# 一维长序列的多维处理

李　均

**提　要**　人们经常需要对一维长序列（数字信号）进行变换或卷积。如嫌序列太长，则可采用多维处理。本文列出几种常用的一维化多维的处理方法，并给出简明的处理步骤。最后，提出一种 WFTA—FNT 复合算法，充分利用了 FNT 的简便性、快速性和精确性，而 FNT 在变换长度和字长较长时，输出模溢出而造成数值模糊的缺点，则由 WFTA 加以补偿。

对序列（或称数字信号）进行变换或卷积是最常用的信号处理方式之一。用这种方式可以求频谱、功谱、计算自相关和互相关函数，设计和实现有限脉冲响应 (FIR)和无限脉冲响应(IIR)数字滤波器，还可解差分方程等等。

有时序列很长，或者说点数很多，进行一維处理有困难，需要进行多維处理。例如需对长度为 $2^{2n}$ 的序列进行变换求频谱，而暂存器容量限定为 $2^n$ 个点，则可将序列排成方阵，每次处理 $2^n$ 个点，这时暂存器访问主存器的次数仅为一維变换时的$\frac{2}{n+1}$倍[1]。又如在费马数论变换 (FNT) 中变换长度一般规定为字长的 2（或 4）倍，当变换长度太长时，就可采用多維处理来解决此矛盾[2]。再如，近年来出现一些短序列的快速变换和卷积方法，诸如 Winograd 付氏变换算法 (WFTA)、Cook-Toom 短卷积算法和最佳卷积算法等，在短卷积情况下运算次数很少，且都有具体的计算公式和步骤，当用这些算法来处理一維长序列时，也需要将一維序列化成多維[3][4]。有时也有相反的情况，即将多維信号化成一維进行处理反而更简单些。

现在概要地介绍几种常用的一維化多維的方法和步骤，并试作比较，力求简明扼要，关于数学推导和证明，請查有关资料。最后试讨论一种适用于输出为脉冲信号的FNT和WFTA 复合卷积算法，可用于雷达等信号的脉冲压缩匹配滤波器中。

## 一、FFT 法[5]

我们知道两个序列卷积（本文中提到的卷积都指循环卷积）的 DFT 等于它们分别 DFT 的乘积，因此求序列的卷积常需先求其 DFT。在求一維 DFT 时，如序列长度为

本文1981 年 9 月 15 日收到

复合数则可将它化成多維变换，如长度是高度复合数，用此法可得到 FFT 结构。现以二維为例：设序列 $x(n), n=0,1,\cdots,N-1$. 而 $N=L\cdot M$，其 DFT 为：

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{nk} \tag{1}$$

式中 $$W_N=e^{-j\frac{2\pi}{N}}$$

用二維形式求 $X(k)$ 的步驟如下：

1. 先由 $x(n)$ 和 $X(k)$ 定义成 $L\cdot M$ 的数阵 $\hat{x}(n_1,n_2)$ 和 $\hat{X}(k_1,k_2)$，

其中
$$\begin{aligned} &n=n_1+Ln_2,\quad k=Mk_1+k_2 \\ &n_1,k_1=0,1,\cdots,L-1 \\ &n_2,k_2=0,1,\cdots,M-1 \end{aligned} \tag{2}$$

即
$$\begin{aligned} \hat{x}(n_1,n_2)&=\begin{pmatrix} \hat{x}(0,0) & \hat{x}(0,1)\cdots & \hat{x}(0,M-1) \\ \hat{x}(1,0) & \hat{x}(1,1)\cdots & \hat{x}(1,M-1) \\ \vdots & \vdots & \vdots \\ \hat{x}(L-1,0) & \hat{x}(L-1,1)\cdots & \hat{x}(L-1,M-1) \end{pmatrix} \\ &=\begin{pmatrix} x(0) & x(L)\cdots & x(N-L) \\ x(1) & x(L+1)\cdots & x(N-L+1) \\ \vdots & \vdots & \vdots \\ x(L-1) & x(2L-1)\cdots & x(N-1) \end{pmatrix} \end{aligned} \tag{3}$$

由(1)、(2)式可知

$$\begin{aligned} X(Mk_1+k_2)&=\sum_{n_1=0}^{L-1}\sum_{n_2=0}^{M-1}x(n_1+Ln_2)\,W_N^{(n_1+Ln_2)(Mk_1+k_2)} \\ &=\sum_{n_1=0}^{L-1}W_L^{n_1k_1}\,W_N^{n_1k_2}\sum_{n_2=0}^{M-1}x(n_1+Ln_2)W_M^{n_2k_2} \end{aligned} \tag{4}$$

2. 计算 $\sum_{n_2=0}^{M-1}x(n_1+Ln_2)W_M^{n_2k_2}=X'(n_1,k_2)$，即对 $x(n_1,n_2)$ 的各行进行 $L$ 次 $M$ 点的一維 DFT.

3. 将 $X'(n_1,k_2)$ 乘上旋转因子 $W_N^{n_1k_2}$.

4. 对 $W_N^{n_1k_2}X'(n_1,k_2)$ 的各列进行 $M$ 次 $L$ 点的 DFT 得数阵 $\hat{X}(k_1,k_2)$.

5. 按 (2) 式由 $\hat{X}(k_1,k_2)$ 得 $X(k)$.

这种方法将 $N$ 点长序列的 DFT 化为 $M$ 次 $L$ 点短序列的 DFT 和 $L$ 次 $M$ 点短序列的 DFT。其优点是 $L$ 和 $M$ 可为任意数，在进行二（或多）維变换时不用另外对序列补零，因此多維运算时并不增加数据处理量，而且所得到的结果是一維频谱，所以实际用得很多，尤其是当 $L$ 或 $M$ 也是高度复合数时，还可继续化短成多維，实际上这就是 FFT 算法。例如 $N=2^s$（$s$ 是正整数），用上法可得到基 $-2^T$ 的FFT 算法（$T$ 为正整数，$1\geqq T>S$）.

这种方法主要的缺点是需要多乘一个旋转因子 $W_N^{n_1k_2}$，增加了运算量。

## 二、补零多维卷积[6]

以二維为例。设所求的序列为 $x$ 与 $h$ 的 $N$ 点卷积 $y$：

$$y\langle n\rangle_N=\sum_{g=0}^{N-1}x\langle g\rangle_N h\langle n-g\rangle_N \tag{5}$$

式中 $\langle\cdot\rangle_N$ 表示对 $\langle\cdot\rangle$ 中的数以 $N$ 为模求余数。

如 $N$ 较大，且 $N=L\cdot M$，则可化一維卷积为二維。首先用 (2)、(3) 式将 $x(n)$，$h(n)$ 和 $y(n)$ 分别定义成数阵 $\hat{x}(n_1,n_2)$，$\hat{h}(n_1,n_2)$ 和 $\hat{y}(n_1,n_2)$.

由(5)式得

$$\hat{y}(n_1,n_2)=\sum_{g_1=0}^{L-1}\sum_{g_2=0}^{M-1}\hat{x}(g_1,g_2)\hat{h}(n_1-g_1,n_2-g_2) \tag{6}$$

我们知道序列 $x$ 和 $h$ 可被看作是以 $N$ 为周期而重复的，因此数阵 $h$ 沿 $n_2$ 方向也是以 $M$ 为周期而重复的。这是因为：

$$\hat{h}(g_1,g_2+M)=h\langle g_1+(g_2+M)L\rangle_N=h(g_1+g_2L)=\hat{h}(g_1,g_2)$$

$x$ 也同样。但它们沿 $n_1$ 方向却不是以 $L$ 为周期重复的，因此当采用重叠舍弃法时，为了实现二維循环卷积，$x$ 和 $h$ 还需按(7)、(8)式的方式扩充成 $2L\times M$ 的数阵 $\hat{\hat{x}}$ 和 $\hat{\hat{h}}$，才能得到满意的结果：

$$\hat{\hat{x}}(n_1,n_2)=\overbrace{\begin{pmatrix}\hat{x}(0,0) & \hat{x}(0,1) & \cdots\hat{x}(0,M-1)\\ \hat{x}(1,0) & \hat{x}(1,1) & \cdots\hat{x}(1,M-1)\\ \vdots & \vdots & \vdots\\ \hat{x}(L-1,0) & \hat{x}(L-1,1) & \cdots\hat{x}(L-1,M-1)\\ 0 & 0 & \cdots\ 0\\ \vdots & \vdots & \vdots\\ 0 & 0 & \cdots\ 0\end{pmatrix}}^{M}\begin{matrix}\left.\begin{matrix}\\ \\ \\ \\ \end{matrix}\right\}L\\ \left.\begin{matrix}\\ \\ \\ \end{matrix}\right\}L\end{matrix}$$

$$=\begin{pmatrix}x(0) & x(L) & \cdots x(N-L)\\ x(1) & x(L+1) & \cdots x(N-L+1)\\ \vdots & \vdots & \vdots\\ x(L-1) & x(2L-1) & \cdots x(N-1)\\ 0 & 0 & \cdots\ 0\\ \vdots & \vdots & \vdots\\ 0 & 0 & \cdots\ 0\end{pmatrix} \tag{7}$$

$$\hat{\hat{h}}(n_1,n_2)=\left\{\begin{array}{c}\overbrace{\begin{array}{ccccc}0 & & 0 & \cdots\cdots & 0\\ \hat{h}(-L+1,0) & & \hat{h}(-L+1,1) & \cdots\cdots & \hat{h}(-L+1,M-1)\\ \hat{h}(-L+2,0) & & \hat{h}(-L+2,1) & \cdots\cdots & \hat{h}(-L+2,M-1)\\ \vdots & & \vdots & & \vdots\\ \hat{h}(-1,0) & & \hat{h}(-1,1) & \cdots\cdots & \hat{h}(-1,M-1)\\ \hat{h}(0,0) & & \hat{h}(0,1) & \cdots\cdots & \hat{h}(0,M-1)\\ \vdots & & \vdots & & \vdots\\ \hat{h}(L-1,0) & & \hat{h}(L-1,1) & \cdots\cdots & \hat{h}(L-1,M-1)\end{array}}^{M}\end{array}\right.\begin{array}{l}\left.\begin{array}{l}\\ \\ \\ \\ \end{array}\right\}L\\ \left.\begin{array}{l}\\ \\ \\ \\ \end{array}\right\}L\end{array}$$

$$=\left(\begin{array}{ccccc}0 & & 0 & \cdots & 0\\ h(N-L+1) & & h(1) & \cdots\cdots & h(N-2L+1)\\ h(N-L+2) & & h(2) & \cdots\cdots & h(N-2L+2)\\ \vdots & & \vdots & & \vdots\\ h(N-1) & & h(-1+L) & \cdots\cdots & h(N-L-1)\\ h(0) & & h(L) & \cdots\cdots & h(N-L)\\ \vdots & & \vdots & & \vdots\\ h(L-1) & & h(2L-1) & \cdots\cdots & h(N-1)\end{array}\right) \tag{8}$$

$\hat{\hat{h}}$ 最上面一行 0 也可以不要，这是为了凑成 $2L$ 行便于计算而加上的。

它们的循环卷积是：

$$\hat{\hat{y}}(n_1,n_2)=\sum_{g_1=0}^{2L-1}\sum_{g_2=0}^{M-1}\hat{\hat{x}}(g_1,g_2)\,\hat{\hat{h}}(n_1-g_1,n_2-g_2),$$

$$\begin{array}{l}n_1=0,1,\cdots,2L-1\\ n_2=0,1,\cdots,M-1\end{array} \tag{9}$$

可以证明从(9)式得到的 $2L\times M$ 数阵 $\hat{\hat{y}}$ 的下面一半，即下面 $L\times M$ 子数阵即为所需的 $\hat{y}$.

(9) 式的二維卷积也可用二維变换来实现，即

$$\hat{\hat{y}}(n_1,n_2)=T^{-1}[T\hat{\hat{x}}(n_1,n_2)\otimes T\hat{\hat{h}}(n_1,n_2)] \tag{10}$$

式中 $T$ 和 $T^{-1}$ 分别是二維正、反变换的标记。而这里的二維变换又可用两组一維变换的乘积来实现，只要这些一維变换必须分别具有 $2L$ 和 $M$ 点的循环卷积特性 (CCP)，

如变换是 DFT，则

$$\left.\begin{aligned} T\hat{\hat{x}}(n_1,n_2)&=\hat{X}(k_1,k_2)=\sum_{n_2=0}^{M-1}\sum_{n_1=0}^{2L-1}\hat{x}(n_1,n_2)W_{2L}^{n_1k_1}W_M^{n_2k_2}\\ T^{-1}\hat{\hat{X}}(k_1,k_2)&=(2N)^{-1}\sum_{k_2=0}^{M-1}\sum_{k_1=0}^{2L-1}\hat{\hat{X}}(k_1,k_2)W_{2L}^{-n_1k_1}W_M^{-n_2k_2}\end{aligned}\right\}\qquad(11)$$

如变换是FNT，且设$F_t=\prod_{i=1}^{s}P_i^{r_i}$，则分别用$Z_{F_t}$上的$2L$和$M$次单位根$\alpha$和$\beta$来代替$W_{2L}$和$W_M$，其中$\alpha$和$\beta$对模$P_i$的次数也分别是$2L$和$M$.

此外，我们还可对(10)式进行推广：

$$\hat{\hat{y}}(n_1,n_2)=C_1C_2[A_2A_1\hat{\hat{h}}(n_1,n_2)\otimes B_2B_1\hat{\hat{x}}(n_1,n_2)]\qquad(12)$$

式中$A_1$、$B_1$和$C_1$是一组在$2L$点上具有CCP的变换，$A_2$、$B_2$和$C_2$是另一组在$M$点上具有CCP的变换。这些变换可以是DFT、FNT或矩形变换等等。其中$A$、$B$是正变换，$C$是反变换。式中$B_2B_1\hat{\hat{x}}(n_1,n_2)$的记号表示对$\hat{\hat{x}}(n_1,n_2)$的各列先做$2L$点$B_1$的变换，然后再对其结果的各行做$M$点$B_2$的变换，当然在实际情况中也可将$ABC$的下标1和2全部互换，这要看怎样的次序可使运算量更节省些[4]。

因此一維循环卷积化为二維实现的步骤可以为：

1. 按(2)、(3)式方法将一维序列$x$和$h$分别化成二維数阵$\hat{x}$和$\hat{h}$.
2. 按(7)、(8)式将二維数阵$\hat{x}$和$\hat{h}$分别扩充成$\hat{\hat{x}}$和$\hat{\hat{h}}$.
3. 按(11)式对$\hat{\hat{x}}$和$\hat{\hat{h}}$分别进行二維变换，考虑到$\hat{\hat{x}}$中有一半的行是全0，故其方法可以是：对$\hat{\hat{x}}$和$\hat{\hat{h}}$的各行先进行$2L$次$M$点的变换，然后对结果的各列再进行$M$次$2L$点的变换，分别得$\hat{\hat{X}}$和$\hat{\hat{H}}$(用矩形变换时，变换次数还要增加)。
4. 将$\hat{\hat{X}}$和$\hat{\hat{H}}$数阵中位置相当的数相乘得$\hat{\hat{Y}}$.
5. 按(11)式对$\hat{\hat{Y}}$进行反变换得$\hat{\hat{y}}$.
6. $\hat{\hat{y}}$的下面一半即$\hat{y}$，用类似(3)式的方法可求得$y(n)$.

用这种方法化成多維的优点是每維长度$L$、$M$可以是任意数，因此适合于$N$是高度复合数如$N=2^r$，同时运算过程中不必乘旋转因子。所以这是采用FNT作多維卷积时常用的一种方法。其缺点是每加一維需将被处理的数据扩展近一倍，且在处理过程中得不到一維频率特性的数据。

## 三、下标重新排列法

无论是求序列的卷积或DFT，当需要将数据由一維化多維进行处理时，都可以根据数论原理，先对输入数据的下标重新排列，然后再处理，最后再对输出数据进行重排就可得到所求的值，用这种方法运算量较节省。

先以一維卷积化二維为例。设处理长度$N=L\cdot M$，此处$L$和$M$必须是互素的整数。根据：

$$n_1=\langle q_1n\rangle_L,\quad n_2=\langle q_2n\rangle_M\qquad(13)$$

将序列$x(n)$和$h(n)$排成二維数阵$\hat{x}(n_1,n_2)$和$\hat{h}(n_1,n_2)$，然后求这两个数阵的二維卷积：

$$\hat{y}(n_1,n_2)=\sum_{g_1=0}^{L-1}\sum_{g_2=0}^{M-1}\hat{x}(g_1,g_2)\hat{h}(\langle n_1-g_1\rangle_L,\langle n_2-g_2\rangle_M) \tag{14}$$

考虑到(13)式的关系，容易证明[5](14)式恰好就是(5)式，用孙子定理：

$$n=\langle n_1 aM+n_2 bL\rangle_N \tag{15}$$

其中

$$\langle q_1 aM\rangle_L=1$$

$$\langle q_2 bL\rangle_M=1$$

可将 $\hat{y}(n_1,n_2)$还原成所求的 $y(n)$。所以如果我们有计算二維卷积的好方法，就可以用于算一維卷积，反之也可用一維卷积来算二維卷积，其关鍵是利用(13)式对 $x$、$h$ 和 $y$ 进行重排。

下面再来看一維化二維 DFT 的情况：

一維序列的 DFT：

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{nk} \tag{16}$$

将 $x(n)$重排成二維数阵$\hat{x}(n_1,n_2)$后，可求得二維DFT即$\hat{X}(k_1,k_2)$。问題是 $n$ 与 $n_1$、$n_2$ 之间以及 $k$ 与 $k_1$、$k_2$ 之间应具有什么关系才能使$X(k)$相当于$\hat{X}(k_1,k_2)$？

容易证明当

$$n_1=\langle q_1 n\rangle_L,\quad n_2=\langle q_2 n\rangle_M \tag{17}$$

以及

$$k_1=\langle ak\rangle_L,\quad k_2=\langle bk\rangle_M \tag{18}$$

式中

$$\langle q_1 aM\rangle_L=1,\quad \langle q_2 bL\rangle_M=1$$

时$X(k)$和$\hat{X}(k_1,k_2)$相当。证明如下：

根据(17)、(18)式和孙子定理

$$n=\langle n_1 aM+n_2 bL\rangle_N,\quad k=\langle k_1 q_1 M+k_2 q_2 L\rangle_N \tag{19}$$

$$\begin{aligned}W^{nk}&=W_N^{\langle n_1 aM+n_2 bL\rangle_N\langle k_1 q_1 M+k_2 q_2 L\rangle_N}\\&=W_N^{\langle n_1 k_1 q_1 aM^2\rangle_N}\cdot W_N^{\langle n_2 k_2 q_2 bL^2\rangle_N}\\&=W_L^{\langle n_1 k_1\rangle_L}\cdot W_M^{\langle n_2 k_2\rangle_M}\\&=W_L^{n_1 k_1}\cdot W_M^{n_2 k_2}\end{aligned} \tag{20}$$

(16)式按(17)、(18)和(20)式可化成：

$$\hat{X}(k_1,k_2)=\sum_{n_2=0}^{M-1}\sum_{n_1=0}^{L-1}\hat{x}(n_1,n_2)W_L^{n_1k_1}W_M^{n_2k_2} \tag{21}$$

这就是二維变换式。就是说，当输入和输出数据分别按(17)和(18)式（或换过来分别按(18)和(17)式也可）排列时，一維变换可化成二維变换形式。当然据此也可将二維变换化成一維来做。以上的变换也可以采用 FNT，这时用 $\alpha$ 和 $\beta$ 来代替 $W_L$ 和$W_M$，也可以是矩形变换，如(12)式所示，但是现在输入数据的数阵是 $L\times M$ 而不是 $2L\times M$.

用这种方法进行循环卷积的步驟是：

1. 按(13)式分别将 $x(n)$和 $h(n)$重新排成$\hat{x}(n_1,n_2)$和$\hat{h}(n_1,n_2)$.

2. 对 $\hat{x}$ 和 $\hat{h}$ 分别进行二維变换，即先对其各行进行 $L$ 次 $M$ 点的一維变换，再对变换结果的各列进行 $L$ 点一維变换（也可先列后行），得$\hat{X}(k_1,k_2)$和$\hat{H}(k_1,k_2)$.

3. 如还需求 $x(n)$的 DFT $X(k)$，则第二步中的变换必须是基于 DFT 的变换，这

时对$\hat{X}(k_1,k_2)$的数据按(19)式重排即可得$X(k)$.

4. 将$\hat{X}(k_1,k_2)$和$\hat{H}(k_1,k_2)$中位置相当的数相乘得$\hat{Y}(k_1,k_2)$.

5. 对$\hat{Y}(k_1,k_2)$作二維反变换，即先对其列作一維的反变换，再对此结果的行作一維反变换（也可先行后列），得$\hat{y}(n_1,n_2)$.

6. 按(15)式将二維数阵复原就得$y(n)$.

这种方法与前两种比较优点突出，即既不用乘旋转因子，又不必补零扩展数据，因此可节省运算量，同时又可获得输入数据的一維频谱。其缺点是$N$的两个因子$L$和$M$必须互素（多維时多个因子间必须互素），因此不能用于类似$N=2^r$的情况，此外还必须对输入和输出数据重新排列。

在具体实现这种方法时也可采用矩阵分块法[3]，但其原理与上述差不多，故不另阐述。

## 四、FNT 和矩形变换的复合算法

FNT 的优点是很吸引人的，它不但沒有运算误差而且可用移位和加法来代替 DFT 中乘$W_N^{nk}$的乘法，运算简便得多。遗憾的是各参数间互相制约较严，如变换长度一般为字长的 2 或 4 倍，因此最大变换长度往往受到字长的限制，而且最小字长又受到输出最大值模溢出的限制[2].

矩形变换用于短序列的卷积，乘法次数很少，运算也很简便，但它只适用于短序列，变换长度很短[4].

在一維卷积化作多維处理时可将这两种变换结合起来，扬长避短。用FNT处理其中的一維数据，其长度根据需要和FNT的可能，尽量长些，其长度不足之数由其他維用矩形变换补足。

现以一維化二維卷积为例，说明其步骤：

1. 选$N=L\cdot M$. 如FNT字长32位，可选$L=4\times32=128$. 选$M$时还应考虑到$L$、$M$互素的条件。

2. 将$x$和$h$按(13)式重排成$\hat{x}$和$\hat{h}$.

3. 作二維卷积：

$$\hat{y}(n_1,n_2)=F_{128}^{-1}C[AF_{128}\hat{h}(n_1,n_2)\otimes BF_{128}\hat{x}(n_1,n_2)] \tag{22}$$

这里对$\hat{x}$和$\hat{h}$先作FNT的正变换，然后再作$A$、$B$矩形变换，是为了减少运算量，否则$\hat{x}$和$\hat{h}$经矩形变换后，数阵被扩大了，这时再作FNT，运算量势必增加。

4. 按(15)式将$\hat{y}(n_1,n_2)$恢复成$y(n)$.

这种方法的优点是：既能利用FNT和矩形变换运算简单、运算精度高的特点，又能使它们克服变换长度短的缺点，即把它们组合起来，扩大长度以满足需要。因FNT和矩形变换都是在实数域上进行的，故更适用于处理实数信号。

现将用不同的变换方法进行卷积时，平均每点所需的加法和乘法次数列表比较如下[4]：虽然FNT法效率的提高主要取决于专用硬件，但从表中也可看出采用FNT复合算法的优越性是很大的。

| FFT(基—2、4、8) | | | 下标重排、矩形变换 | | | 128 点 FNT 与矩形变换复合算法 | | | |
|---|---|---|---|---|---|---|---|---|---|
| $N$ | 实加数/点 | 实乘数/点 | $N$ | 实加数/点 | 实乘数/点 | $N$ | $\frac{N}{128}$ | 另外的实加数/点 | 实乘数/点 |
| | | | $84=4\times3\times7$ | 25.48 | 4.52 | | | | |
| | | | $120=3\times8\times5$ | 27.67 | 4.67 | | | | |
| 128 | 20.51 | 8.03 | | | | 128 | 1 | 0 | 1 |
| | | | $210=2\times3\times5\times7$ | 42.42 | 7.24 | | | | |
| 256 | 23.00 | 9.01 | | | | | | | |
| | | | $360=8\times9\times5$ | 54.75 | 8.56 | 384 | 3 | 3.66 | 1.33 |
| 512 | 25.75 | 10.00 | $504=8\times9\times7$ | 68.81 | 11.61 | 640 | 5 | 7.00 | 2.0 |
| | | | $840=3\times8\times5\times7$ | 75.67 | 12.67 | 896 | 7 | 10.28 | 2.71 |
| 1024 | 28.25 | 12.00 | | | | | | | |
| | | | $1260=4\times9\times5\times7$ | 101.61 | 16.59 | 1152 | 9 | 10.88 | 2.44 |
| 2048 | 31.25 | 13.00 | $2520=8\times9\times5\times7$ | 142.75 | 23.22 | 1920 | $3\times5$ | 13 | 2.66 |

其缺点除了得不到一維频谱外，主要的是并沒有解决 FNT 中字长与模溢出的矛盾，而这往往是 FNT 的致命弱点，关于这点略述如下：

设 FNT 字长为 $b$（含符号位），则模应为 $2^b+1$，为防止输出的多值性，最后输出值的范围不应超出模的范围。如果输出值 $y$ 有正有负，则

$$|y_{max}| \leqq 2^{b-1} \tag{23}$$

例如在脉冲压缩匹配滤波器中，输出 $y$ 是个幅度很大的脉冲，

$$y_{max}=2^{b_x}\cdot 2^{b_h}\cdot\frac{N}{2}. \tag{24}$$

式中：$b_x$ 和 $b_h$ 分别是 $x$ 和 $h$ 的字长（不含符号位）。

由(23)和(24)式得字长和$N$的关系为[7]：

$$b \geqq b_x+b_h+L_{g_2}N \tag{25}$$

在这方法中，最后一个变换是 $FNT^{-1}$，因此它必须滿足 (25) 式。当采用矩形变换来扩大$N$时，此矛盾更显得突出。

## 五、FNT 与 WFTA 的复合算法

现试用图 1 中的方法来解决脉压匹配滤波器模溢出的问题。

该法与前法一样，先将$x$和$h$分别按下标重排法排成 $\hat{x}$ 和 $\hat{h}$，然后做二維卷积得 $\hat{y}$. 但在做二維卷积时与前法有所不同：

1. 用 WFTA 代替矩形变换；
2. 将 FNT 和 WFTA 的位置互换一下；
3. 做$L$点 $WFTA^{-1}$ 或 $DFT^{-1}$ 时，本来需乘 $L^{-1}$，现将它分成$L^{-\frac{1}{2}}$和 $L^{-\frac{1}{2}}$ 分别

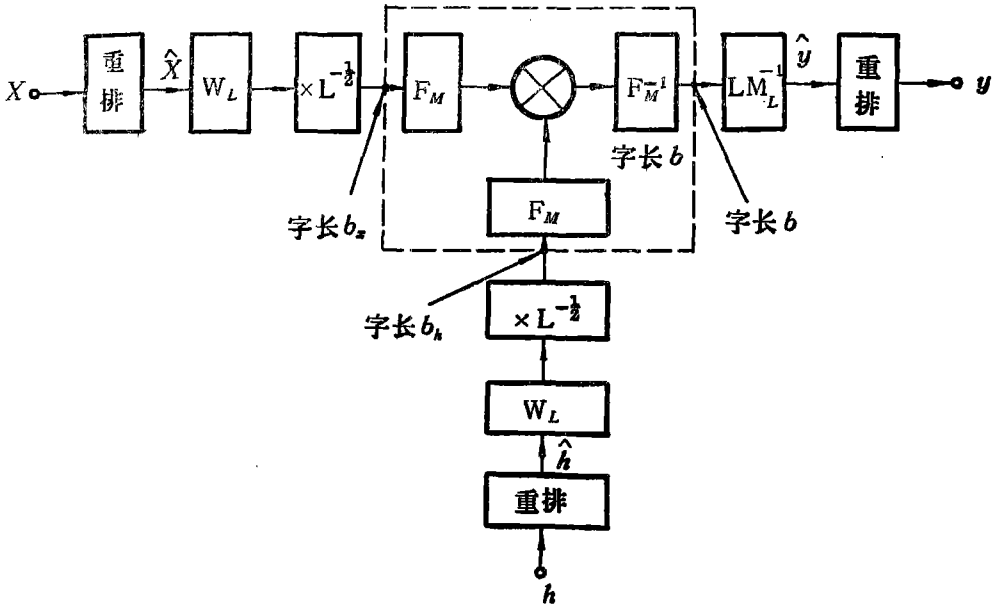



图 1

乘于两个正变换后面，这样对 $\mathrm{WFTA}^{-1}$ 而言，相当于乘上 $L$.

图 1 中的 $\hat{y}$ 可表示为：

$$\hat{y}=L\cdot W_L^{-1}F_M^{-1}\{[F_M(L^{-\frac{1}{2}})W_L\hat{h}]\otimes[F_M(L^{-\frac{1}{2}})W_L\hat{x}]\} \tag{26}$$

式中 $W_L$ 为作 $L$ 点 WFTA 的标记，

$F_M$ 为作 $M$ 点 FNT 的标记。

此式也可写成：

$$L^{-1}\cdot W_L\hat{y}=F_M^{-1}\{[F_M(L^{-\frac{1}{2}})W_L\hat{h}]\otimes[F_M(L^{-\frac{1}{2}})W_L\hat{x}]\} \tag{27}$$

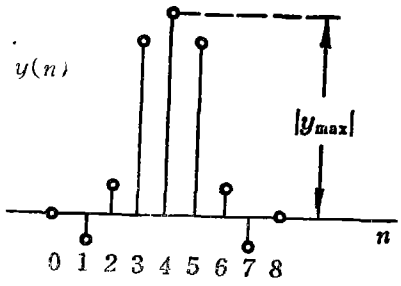



图 2

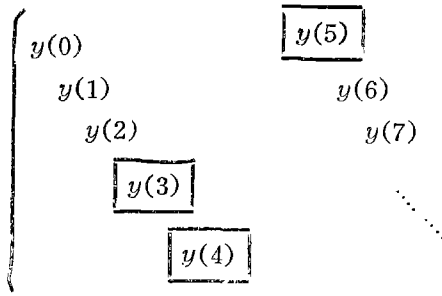
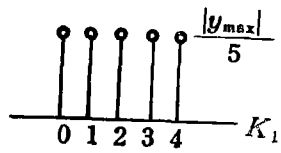



图 3

我们知道 $y$ 是脉压后输出的窄脉冲信号，例如图 2，$\hat{y}$ 是下标重新排列后的数阵。为了利用 FNT 的长处，尽量使 $M$ 大些，$L$ 短些，例如 $L=5$，从图 3 可见在 $\hat{y}$ 的每一列上最多只能有一个幅值很大的数（用□框上），其余



的数值都很小。(27)式等号左面就是对这样的列进行 $L(=5)$ 点 WFTA 并乘$\frac{1}{L}\left(=\frac{1}{5}\right)$，令其为 $Y^1(k_1,n_2)$，则

$$Y^1(k_1,n_2)=\frac{1}{L}\sum_{n_1=0}^{L-1}\hat{y}(n_1,n_2)W_L^{n_1k_1} \tag{28}$$

对固定的 $n_2$ 来说，当 $n_1$ 从 0 到 $L-1$ 时，$\hat{y}(n_1,n_2)$ 中最多只有一个值近于 $y_{max}$，其余值都很小，因此从(28)式可知 $\hat{Y}^1(k_1,n_2)$ 中数据的最大值被限制在$\frac{|y_{max}|}{L}=\left(\frac{|y_{max}|}{5}\right)$左右。如图 4 所示。

再看(27)式等号右面，最后一步是$M$点 $FNT^{-1}$，为防止模溢出，由 (23) 式可知，其输出的最大值应$\leq 2^{b-1}$。从(27)式等号左右两边情况看来，存在如下关系：

$$|y_{max}|\approx 5\cdot 2^{b-1} \tag{29}$$

与(23)式比较，可知输出的最大值可增大 5 倍，即 $L$ 倍左右。

这里选 WFTA 与 FNT 配合是为了在(27)式左面得到 $\hat{y}$ 的列的频谱，以保证 $FNT^{-1}$ 输出的结果大致具有图 4 的形状，幅度均匀分布，而不会有特别大的值，而且 WFTA 也是算短 DFT 的简便方法。

这种方法的优点是能在字长不长的条件下得到较长的 $N$，其中 FNT 和 WFTA 都还可扩大为多維，这样$N$还可扩大；而且运算比较简便，不用乘旋转因子，也不用补零以扩大数阵；FNT和短长度的 WFTA 计算都较简单，用硬件实时处理设备并不复杂；同时计算精度也较高，因 FNT 是无舍入误差的，而 WFTA 的字长可根据需要来选择。其缺点是各維的长度必须互素；从运算的中间结果里得不到一維频谱；由于 WFTA 的计算结果是复数，故适用于复数数据的卷积，否则比较浪费；同时，如有多批目标信号，可能产生混淆。

## 结 束 语

以上概要介绍了几种一維化多維的变换和卷积算法。其中 FFT 法需乘旋转因子。补零多維卷积法每加一維需将处理数据扩充一倍。在这两种方法中，对各維的长度原则上沒有什么限制，但为便于采用 FFT 的结构，常取为 2 的幂。下标重新排列法根据数论孙子定理的原理对输入、输出数据进行排列，然后直接进行多維处理，此法不必乘旋转因子，也不必扩充数据，但要求各維的长度必须互素。FNT 和矩形变换复合算法采用下标重新排列法，用矩形变换来弥补 FNT 处理长度的不足，精度高，运算简便，适用于实数信号的处理，但它沒有解决模溢出问题。最后一种是 FNT 和 WFTA 复合算法，它也采用下标重新排列法，适用于输出是脉冲信号的脉冲压缩匹配滤波器中，其中 $FNT^{-1}$ 输出的数据是最后结果的 DFT 乘以$\frac{1}{L}$($L$是 WFTA 的长度)，这意味着$FNT^{-1}$

输出的幅度仅是最后幅度的$\frac{1}{L}$倍左右，因此在一定程度上可避免模溢出问题，同时此法还较简便、精度高。

## 参 考 文 献


[1] Benjamin Arazi,"Two-dimensional digital processing of one dimensional signal",IEEE Trans. ASSP.,vol. ASSP-22,pp.81-86,Apr.1974.

[2] R.C.Agarwal and C. S. Burrus, "Fast convolution using Fermat number transforms with applications to digital filtering", IEEE Trans.ASSP.,vol.ASSP-22,pp.87-97,Apr.1974.

[3] S. Winograd, "On computing the discrete Fourier transform", Mathematics of Computation, vol.32, no.141, pp. 175-199,Jan.1978.

[4] R.C.Agarwal and J. W. Cooley, "New algorithms for digital convolution",IEEE Trans.ASSP., vol.ASSP-25, pp. 392-410, Oct. 1977.

[5] J.H.McClellan and C.M.Rader, Number Theory in Digital Signal Processing.Prentice-Hall,1979.

[6] R.C. Agarwal and C. S. Burrus, "Fast one—dimensional digital convolution by multidimensional techniques",IEEE Trans.ASSP.,vol. ASSP-22,pp.1-10,Feb.1974.

[7] 数字脉冲压缩及计算机模拟试验，梁甸农、李均，1980年12月.


# Multidimensional Processing of Long One-dimensional Sequence

Li Jun


## Abstract

It is often necessary to perform a long one-dimensional sequence transformation or convolution. If the sequence is too long to process, the multidimensional techniques can be used. This article describes some methods of multidimensional processing to replace the method of one-dimensional processing,and gives out their distinct processing procedure. On the basis of this, this paper provides a compound algorithm WFTA-FNT,it takes good advantages of simplicity,speediness and accuracy of FNT,while the restriction on sequence length imposed by word length or the ambiguous representation of the output is avoided by WFTA.