
C     BI        REAL*8        BORNE INFERIEURE DONNEE EN (4.13)
C     BS        REAL*8        BORNE SUPERIEURE DONNEE EN (4.13)
C
C
C
      COMPLEX*8 X(1024),Y(1024)
      COMPLEX*16 W(1024),Z(1024)
      REAL*8 E,F,NSRDIT,NSRDIF,DELTAF,DELTAP,DELT2F,DELT2P,XI,LN,PO,PI
      REAL*8 D,G,H,NSRDTI,NSRDFI
      REAL*8 BI,BS
      INTEGER T,FI
      DIMENSION FI(1024)
      PI=3.1415926535897932
C
C     CALCUL DES ESPERANCES ET DES VARIANCES DES ERREURS LOCALES
```

```
C
      XI=16.
      LN=DLOG(XI)
      T=6
      PO=0.7
      DELTAF=-(XI-1)/(2*LN*(XI**T))
      DELTAP=(1-PO)*DELTAF
      DELT2F=(XI**2-1)/(6*LN*(XI**(2*T)))
      DELT2P=(1-PO)*DELT2F
      DELT2F=DELT2F-(DELTAF**2)
      DELT2P=DELT2P-(DELTAP**2)
      DO 009 NU=2,9
      N=2**NU
      DO 001 K=1,N
      FI(K)=IFT(K-1,NU)
001   CONTINUE
      READ(25)(X(J),J=1,N)
C
C     CALCUL DES BORNES
C
      BI=NU*DELT2P+(NU**2)*(DELTAP**2)
      BS=(NU-2)*DELT2F+(2*NU-2)*DELT2P+((NU-2)*DELTAF+(2*NU-2)*DELTAP)**
     12
C
C     CALCUL D'UN RAPPORT BRUIT A SIGNAL THEORIQUE ET DE SON
C     CORRESPONDANT EXPERIMENTAL POUR LA DIT
C
      DO 003 J=1,N
      Y(J)=X(J)
      W(J)=CDBLE(X(J))
      Z(J)=W(J)
003   CONTINUE
      CALL SDIT(NU,X,Y,N,PI,.FALSE.)
      CALL DDIT(NU,W,Z,N,PI,.FALSE.)
      D=0.
      E=0.
      F=0.
      G=0.
      H=0.
      DO 004 K=1,N
      D=D+FI(K)*(CDABS(W(K)))**2
      E=E+(CDABS(CDBLE(Y(K))-Z(K)))**2
      F=F+(CDABS(Z(K)))**2
      G=G+((FI(K)*(DELTAF+DELTAP)+NU*DELTAP)**2)*((CDABS(W(K)))**2)
      H=H+(CDABS(W(K)))**2
004   CONTINUE
      NSRDTI=NU*DELT2P+(DELT2P+DELT2F)*D/H+G/H
      NSRDIT=E/F
C
C     CALCUL D'UN RAPPORT BRUIT A SIGNAL THEORIQUE ET DE SON
```

```
101 C       CORRESPONDANT EXPERIMENTAL POUR LA DIF
102 C
103         DO 005 J=1,N
104         Y(J)=X(J)
105         Z(J)=W(J)
106  005    CONTINUE
107         CALL SDIF(NU,X,Y,N,PI,.FALSE.)
108         CALL DDIF(NU,W,Z,N,PI,.FALSE.)
109         D=0.
110         E=0.
111         F=0.
112         G=0.
113         DO 006 K=1,N
114         D=D+FI(K)*(CDABS(Z(K)))**2
115         E=E+(CDABS(CDBLE(Y(K))-Z(K)))**2
116         F=F+(CDABS(Z(K)))**2
117         G=G+((FI(K)*(DELTAF+DELTAP)+NU*DELTAP)**2)*((CDABS(Z(K)))**2)
118  006    CONTINUE
119         NSRDIF=E/F
120         NSRDFI=NU*DELT2P+(DELT2P+DELT2F)*D/F+G/F
121 C
122 C       IMPRESSIONS
123 C
124         PRINT 007,N,BI,BS,NSRDTI,NSRDIT,NSRDFI,NSRDIF
125  007    FORMAT(1H1,'NOMBRE DE POINTS',I4,////,1X,'BORNE INFERIEURE ',D40.1
126        16,///,1X,'BORNE SUPERIEURE ',D40.16,////,1X,20X,'VALEURS THEORIQUE
127        2S ',20X,'VALEURS EXPERIMENTALES ',///,1X,'DIT',2D40.16,///,1X,'DIF
128        3',2D40.16,///)
129  009    CONTINUE
130         STOP
131         END
```

```
      PROGRAM FFT
C
C
C
C     CE PROGRAMME COMPARE LES VALEURS THEORIQUES ET EXPERIMENTALES
C     DU RAPPORT BRUIT A SIGNAL POUR LES ALGORITHMES FFT DANS LE CAS OU
C     N = 2**NU.
C
C     ON CONSIDERE LE PROBLEME PARTICULIER DU BRUIT BLANC :
C     LE FICHIER 25 CONTIENT DES NOMBRES COMPLEXES DONT LES PARTIES
C     REELLES ET IMAGINAIRES SONT DISTRIBUEES UNIFORMEMENT SUR
C     L'INTERVALLE  (-1, +1).
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE  TYPE          SIGNIFICATION
C
C     W         COMPLEX*16    TABLEAU D'ENTREE DE LA TRANSFORMEE
C     Z         COMPLEX*16    TABLEAU DE SORTIE DE LA TRANSFORMEE
C     X         COMPLEX*8     TABLEAU D'ENTREE DE LA TRANSFORMEE
C     Y         COMPLEX*8     TABLEAU DE SORTIE DE LA TRANSFORMEE
C     N         INTEGER       NOMBRE DE POINTS DE LA TRANSFORMEE
C     T         INTEGER       LE NOMBRE DE CHIFFRES DE LA MANTISSE
C     INVERS    LOGICAL       SI VRAI, ALORS TRANSFORMEE INVERSE
C                             SI FAUX, ALORS TRANSFORMEE DIRECTE
C     PI        REAL*8        LE NOMBRE PI
C     XI        REAL*8        LA BASE DE L'ORDINATEUR
C     DELTAF    REAL*8        ESPERANCE D'UNE ERREUR DE MULTIPLICATION
C     DELTAP    REAL*8        ESPERANCE D'UNE ERREUR D'ADDITION
C     DELT2F    REAL*8        VARIANCE D'UNE ERREUR DE MULTIPLICATION
C     DELT2P    REAL*8        VARIANCE D'UNE ERREUR D'ADDITION
C     NSRDIT    REAL*8        D'ABORD : VALEUR THEORIQUE DU RAPPORT
C                             PUIS    : MOYENNE DES VALEURS EXPERIMENTA-
C                                       LES DU RAPPORT
C                             BRUIT A SIGNAL POUR LA DIT
C     NSRDIF    REAL*8        D'ABORD : VALEUR THEORIQUE DU RAPPORT
C                             PUIS    : MOYENNE DES VALEURS EXPERIMENTA-
C                                       LES DU RAPPORT
C                             BRUIT A SIGNAL POUR LA DIF
C
C
C
      COMPLEX*8 X(1024),Y(1024)
      COMPLEX*16 W(1024),Z(1024)
      REAL*8 E,F,NSRDIT,NSRDIF,DELTAF,DELTAP,DELT2F,DELT2P,XI,LN,PO,PI
      INTEGER T
      PI=3.1415926535897932
C
C     CALCUL DES ESPERANCES ET DES VARIANCES DES ERREURS LOCALES
```

```
C
      XI=16.
      LN=DLOG(XI)
      T=6
      PO=0.7
      DELTAF=-(XI-1)/(2*LN*(XI**T))
      DELTAP=(1-PO)*DELTAF
      DELT2F=(XI**2-1)/(6*LN*(XI**(2*T)))
      DELT2P=(1-PO)*DELT2F
      DELT2F=DELT2F-(DELTAF**2)
      DELT2P=DELT2P-(DELTAP**2)
      DO 009 NU=2,9
      N=2**NU
C
C     CALCUL DE LA VALEUR THEORIQUE DU RAPPORT BRUIT A SIGNAL
C
      NSRDIT=((NU-3)/2+2/N)*DELT2F+((3*NU-3)/2+2/N)*DELT2P+(NU**2)/4*((D
     1ELTAF**2)+6*DELTAF*DELTAP+9*DELTAP**2)-NU/4*(3*DELTAF**2+14*DELTAP
     2*DELTAF+11*DELTAP**2)+((DELTAF+DELTAP)**2)/2
      NSRDIF=NSRDIT
C
C     IMPRESSIONS
C
      PRINT 008,N
  008 FORMAT(1X,'NOMBRE DE POINTS = ',I4,//,1X,23('*')///)
      PRINT 010,NSRDIT,NSRDIF
  010 FORMAT(1X,'VALEURS THEORIQUES  ',2(D40.16)///)
C
C     CALCUL DE LA MOYENNE DE 50 RAPPORTS BRUIT A SIGNAL EXPERIMENTAUX
C
      NSRDIT=0.
      NSRDIF=0.
      DO 001 I=1,50
      READ(25)(X(J),J=1,N)
      DO 003 J=1,N
      Y(J)=X(J)
      W(J)=CDBLE(X(J))
      Z(J)=W(J)
  003 CONTINUE
      CALL SDIT(NU,X,Y,N,PI,.FALSE.)
      CALL DDIT(NU,W,Z,N,PI,.FALSE.)
      E=0.
      F=0.
      DO 004 K=1,N
      E=E+(CDABS(CDBLE(Y(K))-Z(K)))**2
      F=F+(CDABS(Z(K)))**2
  004 CONTINUE
      NSRDIT=NSRDIT+E/F
      DO 005 J=1,N
      Y(J)=X(J)
```

```
      Z(J)=W(J)
005   CONTINUE
      CALL SDIF(NU,X,Y,N,PI,.FALSE.)
      CALL DDIF(NU,W,Z,N,PI,.FALSE.)
      E=0.
      F=0.
      DO 006 K=1,N
      E=E+(CDABS(CDBLE(Y(K))-Z(K)))**2
      F=F+(CDABS(Z(K)))**2
006   CONTINUE
      NSRDIF=NSRDIF+E/F
001   CONTINUE
      NSRDIT=NSRDIT/DFLOAT(50)
      NSRDIF=NSRDIF/DFLOAT(50)
C
C     IMPRESSIONS
C
      PRINT 007,NSRDIT,NSRDIF
007   FORMAT(1X,22X,2D40.16/)
      REWIND 25
009   CONTINUE
      STOP
      END
```

```
      PROGRAM ARROND
C
C
C
C     CE PROGRAMME CALCULE LES BORNES INFERIEURES ET SUPERIEURES (4.12)
C     POUR LE RAPPORT BRUIT A SIGNAL A LA SORTIE DE L'ALGORITHME DE LA
C     TRANSFORMEE DE FOURIER RAPIDE LORSQUE LE MODE D'ARRONDI EST
C     UTILISE. IL CALCULE EGALEMENT LA VALEUR THEORIQUE DU RAPPORT
C     BRUIT A SIGNAL PROPOSE EN (4.20).
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE  TYPE          SIGNIFICATION
C
C     N         INTEGER       NOMBRE DE POINTS DE LA TRANSFORMEE
C     T         INTEGER       LE NOMBRE DE CHIFFRES DE LA MANTISSE
C     XI        REAL*8        LA BASE DE L'ORDINATEUR
C     DELT2F    REAL*8        VARIANCE D'UNE ERREUR DE MULTIPLICATION
C     DELT2P    REAL*8        VARIANCE D'UNE ERREUR D'ADDITION
C     BI        REAL*8        BORNE INFERIEURE DONNEE EN (4.12)
C     BS        REAL*8        BORNE SUPERIEURE DONNEE EN (4.12)
C     NSR       REAL*8        VALEUR THEORIQUE DU RAPPORT BRUIT A SIGNAL
C                             DONNEE EN (4.20)
C
C
C
      REAL*8 BI,BS,DELT2F,DELT2P,XI,P0,LN,NSR
      INTEGER T
C
C     CALCUL DES VARIANCES DES ERREURS LOCALES
C
      P0=0.7
      XI=16.
      LN=DLOG(XI)
      T=6
      DELT2F=(XI**2-1)/(24*LN*(XI**(2*T)))
      DELT2P=(1-P0)*DELT2F
C
C     IMPRESSIONS
C
      PRINT 001
  001 FORMAT(1H1,'NBRE DE POINTS',20X,'BORNE INFERIEURE',20X,'BORNE SUPE
     1RIEURE',20X,'VALEUR THEORIQUE'////)
      DO 009 NU=2,9
      N=2**NU
C
C     CALCUL DES BORNES
C
      BI=NU*DELT2P
```

```
      BS=(NU-2)*DELT2F+(2*NU-2)*DELT2P
C
C     CALCUL DE LA VALEUR THEORIQUE DU RAPPORT BRUIT A SIGNAL
C     EN CAS DE BRUIT BLANC
C
      NSR=((NU-3)/2+2/N)*DELT2F+((3*NU-3)/2+2/N)*DELT2P
C
C     IMPRESSIONS
C
      PRINT 002,N,BI,BS,NSR
  002 FORMAT(1X,I10,3D40.16/)
  009 CONTINUE
      STOP
      END
```

```
      SUBROUTINE DDIT(NU,X,Y,N,PI,INVERS)
C
C
C
C     CETTE SOUS-ROUTINE CALCULE LA TRANSFORMEE DE FOURIER DISCRETE
C     EN DOUBLE PRECISION PAR L'ALGORITHME DE COOLEY ET TUKEY
C     DANS LE CAS OU  N = 2**NU
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE   TYPE          SIGNIFICATION
C
C     X          COMPLEX*16    TABLEAU D'ENTREE DE LA  TRANSFORMEE
C     Y          COMPLEX*16    TABLEAU DE SORTIE DE LA TRANSFORMEE
C     N          INTEGER       NOMBRE DE POINTS DE LA TRANSFORMEE
C     PI         REAL*8        LE NOMBRE PI
C     INVERS     LOGICAL       SI VRAI, ALORS TRANSFORMEE INVERSE
C                              SI FAUX, ALORS TRANSFORMEE DIRECTE
C
C
C
      COMPLEX*16 X(1024),Y(1024),T,E
      REAL*8 ARG,DC,DS,PI
      LOGICAL INVERS
      N2=1
C
C     ON INVERSE BINAIREMENT L'ORDRE DES ENTREES
C
      N1=N-1
      DO 101 K=2,N1
      K1=K-1
      I=IBITR(K1,NU)+1
      IF(I.LE.K)  GOTO 101
      Y(K)=X(I)
      Y(I)=X(K)
  101 CONTINUE
      DO 103 NU1=1,NU
      K=1
      KN2=K+N2
      K1=K-1
  108 DO 104 I=1,N2
      L=LDIT(NU1,K1,N,INVERS)
      IF (L.NE.0)  GOTO 105
C
C     LE FACTEUR MULTIPLICATIF = 1
C
      T=Y(KN2)
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
```

```
      GOTO 106
  105 IF(IABS(L).NE.N/2) GOTO 111
C
C     LE FACTEUR MULTIPLICATIF = -1
C
      T=-Y(KN2)
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
      GOTO 106
  111 IF(IABS(L).NE.3*N/4) GOTO 112
C
C     LE FACTEUR MULTIPLICATIF = J
C
      E=DCMPLX(0.D0,1.D0)
      IF(INVERS) E=DCMPLX(0.D0,-1.D0)
      T=Y(KN2)*E
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
      GOTO 106
  112 IF(IABS(L).NE.N/4) GOTO 107
C
C     LE FACTEUR MULTIPLICATIF = -J
C
      E=DCMPLX(0.D0,-1.D0)
      IF(INVERS) E=DCMPLX(0.D0,1.D0)
      T=Y(KN2)*E
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
      GOTO 106
C
C     LE FACTEUR MULTIPLICATIF EST QUELCQUONQUE
C
  107 ARG=2*PI*DFLOAT(L)/DFLOAT(N)
      DC=DCOS(ARG)
      DS=-DSIN(ARG)
      E=DCMPLX(DC,DS)
      T=Y(KN2)*E
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
  106 K=K+1
      K1=K-1
      KN2=KN2+1
  104 CONTINUE
      K=K+N2
      K1=K-1
      KN2=KN2+N2
      IF(K.LE.N)  GOTO 108
      N2=N2*2
  103 CONTINUE
      IF(.NOT.INVERS) GOTO 100
```

```
101 C
102 C       POUR CALCULER LA TRANSFORMEE INVERSE, IL FAUT DIVISER LES
103 C       RESULTATS PAR N
104 C
105         DO 115 I=1,N
106         Y(I)=Y(I)/DFLOAT(N)
107   115   CONTINUE
108   10C   RETURN
109         END
```

```
      SUBROUTINE SDIT(NU,X,Y,N,PI,INVERS)
C
C
C
C     CETTE SOUS-ROUTINE CALCULE LA TRANSFORMEE DE FOURIER DISCRETE
C     EN SIMPLE PRECISION PAR L'ALGORITHME DE COOLEY ET TUKEY
C     DANS LE CAS OU  N = 2**NU
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE   TYPE          SIGNIFICATION
C
C     X          COMPLEX*8     TABLEAU D'ENTREE DE LA TRANSFORMEE
C     Y          COMPLEX*8     TABLEAU DE SORTIE DE LA TRANSFORMEE
C     N          INTEGER       NOMBRE DE POINTS DE LA TRANSFORMEE
C     PI         REAL*8        LE NOMBRE PI
C     INVERS     LOGICAL       SI VRAI, ALORS TRANSFORMEE INVERSE
C                              SI FAUX, ALORS TRANSFORMEE DIRECTE
C
C
C
      COMPLEX*8 X(1024),Y(1024),T
      COMPLEX*16 E
      REAL*8 DC,DS,ARG,PI
      LOGICAL INVERS
      N2=1
C
C     ON INVERSE BINAIREMENT L'ORDRE DES ENTREES
C
      N1=N-1
      DO 201 K=2,N1
      K1=K-1
      I=IBITR(K1,NU)+1
      IF(I.LE.K)  GOTO 201
      Y(K)=X(I)
      Y(I)=X(K)
  201 CONTINUE
      DO 203 NU1=1,NU
      K=1
      KN2=K+N2
      K1=K-1
  208 DO 204 I=1,N2
      L=LDIT(NU1,K1,N,INVERS)
      IF (L.NE.0)  GOTO 205
C
C     LE FACTEUR MULTIPLICATIF = 1
C
      T=Y(KN2)
      Y(KN2)=Y(K)-T
```

```
      Y(K)=Y(K)+T
      GOTO 206
  205 IF(IABS(L).NE.N/2) GOTO 211
C
C     LE FACTEUR MULTIPLICATIF = -1
C
      T=-Y(KN2)
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
      GOTO 206
  211 IF(IABS(L).NE.3*N/4) GOTO 212
C
C     LE FACTEUR MULTIPLICATIF = J
C
      E=DCMPLX(0.D0,1.D0)
      IF(INVERS) E=DCMPLX(0.D0,-1.D0)
      T=Y(KN2)*E
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
      GOTO 206
  212 IF(IABS(L).NE.N/4) GOTO 207
C
C     LE FACTEUR MULTIPLICATIF = -J
C
      E=DCMPLX(0.D0,-1.D0)
      IF(INVERS) E=DCMPLX(0.D0,1.D0)
      T=Y(KN2)*E
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
      GOTO 206
C
C     LE FACTEUR MULTIPLICATIF EST QUELCQUONQUE
C
  207 ARG=2*PI*DFLOAT(L)/DFLOAT(N)
      DC=DCOS(ARG)
      DS=-DSIN(ARG)
      E=DCMPLX(DC,DS)
      T=Y(KN2)*E
      Y(KN2)=Y(K)-T
      Y(K)=Y(K)+T
  206 K=K+1
      K1=K-1
      KN2=KN2+1
  204 CONTINUE
      K=K+N2
      K1=K-1
      KN2=KN2+N2
      IF(K.LE.N)  GOTO 208
      N2=N2*2
  203 CONTINUE
```

```
      IF(.NOT.INVERS) GOTO 200
C
C     POUR CALCULER LA TRANSFORMEE INVERSE, IL FAUT DIVISER LES
C     RESULTATS PAR N
C
      DO 215 I=1,N
      Y(I)=Y(I)/DFLOAT(N)
  215 CONTINUE
  200 RETURN
      END
```

```
      SUBROUTINE DDIF(NU,X,Y,N,PI,INVERS)
C
C
C
C     CETTE SOUS-ROUTINE CALCULE LA TRANSFORMEE DE FOURIER DISCRETE
C     EN DOUBLE PRECISION PAR L'ALGORITHME DE SANDE ET TUKEY
C     DANS LE CAS OU  N = 2**NU
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE  TYPE          SIGNIFICATION
C
C     X         COMPLEX*16    TABLEAU D'ENTREE DE LA TRANSFORMEE
C     Y         COMPLEX*16    TABLEAU DE SORTIE DE LA TRANSFORMEE
C     N         INTEGER       LE NOMBRE DE POINTS DE LA TRANSFORMEE
C     PI        REAL*8        LE NOMBRE PI
C     INVERS    LOGICAL       SI VRAI, ALORS TRANSFORMEE INVERSE
C                             SI FAUX, ALORS TRANSFORMEE DIRECTE
C
C
C
      COMPLEX*16 X(1024),Y(1024),T,E
      LOGICAL INVERS
      REAL*8 DC,DS,ARG,PI
      N2=N
      DO 401 NU1=1,NU
      MIN=1-N2
      N2=N2/2
      NU2=2**(NU1-1)
      DO 402 I=1,NU2
      MIN=MIN+2*N2
      MAX=MIN+N2-1
      DO 403 K=MIN,MAX
      K1=K-1
      KN2=K+N2
      L=LDIF(NU1,K1,N,INVERS)
      IF(L.NE.0) GOTO 405
C
C     LE FACTEUR MULTIPLICATIF = 1
C
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=T-Y(KN2)
      GOTO 403
  405 IF(IABS(L).NE.N/2) GOTO 411
C
C     LE FACTEUR MULTIPLICATIF = -1
C
      T=Y(K)
```

```
      Y(K)=T+Y(KN2)
      Y(KN2)=Y(KN2)-T
      GOTO 403
  411 IF(IABS(L).NE.3*N/4) GOTO 412
C
C     LE FACTEUR MULTIPLICATIF = J
C
      E=DCMPLX(0.D0,1.D0)
      IF(INVERS) E=DCMPLX(0.D0,-1.D0)
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=(T-Y(KN2))*E
      GOTO 403
  412 IF(IABS(L).NE.N/4) GOTO 407
C
C     LE FACTEUR MULTIPLICATIF = -J
C
      E=DCMPLX(0.D0,-1.D0)
      IF(INVERS) E=DCMPLX(0.D0,1.D0)
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=(T-Y(KN2))*E
      GOTO 403
C
C     LE FACTEUR MULTIPLICATIF EST QUELCQUONQUE
C
  407 ARG=2*PI*DFLOAT(L)/DFLOAT(N)
      DC=DCOS(ARG)
      DS=-DSIN(ARG)
      E=DCMPLX(DC,DS)
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=(T-Y(KN2))*E
  403 CONTINUE
  402 CONTINUE
  401 CONTINUE
      IF(.NOT.INVERS) GOTO 400
C
C     POUR CALCULER LA TRANSFORMEE INVERSE, IL FAUT DIVISER LES
C     RESULTATS PAR N
C
      DO 410 I=1,N
      Y(I)=Y(I)/DFLOAT(N)
  410 CONTINUE
C
C     ON INVERSE BINAIREMENT L'ORDRE DES RESULTATS
C
  400 N1=N-1
      DO 404 K=2,N1
      K1=K-1
```

```
101         I=IBITR(K1,NU)+1
102         IF(I.LE.K) GOTO 404
103         T=Y(K)
104         Y(K)=Y(I)
105         Y(I)=T
106   404   CONTINUE
107         RETURN
108         END
```

```
      SUBROUTINE SDIF(NU,X,Y,N,PI,INVERS)
C
C
C
C     CETTE SOUS-ROUTINE CALCULE LA TRANSFORMEE DE FOURIER DISCRETE
C     EN SIMPLE PRECISION PAR L'ALGORITHME DE SANDE ET TUKEY
C     DANS LE CAS OU  N = 2**NU
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE  TYPE          SIGNIFICATION
C
C     X         COMPLEX*8     TABLEAU D'ENTREE DE LA TRANSFORMEE
C     Y         COMPLEX*8     TABLEAU DE SORTIE DE LA TRANSFORMEE
C     N         INTEGER       LE NOMBRE DE POINTS DE LA TRANSFORMEE
C     PI        REAL*8        LE NOMBRE PI
C     INVERS    LOGICAL       SI VRAI, ALORS TRANSFORMEE INVERSE
C                             SI FAUX, ALORS TRANSFORMEE DIRECTE
C
C
C
      COMPLEX*8 X(1024),Y(1024),T
      COMPLEX*16 E
      LOGICAL INVERS
      REAL*8 DC,DS,ARG,PI
      N2=N
      DO 501 NU1=1,NU
      MIN=1-N2
      N2=N2/2
      NU2=2**(NU1-1)
      DO 502 I=1,NU2
      MIN=MIN+2*N2
      MAX=MIN+N2-1
      DO 503 K=MIN,MAX
      K1=K-1
      KN2=K+N2
      L=LDIF(NU1,K1,N,INVERS)
      IF(L.NE.0) GOTO 505
C
C     LE FACTEUR MULTIPLICATIF = 1
C
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=T-Y(KN2)
      GOTO 503
  505 IF(IABS(L).NE.N/2) GOTO 511
C
C     LE FACTEUR MULTIPLICATIF = -1
C
```

```
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=Y(KN2)-T
      GOTO 503
  511 IF(IABS(L).NE.3*N/4) GOTO 512
C
C     LE FACTEUR MULTIPLICATIF = J
C
      E=DCMPLX(0.D0,1.D0)
      IF(INVERS) E=DCMPLX(0.D0,-1.D0)
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=(T-Y(KN2))*E
      GOTO 503
  512 IF(IABS(L).NE.N/4) GOTO 507
C
C     LE FACTEUR MULTIPLICATIF = -J
C
      E=DCMPLX(0.D0,-1.D0)
      IF(INVERS) E=DCMPLX(0.D0,1.D0)
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=(T-Y(KN2))*E
      GOTO 503
C
C     LE FACTEUR MULTIPLICATIF EST QUELCQUONQUE
C
  507 ARG=2*PI*DFLOAT(L)/DFLOAT(N)
      DC=DCOS(ARG)
      DS=-DSIN(ARG)
      E=DCMPLX(DC,DS)
      T=Y(K)
      Y(K)=T+Y(KN2)
      Y(KN2)=(T-Y(KN2))*E
  503 CONTINUE
  502 CONTINUE
  501 CONTINUE
      IF(.NOT.INVERS) GOTO 500
C
C     POUR CALCULER LA TRANSFORMEE INVERSE, IL FAUT DIVISER LES
C     RESULTATS PAR N
C
      DO 510 I=1,N
      Y(I)=Y(I)/DFLOAT(N)
  510 CONTINUE
C
C     ON INVERSE BINAIREMENT L'ORDRE DES RESULTATS
C
  500 N1=N-1
      DO 504 K=2,N1
```

```
      K1=K-1
      I=IBITR(K1,NU)+1
      IF(I.LE.K) GOTO 504
      T=Y(K)
      Y(K)=Y(I)
      Y(I)=T
504   CONTINUE
      RETURN
      END
```

```
1         FUNCTION IBITR (J,NU)
2   C
3   C
4   C
5   C       CETTE FONCTION CALCULE LE NOMBRE BINAIRE INVERSE DE J,
6   C       PAR RAPPORT A NU COMPOSANTES
7   C
8   C
9   C
10        J1=J
11        IBITR=J
12        DO 300 I=1,NU
13        J2=J1/2
14        IBITR=IBITR*2+(J1-2*J2)
15  300   J1=J2
16        RETURN
17        END
```

```
      FUNCTION LDIT(NU1,K1,N,INVERS)
C
C
C
C     CETTE FONCTION SERT A CALCULER LES EXPOSANTS DES FACTEURS
C     MULTIPLICATIFS DANS L'ALGORITHME DE COOLEY ET TUKEY.
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE   TYPE          SIGNIFICATION
C
C     K1         INTEGER       REPERE LE PAPILLON TRAITE
C     NU1        INTEGER       NUMERO DE L'ETAPE
C     N          INTEGER       NOMBRE DE POINTS DE LA TRANSFORMEE
C     INVERS     LOGICAL       SI VRAI, ALORS TRANSFORMEE INVERSE
C                              SI FAUX, ALORS TRANSFORMEE DIRECTE
C
C
C
      LOGICAL INVERS
      K2=K1*N/2**NU1
      LDIT=MOD(K2,N)
      IF(INVERS) LDIT=-LDIT
      RETURN
      END
```

```
      FUNCTION LDIF(NU1,K1,N,INVERS)
C
C
C
C     CETTE FONCTION SERT A CALCULER LES EXPOSANTS DES FACTEURS
C     MULTIPLICATIFS DANS L'ALGORITHME DE SANDE ET TUKEY
C
C
C     TABLE DES VARIABLES PRINCIPALES
C
C     VARIABLE  TYPE          SIGNIFICATION
C
C     K1        INTEGER       REPERE LE PAPILLON TRAITE
C     NU1       INTEGER       NUMERO DE L'ETAPE
C     N         INTEGER       NOMBRE DE POINTS DE LA TRANSFORMEE
C     INVERS    LOGICAL       SI VRAI, ALORS TRANSFORMEE INVERSE
C                             SI FAUX, ALORS TRANSFORMEE DIRECTE
C
C
C
      LOGICAL INVERS
      K2=K1*2**(NU1-1)
      LDIF=MOD(K2,N)
      IF(INVERS) LDIF=-LDIF
      RETURN
      END
```

```
1         FUNCTION IFT(P,N)
2   C
3   C
4   C
5   C     CETTE FONCTION CALCULE LA VALEUR DE LA FONCTION
6   C     F(P) DEFINIE EN (4,9)
7   C
8   C
9   C
10        PE=P
11        L=0
12        N2=N-2
13        IF(N2.LE.0) GOTO 603
14        DO 601 I=1,N2
15        PDEMI=PE/(2**I)
16        J=0
17        IF(PE.EQ.(2**I)*PDEMI) GOTO 602
18        J=1
19        L=L+1
20  602   PE=PE-J*2**(I-1)
21  601   CONTINUE
22  603   IFT=L
23        RETURN
24        END
```

BIBLIOGRAPHIE

[1] E.O. Brigham, The Fast Fourier Transform, Englewood Cliffs, NJ : Prentice-Hall, 1974.

[2] W.T. Cochran, et al., What Is The Fast Fourier Transform ?, IEEE Transactions on Audio Electroacoustics,vol. AU-15, pp. 45-55, June 1967.

[3] J.W. Cooley & J.W. Tukey, An Algorithm For The Machine Calculation Of Complex Fourier Series, Math. Comp., vol. 19, pp. 297-301, April 1965.

[4] W.M. Gentleman & G. Sande, Fast Fourier Transform For Fun And Profit, in 1966 Fall Joint Computer Conf., AFIPS Conf. Proc., vol. 29, pp. 563-578.

[5] R.W. Hamming, On The Distribution Of Numbers, Bell System Technical Journal, vol. 49, No 8, pp. 1609-1626, Oct. 1970.

[6] Hoel, Port & Stone, Introduction To Probability Theory, Boston, Houghton Mifflin Company, 1971.

[7] T. Kaneko & B. Liu, Accumulation Of Roundoff Error In Fast Fourier Transforms, Journal of the ACM, vol. 17, pp. 637-656, Oct. 1970.

[8] T. Kaneko & B. Liu, On Local Roundoff Errors In Floating Point Arithmetic, Journal of the ACM, vol. 20, pp. 391-398, July 1973.

[9] B. Liu & T. Kaneko, Roundoff Error In Fast Fourier Transforms (Decimation In Time), IEEE Proceedings, vol. 63, pp. 991-992, June 1975.

[10] B. Liu & A. Peled, A New Hardware Realization Of High Speed Fast Fourier Transform, IEEE Trans. Acoustics Speech Signal Processing, vol. ASSP-23, pp. 543-547, Dec. 1975.

[11] A.V. Oppenheim & R.W. Schafer, Digital Signal Processing, Englewood Cliffs, NJ : Prentice-Hall, 1975.

[12] A.V. Oppenheim & C.J. Weinstein, Effects Of Finite Register Length In Digital Filtering And The Fast Fourier Transform, Proc. IEEE, pp. 957-976, Aug. 1972.

[13] M. Sundaramurthy & V. Umapathi Reddy, Some Results In Fixed-Point Fast Fourier Transform Error Analysis, IEEE Trans. on Computers, Correspondence, pp. 305-308, March 1977.

[14] D.W. Sweeney, An Analysis Of Floating-Point Addition, IBM Syst. J., 4,1, pp. 31-42, 1965.

[15] J.P. Thiran, Error Bounds For Floating-Point Addition Using Guard Digits, to be presented at the 1978 IEEE International Symposium on Circuits and Systems, New-York, May 1978.

[16] Tran-Thong & B. Liu, Fixed-Point Fast Fourier Transform Error Analysis, IEEE Transactions on Acoustics, Speech and Signal Processing, vol. ASSP-24, No 46, pp. 563-573, Dec. 1976.

[17] Tran-Thong & B. Liu, Accumulation Of Roundoff Errors In Floating-Point Fast Fourier Transform, IEEE Transactions on Circuits and Systems, vol. CAS-24, No 3, pp. 132-143, March 1977.

[18] C.J. Weinstein, Quantization Effects In Digital Filters, Ph. D. Dissertation, Department of Electrical Engineering, M.I.T., July 1969.

[19] C.J. Weinstein, Roundoff Noise In Floating-Point Fast Fourier Transform Computation, IEEE Transactions on Audio Electroacoustics, vol. AU-17, pp. 209-215, Sept. 1969.

[20] P.D. Welch, A Fixed-Point Fast Fourier Transform Error Analysis, IEEE Transactions on Audio Electroacoustics, vol. AU-17, pp. 153-157, June 1969.

TABLE DES MATIERES

INTRODUCTION 1

1 TRANSFORMEE DE FOURIER RAPIDE (FFT) 4

1.1 Transformée de Fourier Discrète 5

1.2 Algorithmes FFT dans le cas $N=2^M$ 7

A Algorithme de Cooley et Tukey 8

B Algorithme de Sande et Tukey 14

1.3 Algorithmes FFT dans le cas $N=r^M$ 20

A Algorithme de Cooley et Tukey 21

B Algorithme de Sande et Tukey 24

1.4 Algorithmes FFT dans le cas $N=r_1.r_2$ 26

A Algorithme de Cooley et Tukey 26

B Algorithme de Sande et Tukey 27

2 MODELES STATISTIQUES POUR LES ERREURS NUMERIQUES 28

2.1 Théorie statistique élémentaire des erreurs numériques 29

2.1.1 Introduction 29

2.1.2 Processus aléatoires 30

2.1.3 Modèle statistique pour les erreurs numériques 32

2.2 Analyse des erreurs en virgule fixe 34

2.2.1 Arithmétique en virgule fixe 34

A Représentation des nombres binaires 34

B Opérations élémentaires 35

C Erreurs produites par la troncature ou l'arrondi 36

D Erreurs produites par le dépassement 39

2.2.2 Application des hypothèses du modèle statistique 40

2.3 Analyse des erreurs en virgule flottante 41

2.3.1 Arithmétique et bornes d'erreur 41

A Représentation numérique en virgule flottante 41
B Multiplication à l'aide d'un registre de longueur double 43
C Addition à l'aide de chiffres de garde 45
2.3.2 Modèles statistiques pour les erreurs 47
A Distribution de la mantisse 48
B Erreur de représentation numérique 48
C Erreur de multiplication 51
D Erreur d'addition 52

3 ANALYSE DES ERREURS NUMERIQUES DANS LES ALGORITHMES DE LA TRANSFORMEE DE FOURIER RAPIDE EN VIRGULE FIXE 56
Introduction 57
3.1 Résultats de l'analyse de l'algorithme de Cooley et Tukey 58
3.1.1 Modèle statistique 58
3.1.2 Propriétés de l'algorithme de Cooley et Tukey 61
3.1.3 Etude des erreurs numériques 66
3.1.3.1 Arrondi - Sans déplacement 66
3.1.3.2 Troncature - Sans déplacement 69
3.1.3.3 Arrondi - Déplacement à chaque étape 73
3.1.3.4 Troncature - Déplacement à chaque étape 77
3.1.3.5 Remarques 78
3.2 Résultats de l'analyse de l'algorithme de Sande et Tukey 79
3.2.1 Modèle statistique 79
3.2.2 Propriétés de l'algorithme de Sande et Tukey 80
3.2.3 Etude des erreurs numériques 81
3.2.3.1 Arrondi - Sans déplacement 81
3.2.3.2 Troncature - Sans déplacement 82
3.2.3.3 Arrondi - Déplacement à chaque étape 85

3.2.3.4 Troncature - Déplacement à chaque étape 87
3.3 Algorithmes pour des bases composites 88
3.3.1 Algorithme de Cooley et Tukey 88
3.3.2 Algorithme de Sande et Tukey 90

4 ANALYSE DES ERREURS NUMERIQUES DANS LES ALGORITHMES DE LA TRANSFORMEE DE FOURIER RAPIDE EN VIRGULE FLOTTANTE 93
4.1 Généralités 93
4.2 Algorithme de Cooley et Tukey 96
4.3 Algorithme de Sande et Tukey 106
4.4 Cas particuliers 112
4.4.1 $N=2^M$ 113
A Cas de l'arrondi 113
B Cas de la troncature 115
4.4.2 $N=r^M$, r quelconque 117
A Algorithme de Cooley et Tukey 117
B Algorithme de Sande et Tukey 121
4.4.3 $N=r_1.r_2$, r_1 et r_2 quelconques 125
A Algorithme de Cooley et Tukey 125
B Algorithme de Sande et Tukey 128

5 RESULTATS NUMERIQUES 131
5.1 Résultats numériques pour la virgule fixe 132
Annexe : Programmes et sous-programmes FORTRAN 138
5.2 Résultats numériques pour la virgule flottante 151
Annexe : Programmes et sous-programmes FORTRAN 163

BIBLIOGRAPHIE 188

TABLE DES MATIERES 191