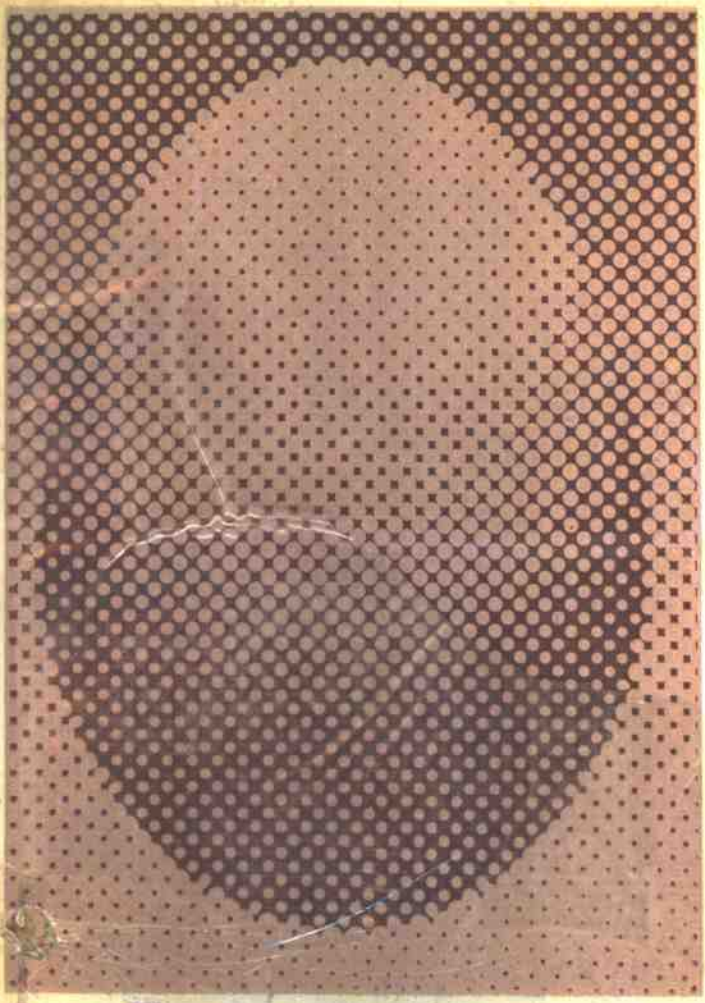
FFTPACK



颜宝勇 编著

快速富氏变换库程序

——算法汇编与程序设计

国防科技大学出版社

颜宝勇 编著

快速富氏变换库程序

——算法汇编与程序设计

国防科技大学出版社

内 容 简 介

本书提供基 2、基 3、基 4、基 5 和任意质因子的快速富氏变换（FFT）算法，以及用标准 FORTRAN77 书写的各种数据类型的 FFT 程序．这些程序在各种大、中、小型计算机上是高效的；在巨型向量计算机上能充分发挥向量运算部件的功能，使之高度并行运算．

全书共分五章：复数据快速富里叶变换；实数据快速富里叶变换；实数据快速正弦变换；实数据快速余弦变换；二维快速富里叶变换．

本书可供广大科研人员、工程技术人员、高等院校高年级学生和硕士研究生阅读和参考，亦可为配有巨、大、中、小型计算机的单位提供快速富氏变换库程序．

快速富氏变换库程序
——算法汇编与程序设计

顾宝嘉　编著

责任编辑　何　晋

封面设计　王建华

国防科技大学出版社出版发行

国防科技大学印刷厂印装

开本：850×1168　1/32　印张：8.75　字数：236 千字

1990 年 2 月第 1 版第 1 次印刷　印数：1–2,000 册

ISBN　7－81024－073－0

TP・14　定价：2.00元

前　言

快速富里叶变换（简写为 FFT）和它的方案已成为机器计算时间采样信号或其它数据序列富氏级数的主要方法．这种方法已经能广泛地应用于高速卷积和相关，功率谱估计，计算微分方程和仿真滤波器等等的目的上。

七十年代中期后，许多学者又进一步研究了 FFT 的并行计算方案，使得并行计算的 FFT 方案又比串行计算提高计算效率几倍至几十倍。

自八十年代初我国有了自己研制的巨型向量计算机后，为适应石油勘探和天气预报等方面需要，广大科研人员对 FFT 进行了广泛深入的研究，设计和开发了一批适合于计算任意点数的一维、二维快速富氏变换程序．这些程序除具有内容丰富，品种齐全，结构清晰，实用性强，通用性好的特点外，更为重要的是运行效率高．

为减少用户不必要的重复工作和提高计算机的计算效率，我们将这些程序组成通用库程序，命名为快速富里叶变换程序包(FFTPACK)，或简称为快速富氏变换库，提供用户使用．

本库程序共有 64 个模块，其中直接提供用户使用的模块为 27 个，每个模块具有一个或多个功能．这些程序由于采用快速算法，故在串行计算机上是高效的．又因为在标量程序中加入强行向量化指导符，故对具有向量识别编译器的向量机，它能自动在指定的 DO 循环中，变标量计算为向量计算．若向量机配有向量 FORTRAN 编译器，则可直接使用向量程序，这些程序能充分发挥向量功能部件并行操作特点，以达到更加高效的目的．经测试统计表明，这些程序在 YH-1 机上，向量运算成份平均达 90%以上．

整个库程序由下述五部分内容组成：

·复数据快速富氏变换
·实数据快速富氏变换
·实数据快速正弦变换
·实数据快速余弦变换
·二维快速富氏变换

由于书中提供的源程序是采用标准 FORTRAN 77 书写的，故这些程序在各种巨、大、中、小型计算机上均能高效运行。对配有向量 FORTRAN 编译器的 YH-1 计算机，可直接采用书中提供的向量程序，它更能发挥向量运算部件并行操作特点，使之高度并行运算。

本书属于通用库程序之一，我们计划在今后 2～3 年内推出若干个通用库程序。在本书编写过程中，得到了陈火旺教授、陈立杰教授、李晓梅副教授的指导和帮助;得到了国防科工委并行算法应用基础研究课题、国防科技大学计算机研究所银河-II 工程的资助;此外，王丽萍同志参加了本库程序的调试工作，编者谨在此致以诚挚的谢意。

由于水平所限，错误难免，欢迎专家、读者及用户批评指正。

编　者

1988 年 10 月

目　　录

1 复数据快速富里叶变换

本章包括五个程序，其中头一个和第四个程序实现对采样点数N进行质因子分解并求变换所需的三角函数值；其余三个程序用来计算复数据到复数据的快速离散富氏变换及其逆变换。这些程序是属于非标准化的变换程序。

一般而言，采用点数可以不受2的幂次方限制而取任意的整数值。但当选择采样点数为小质数乘积时，算法更为高效。从并行计算角度出发，CFFT99程序可多道同时并行计算。因此，它的计算效率比CFFTF或CFFTB高。此外，CFFT99程序更常用于实现二维复数据快速富氏变换。然而，当每道采样点数较大(比如超过10000)时，CFFTF或CFFTB程序也许更有利于解决问题，故用户可根据每道采样点数大小，灵活选用有关程序进行富氏变换。

本章给出了任意因子、基2、基3、基4、基5之快速富氏变换算法公式。

1.1 CFFTI 对CFFTF或CFFTB预置初值

1.1.1 功 能

本子程序为CFFTF或CFFTB提供初始值，即对采样点N进行因子分解，并计算三角函数$\mathrm{COS}(\frac{2\pi}{N}p)$及$\mathrm{SIN}(\frac{2\pi}{N}p)$(p与N有关)的值。

1.1.2 算法简介

对采样点数 N 进行因子分解，即首先判别 N 是否有 3 的因子，若有，就将它从 N 中分解出来，直到 N 中不再包含 3 的因子为止；第二，从剩余的 N 中分解出 4 的因子、2 的因子(易知，若有 2 的因子，最多只有一个)，以及 5 的因子；第三，再从剩余 N 中分解出 5 以上的质因子 7，11，13，…；最后，将分解出的因子按升序顺序排列．这样一来，N 被分解为：

$$N = r_1 r_2 \cdots r_m$$

其中：r_j (j= 1,2,…,m)为 2,3,4,5,7,…及其它质数。

计算三角函数$COS(\frac{2\pi}{N}p)$与$SIN(\frac{2\pi}{N}p)$值，实际上就是求加权系数：

$$W_N^p = COS(\frac{2\pi}{N}p) - jSIN(\frac{2\pi}{N}p)$$

这里 p 与 N 的分解因子有关。

1.1.3 使用说明

1.1.3.1 调用形式

```
CALL CFFTI(N,WSAVE)
```

1.1.3.2 参数说明

N	整变量，输入参数，采样点数。
WSAVE	具有4N+15个元素的一维实数组，输出参数．其中 WSAVE(2N+1)～WSAVE(4N)存放三角函数值。WSAVE(4N+1)～WSAVE(4N+15)位置存放 N 的分解因子，它对应于整数组 IFAC。只要 N 不变，则所提供的 WSAVE 数组之值也不变。

1.1.3.3 引用外部过程

CFFTI1

1.1.3.4 出错条件和返回信息

(无)

1.1.3.5 适用性和限制

本子程序为 CFFTF 或 CFFTB 提供初始值。进行富氏变换时，必须先调用本子程序。过后，只要采样点数 N 不变，就不必重新计算其加权系数。

1.1.4 程 序

```
      SUBROUTINE CFFTI(N,WSAVE)
      DIMENSION WSAVE(*)
      IF(N EQ 1)RETURN
      IW1=N+N+1
      IW2=IW1+N+N
      CALL CFFTI1(N,WSAVE(IW1),WSAVE(IW2))
      RETURN
      END
C
      SUBROUTINE CFFTI1(N,WA,IFAC)
      DIMENSION WA(*),IFAC(*),NTRYH(4)
      DATA NTRYH(1),NTRYH(2),NTRYH(3),NTRYH(4)/3,4,2,5/
      NL=N
      NF=0
      J=0
101   J=J+1
      IF(J-4)102,102,103
102   NTRY=NTRYH(J)
      GOTO 104
103   NTRY=NTRY+2
104   NQ=NL/NTRY
      NR=NL-NTRY*NQ
      IF(NR)101,105,101
105   NF=NF+1
      IFAC(NF+2)=NTRY
```

```
      NL=NQ
      IF(NTRY.NE.2) GOTO 107
      IF(NF.EQ.1) GOTO 107
      DO 106 I=2,NF
      IB=NF-I+2
      IFAC(IB+2)=IFAC(IB+1)
  106 CONTINUE
      IFAC(3)=2
  107 IF(NL.NE.1) GOTO 104
      IFAC(1)=N
      IFAC(2)=NF
      TPI=6.28318530717959
      ARGH=TPI/FLOAT(N)
      I=2
      L1=1
      DO 110 K1=1,NF
      IP=IFAC(K1+2)
      LD=0
      L2=L1*IP
      IDO=N/L2
      IDOT=IDO+IDO+2
      IPM=IP-1
      DO 109J=1,IPM
      I1=I
      WA(I-1)=1.
      WA(I)=0.
      LD=LD+L1
      FI=0.
      ARGLD=FLOAT(LD)*ARGH
      DO 108 II=4,IDOT,2
      I=I+2
      FI=FI+1.
      ARG=FI*ARGLD
      WA(I-1)=COS(ARG)
      WA(I)=SIN(ARG)
  108 CONTINUE
      IF(IP.LE.5) GOTO 109
      WA(I1-1)=WA(I-1)
      WA(I1)=WA(I)
  109 CONTINUE
      L1=L2
  110 CONTINUE
      RETURN
      END
```

CFFTI1 模块对应的向量程序为:

```
      SUBROUTINE CFFTI1(N,WA,IFAC)
      DIMENSION WA( * ),IFAC( * ),NTRYH(4)
      DIMENSION JI(65540)
      DATA NTRYH(1),NTRYH(2),NTRYH(3),
     *   NTRYH(4) / 3,4,2,5 /
      NL = N
      NF = 0
      J = 0
  101 J = J+1
      IF(J-4)102,102,103
  102 NTRY = NTRYH(J)
      GOTO 104
  103 NTRY = NTRY+2
  104 NQ = NL / NTRY
      NR = NL-NTRY * NQ
      IF(NR)101,105,101
  105 NF = NF+1
      IFAC(NF+2) = NTRY
      NL = NQ
      IF(NTRY.NE.2) GOTO 107
      IF(NF.EQ.1) GOTO 107
      DO 106 I = 2,NF
      IB = NF-I+2
      IFAC(IB+2) = IFAC(IB+1)
  106 CONTINUE
      IFAC(3) = 2
  107 IF(NL.NE.1) GOTO 104
      IFAC(1) = N
      IFAC(2) = NF
      TPI = 6.28318530717959
      ARGH = TPI / FLOAT(N)
      I = 2
      L1 = 1
      DO 110 K1 = 1,NF
      IP = IFAC(K1+2)
      LD = 0
      L2 = L1 * IP
      IDO = N / L2
      IDOT = IDO+IDO+2
      IPM = IP-1
      DO 109J = 1,IPM
      JI = I
      WA(I-1) = 1.
      WA(I) = 0.
      LD = LD+L1
```

```
      ARGLD=FLOAT(LD)*ARGH
      II1=0
      DO 108 J1=4,IDOT,2
      II1=II1+1
108   JI(II1)=II1
      I=I+2
      CONGRUENT 5
      #I001=JI(1:(IDOT-4)/2+1)
      #R002=FLOAT(#I001)
      #R003=#R002*ARGLD
      WA(I-1:I+IDOT-5:2)=COS(#R003)
      WA(I:I+IDOT-4:2)=SIN(#R003)
      I=I+(IDOT-4)/2*2
      IF(IP.LE.5) GOTO 109
      WA(I1-1)=WA(I-1)
      WA(I1)=WA(I)
109   CONTINUE
      L1=L2
110   CONTINUE
      RETURN
      END
```

1.1.5 实 例

取 N=53760，调用本程序获得 N 的分解因子如下：

IFAC(1)=53760　　IFAC(2)=8

IFAC(3)=2　　IFAC(4)=3

IFAC(5)=4　　IFAC(6)=4

IFAC(7)=4　　IFAC(8)=4

IFAC(9)=5　　IFAC(10)=7

其中：IFAC(1)存放点数 N，IFAC(2)存放分解因子个数，IFAC(3)~IFAC(10)存放分解因子。

N 个加权系数存放在 WSAVE 数组中，这里不一一列出。

1.2 CFFTF 复周期序列富里叶正变换

1.2.1 功 能

本子程序用来计算复的离散富里叶正变换。换句话说，CFFTF 用于计算任意点数的复周期序列的富里叶系数。

1.2.2 算法简介

众知，N 点离散富氏变换(DFT)定义为:

$$X_k = \sum_{m=0}^{N-1} x_m W^{mk} \quad (k = 0,1,\cdots,N-1) \tag{1.2.1}$$

这里: $W = e^{-j\,2\pi/N}$。

$$j = \sqrt{-1}$$

当 N 为两个因子的乘积，即 N = N1 * N2 时，则可重新定义标号 m 及 K:

$$\begin{cases} m = N_1 m_2 + m_1, & m_1, K_1 = 0,1,\cdots,N_1 - 1 \\ K = N_2 K_1 + K_2, & m_2, K_2 = 0,1,\cdots,N_2 - 1 \end{cases} \tag{1.2.2}$$

将(1.2.2)式代入得:

$$X_{N_2 K_1 + K_2} = \sum_{m_1=0}^{N_1-1} W^{N_2 m_1 k_1} W^{m_1 k_2} \sum_{m_2=0}^{N_2-1} x_{N_1 m_2 + m_1} W^{N_1 m_2 K_2} \tag{1.2.3}$$

记 $W^{N_2} = e^{-j2\pi/N_1} = W_1$，$W^{N_1} = e^{-j2\pi/N_2} = W_2$，得:

$$\mathbf{X}_{N_2k_1+k_2} = \sum_{m_1=0}^{N_1-1} \mathbf{W}_1^{m_1k_1} \mathbf{W}^{m_1k_2} \sum_{m_2=0}^{N_2-1} \mathbf{x}_{N_1m_2+m_1} \mathbf{W}_2^{m_2k_2}$$

(1.2.4)

于是，利用(1.2.4)式计算 $\mathbf{X}_k$ 实际上可分成三步．第一步计算 N_1 个 DFT $\mathbf{Y}_{m_1,k_2}$，它们对应于 m_1 的 N_1 个不同值：

$$\mathbf{Y}_{m_1,k_2} = \sum_{m_2=0}^{N_2-1} \mathbf{x}_{N_1m_2+m_1} \mathbf{W}_2^{m_2k_2} \tag{1.2.5}$$

然后，将 $\mathbf{Y}_{m_1,k_2}$ 乘以旋转因子 $\mathbf{W}^{m_1k_2}$；最后，对 N_2 个输入序列 $\mathbf{Y}_{m_1,k_2}\mathbf{W}^{m_1k_2}$ 计算 N_2 个 N_1 点的 DFT，便可得到 $\mathbf{X}_k$，即

$$\mathbf{X}_{N_2k_1+k_2} = \sum_{m_1=0}^{N_1-1} \mathbf{Y}_{m_1,k_2} \mathbf{W}^{m_1k_2} \mathbf{W}_1^{m_1k_1} \tag{1.2.6}$$

应指出，计算步骤可以按相反次序进行，即乘以旋转因子的计算可以在计算前一组 DFT 之前进行，而不是在计算这些 DFT 之后进行．对这种情况，可写成：

$$\mathbf{X}_{N_2k_1+k_2} = \sum_{m_2=0}^{N_2-1} \mathbf{W}_2^{m_2k_2} \sum_{m_1=0}^{N_1-1} (\mathbf{x}_{N_1m_2+m_1} \mathbf{W}^{m_1k_2}) \mathbf{W}_1^{m_1k_1}$$

(1.2.7)

因此，对快速富氏变换(FFT)算法一般有两种不同的形式，它们从计算的复杂性考虑是等价的。

FFT 算法是利用几个小的 DFT 的计算来代替一个大的 DFT 的计算而达到计算效率的提高的。因为当 N 分解成许多小的 DFT 时，运算次数迅速减少。易知，若按(1.2.5)及(1.2.6)式用 FFT 算法来计算这个 DFT 时，计算分解成 N_1 个 N_2 项的 DFT，N_2 个 N_1

项的 DFT，再加 N_1N_2 次乘以旋转因子的乘法。总的乘法次数为：

$$M = N_1N_2^2 + N_2N_1^2 + N_1N_2 = N_1N_2(N_1 + N_2 + 1) \tag{1.2.8}$$

上式显然小于直接计算 N 点 DFT 的乘法次数$N_1^2N_1^2$。这就是 N 为两个因子乘积的 FFT 算法，以下推广到一般情况。

实际上，当 N 为大的复合数时，可以反复多次使用上述步骤。在这种情况下，当 N_1 及 N_2 为复合数时，可以利用上述讨论的两因子分解法，而长度为 N_1 和 N_2 的 DFT 再一次用一个 FFT 程序来计算。利用这种方法，每一级计算都能使运算次数进一步减少。因此，当 N 为大的复合数时，这种算法非常有效。

综上可知，当 N 为大的复合数时，它总可以分解成：

$$N = r_1r_2\cdots r_m$$

其中 r_j (j= 1,2,…,m)为 2，3，4，5，7，…及其它质数。为了进一步提高计算效率，一般选取 N 为小质数乘积为佳。有时为了得到一个有规则的计算结构，可选取长度 N 为整数幂的 DFT 计算。因而出现诸如基 2 的 FFT 算法，基 4 的 FFT 算法，基 3 的 FFT 算法等等。为方便用户使用，我们选择采样点数 N 为任意的。

1.2.3 使用说明

1.2.3.1 调用形式

```
CALL  CFFTF(N,C,WSAVE)
```

1.2.3.2 参数说明

N　　整变量，输入参数，变换点数。当N为小质数乘积时，本算法高效。

C　　具有N个元素的一维复数组，输入、输出参数。输入时，存放变换的复序列。输出时，存放变换

后的结果。即

$$C(n)=\sum_{k=0}^{N-1}C(k)e^{-jnk2\pi/N}\quad(n=0,1,2,\cdots,N-1)$$

这里：　$j=\sqrt{-1}$

WSAVE　具有4N+15个元素的一维实数组，输入参数，存入三角函数值及 N 的分解因子，是 CFFTI 的对应输出结果。

1.2.3.3　引用外部过程

CFFTF1，PASSF2，PASSF3，PASSF4，PASSF5，PASSF

1.2.3.4　出错条件和返回信息

(无)

1.2.3.5　适用性和限制

本子程序可实现对任意点数之复序列的离散富氏正变换，其变换结果是复的。在调用本子程序之前，应先调用 CFFTI 程序求初始值。

由于本变换是非标准化的，故调用 CFFTF 后紧接着调用逆变换 CFFTB 程序，其结果将比原输入序列扩大 N 倍。为获得标准变换结果，可将本变换的结果除以 N。

若选取 N 为许多小质数乘积时，本算法能显著地提高计算效率。

考虑到内存容量，本程序限制 N 不得大于 2^{17}。