AN ALGORITHM TO REDUCE THE COMPLEXITY REQUIRED TO CONVOLVE FINITE LENGTH SEQUENCES USING THE HIRSCHMAN OPTIMAL TRANSFORM (HOT)

Victor DeBrunner

School of Electrical and Computer Engineering
The University of Oklahoma
202 West Boyd Street, Room 219
Norman, OK 73019 USA
vdebrunn@ou.edu

Ewa Matusiak

Department of Mathematics
The University of Oklahoma
601 Elm Avenue, Room 423
Norman, OK 73019 USA
ematusiak@ou.edu

ABSTRACT

We develop an algorithm suitable for convolving two finite length sequences of uneven length that is more efficient than its FFT-based competitor. In particular, we present a method for computing a fast linear convolution of the finite length sequences h and x where the length of x is much greater than the length of h using the Hirschman Optimal Transform (HOT). When compared to the most efficient methods using the DFT and its fast FFT implementation, our method can reduce the computational complexity by a third.

1. INTRODUCTION

The DFT (Discrete Fourier Transform) has the well-known convolution property that convolution in time is multiplication in the frequency domain, though, of course, the DFT performs what is most typically called "circular convolution" in the literature. However, the linear convolution of 2 finite duration sequences, define them as the N_1-point sequence h and the N_2-point sequence x, can be computed by evaluating the $(N = N_1 + N_2 - 1)$-point DFT of H with that of X of the appropriately zero-padded sequences h and x respectively. This is accomplished by multiplying the respective N DFT coefficients, followed by the computation of the N-point IDFT (Inverse DFT) of that element by element product $H. * X$, where we have used the definition that

$$H. * X \equiv H(k)\,X(k)\ \forall\, 1 \le k \le N$$

Hence we have

$$c = DFT^{-1}\left\{DFT\left\{h\right\}. * DFT\left\{x\right\}\right\}$$

for

$$c(l) = \sum_{n=1}^{N} h(n)x(l-n) \qquad l = 1, \ldots, N.$$

The required number of multiplications M and additions A are (assuming that DFT H of h is known *a priori*) are

$$M = 2\frac{N}{2}\log_2 N + N = N\log_2 N + N$$

$$A = 2N\log_2 N.$$

These formulas are taken directly from [3]. This fundamental concept forms the basis of many digital signal processing systems. In fact, some effort has been made to make this more efficient. Recently, [1] developed a method of "bit packing" that may reduce the computational complexity to that of calculating the DFT of 2 $\max(N_1, N_2)$-point sequences. This method will work on any circular convolution method, but the performance of their proposed method degrades as the number of bits increases and as the lengths of the DFT increase.

The method that we propose and study in this paper does not suffer from these drawbacks. In fact, because our method uses circular convolution of smaller sub-sequences, application of their method to our proposed method could result in even more computational savings than we report here. We will leave that to future work. However, it is important to remember that the method of linear convolution that we describe in the subsequent sections of this paper requires as little as 69% of the multiplications and 65% of the additions without any loss for any reason! These reported savings require the length of the DFT to be $N = 2^{12}$, but the savings of approximately 10% in both multiplications and additions is present for lengths on the order of $N = 2^6$. Our method also obtains more substantial savings as the disparity in the lengths of the sequence increases, i.e. as $N_2 \gg N_1$. To understand how this savings results, we first present the HOT (Hirschman Optimal Transform). Then we develop our linear convolution procedure using this transform, and then we provide some examples that show how it performs compared to the DFT implementation of linear

0-7803-7663-3/03/$17.00 ©2003 IEEE

convolution. Finally, we conclude and discuss our future work.

2. THE HIRSCHMAN OPTIMAL TRANSFORM

We repeat a few of the salient results from [2] regarding the HOT just to aid the reader. We use the K-dimensional (K-point) DFT as the originator signal for the $N \equiv LK$-dimensional (N-point) HOT basis. Each of these basis functions must then be interpolated (by K or L) and then circularly shifted to produce the complete set of orthogonal basis functions that define the HOT. As an example of this process of interpolation and shifting, we detail the process for an $N = 9$-point HOT. To start, consider the 3-point DFT

$$\begin{bmatrix} X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \omega_3 & \omega_3^2 \\ 1 & \omega_3^2 & \omega_3^4 \end{bmatrix} \begin{bmatrix} x[1] \\ x[2] \\ x[3] \end{bmatrix}$$

where $\omega_3 = e^{-j\frac{2\pi}{3}}$.

This 3-point DFT yields the 9-point HOT

$$[H(1), H(2), \ldots, H(7), H(8), H(9)]^T =$$

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & \omega_3 & 0 & 0 & \omega_3^2 & 0 & 0 \\ 0 & 1 & 0 & 0 & \omega_3 & 0 & 0 & \omega_3^2 & 0 \\ 0 & 0 & 1 & 0 & 0 & \omega_3 & 0 & 0 & \omega_3^2 \\ 1 & 0 & 0 & \omega_3^2 & 0 & 0 & \omega_3^4 & 0 & 0 \\ 0 & 1 & 0 & 0 & \omega_3^2 & 0 & 0 & \omega_3^4 & 0 \\ 0 & 0 & 1 & 0 & 0 & \omega_3^2 & 0 & 0 & \omega_3^4 \end{bmatrix} \cdot x$$

where $x = [x[1], x[2], \ldots, x[7], x[8], x[9]]^T$.

In general we have the transform relationship

$$H(L(r-1)+l) = \frac{1}{\sqrt(K)} \sum_{n=1}^{K} x[L(n-1)+l]\omega_K^{(n-1)(r-1)}$$

where $1 \leq r, l \leq K$, and its inverse

$$x[L(n-1)+l] = \frac{1}{\sqrt(K)} \sum_{r=1}^{K} H(L(r-1)+l)\omega_K^{(n-1)(r-1)}$$

where $1 \leq n, l \leq K$.

Because the HOT is based on periodic shifts of the DFT, the $N = LK$-point HOT can be accomplished using L separate K-point DFT computations. When K is a power of 2 this becomes (M multiplications and A additions):

$$M = L\frac{K}{2}\log_2 K = \frac{N}{2}\log_2 K$$

$$A = LK\log_2 K = N\log_2 K.$$

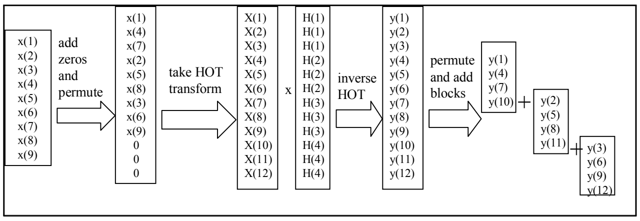

Fig. 1. Figure 1. Fast Convolution of HOT for a 9- and 2-point sequence.

3. THE MAIN IDEA

Since the HOT transform of any sequence is calculated using the DFT of several sub-sequences, we can apply concepts of the overlap-and-add method to compute the linear convolution of h and x where the length of x is greater than the length of h. Suppose that the length of the sequence h is L and that the sequence x is length Km, where $K \geq L$. Of course both K and m are integers. First we rearrange (index) the sequence x and compute its Km-point HOT transform. We also compute the K-point DFT of h, $H = DFT\{h\}$, thus h is typically zero-padded. The next step is to multiply each successive subgroup of K elements of the calculated HOT $X = HOT\{x\}$ coefficient by coefficient H. Then, we should take the Km-point inverse HOT (IHOT) of the resultant sequence and rearrange (undue the effects of the indexing of x) the terms. We can think of this vector as a collection of Km-point circular convolutions. To obtain a linear convolution of x and h we must add the last $L-1$ samples of the n^{th} K-point segment to the first $L-1$ samples of the $(n+1)^{st}$ segment, for each of the subgroup segments $n = 0, \ldots, m-1$.

This process is best exemplified for the convolution of a 9-point x with a 2-point h. Consider the flow given in Figure 1. Notice that to get the lengths correct, we must add 3 zeros to the end of x (zero pad). The multiplication in the center of the figure is an element by element multiplication. The indexing on the DFT shows the re-use of the DFT coefficients K times.

4. FAST CONVOLUTION OF TWO SEQUENCES OF FINITE LENGTH

Now lets look at the details and generalize our concept. Consider the two sequences h and x. The length of h is equal to L and length of x is equal to S. Let $m = \frac{S}{n}$ where $n > L$ and $K = L + n - 1$ and $m, n, L, K, S \in \mathbb{Z}$, i.e. they are all integers. Now, pad h with $n-1$ zeros and compute its DFT, call it $H = DFT\{h\}$. We will use the idea presented in the preceding section. Between every n elements of x and at the end of the sequence insert $L-1$ zeros to get a sequence y of length $N = nm + m(L-1) = m(n+L-1) = mK$ where $K = n + L - 1$. Arrange the elements of y by applying a permutation matrix P of size $mK \times mK$ which gives a new vector $\tilde{y}$ such that

$$\tilde{y}(b + 1 + (a-1)m) = y(a + Kb),$$
$$a = 1, \ldots, K \quad \text{and} \quad b = 0, \ldots, m-1.$$

Hence P has ones at the coordinate pairs $(b + 1 + (a-1)m, a + Kb)$ where $a = 1, \ldots, K$ and $b = 0, \ldots, m-1$ and zeros elsewhere. Compute the N-dimensional HOT of $\tilde{y}$, call it $\widetilde{Y}$, and then multiply the elements of $\widetilde{Y}$ with the elements of the expanded DFT (where each $H(i)$, $i = 1, \ldots, K$, is repeated m times)

$$\widetilde{H} = \begin{bmatrix} H(1) & \cdots & H(1) & \cdots & H(K) & \cdots & H(K) \end{bmatrix}^T$$

Now, we should evaluate the IHOT of that product $\widetilde{H}. \times \widetilde{Y}$ to get $\widetilde{Z}$. Finally, applying the inverse of the permutation P^T to $\widetilde{Z}$ we obtain z. The result of this process are several mK-point convolutions of h with n-point long sections of x put together one after the other to form z. Thus, z is a vector in $\mathbb{C}^{mK}$ and we have to project it onto the space $\mathbb{C}^{nm+L-1}$ where a convolution vector $c = h * x$ lives. Consider the linear map $T : \mathbb{C}^{mK} \to \mathbb{C}^{nm+L-1}$ given by the $mK \times (nm + L - 1)$ matrix

$$T = \begin{bmatrix} I_n & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & A & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{2n-K} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & A & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{2n-K} & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & A & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_n \end{bmatrix}$$

where I_n and I_{2n-K} are the identity matrices of dimensions $n \times n$ and $(2n-K) \times (2n-K)$ respectively and A is a $(K-n) \times (2K-2n)$ matrix of the form

$$A = \begin{bmatrix} 1 & 0 & \ldots & 0 & 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 & 0 & 1 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & & \vdots \\ 0 & 0 & \ldots & 1 & 0 & 0 & \ldots & 0 & 1 \end{bmatrix}$$

where in every row there are two ones with $K - n - 1$ zeros between them. Evaluating T at z yields the linear convolution of h with x, i.e. $c = Tz = h * x$.

5. EXAMPLE

To see that this method works without loss, we compute an example. Consider two sequences $h = [1, 2]$ and $x = [1, 3, 5, -2, -1, 7, 1, 1, 5]$. Their linear convolution

$$c = h * x = [1, 5, 11, 8, -5, 5, 15, 3, 7, 10]$$

Since the length of x is $9 = 3 \cdot 3$, we have $n = m = 3$, $K = 3 + 2 - 1 = 4$ and $N = 3 \cdot 4 = 12$. Let

$$H = DFT\left\{ \begin{bmatrix} 1 \\ 2 \\ 0 \\ 0 \end{bmatrix} \right\} = \begin{bmatrix} 1.5 \\ 0.5 - i \\ -0.5 \\ 0.5 + i \end{bmatrix}.$$

Now pad x with zeros to obtain the sequence

$$y = [1, 3, 5, 0, -2, -1, 7, 0, 1, 1, 5, 0]$$

Before applying the 12-point HOT we permute it with P,

$$P = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1 \end{bmatrix}$$

The outcome is

$$\tilde{y} = Py^T = [1, -2, 1, 3, -1, 1, 5, 7, 5, 0, 0, 0]^T$$

The 12-point HOT transform coefficients $\widetilde{Y}$ and the expansion of H are

$$\widetilde{Y} = \begin{bmatrix} 4.5 \\ 2 \\ 3.5 \\ -2 - 1.5i \\ -4.5 + 0.5i \\ -2 - 0.5i \\ 1.5 \\ 3 \\ 2.5 \\ -2 + 1.5i \\ -4.5 - 0.5i \\ -2 + 0.5i \end{bmatrix} \quad \text{and} \quad \widetilde{H} = \begin{bmatrix} 1.5 \\ 1.5 \\ 1.5 \\ 0.5 - i \\ 0.5 - i \\ 0.5 - i \\ -0.5 \\ -0.5 \\ -0.5 \\ 0.5 + i \\ 0.5 + i \\ 0.5 + i \end{bmatrix},$$

Then, we multiply $\widetilde{H}$ term by term with $\widetilde{Y}$ and compute the IHOT of $\left(\widetilde{H}. \times \widetilde{Y}\right)$

$$\tilde{z} = [1, -2, 1, 5, -5, 3, 11, 5, 7, 10, 14, 10]^T.$$

We want to rearrange the sequence into its normal order, so compute

$$z = P^T \tilde{z} = [1, 5, 11, 10, -2, -5, 5, 14, 1, 3, 7, 10]^T$$

The last step is to multiply z with the matrix T, which in this case has the form

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Consequently, we get the convolution of h with x:

$$\begin{aligned} c &= Tz = [1, 5, 11, 8, -5, 5, 15, 3, 7, 10]^T \\ &= h * x \end{aligned}$$

It is not too difficult to prove that this method works in general.

6. COMPUTATIONAL COMPLEXITY

Now, we want to consider the application of this method in a block-processing implementation of FIR filtering. In that case, we may assume that the DFT of h is known since it is the fixed filter. Thus, we need only count the computations involving x. Suppose x is an infinite-length sequence, i.e. suppose that x is the input sequence that is very long. We segment x into blocks of length nm and take the $N = mK$-point HOT . Then the number of multiplications and additions, respectively, are

$$M = mK \log_2 K + mK$$

$$A = 2mK \log_2 K + (m-1)(K-n)$$

Tables 1 and 2 list the number of operations corresponding to the calculations of the convolutions of one block of x by the DFT and HOT respectively. Note that the block lengths are slightly different for each method (DFT and HOT) due to each algorithms peak performance lengths. Consequently, we have calculated the number of multiplications M and additions A per output sample as the fairest comparison available. We have assumed that the length of h (the shorter sequence) is 15 to generate the tables. We see that the HOT algorithm reduces the number of operations (both multiplications and additions) per point when the segment length of the input sequence x is as short as 2^7. We have also noted that the computational load reduction of the HOT algorithm improves as the difference between the length of h and the length of the segments of x increases.

L (x)	M/pt.	A/pt.
64 (x=50)	8.96	15.36
128 (x=114)	8.98	15.72
256 (x=242)	9.52	16.92
512 (x=498)	10.28	18.5
1024 (x=1010)	11.15	20.27
2048 (x=2034)	12.08	22.15
4096 (x=4082)	13.04	24.08

Table 1. DFT Algorithm Computational Complexity

L (x)	N	M/pt.	A/pt.
68 (x=54)	$3 \cdot 2^5$	10.66	18.296
164 (x=150)	$3 \cdot 2^6$	8.96	15.546
264 (x=250)	$5 \cdot 2^6$	8.96	15.584
514 (x=500)	$10 \cdot 2^6$	8.96	15.612
1064 (x=1050)	$21 \cdot 2^6$	8.96	15.624
2064 (x=1050)	$41 \cdot 2^6$	8.96	15.633
4114 (x=4100)	$82 \cdot 2^6$	8.96	15.636

Table 2. HOT Algorithm Computational Complexity

7. CONCLUSIONS AND FUTURE WORK

We see that the HOT algorithm has the potential to reduce the computational burden of block processing FIR implementations by as much as 30% of the multiplications and 35% of the additions. This amazing reduction in the computational complexity may be accomplished because of the dependence of the HOT on the DFT, and the natural overlap and add nature of the sub-segments of the HOT. We are currently developing a convolutional theory of the HOT that could potentially improve our performances shown here. We are also examining overlap and save algorithms.

8. REFERENCES

[1] Z. BABIC AND D. MANDIC, "A Fast Algorithm for Linear Convolution of Discrete Time Signals," *TELSIKS 2001*, Sep 2001.

[2] T.PRZEBINDA, V.DEBRUNNER AND M.ÖZAYDIN, "The Optimal Transform For The Discrete Hirschman Uncertainty Principle," *IEEE Transactions on Information Theory*, vol. 47, no. 5, pp. 2086-2090, Jul 2001.

[3] H.J.NUSSBAUMER, *Fast Fourier Transform and Convolution Algorithms*, Springer Verlag , 1981.

[4] S. K. MITRA, *Digital Signal Processing*, The McGraw-Hill Companies, Inc., 1998.