Lesser Known FFT Algorithms

R. Tolimieri and M. An

ABSTRACT. The introduction of the Cooley-Tukey Fast Fourier transform (C-T FFT) algorithm in 1968 was a critical step in advancing the widespread use of digital computers in scientific and technological applications. Initial efforts focused on realizing the potential of the immense reduction in arithmetic complexity afforded by the FFT for computing the finite Fourier transform and convolution. On existing serial butterfly architectures, this limited implementations of the FFT to transform sizes a power of two.

The original C-T FFT algorithms and its many extensions rested mainly on additive structures of the data indexing set and divide-and-conquer strategies for reducing complexity. The relative cost of multiplications as compared with additions on these early machines motivated the study of new algorithms for reducing multiplications usually at the cost of increasing additions. These new algorithms were based on multiplicative structures of the data indexing set. They resulted in greater flexibility as compared to earlier efforts by allowing for fast small prime size computations which could be embedded in a wide collection of large transform sizes.

During the 1980s, the increasing importance of RISC and parallel computers removed some of the initial motivation for these multiplicative FFTs. Many RISC architectures featured a hardwired multiply and accumulate which permitted multiplications to be nested in additions. The goal was to arrange the computations so that most multiplications were followed by an addition. Parallel architectures placed the major algorithmic burden on controlling the data flow. The exotic data flow of the multiplicative FFTs were often incompatible or at least required an immense coding effort for efficient parallel computation. Only the need for flexible transform sizes justified continued efforts.

The recent importance of FPGAs and reconfigurable hardware renews the need to reduce multiplication counts. In this talk, we will review some of the work on multiplicative approaches introduced mainly at IBM in the 1970s and 1980s and place these results within the realm of harmonic analysis.

1. Introduction

The fast Fourier transform (FFT) algorithm has a long history dating from Gauss but has over the years been rediscovered in a variety of contexts. It is closely related to the Poisson

summation formula and the Chinese remainder theorem for polynomials. In the latter it reflects the semi-simplicity of the complex group algebra of $\mathbf{Z}/N$, $N \geqslant 2$.

In 1965 the FFT was introduced to IBM by James Cooley [7] and was the joint work of James Cooley and John Tukey. Although other FFT algorithms such as the Good-Thomas prime factor algorithm had been in use, especially in geophysics [10, 16], the great reduction in computation time for large size problems, flexibility, ease of coding, and widespread applicability of the Cooley-Tukey (CT) FFT algorithm was quickly recognized by IBM management. They saw it as the key tool for the rapidly expanding use of digital computers.

The success of the CT FFT algorithm often hid several of its disadvantages in scientific and engineering applications. In fact, from the very beginning, especially at IBM, other algorithms for computing the finite Fourier transform had been discovered and sporadically found use. Typically these algorithms increase the number of additions and decrease the number of multiplications as compared with the CT FFT.

During the 1980s, with the introduction of reduced instruction set computer (RISC) processors which are capable of nesting multiplications inside additions (pipelined dual ops), these approaches were ignored.

We will discuss two historically neglected approaches which, driven by recent technological advances, have become increasingly important, the Rader-Winograd multiplicative FFT algorithms and the reduced-polynomial transform algorithms.

Much of this work was carried out during the late 1970s and early 1980s by S. Winograd [19–22] and H. J. Nussbaumer [12, 13] and placed in a mathematical framework by several of their collaborators including L. Auslander, E. Feig [3] and at a later time by C. S. Burrus [5, 6], I. Gertner [9], and M. Rofheart [15]. Additional results and references can be found in texts [4, 8, 11, 17, 18].

Proofs of theorems stated in this presentation are in [18].

2. Multiplicative Theory

2.1. *Introduction*

Standard divide-and-conquer FFT algorithms are based on the existence of nontrivial subgroups of the natural additive group structure of the underlying indexing set. The specification of a subgroup decomposes the indexing set into cosets and the computation is decomposed relative to the coset partitioning. The importance of two-to-a-power size indexings in early programming efforts is due to the plentitude of subgroups and the simplicity of implementing the two-point Fourier transform (butterfly). Data readdressing is especially regular (striding) and was eventually hardwired into many architectures (LOAD-STORE).

The immense speed-up in run time of two-to-a-power FFT codes as compared with direct computation resulted in significant compromises in many applications. The need to zero-pad from a natural computation size to the nearest two-to-a-power size introduced increased memory requirements and less accuracy, especially in multidimensional problems. In some

applications, such as X-ray crystallography, symmetry relations on data which had played an essential role before the FFT became difficult to fully exploit inside standard FFT codes.

Moreover, the regularity of these codes caused memory call conflicts on many large vector and parallel processors. The need for fast Fourier transform codes acting on more flexible data set sizes became, by the end of the 1970s, an increasingly important goal. The most important step in achieving this goal was quickly seen to be the designing of algorithms for small prime size Fourier transforms which could then be nested in standard FFT codes.

C. Rader [14] and independently S. Winograd had developed such algorithms in the early 1970s, but for the most part these algorithms were viewed as mathematical oddities. These algorithms suffered from several disadvantages which at the time precluded their widespread adoption. Coding is difficult, with each size requiring a machine dependent special, time-consuming coding effort to implement complex data readdressing. On reduced instruction set computer (RISC) architectures the instruction set for these codes can use significant memory.

From the late 1980s codes for small prime size Fourier transforms have increasingly become part of standard programming packages. In this section we will first describe the original derivations of C. Rader and S. Winograd and then place them in a harmonic analysis framework.

2.2. *Rader algorithm*

The additive group $\mathbf{Z}/p$, p a prime, has no nontrivial subgroups, but the multiplicative group $U(p)$ of nonzero elements is cyclic. The main idea underlying the p-point FFT as described by C. Rader is that the p-point Fourier transform relative to input and output data, ordered by powers of any generator of $U(p)$, has an especially simple form: the p-point Fourier transform becomes a $(p-1)$-point cyclic convolution.

For odd prime p, the group of units $U(p^m)$ of $\mathbf{Z}/p^m$ is again a cyclic group. Using this result S. Winograd independently extended Rader's result to odd prime power sizes. Fourier transform algorithms over finite fields can be developed in a similar manner. These last FFT algorithms are important in number theoretic transforms and in error-correcting coding [1,2].

Suppose p is an odd prime and $t = p-1$. The *unit group* $U(p)$ of $\mathbf{Z}/p$ is a cyclic group of order t. Choosing a generator g of $U(p)$, we have

$$U(p) = \{1,\ g,\ \ldots,\ g^{t-1}\}.$$

Denote by π_g the permutation of $\mathbf{Z}/p$ defined by

$$\pi_g = \left(0\ 1\ g\ \cdots\ g^{t-1}\right),$$

and by P_g the $p \times p$ permutation matrix defined by

$$P_g = [\mathbf{e}_0\ \mathbf{e}_1\ \mathbf{e}_g\ \cdots\ \mathbf{e}_{g^{t-1}}]$$

where $\mathbf{e}_n$ is the p-tuple having 1 in the n-th position and 0 otherwise, $0 \leqslant n < p$. The main result of C. Rader is contained in the following theorem.

THEOREM 1.

$$P_g^{-1}F(p)P_g = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & & & \\ \vdots & & C(p) & \\ 1 & & & \end{bmatrix},$$

where $C(p)$ is the $(p-1)\times(p-1)$ skew-circulant matrix

$$C(p) = \left[v^{g^{j+k}}\right] = \begin{bmatrix} v & v^g & \cdots \\ v^g & v^{g^2} & \\ \vdots & & \\ v^{g^{t-1}} & & \end{bmatrix}, \quad v = e^{2\pi i\frac{1}{p}}.$$

EXAMPLE 1. *For $p=5$ and $g=2$,*

$$P_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

and

$$P_2^{-1}F(5)P_2 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & v & v^2 & v^4 & v^3 \\ 1 & v^2 & v^4 & v^3 & v \\ 1 & v^4 & v^3 & v & v^2 \\ 1 & v^3 & v & v^2 & v^4 \end{bmatrix}.$$

Set

$$\mathbf{H} = F(t)\begin{bmatrix} v \\ v^g \\ \vdots \\ v^{g^{t-1}} \end{bmatrix}.$$

THEOREM 2.

$$\left(1 \otimes F(t)^{-1}\right) P_g^{-1} F(p) P_g \left(1 \oplus F(t)\right) = \begin{bmatrix} 1 & t & & & \\ & & & & \\ 1 & -1 & & \mathbf{0} & \\ & & & & H_{t-1} \\ & & & & \ddots \\ & \mathbf{0} & & & \\ & & & & \\ & & & H_1 & \end{bmatrix}.$$

Consider the functions χ_n, $0 \leqslant n < t$, on $\mathbf{Z}/p$ defined by the following table.

	χ_0	χ_1	$\cdots$	χ_{t-1}
0	0	0		0
1	1	1		1
g	1	w		w^{t-1}
g^2	1	w^2		$w^{2(t-1)}$
$\vdots$				
g^{t-1}	1	w^{t-1}		$w^{(t-1)(t-1)}$

$w = e^{2\pi i \frac{1}{t}}$.

TABLE 1. Multiplicative characters

Since H_n, $0 \leqslant n < t$, can be written as

$$H_n = \begin{bmatrix} 1 & v & v^g & \dots & v^{g^{t-1}} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ w^n \\ \vdots \\ w^{(t-1)n} \end{bmatrix},$$

we have the following result.

THEOREM 3.

$$H_n = F(p)(\chi_n)(1), \quad 0 \leqslant n < t.$$

2.3. *Multiplicative characters*

Table 1 describes the multiplicative characters of $\mathbf{Z}/p$. Such characters were introduced by Gauss and studied in his work on reciprocity laws. The complex constants H_n, $0 \leqslant n < t$, are called the Gauss sums of multiplicative characters.

The general definition is given as follows. A mapping $\chi : U(p) \to \mathbf{C}$ is called a multiplicative character mod p if χ is a homomorphism from the multiplicative group $U(p)$ into the multiplicative group of nonzero complex numbers,

$$\chi(ab) = \chi(a)\chi(b), \quad a, b \in U(p).$$

A multiplicative character mod p is completely determined by its value on the generator g of $U(p)$ by

$$\chi(g^m) = \chi(g)^m, \quad 0 \leqslant m < t.$$

Since $g^t = 1$, $\chi(g)$ is a t-th root of unity. For $0 \leqslant n < t$, denote by χ_n the multiplicative character defined by

$$\chi_n(g) = w^n \tag{2.1}$$

and observe that the multiplicative characters χ_n, $0 \leqslant n < t$, defined by (2.1) coincide with the functions in Table 1.

Denote by $L^2(\mathbf{Z}/p)$, the inner product space of all complex valued functions on $\mathbf{Z}/p$ with inner product

$$\langle f, g\rangle = \sum_{n=0}^{p-1} f(n)g^*(n), \quad f,\ g \in L(\mathbf{Z}/p),$$

where $*$ denotes complex conjugation.

Set $r = \frac{t}{2}$.

THEOREM 4.

$$\chi_r(g) = -1$$

and

$$\chi_n^* = \chi_{t-n}, \quad 0 \leqslant n < t.$$

THEOREM 5. *For any two multiplicative characters χ and χ', we have*

$$\langle \chi, \chi'\rangle = \begin{cases} t, & \chi = \chi' \\ 0, & \chi \neq \chi'. \end{cases}$$

Extend the domain of definition of each multiplicative characters χ to $\mathbf{Z}/p$ by setting $\chi(0) = 0$. Set $\mu_n = t^{-\frac{1}{2}}\chi_n$, $0 \leqslant n < t$.

THEOREM 6. *The set*

$$e_0,\ \mu_0,\ \mu_1,\ \ldots,\ \mu_{t-1}$$

is an orthonormal basis of $L^2(\mathbf{Z}/p)$.

Since $F(n)^2 f(x) = nf(-x)$ we have

$$F(p)^2\chi(x) = p\chi(-x) = p\chi(-1)\chi(x),$$

implying that the multiplicative characters are eigenvectors of $F(p)^2$,

$$F(p)^2\chi = p\chi(-1)\chi$$

and that the space spanned by

$$\{\chi, F(p)\chi\}$$

is invariant under the action of $F(p)$. The action of $F(p)$ on the multiplicative characters is described in the next two theorems.

THEOREM 7.

$$\begin{aligned} F(p)e_0 &= e_0 + \chi_0. \\ F(p)\chi_0 &= te_0 - \chi_0. \end{aligned}$$

For any nontrivial multiplicative character χ, we have

THEOREM 8.

$$F(p)\chi = \sqrt{p}G_p(\chi)\chi^*,$$

where

$$F(p)\chi(1) = \sqrt{p}G_p(\chi).$$

Proof

$$\begin{aligned} F(p)\chi(x) &= \sum_{y\in U(p)} \chi(y)e^{2\pi i\frac{xy}{p}} \\ &= \sum_{y\in U(p)} \chi(x^{-1}y)e^{2\pi i\frac{y}{p}} \\ &= \chi(x^{-1})\sum_{y\in U(p)} \chi(y)e^{2\pi i\frac{y}{p}} \\ &= \chi(x^{-1})F(p)\chi(1). \end{aligned}$$

By $||F(p)f||^2 = p||f||^2$, we have

$$|G_p(\chi)| = 1.$$

Relative to the orthonormal basis e_0, μ_0, μ_1, ..., μ_t of $L^2(\mathbf{Z}/p)$, we have

$$\begin{aligned} F(p)e_0 &= e_0 + t^{\frac{1}{2}}\mu_0, \\ F(p)\mu_0 &= t^{\frac{1}{2}}e_0 - \mu_0 \end{aligned}$$

and

$$F(p)\mu_n = \sqrt{p}G_p(\chi)u_{t-n}, \quad 1 \leqslant n < t.$$

The matrix of $F(p)$ relative to this basis is given by

$$\begin{bmatrix} 1 & t^{\frac{1}{2}} & & & & \\ t^{\frac{1}{2}} & -1 & & 0 & & \\ & & & & & \sqrt{p}G_p(\chi_{t-1}) \\ & & & & \iddots & \\ & 0 & & \iddots & & \\ & & \sqrt{p}G_p(\chi_1) & & & \end{bmatrix}.$$

This matrix representation of $F(p)$ is essentially the same as that in theorem 2. It is the first step in several studies into the p-point Fourier transform [3]. Other studies include:

- construction of an orthonormal basis diagonalizing $F(p)$,
- determining rational subspaces of $F(p)$.

3. Multidimensional Algorithm

3.1. *Introduction*

The reduced transform algorithms (RTA) [9, 15, 17] compute a multidimensional Fourier transform by first projecting the multidimensional input data onto lines (more generally, lower dimensional planes) and then computing the one-dimensional Fourier transform of the projected data. By the periodic version of the projection slice theorem from tomography, these one-dimensional Fourier transforms compute the multidimensional Fourier transform. The RTA can be viewed as periodic Radon transforms.

Polynomial rings play many roles in algorithm design. Linear convolution can be defined as polynomial product and cyclic convolution mod N can be defined as polynomial product mod $x^N - 1$. Fourier transform can be defined over certain quotient polynomial rings. The Chinese remainder theorem (CRT) for polynomial rings is especially important in the Cook-Toom and Winograd convolution algorithms [18, 22]. The convolution theorem which diagonalizes cyclic convolution mod N relative to the N-point Fourier transform basis can be viewed as the CRT applied to the quotient polynomial ring $\mathbf{C}[x]/x^N - 1$,

$$\mathbf{C}[x]/x^N - 1 \tilde{\equiv} \mathbf{C}[x]/x - 1 \oplus \mathbf{C}[x]/x - v \oplus \cdots \oplus \mathbf{C}[x]/x - v^{N-1},$$

where $v = e^{2\pi i \frac{1}{N}}$.

Polynomial ring theory is central to many of the results in the works of H. J. Nussbaumer [12, 13], both for multidimensional convolution and Fourier transform (polynomial transform). One goal in these works is to rely as much as possible on shifts and cyclic shifts

to carry out these computations. Polynomial transforms compute multidimensional convolutions by first using the CRT to decompose the computation into subcomputations, some of which can be computed by Fourier transform computations over rings and then completed by using the inverse CRT isomorphism to combine the subcomputations.

The RTA can be adapted to computing multidimensional convolution and provides a geometric setting for the Nussbaumer polynomial transform. A multidimensional convolution can be computed by RTA which first projects the multidimensional data onto lines, computes the convolution of the one-dimensional data and then combines the subcomputations. It differs from the RTA for multidimensional Fourier transform in that the last step does not rely on the projection slice theorem. We will address this last problem by defining an inverse projection algorithm which reconstructs multidimensional data from its one-dimensional order projections.

In some ways this is a curious result since, in the continuous case, the inversion requires either Fourier transform computations or convolutions and back projections and can be extremely costly. However a close inspection of these stages in the Nussbaumer polynomial transform algorithm shows that this step can be accomplished using only additions. We will show, using the constructions underlying the RTA, how this last step can be implemented using integer matrix multiplication.

3.2. *Periodic back projection*

Consider $\mathbf{Z}/3 \times \mathbf{Z}/3$. By a line in $\mathbf{Z}/3 \times \mathbf{Z}/3$, we mean a maximal cyclic subgroup of $\mathbf{Z}/3 \times \mathbf{Z}/3$. There are 4 distinct lines in $\mathbf{Z}/3 \times \mathbf{Z}/3$,

$$\begin{aligned} L(1,j) &= \{(0,0),\ (1,j),\ (2,j)\}, \quad 0 \leqslant j < 3. \\ L(0,1) &= \{(0,0),\ (0,1),\ (0,2)\}. \end{aligned}$$

Suppose $f \in L(\mathbf{Z}/3 \times \mathbf{Z}/3)$. The periodizations of f orthogonal to the 4 lines can be constructed as follows. Denote by $\mathbf{f}$ the lexicographically ordered two dimensional array f

$$\mathbf{f} = \begin{bmatrix} f(0,0) \\ f(1,0) \\ f(2,0) \\ f(0,1) \\ f(1,1) \\ f(2,1) \\ f(0,2) \\ f(1,2) \\ f(2,2) \end{bmatrix}.$$

The periodization orthogonal to $L(0,1)$ consists of the sums

$$f(0,0) + f(1,0) + f(2,0),$$

$$f(0,1) + f(1,1) + f(2,1),$$
$$f(0,2) + f(1,2) + f(2,2),$$

which we can write as

$$(I_3 \otimes 1_3^t)\mathbf{f}.$$

The periodization orthogonal to $L(1,0)$ consists of the sums

$$f(0,0) + f(0,1) + f(0,2),$$
$$f(1,0) + f(1,1) + f(1,2),$$
$$f(2,0) + f(2,1) + f(2,2),$$

which we can write as

$$(I_3 \otimes 1_3^t)P(9,3)\mathbf{f} = (1_3^t \otimes I_3)\mathbf{f}.$$

Set

$$S_3 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

and $Z_3 = I_3 \oplus S_3 \oplus S_3^2$. The periodization orthogonal to $L(1,1)$ is given by

$$(I_3 \otimes 1_3^t)P(9,3)Z_3\mathbf{f} = (1_3^t \otimes I_3)Z_3\mathbf{f}$$

and the periodization orthogonal to $L(1,2)$ is given by

$$(I_3 \otimes 1_3^t)P(9,3)Z_3^2\mathbf{f} = (1_3^t \otimes I_3)Z_3^2\mathbf{f}.$$

We assume that we know

$$\begin{bmatrix} I_3 \otimes 1_3^t \\ 1_3^t \otimes I_3 \\ (1_3^t \otimes I_3)Z_3 \\ (1_3^t \otimes Z_3^2 \end{bmatrix} \mathbf{f} = A\mathbf{f}$$

and we want to compute $\mathbf{f}$. We will show

$$A^T A\mathbf{f} = (9I_3 + E_3 \otimes E_3)\mathbf{f},$$

where E_3 is the 3×3 matrix of all ones.

Since $(E_3 \otimes E_3)\mathbf{f}$ is known from the sum

$$\sum_{j=0}^{2}\sum_{k=0}^{2} f(j,k),$$

We can compute $\mathbf{f}$ from $A^T A\mathbf{f}$.

The following table lists the matrices in A, their transposes and the products of the matrices with their transposes. $A^T A$ is the sum of the matrices in the last column.

X	X^T	X^TX
$I_3 \otimes 1_3^t$	$I_3 \otimes 1_3$	$I_3 \otimes E_3$
$1_3^t \otimes I_3$	$1_3 \otimes I_3$	$E_3 \otimes I_3$
$(1_3^t \otimes I_3)Z_3$	$Z_3^2(1_3 \otimes I_3)$	$\begin{bmatrix} I_3 & S_3 & S_3^2 \\ S_3^2 & I_3 & S_3 \\ S_3 & S_3^2 & I_3 \end{bmatrix}$
$(1_3^t \otimes I_3)Z_3^2$	$Z_3(1_3 \otimes I_3)$	$\begin{bmatrix} I_3 & S_3^2 & S_3 \\ S_3 & I_3 & S_3^2 \\ S_3^2 & S_3 & I_3 \end{bmatrix}$

Since $I + S_3 + S_3^2 = E_3$, A^TA is as claimed.

References

[1] R. C. Agarwal and C. S. Burrus, "Fast convolution using Fermat number transforms with applications to digital filtering," *IEEE Trans. ASSP* **ASSP22**, 87-97, 1974.

[2] R. C. Agarwal and C. S. Burrus, "Number theoretic transforms to implement fast digital convolution," *Proc. IEEE* **63**, 550-560, 1975.

[3] L. Auslander, E. Feig and S. Winograd, "The multiplicative complexity of the discrete Fourier transform," *Adv. in Appl. Math.* (5), 31-55, 1984.

[4] R. E. Blahut, *Fast Algorithms for Digital Signal Processing*, Addison-Wesley, Reading, MA 1985.

[5] C. S. Burrus, "An in-place, in-order prime factor FFT algorithm," *IEEE Trans. ASSP*, **ASSP29**, 806-817, 1979.

[6] C. S. Burrus and T. W. Park, *DFT/FFT and Convolution Algorithms*, Wiley and Sons, New York, 1985.

[7] J. Cooley and J. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comp.*, **19** 297-301, 1965.

[8] Hari Krishna Garg, *Digital Signal Processing: Number theory, convolution, fast Fourier Transforms, and applications*, CRC Press, London, 1998.

[9] I. Gertner, "A new efficient algorithm to compute the two-dimensional discrete Fourier transform," *IEEE Trans. ASSP* **ASSP36**(7), 1036-1050, 1988.

[10] I. J. Good, "The interaction algorithm and practical Fourier analysis," *J. Royal Statistics*, **B**(2), 361-375, 1958.

[11] J. H. McClellan and C. M. Rader, *Number theory in Digital Signal Processing*, Prentice Hall, 1979.

[12] H. J. Nussbaumer, "Fast polynomial transform algorithms for digital convolution," *IEEE Trans. ASSP* **ASSP28**, 205-215, 1980.

[13] H. J. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms*, Springer-Verlag, Berlin, Heidelberg and New York, 1981.

[14] C. Rader, "Discrete Fourier transform when the number of data samples is prime," *Proc. IEEE*, **56**, 1107-1108, 1968.
[15] M. Rofheart, *Algorithms and Methods for Multidimensional Digital Signal Processing*, PhD Thesis, the City University of New York, 1991.
[16] L. H. Thomas, "Using computers to solve problems in physics," *Applications of Digital Computers*, Ginn and Co., Boston, 1963.
[17] R. Tolimieri, M. An and C. Lu, *Mathematics of Multidimensional Fourier Transform Algorithms 2nd ed.*, Springer, New York 1997.
[18] R. Tolimieri, M. An and C. Lu, *Algorithms for Discrete Fourier Transform and Convolution, 2nd ed.*, Springer, New York 1997.
[19] S. Winograd, "On computing the discrete Fourier transform," *Math. of Computation*, **32**(141) 175-199, 1978.
[20] S. Winograd, "On computing the discrete Fourier transform," *Proc. Nat. Acad. Science, USA*, **73**(4), 1005-1006, 1976.
[21] S. Winograd, "Complexity of computations," *CBMS Regional Conf. Ser. in Math.*, **33** Philadelphia, 1980.
[22] S. Winograd, "On multiplication of polynomials modulo a polynomial," *SIAM J. on Computing* **9**(2), 225-229, 1980.