NEUVIEME COLLOQUE SUR LE TRAITEMENT DU SIGNAL ET SES APPLICATIONS

NICE du 16 au 20 MAI 1983

PROGRAMMATION PERFORMANTE DE LA TRANSFORMATION DE FOURIER PAR ENCHAINEMENT DES ALGORITHMES DE WINOGRAD

J.R MASSE

CENTRE DE CALCUL SCIENTIFIQUE DE L'ARMEMENT 35170 BRUZ

D. CANTE

Laboratoire d'automatique
INSA 35042 RENNES CEDEX

RESUME

Cette étude porte sur les nouvelles procédures de transformation de Fourier rapide par enchainement des algorithmes de base de S.WINOGRAD [4] [5].

Dans de nombreuses applications, (convolution, filtrage, identification) il est utile de pouvoir effectuer les transformations directe et inverse à l'aide d'un même programme.

Dans ce but nous exploitons les propriétés d'isomorphismes de groupe des indiçages utilisés dans les procédures d'enchainement pour construire des implantations de l'option inverse. L'une d'elles est originale et nous montrons expérimentalement (place mémoire, rapidité,...) qu'il y a lieu de l'utiliser.

Les programmes obtenus sont dans le Logiciel scientifique de base du Centre de Calcul Scientifique de l'Armement.

SUMMARY

This paper is mainly descriptive (how to program) and experimental (timings and comparisons).

The problem is to build a general-N WINOGRAD D.F.T program able to compute both forward and inverse D.F.T.. This kind of program is useful in many applications (convolution, filtering, identification).

We propose a new way of implementing the inverse computation as an option of the program. We compare this to more classical ways and we show that this is preferable in some cases.

This extends to inverse option D.F.T works previously published ([4] [5])

PROGRAMMATION PERFORMANTE DE LA TRANSFORMATION DE FOURIER PAR ENCHAINEMENT DES ALGORITHMES DE WINOGRAD

INTRODUCTION

Les travaux de S.WINOGRAD [1] permettent de connaitre le nombre minimum de multiplications pour calculer une transformée de Fourier discrete (TFD) d'ordre N en puissance d'un nombre premier et de construire de tels algorithmes pour N petit. S.WINOGRAD propose de les enchainer pour obtenir des T.F.D d'ordres plus élevés. Deux approches sont suggerées : l'enchainement des facteurs premiers de I.J GOOD [2] et l'enchainement des facteurs premiers par emboitements de S.WINOGRAD.

En 1979 J.H Mc CLELLAN publie un programme FORTRAN général (WFTA) selon l'enchainement par emboitements [3]. En 1981 C.S BURRUS publie un programme FORTRAN (PFA1) plus simple et plus rapide utilisant "sur place" l'algorithme des facteurs premiers [4]. En 1982 J.H ROTHWEILER publie une version "sur place" et "dans l'ordre" de ce programme (PFA2) plus sobre en mémoire [5].

Ces deux derniers ne comportent pas d'option de T.F.D inverse. Or, dans de nombreuses applications, il est utile de pouvoir effectuer les transformations directe et inverse à l'aide d'un même programme. Dans ce qui suit de telles implantations sont déduites systématiquement des propriétés des indiçages des facteurs premiers.

INDICAGES

D'après S.BURRUS [6] les indiçages utilisés dans les algorithmes des facteurs premiers sont de la forme : $(n_1, \ldots, n_m) \longleftrightarrow$

$n = \alpha_1 (N/N_1) n_1 + \ldots + \alpha_m (N/N_m) n_m$ modulo N

où N est décomposable en facteurs premiers entre eux:

$N = N_1 \times \ldots \times N_m$

et où α_i et N_i sont premiers entre eux pour tout i.

De tels indiçages définissent des isomorphismes de groupes additifs entre Z/NZ et $(Z/N_1Z) \times \ldots \times (Z/N_mZ)$

C'est en particulier le cas pour l'isomorphisme d'anneaux CHINOIS : $\alpha_i = (N/N_i)^{-1}$ modulo N_i et pour l'isomorphisme de groupe dit RURITANIEN [2] : $\alpha_i = 1$, utilisés dans les algorithmes ci-dessous.

ALGORITHMES

Cas général

La T.F.D. d'ordre N est définie par :

$C(k) = \sum_n x(n) W_N^{nk} \qquad 0 \leqslant n,k \leqslant N-1$

où $W_N = e^{-j2\pi/N}$

Les indiçages précédents sont appliqués à n et k :

$n = \alpha_1 (N/N_1) n_1 + \ldots + \alpha_m (N/N_m) n_m$ modulo N

$k = \beta_1 (N/N_1) k_1 + \ldots + \beta_m (N/N_m) k_m$ modulo N

soient $\hat{X}(n_1, \ldots, n_m) = X(n)$ et $\hat{C}(k_1, \ldots, k_m) = X(k)$

on obtient : $\hat{C}(k_1, \ldots, K_m) =$

$$\sum_{n_1} \cdots \sum_{n_m} \hat{X}(n_1, \ldots n_m) W_{N_1}^{\alpha_1 \beta_1 (N/N_1) n_1 k_1} \cdots W_{N_m}^{\alpha_m \beta_m (N/N_m) n_m k_m}$$

Algorithme de BURRUS

Dans le sous-programme PFA1 "sur place"[4] les indiçages CHINOIS et RURITANIEN sont appliqués respectivement à k et n. L'expression ci-dessus devient :

$$\hat{C}(k_1, \ldots, k_m) = \sum_{n_m} W_{N_m}^{n_m k_m} \cdots \underbrace{\sum_{n_1} W_{N_1}^{n_1 k_1} \hat{X}(n_1, \ldots, n_m)}_{1^{re}\text{ étape : } N/N_1 \text{ T.F.D. d'ordre } N_1}$$

m^e étape : N/N_m T.F.D d'ordre N_m

Les m étapes sont générées par la boucle DO 10 (fig.1.0)

Les calculs de T.F.D d'ordres N_i sont faits sur place. Le résultat final doit donc être réordonné. La permutation à appliquer est la composée de l'isomorphisme RURITANIEN par l'isomorphisme CHINOIS qui s'exprime par [4] : $n \longmapsto (nx\text{UNSC})$ modulo N où UNSC = $(N/N_1 + \ldots + N/N_m)$ modulo N

Algorithme de BURRUS - ROTHWEILER

Dans le sous-programme PFA2 "sur place" et "dans l'ordre" [4] [5] les calculs sont également effectués sur place. De plus le même indiçage RURITANIEN est appliqué à n et k. Le résultat final n'a donc plus à être réordonné. L'expression générale devient :

$$\hat{C}(K_1, \ldots, K_m) = \sum_{n_m} W_{N_m}^{(N/N_m) n_m k_m} \cdots \sum_{n_1} W_{N_1}^{(N/N_1) n_1 k_1} \hat{X}(n_1, \ldots, n_m)$$

N/N_i et N_i étant premiers entre eux $(N/N_i)\, n_i$

ou $(N/N_i) k_i$ modulo N_i définissent des permutations

[4] [3]. En définitive le caractère "dans l'ordre" se traduit par une permutation d'indice au début ou à la fin de chaque petit algorithme de base de WINOGRAD. Les permutations sont calculées une fois pour toutes au début de chaque étape dans la boucle DO 15 (fig.2.0)

PROGRAMMATION PERFORMANTE DE LA TRANSFORMATION DE FOURIER PAR ENCHAINEMENT DES ALGORITHMES DE WINOGRAD

IMPLANTATION DE L'INVERSE

Cas Général

La T.F.D inverse est définie par :

$$B(k) = N^{-1} \sum_{n} x(n) W_N^{-nk}$$

Implantations externes Z/NZ étant un anneau

ceci peut s'écrire : $B(k) = N^{-1} \sum_{n} x(n) (W_N^{-1})^{nk}$ (c.e)

$$B(k) = N^{-1} \sum_{n} X(n) W_N^{(-n)k} = N^{-1} \sum_{n} X(n) W_N^{n(-k)} \quad \text{(p.e)}$$

Pour $W = e^{-j2\pi/N}$ (W_N^{-1}) est le conjugué complexe $\overline{W_N}$ de W_N et (c.e) devient $B(k) = N^{-1} \overline{\sum_{n} \overline{X(n)}\, W_N^{nk}}$.

Ainsi, l'inverse peut être implanté par conjugaison de l'entrée et de la sortie normalisée par N^{-1} . Ceci est d'utilisation courante notemment dans le programme FFT842 [7] et nous appellerons "conjugaison externe" cette implantation.

L'expression (p.e) conduit à un autre type d'implantation. Z/NZ étant un groupe additif $(-n) = (N-n)$ modulo N définit une permutation. Ceci est utilisé dans le programme WFTA de Mc CLELLAN [7] par enchainement des facteurs premiers par emboitements. Nous appellerons "permutation externe" cette implantation.

Implantations internes Soit $\hat{B}(k_1,\ldots,k_m) = B(k)$

comme précédemment. Les indiçages utilisés étant des isomorphismes de groupe, -n correspond à $(-n_1,\ldots,-n_m)$

et (p.e) et (c.e) deviennent : $\hat{B}(k_1,\cdots,k_m) =$

$$N^{-1} \sum_{n_1} \cdots \sum_{n_m} \hat{X}(n_1,\cdots,n_m) W_{N_1}^{\alpha_1\beta_1(N/N_1)(-n_1)k_1} \cdots W_{N_m}^{\alpha_m\beta_m(N/N_m)(-n_m)k_m} \quad \text{(p.i)}$$

$$= N^{-1} \sum_{n_1} \cdots \sum_{n_m} \hat{X}(n_1,\cdots,n_m) \left(W_{N_1}^{-1}\right)^{\alpha_1\beta_1(N/N_1)n_1k_1} \cdots \left(W_{N_m}^{-1}\right)^{\alpha_m\beta_m(N/N_m)n_mk_m} \quad \text{(c.i)}$$

Dans ce qui suit (c.i) et (p.i) correspondent à des conjugaisons et des permutations appliquées aux petits algorithmes de base de WINOGRAD. Nous appellerons ceci "conjugaison interne" et "permutation interne".

Algorithme de BURRUS

Dans PFA1 il est possible de combiner la permutation externe avec la remise en ordre finale qui devient alors : n (-n x UNSC) = (n x (N-UNSC)) modulo N. Ceci est montré en figure (1.p.e)

En ce qui concerne la conjugaison interne, l'expression (c.i.) devient $\hat{B}(k_1,\ldots,k_m) =$

$$N^{-1} \sum_{n_m} \left(W_{N_m}^{-1}\right)^{n_m k_m} \cdots \sum_{n_1} \left(W_{N_1}^{-1}\right)^{n_1 k_1} \hat{X}(n_1,\cdots,n_m)$$

Pour éviter d'avoir à conjuguer entrée et sortie à chaque application d'un petit module de base de WINOGRAD, on peut conjuguer une fois pour toutes les constantes multiplicatives immaginaires, y compris la constante triviale j, de ces modules. Ceci est explicité en figure (1.c.i.)

Quant à (p.i.) elle devient $\hat{B}(k_1,\ldots,k_m) =$

$$N^{-1} \sum_{n_m} W_{N_m}^{n_m(-k_m)} \cdots \sum_{n_1} W_{N_1}^{n_1(-k_1)} \hat{X}(n_1,\cdots,n_m)$$

qui correspond à une permutation d'indice à l'issue de chaque petit module de base. Ceci conduit à l'architecture de PFA2.

Algorithme de BURRUS - ROTHWEILER

PFA2 ayant été construit pour effectuer une permutation d'indice dans les petits modules de base, ceci permet d'intégrer très simplement la permutation interne. La permutation d'indice précalculée dans la boucle DO 15 devient : $k_i \mapsto ((N/N_i)(-k_i)) = ((N-(N/N_i))\, k_i)$ modulo N_i . Ceci s'implante par une ligne fortran (fig 2.P.I.)

COMPARAISONS

Algorithme de BURRUS

Tableau 1 - PFA1 - place mémoire (en mots) occupée par les instructions relatives à l'option inverse en FORTRAN ASCII 8R1 sur UNIVAC 1100/83 sous EXEC 37R2

conjugaison externe	70
conjugaison interne	192
permutation externe	50

Tableau 2 - PFA1 - Implantations de l'option inverse Temps moyen pour 10 exécutions (en m.s. écarts types observes inférieurs à 0,2 m.s.) - FORTRAN ASCII 8R1 UNIVAC 1100/83 EXEC 37R2

	Ordre N		1008	1260	1680	2520	5040
Conjugaison externe		D	67.0	98.0	133.2	202.2	421.1
		I	74.9	107.4	140.6	220.8	458.4
Conjugaison interne		D	67.7	99.0	134.3	204.3	424.1
		I	71.5	103.2	140.5	212.7	441.1
Permutation externe		D	66.9	97.9	133.2	202.0	421.0
		I	70.8	102.2	139.4	210.6	438.0

D : T.F.D. Directe - I : T.F.D. inverse.

D'après les tableaux 1 et 2, les conjugaisons sont les implantations les moins performantes. La conjugaison interne affecte les calculs directs et inverses du fait de l'initialisation des constantes multiplicatives et des multiplications triviales supplémentaires dans les modules d'ordres 4, 8, et 16. La permutation externe est légèrement plus rapide et moins couteuse en mémoire.

Algorithme de BURRUS - ROTHWEILER

Tableau 3 - PFA2 - Place mémoire (en mots) occupée par les instructions relatives à l'option inverse en FORTRAN ASCII 8R1 sur UNIVAC 1100/83 sous EXEC 37R2

Conjugaison externe	54
Conjugaison interne	175
Permutation externe	66
Permutation interne	44

Tableau 4 - PFA2 - Implantations de l'option inverse Temps moyens pour 10 exécutions (en m.s écarts types observés inférieurs à 0.3 m.s.) - FORTRAN ASCII 8R1 UNIVAC 1100/83 EXEC 37R2

Ordre N		1008	1260	1680	2520	5040
Conjugaison externe	D	59.6	88.8	120.9	183.9	386.3
	I	73.1	105.0	142.9	216.9	451.0
Conjugaison interne	D	60.3	89.8	122.0	186.7	390.0
	I	69.3	100.9	137.4	208.9	434.4
Permutation externe	D	59.7	88.8	120.9	184.4	386.5
	I	68.9	99.7	135.9	206.3	430.2
Permutation interne	D	59.6	88.8	120.9	184.7	386.6
	I	69.0	99.9	136.3	206.6	431.1

Tableau 5 - PFA2 - Implantations de l'option inverse Temps d'exécution (m.s) - FORTRAN 77 ANSI X3.0 1978 HP 1000 21/17-F-RTE 6

Ordre N		1008	1260	1680	2520	5040
Permutation externe	D	727	1055	1429	2228	4691
	I	807	1154	1560	2425	5084
Permutation interne	D	727	1054	1428	2225	4688
	I	801	1146	1551	2408	5053

D'après les tableaux 3 et 4 ce qui concerne les conjugaisons s'applique toujours.

Les vitesses relatives pouvant dépendre de la configuration informatique, le tableau 5 concerne un autre système (Laboratoire d'automatique INSA de RENNES

La permutation interne semble intéressante dans les deux cas.

Cas Général

Tableau 6 - Temps d'exécution d'une T.F.D directe suivie d'une T.F.D inverse (m.s) - FORTRAN ASCII 8R1 sous EXEC 37R2 sur UNIVAC 1100/83

N	PFA2	FFT842	FFT	WFTA
1008	129		323	208
1024		228	247	
1260	188		453	280
1680	257		591	385
2048		464	560	
2520	390		963	592
4096		961	1103	
5040	817		1927	

Tableau 7 : Occupation mémoire du programme et des données (mots) - FORTRAN ASCII 8R1 sous EXEC 37R2 sur UNIVAC 1100/83

	PROGRAMME	DONNEES
PFA1	2230	4 x N
PFA2	2340	2 x N
WFTA	4600	$\geq$ 6 x N
FFT	2050	2 x N
FFT842	1300	2 x N

Cette étude s'achève par une comparaison avec des programmes types de bibliothèques [7] : Le programme WFTA de J.H Mc CLELLAN selon l'enchainement par emboitements des facteurs premiers, le programme FFT à bases multiples de R.C SINGLETON et le programme FFT842 de G.D BERGLAND à bases 2, 4 et 8.

Il convient de noter que WFTA comprend une étape dite d'initialisation au cours de laquelle sont effectués des calculs, tels que la décomposition de N, des calculs d'indiçages et des constantes multiplicatives. Cette étape peut être effectuée une fois pour toutes pour N fixé. Elle n'a pas été prise en compte dans les temps d'exécution. En ce qui concerne PFA1 et PFA2 une telle étape se limite à la décomposition de N et éventuellement au calcul de la constante UNSC et l'on constate que le temps d'exécution est insignifiant ce qui n'est pas le cas pour WFTA.

Les tableaux 6 et 7 montrent que PFA2 est réellement rapide pour un encombrement mémoire raisonnable.

La structure très modulaire de PFA1 et PFA2 permet d'ajouter ou de retrancher des algorithmes de base en fonction des besoins ce qui donne une souplesse d'utilisation se rapprochant de celle des algorithmes à bases multiples.

Ceci confirme et étend à la TFD avec option inverse les travaux de C.S BURRUS [4] J.D BLANKEN [8] et J.H ROTHWEILER [5]

CONCLUSION

Trois types d'implantation de l'option inverse peuvent se déduire des propriétés des indiçages utilisés dans les enchainements des facteurs premiers : la conjugaison interne, la permutation externe et la permutation interne.

La conjugaison est d'utilisation courante en transformation de Fourier rapide. On trouve la permutation externe dans WFTA. Il y a lieu de l'utiliser dans PFA1. Le troisième type d'implantation est original et il y a lieu de l'utiliser dans PFA2.

Une étude comparative, incluant des programmes types de bibliothèques, confirme l'intérêt de PFA2 dont la rapidité dépasse celle de programmes à bases en puissances de deux pour une occupation mémoire et une souplesse d'utilisation se rapprochant de celles des programmes à bases multiples.

Les procédures d'enchainement des algorithmes de WINOGRAD sont désormais performantes.

```
: FIG.1.0. EXTRAIT DU PROGRAMME 'SUR PLACE'
:          INITIAL (C.F. [4] ET [5])
:
:      SUBROUTINE PFA1(X,Y,A,B,N,M,NI,UNSC)
:      DIMENSION X(1) ,Y(1),A(1),B(1),NI(1)
:      INTEGER I(16),UNSC
:      EQUIVALENCE (I(1),I1) , (I(2),I2)
:     +, (I(3),I3)
:C     A COMPLETER POUR LES AUTRES FACTEURS
:      DATA C31,C32/0.86602540,0.5000000 /
:C     A COMPLETER POUR LES AUTRES FACTEURS
:C-------------M ETAPES--------------------
:      DO 10 K = 1 , M
:      N1 = NI(K)
:      N2 = N/N1
:         DO 20 J = 1,N,N1
:         I(1) = J
:         IT = J
:            DO 30 L = 2 , N1
:            TT = IT + N2
:            IF (IT.GT.N) IT =IT - N
:            I(L) = IT
:   30       CONTINUE
:         GO TO (20,102,103) , N1
:C     A COMPLETER POUR LES AUTRES FACTEURS
:   20    CONTINUE
:   10 CONTINUE
:C-------------MISE EN ORDRE----------------
:      L = 1
:      DO 2 K = 1 ,N
:      A(K) = X(L)
:      B(K) = Y(L)
:      L = L + UNSC
:      IF (L.GT.N) L = L - N
:    2 CONTINUE
:C-------------RETOUR-----------------------
:   60 RETURN
:C-------------T.F.D. D'ORDRE N1 = 2---------
:  102 T1  = X(I1)
:      X(I1) = T1 + X(I2)
:      X(I2) = T1 - X(I2)
:      T1       = Y(I1)
:      Y(I1) = T1 + Y(I2)
:      Y(I2) = T1 - Y(I2)
:      GO TO 20
:C-------------T.F.D. D'ORDRE N1 = 3---------
:  103 T1= ( X(I2) - X(I3) ) * C31
:      U1 = ( Y(I2) - Y(I3) ) * C31
:      R1 = X(I2) + X(I3)
:      S1 = Y(I2) + Y(I3)
:      T2 = X(I1) - R1 * C32
:      U2 = Y(I1) - S1 * C32
:      X(I1) = X(I1) + R1
:      Y(I1) = Y(I1) + S1
:      X(I2) = T2 + U1
:      X(I3) = T2 - U1
:      Y(I2) = U2 - T1
:      Y(I3) = U2 + T1
:      GO TO 20
:C     A COMPLETER POUR LES AUTRES FACTEURS
:      END
```

```
: FIG. 1.C.I. ADJONCTIONS POUR LA
:             CONJUGAISON INTERNE
:
:      SUBROUTINE PFA1(X,Y,A,B,N,M,NI,UNSC,
:     +INVRS)
:                  .../...
:      DATA C31,C32/0.86602540,0.5000000 /
:C     A COMPLETER POUR LES AUTRES FACTEURS
:      IF (INVRS .NE. 1) GO TO 50
:      CI31 = -C31
:      CI32 = -C32
:C     A COMPLETER POUR LES AUTRES FACTEURS
:      GO TO 70
:   50 CI31 = C31
:      CI32 = C32
:C     A COMPLETER POUR LES AUTRES FACTEURS
:   70 CONTINUE
:C-------------M ETAPES--------------------
:                  .../...
:C-------------MISE EN ORDRE----------------
:      IF (INVRS.EQ.1) GO TO 90
:    2 CONTINUE
:      GO TO 60
:   90 FN = FLOAT(N)
:      L = 1
:      DO 22 K = 1 , N
:      A(K) = X(L) / FN
:      B(K) = Y(L) / FN
:      L = L + UNSC
:      IF ( L .GT. N ) L = L - N
:   22 CONTINUE
:                  .../...
:C-------------T.F.D. D'ORDRE N1 = 3---------
:  103 T1= ( X(I2) - X(I3) ) * CI31
:      U1 = ( Y(I2) - Y(I3) ) * CI31
:      R1 = X(I2) + X(I3)
:      S1 = Y(I2) + Y(I3)
:      T2 = X(I1) - R1 * C32
:      U2 = Y(I1) - S1 * C32
:      X(I1) = X(I1) + R1
:      Y(I1) = Y(I1) + S1
:      X(I2) = T2 + U1
:      X(I3) = T2 - U1
:      Y(I2) = U2 - T1
:      Y(I3) = U2 + T1
:      GO TO 20
:                  .../...
```

```
: FIG. 1.P.E. ADJONCTIONS POUR LA
:             PERMUTATION EXTERNE
:
:      SUBROUTINE PFA1(X,Y,A,B,N,M,NI,UNSC,
:     +INVRS)
:                  .../...
:C-------------MISE EN ORDRE----------------
:      IF (INVRS.EQ.1) GO TO 90
:      L = 1
:                  .../...
:    2 CONTINUE
:      GO TO 60
:   90 FN = FLOAT(N)
:      L = 1 + UNSC
:      DO 22 K = N , 1 , -1
:      A(K) = X(L) / FN
:      B(K) = Y(L) / FN
:      L = L + UNSC
:      IF ( L .GT. N ) L = L - N
:   22 CONTINUE
:C-------------RETOUR-----------------------
:                  .../...
```

```
: FIG. 2.0. EXTRAIT DU PROGRAMME 'SUR PLACE'
:           ET 'DANS L'ORDRE' INITIAL
:           (C.F. [4] ET [5])
:
:      SUBROUTINE PFA2(X,Y,N,M,NI)
:      DIMENSION X(1) ,Y(1),NI(1)
:      INTEGER I(16),NUMAP(16),IR(16),LOC
:      EQUIVALENCE (I(1),I1,IR1) , (I(2),I2)
:     +, (I(3),I3)
:      EQUIVALENCE(IR(2),IR2) , (IR(3),IR3)
:C     A COMPLETER POUR LES AUTRES FACTEURS
:      DATA C31,C32/0.86602540,0.5000000 /
:C     A COMPLETER POUR LES AUTRES FACTEURS
:C--------------M ETAPES---------------------
:      DO 10 K = 1 , M
:      N1 = NT(K)
:      N2 = N/N1
:C--------------CALCUL DES INDICES DE--------
:C--------------PERMUTATION INTERNE----------
:      LOC = 0
:      DO 15  J = 1 , N1
:      NUMAP(J) = MOD(LOC,N1) + 1
:   15 LOC = LOC + N2
:C---------------------------------------------
:         DO 20 J = 1,N,N1
:         I(1) = J
:         IT = J
:            DO 30 L = 2 , N1
:            IT = IT + N2
:            IF (IT.GT.N) IT =IT - N
:            I(L) = IT
:   30       CONTINUE
:         GO TO (20,102,103) , N1
:C     A COMPLETER POUR LES AUTRES FACTEURS
:   20    CONTINUE
:   10 CONTINUE
:C--------------RETOUR-----------------------
:   60 RETURN
:C--------------T.F.D. D'ORDRE N1 = 2--------
:  102 T1  = X(I1)
:      X(IR1) = T1 + X(I2)
:      X(IR2) = T1 - X(I2)
:      T1       = Y(I1)
:      Y(IR1) = T1 + Y(I2)
:      Y(IR2) = T1 - Y(I2)
:      GO TO 20
:C--------------T.F.D. D'ORDRE N1 = 3--------
:  103 T1= ( X(I2) - X(I3) ) * C31
:      U1 = ( Y(I2) - Y(I3) ) * C31
:      R1 = X(I2) + X(I3)
:      S1 = Y(I2) + Y(I3)
:      T2 = X(I1) - R1 * C32
:      U2 = Y(I1) - S1 * C32
:      X(IR1) = X(I1) + R1
:      Y(IR1) = Y(I1) + S1
:      X(IR2) = I2 + U1
:      X(IR3) = T2 - U1
:      Y(IR2) = U2 - T1
:      Y(IR3) = U2 + T1
:      GO TO 20
:C     A COMPLETER POUR LES AUTRES FACTEURS
:      END
```

```
: FIG. 2.P.E.  ADJONCTIONS POUR LA
:              PERMUTATION EXTERNE
:
:      SUBROUTINE PFA2(X,Y,N,M,NI,INVRS)
:      DIMENSION X(1) ,Y(1),NI(1)
:                  .../...
:   10 CONTINUE
:      IF (INVRS .NE. 1)  GO TO 60
:      FN = FLOAT(N)
:      X(1) = X(1) / FN
:      Y(1) = Y(1) / FN
:      L = N + 2
:      DO 90 J = 2 , N/2 +1
:      K = L - J
:      A = X(J)/FN
:      X(J) = X(K) / FN
:      X(K) = A
:      B = Y(J)/FN
:      Y(J) = Y(K) / FN
:      Y(K) = B
:   90 CONTINUE
:C--------------RETOUR-----------------------
:                  .../...
: FIG. 2.P.I.  ADJONCTIONS POUR LA
:              PERMUTATION INTERNE
:
:      SUBROUTINE PFA2(X,Y,N,M,NI,INVRS)
:      DIMENSION X(1) ,Y(1),NI(1)
:                  .../...
:C--------------CALCUL DES INDICES DE--------
:C--------------PERMUTATION INTERNE----------
:      LOC = 0
:      IN2=N2
:      IF (INVRS.EQ.1) IN2=N1-MOD(N2,N1)
:      DO 15  J = 1 , N1
:      NUMAP(J) = MOD(LOC,N1) + 1
:   15 LOC = LOC + IN2
:C---------------------------------------------
:                  .../...
:   10 CONTINUE
:      IF (INVRS .NE. 1)  GO TO 60
:      FN = FLOAT(N)
:      DO 90 J=1,N
:      X(J)=X(J)/FN
:   90 Y(J)=Y(J)/FN
:C--------------RETOUR-----------------------
:                  .../...
```

BIBLIOGRAPHIE

[1] S.WINOGRAD "On Computing the discrete Fourier Transform" Math. Comp. vol 32 pp 175-199 Jan.1978

[2] I.J GOOD "The relationship between two fast Fourier Transforms" IEEE Trans.Comp. Vol C-20 Mar.1971 pp 310-317

[3] J.H Mc CLELLAN, C.M RADER "Number theory in digital signal processing" Prentice Hall 1979

[4] C.S BURRUS, P.W ESCHENBACHER "An in-place, in-order prime factor FFT algorithm" IEEE Trans. on ASSP pp 806-817 August 1981

[5] J.H ROTHWEILER "Implementation of the in-order prime factor transform for variable sizes" IEEE Trans. on ASSP 1982 pp 105-107

[6] C.S BURRUS "Index Mappings for multidimensionnal formulation of the D.F.T and convolution" IEEE Trans. on ASSP 1977 pp 239-242

[7] "Programs for digital signal processing" IEEE Press New York 1979

[8] J.D BLANKEN, P.L RUSTAN "Selection criteria for efficient implementation of FFT algorithms" IEEE Trans. on ASSP 1982 pp 107-109

[9] H.J NUSSBAUMER "Fast Fourier transform and convolution algorithms" Springer-Verlag 1981

[10] M.BELLANGER "Traitement Numérique du signal" Masson 1981

[11] A.YGER "Nouveaux algorithmes de FFT" Note L.A N°50 du Centre de Calcul Scientifique de l'Armement 1980

[12] J.R MASSE, D.CANTE "General-N WINOGRAD D.F.T Programs with inverse option" IEEE International Conf. On ASSP Boston April 1983.