A probabilistic algorithm for verifying polynomial middle product in linear time

Pascal Giorgi

A probabilistic algorithm for verifying polynomial middle product in linear time

Pascal Giorgi

LIRMM, University of Montpellier, CNRS, Montpellier, FRANCE

Abstract

Polynomial multiplication and its variants are a key ingredient in effective computer algebra. While verifying a polynomial product is a well known task, it was not yet clear how to do a similar approach for its middle product variant. In this short note, we present a new algorithm that provides such a verification with the same complexity and probability that for the classical polynomial multiplication. Furthermore, we extend our algorithm to verify any operations that compute only a certain chunk of the product, which is the case for instance of the well known short product operation.

Keywords: probabilistic algorithm; design of algorithm; computation on polynomials; polynomial multiplication;

1. Introduction

Polynomial multiplication is a fundamental tool in computer algebra as it often plays a central role in most efficient algorithms. In some cases, one may not need to compute the whole result of the product and this can be taken into account to speed up the computation. For instance, when dealing with truncated power series one need to only compute the lowest part of the polynomial multiplication. The latter operation is also referenced as short product in [1]. Another situtation occurs within polynomial division or inversion where only the middle terms of a specific product are needed [2, 3, 4]. This specific operation is called the middle product in [2].

Let $F, G \in \mathbb{K}[X]$ be two polynomials defined over a field $\mathbb{K}$ such that $\deg F = s-1, \deg G = 2s-2$. The middle product of FG denoted by $\mathsf{MP}_s(F,G)$ corresponds to the coefficients of degree $s-1$ to $2s-2$ from the product FG. Let $FG = \sum_{i=0}^{3s-3} h_i X^i$ then $\mathsf{MP}_s(F,G) = h_{s-1} + h_s X + h_{s+1}X^2 + \cdots + h_{2s-2}X^{s-1}$. Let $M(n)$ denote the complexity function for the multiplication of two polynomials of $\mathbb{K}[X]$ of degree at most n. Computing $\mathsf{MP}_s(F,G)$ through a full product requires $2M(s)+O(s)$ operations in $\mathbb{K}$. As shown in [2], dedicated algorithms can compute $\mathsf{MP}_s(F,G)$ twice faster. One remarkable property of middle product is to be the transposed problem of polynomial multiplication using the Tellegen principle [5]. This strong result tells us that every polynomial multiplication algorithm can be turned into an algorithm for middle product with the same asymptotic complexity, i.e. $M(s)+O(s)$. Since the seminal work of Karatsuba [6], many fast polynomial multiplication algorithms have been designed in order to reach a quasi-linear time complexity [7, Chapter 8]. As of today, the best result over finite fields is $O(d \log d \, 8^{\log^* d} \log p)$ operations[1] for the product of degree d polynomials [8]. A common feature of all these algorithms is to be much more complex than the naive product, meaning their implementation could be complicated and errors prone. Using Tellegen principle to derive a middle product algorithm introduces another level of difficulty that might further complicate its implementations.

A classic way to check computations is to use *a posteriori* verification. The idea is to provide an algorithm that can check the result with an asymptotically better complexity than the operation itself. The simplicity of the algorithm must ensure its implementation's robustness. Such a verification is of great interest when one wants to check a computation from an untrusted cloud server. In order to check a polynomial product FG one can pick a random point α and check that $F(\alpha)G(\alpha) = (FG)(\alpha)$. If not, it is clear that the product is wrong. If the results agree, it is well known through Zippel-Schwartz-Lipton-DeMillo lemma [9, 10, 11] that the product FG is correct with a probability greater than $1-\frac{d}{N}$ where N corresponds to the number of sampling points for α and $\deg FG < d$. Assuming $N > d$, one can decrease the probability to $1-\frac{d^k}{N^k}$ by picking k different points. One advantage of this verification is that polynomial evaluation has a linear time complexity and can be implemented easily through Horner's rules.

To the best of our knowledge, the verification of the middle product has not been investigated yet and we provide a similar linear time algorithm for it. One motivation of this work came from our experiment to compute the kernel of a large sparse matrix arising in discrete logarithm computation. In particular, one part of the computation was relying on polynomial middle product with matrix coefficients [12]. Unfortunately, our code failed to produce correct results when polynomial degrees were above 500 000. Since quadratic time verification was not feasible, we decided to develop a fast approach. Note that our algorithm might also be of interest for the recent Middle-Product Learning With Error problem [13].

We start the next section by giving a matrix interpretation to the verification of polynomial product. Using this interpretation, we will define in the following sections

[1] $\log^*$ is the iterated logarithm function

our probabilistic verification for the middle product. Finally, in the last section we show how our method easily extends to the short product and any other operations that compute any partial chunk of a polynomial product.

2. Certifying Polynomial Multiplication

Let $F, G \in \mathbb{K}[X]$ where $F = f_0 + f_1X + \cdots + f_{m-1}X^{m-1}$ and $G = g_0 + g_1X + \cdots + g_{n-1}X^{n-1}$. Assuming F is fixed, the product $H = FG = \sum_{i=0}^{m+n-2} h_iX^i$ can be described through a linear application from $\mathbb{K}^n$ to $\mathbb{K}^{m+n}$. The matrix for this application corresponds to a Toeplitz matrix built from the coefficients of F. Let us denote $\mathcal{A}_F$ such a matrix, the product of F by G correspond to the following matrix-vector product :

$$\underbrace{\begin{pmatrix} f_0 & & \\ f_1 & \ddots & \\ \vdots & \ddots & f_0 \\ f_{m-1} & & f_1 \\ & \ddots & \vdots \\ & & f_{m-1} \end{pmatrix}}_{\mathcal{A}_F} \times \underbrace{\begin{pmatrix} g_0 \\ g_1 \\ \vdots \\ g_{n-1} \end{pmatrix}}_{v_G} = \underbrace{\begin{pmatrix} h_0 \\ \\ h_1 \\ \\ \vdots \\ \\ h_{m+n-2} \end{pmatrix}}_{v_H} \tag{1}$$

where $\mathcal{A}_F \in \mathbb{K}^{(m+n-1)\times n}$, $v_G \in \mathbb{K}^n$ and $v_H \in \mathbb{K}^{m+n-1}$.

A classic way to certify the product $H = FG$ is to choose a random α from a finite subset $S \subset \mathbb{K}$ and to check $H(\alpha) = F(\alpha)G(\alpha)$. Of course, some values of α may lead to a positive answer while $H \neq FG$. However, the number of such α is at most $\deg H$ as they correspond to the roots of the polynomial $(H - FG) \neq 0$ over the field $\mathbb{K}$. The probability of success is then greater than $1 - \frac{\deg H}{|S|}$, which corresponds exactly to the Zippel-Schwartz-Lipton-DeMillo lemma [9, 10, 11] on univariate polynomials. This approach reduces the verification to three polynomial evaluations and one product and thus has a linear time complexity of $O(\deg F + \deg G + \deg H)$.

Using the matrix version for polynomial product depicted in Equation (1), this latter approach corresponds exactly to multiplying both parts of the equation on the left by the row vector $\vec{\alpha} = [1, \alpha, \alpha^2, \ldots, \alpha^{m+n-2}]$. By definition of v_H, we clearly have $\vec{\alpha} \cdot v_H = H(\alpha)$. Using the Toeplitz structure of the matrix $\mathcal{A}_F$ we have $\vec{\alpha}\mathcal{A}_F = F(\alpha)[1, \alpha, \ldots, \alpha^{n-1}]$, which gives $(\vec{\alpha}\mathcal{A}_F) \cdot v_G = F(\alpha)G(\alpha)$. The probability result can be retrieved with the specific Freivalds certificate for matrix multiplication given in [14].

3. Certifying Middle Product

In order to illustrate our strategy we start this section with an example. Let A, B be two polynomials of $\mathbb{K}[X]$ of degree respectively 3 and 6, with $A = a_0 + a_1X + a_2X^2 + a_3X^3$ and $B = b_0 + b_1X + b_2X^2 + b_3x^3 + b_4X^4 + b_5X^5 + b_6X^6$. We want to compute $C_M = c_3 + c_4X + c_5X^2 + c_6X^3$ where $C = AB = \sum_{i=0}^{9} c_iX^i$. Using Equation (1) one can easily remark that the middle product operation corresponds to using only certain rows of the linear application for the full multiplication by A. Equation (2) illustrates this remark on our example. The grey area highlights the rows used by the middle product operation. One may note that this is an important observation in Tellegen transposition principle for the middle product [5].

$$\begin{pmatrix} a_0 & & & & & & \\ a_1 & a_0 & & & & & \\ a_2 & a_1 & a_0 & & & & \\ a_3 & a_2 & a_1 & a_0 & & & \\ & a_3 & a_2 & a_1 & a_0 & & \\ & & a_3 & a_2 & a_1 & a_0 & \\ & & & a_3 & a_2 & a_1 & a_0 \\ & & & & a_3 & a_2 & a_1 \\ & & & & & a_3 & a_2 \\ & & & & & & a_3 \end{pmatrix} \times \underbrace{\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{pmatrix}}_{v_B} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{pmatrix} \tag{2}$$

In order to certify the coefficients of the middle product $\mathsf{MP}_4(A, B) = c_3 + c_4X + c_5X^2 + c_6X^3$, one can multiply the grey part of equation (2) with the vector $[1, \alpha, \alpha^2, \alpha^3]$ with $\alpha \in \mathbb{K}$. In particular, this corresponds to certifying that $[1, \alpha, \alpha^2, \alpha^3] \cdot [c_3, c_4, c_5, c_6]^T = c_M(\alpha)$ is equal to

$$\gamma = \begin{pmatrix} 1 \\ \alpha \\ \alpha^2 \\ \alpha^3 \end{pmatrix}^T \times \begin{pmatrix} a_3 & a_2 & a_1 & a_0 & & \\ & a_3 & a_2 & a_1 & a_0 & \\ & & a_3 & a_2 & a_1 & a_0 \\ & & & a_3 & a_2 & a_1 \end{pmatrix} v_B. \tag{3}$$

More generally, let $F, G, H \in \mathbb{K}[X]$ such that $\deg F = \deg H = s - 1$, $\deg G = 2s - 2$ and $H = \mathsf{MP}_s(F, G)$. As for polynomial multiplication, fixing the polynomial F, we can define the middle product as a linear application from $\mathbb{K}^{2s-1}$ to $\mathbb{K}^s$ with the matrix $\mathcal{B}_F \in \mathbb{K}^{s\times 2s-1}$ such that

$$\mathcal{B}_F = \begin{pmatrix} f_{s-1} & f_{s-2} & \cdots & f_0 & \\ & \ddots & \ddots & & \ddots \\ & & f_{s-1} & f_{s-2} & \cdots & f_0 \end{pmatrix}.$$

Let v_G and v_H be the vector of the coefficients of G and H. By definition of the middle product we have $v_H = \mathcal{B}_F v_G$. To certify this middle product it suffices to pick a random α from a finite subset $S \subset \mathbb{K}$ and set $\vec{\alpha}_s = [1, \alpha, \ldots, \alpha^{s-1}] \in \mathbb{K}^{1\times s}$, then check the following equation:

$$H(\alpha) = (\vec{\alpha}_s\mathcal{B}_F) \cdot v_G \tag{4}$$

Lemma 3.1. *For a random $\alpha \in S \subset \mathbb{K}$, the probability Equation 4 is correct while $H \neq \mathsf{MP}_s(F, G)$ is strictly less than $\frac{s}{|S|}$.*

Proof. The correctness of Equation 4 comes from the following equality $H(\alpha) = \vec{\alpha} \cdot v_H = \vec{\alpha}(\mathcal{B}_F \cdot v_G)$. The proof of Lemma 3.1 is a direct consequence of the Zippel-Schwartz-Lipton-DeMillo lemma [9, 10, 11] remarking that both sides of Equation 4 are distinct polynomials in α with degrees bounded by s. □

The computation of $(\vec{\alpha}_s\mathcal{B}_F)\cdot v_G$ does not correspond to the product of evaluations involving both F and G. However, using the Toeplitz structure of $\mathcal{B}_F$, we are able to derive a simple algorithm that only need a linear number of operations, as explained in the next section.

4. Toeplitz Matrix-Vector Product with powers

Let $F \in \mathbb{K}[X]$ of degree $s-1$, we denote L_F and U_F the following triangular Toeplitz matrices:

$$\underbrace{\begin{pmatrix} f_{s-1} & f_{s-2} & \cdots & f_0 \\ & \ddots & \ddots & \vdots \\ & & \ddots & f_{s-2} \\ & & & f_{s-1} \end{pmatrix}}_{U_F}, \quad \underbrace{\begin{pmatrix} f_0 & & & \\ f_1 & \ddots & & \\ \vdots & \ddots & \ddots & \\ f_{s-1} & \cdots & f_1 & f_0 \end{pmatrix}}_{L_F}$$

where $F = f_0 + f_1X + \cdots + f_{s-1}X^{s-1}$ and $U_F, L_F \in \mathbb{K}^{s\times s}$.

Lemma 4.1. *Let $\vec{\alpha}_s = [1, \alpha, \ldots, \alpha^{s-1}] \in \mathbb{K}^{1\times s}$. The matrix-vector products $\vec{\alpha}_sU_F$ and $\vec{\alpha}_sL_F$ can be computed in $O(s)$ operations in $\mathbb{K}$.*

Proof. It obvious that the lemma is correct for $s = 1$. Let us assume the lemma correct for dimension $s-1$ and write $F = f_0 + X\hat{F}$, with $f_0 \in \mathbb{K}$ and $\hat{F} \in \mathbb{K}[X]$ of degree $s-2$. One can rewrite U_F as follow:

$$U_F = \begin{pmatrix} U_{\hat{F}} & f_0 \\ & \vdots \\ & f_{s-1} \end{pmatrix}.$$

There, multiplying a vector $\vec{\alpha}_s$ by U_F is equivalent to compute the row vector $[\ \vec{\alpha}_{s-1}U_{\hat{F}}\ ,\ \vec{\alpha}_s\cdot[f_0,\ldots,f_{s-1}]^T\ ]$.

From the Toeplitz structure of U_F it is easy to see that $\vec{\alpha}_s\cdot[f_0,\ldots,f_{s-1}]^T$ is equal to $\alpha y + f_0$ where y is the last column of $\vec{\alpha}_{s-1}U_{\hat{F}}$. By induction, it follows immediately that the complexity is linear in the matrix dimension s. For the matrix L_F the proof is similar remarking that

$$L_F = \begin{pmatrix} f_0 & \\ \vdots & \\ f_{s-1} & L_{(F \bmod X^{s-1})} \end{pmatrix}$$

and that $\vec{\alpha}_s\cdot[f_0,\ldots,f_{s-1}]^T = \alpha^{-1}y + \alpha^{s-1}f_{s-1}$ where $y = \vec{\alpha}_{s-1}L_{(F \bmod X^{s-1})}$. □

One may remark that computing $\vec{\alpha}U_F$ performs exactly the same operations as calculating $f(\alpha)$ using Horner's rule. The same remark applied for $\vec{\alpha}L_F$ but with the evaluation of the polynomial $\alpha^{s-1}F(1/X)X^{s-1}$ in $X = 1/\alpha$.

Corollary 4.2. *The transposed operations $U_F\vec{\alpha}^T$ and $L_F\vec{\alpha}^T$ can also be computed in $O(s)$ operations in $\mathbb{K}$.*

Indeed, by transposed matrix product we have $(U_F\vec{\alpha}^T)^T = \vec{\alpha}L_{rev(F)}$ and $(L_F\vec{\alpha}^T)^T = \vec{\alpha}U_{rev(F)}$ where $rev(F)$ is the polynomial reversal of F i.e. $rev(F) = F(1/X)X^{\deg F}$.

Corollary 4.3. *Let T_F be a full Toeplitz matrix, one can compute $T_F\vec{\alpha}^T$ or $\vec{\alpha}T_F$ in $O(s)$ operations rather than $M(s)$ operations with the classical fast approach [15].*

5. A linear time verification algorithm

Let $F, G, H \in \mathbb{K}[X]$ such that $\deg F = \deg H = s-1$, $\deg G = 2s-2$. The following algorithm provides a probabilistic verification for $H = \mathsf{MP}_s(F,G)$ that requires a linear number of operations.

Algorithm **VerifyMP**(F,G,H) :

1. choose a random α from a finite subset $S \subset \mathbb{K}$ and set $\vec{\alpha}_s \leftarrow [1, \alpha, \ldots, \alpha^{s-1}]$
2. $y_1 \leftarrow (\vec{\alpha}_sU_F)\cdot[g_0,\ldots,g_{s-1}]^T$
3. $y_2 \leftarrow \alpha(\vec{\alpha}_{s-1}L_{(F \bmod X^{s-1})})\cdot[g_s,\ldots,g_{2s-2}]^T$
4. return true if $H(\alpha) = y_1 + y_2$, false otherwise

Lemma 5.1. *Algorithm* **VerifyMP**(F,G,H) *ensures that $H = \mathsf{MP}_s(F,G)$ with a probability greater or equal to $1 - s/|S|$. The algorithms uses $O(s)$ operations in $\mathbb{K}$ and $\lceil\log_2|S|\rceil$ random bits.*

Proof. The correctness of algorithm **VerifyMP** comes from the definition of $\mathsf{MP}_s(F,G)$ as a linear application when F is fixed. Indeed, this corresponds to a linear application from $\mathbb{K}^{2s-1} \longrightarrow \mathbb{K}^s$ where its matrix representation in the canonical basis of $\mathbb{K}[X]$ is:

$$\mathcal{B}_F = \underbrace{\begin{pmatrix} f_{s-1} & f_{s-2} & \cdots & f_0 & \\ & \ddots & \ddots & \vdots & \\ & & \ddots & f_{s-2} & \\ & & & f_{s-1} & \end{pmatrix}}_{U_F} \underbrace{\begin{pmatrix} f_0 & & \\ \vdots & \ddots & \\ f_{s-2} & \cdots & f_0 \end{pmatrix}}_{L_{(F \bmod X^{s-1})}}.$$

Let $v_H = \mathcal{B}_F[g_0, g_1, \ldots, g_{2s-2}]^T$, one can read the coefficients of $H = \mathsf{MP}_s(F,G)$ from v_H. Splitting $\mathcal{B}_F$ and G in two parts, we get

$$v_H = U_F\begin{pmatrix} g_0 \\ \vdots \\ g_{s-1} \end{pmatrix} + \begin{pmatrix} 0 & \cdots & 0 \\ & L_{(f \bmod X^{s-1})} & \end{pmatrix}\begin{pmatrix} g_s \\ \vdots \\ g_{2s-2} \end{pmatrix}$$

Therefore, multiplying this equation on the left by $\vec{\alpha}_s$ gives $H(\alpha) = y_1 + y_2$ and proves the correctness of our algorithm. Using Lemma 3.1, the probability that $H(\alpha) = y_1 + y_2$ when $H \neq \mathsf{MP}_s(F,G)$ is less than $\frac{s}{|S|}$, which then gives a probability of success greater than $1 - \frac{s}{|S|}$ as promised. From Lemma 4.1 and the cost of dot product, one can deduce the complexity of $O(s)$. Since the bitsize of α is less than $\log_2|S|$, this concludes the proof. □

Remark 1. Assuming $|S| > 2s$, one can run k times Algorithm **VerifyMP**(F,G,H) on same inputs to raise the probability to $1 - \frac{1}{2^k}$.

6. A more general result

Following our approach we generalize our algorithm to certify any operations that compute only a certain consecutive chunk of a polynomial product. This is for instance the case for the so-called short product operation [1, 16].

Let $F, G \in \mathbb{K}[X]$ of degree $s-1$, the short product of F and G is denoted by $\mathsf{SP}_s(F,G) = FG \bmod X^s$. Similarly, one can define the high short product of F and G to be $\mathsf{HP}_s(F,G) = FG \operatorname{div} X^{s-1}$, corresponding to the s highest terms of the product FG. Assuming F is fixed, one can define these two operations as linear applications from $\mathbb{K}^s \to \mathbb{K}^s$ with the matrix L_F for $\mathsf{SP}_s(F,G)$ and the matrix U_F for $\mathsf{HP}_s(F,G)$. As before, picking a random element $\alpha \in S \subset \mathbb{K}$, one can check the two short product operations by checking respectively $H(\alpha) = (\vec{\alpha}L_F) \cdot v_G$ or $H(\alpha) = (\vec{\alpha}U_F) \cdot v_G$. Indeed, using Lemma 4.1 one can achieve a complexity of $O(s)$ operations in $\mathbb{K}$ and a probability of success greater than $1 - s/|S|$.

Without loss of generality, assuming that $\deg F = m \geq \deg G = n$ and $s \mid n$. One can define a partial product operation on F and G as $\mathsf{PP}_s(F,G,i) = (FG \operatorname{div} X^i) \bmod X^s$. This operation corresponds to extracting the s consecutive terms of the product FG starting from the monomial X^i. Assuming F is fixed, this operation is a linear application from $\mathbb{K}^n \to \mathbb{K}^s$ where its matrix has the form

$$\mathcal{C}_F = \begin{pmatrix} T_{\bar{F}_0} & T_{\bar{F}_1} & \dots & T_{\bar{F}_{n/s-1}} \end{pmatrix} \in \mathbb{K}^{s \times n}$$

such that each $T_{\bar{F}_k} \in \mathbb{K}^{s \times s}$ for $k \in [0, \dots, n/s-1]$ is a Toeplitz matrix formed from the coefficients of the polynomial F. More precisely, we have

$$T_{\bar{F}_k} = \begin{pmatrix} f_{i-ks} & f_{i-ks-1} & \cdots & f_{i-(k+1)s+1} \\ f_{i-ks+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & f_{i-ks-1} \\ f_{i-(k-1)s-1} & \cdots & f_{i-ks+1} & f_{i-ks} \end{pmatrix}$$

with $f_j = 0$ when $j < 0$ or $j > m$ and f_j is the coefficient of the polynomial F at X^j otherwise. Let $H = \mathsf{PP}_s(F,G,i)$ and v_G, v_H be the vector of the coefficients of the polynomial G and H. By definition of $\mathcal{C}_F$ we have $v_H = \mathcal{C}_F v_G$. Here again, applying a vector $\vec{\alpha}$ to this equality provides us a way to certify the partial product operation. The following algorithm provides a probabilistic verification for $H = \mathsf{PP}_s(F,G,i)$ with a complexity of $O(n)$:

Algorithm **VerifyPP**(F,G,H,s,i) :

1. choose a random α from a finite subset $S \subset \mathbb{K}$ and set $\vec{\alpha}_s \leftarrow [1, \alpha, \dots, \alpha^{s-1}]$
2. for k from 0 to n/s
   $y_k \leftarrow (\vec{\alpha}_s T_{\bar{F}_k}) \cdot [g_{ks}, \dots, g_{(k+1)s-1}]^T$
3. return true if $H(\alpha) = \sum_{k=0}^{n/s-1} y_k$, false otherwise

Lemma 6.1. *Algorithm* **VerifyPP**(F,G,H,s,i) *ensures that* $H = \mathsf{PP}_s(F,G,i)$ *with a probability greater or equal to* $1 - s/|S|$. *The algorithms uses* $O(n)$ *operations in* $\mathbb{K}$ *and* $\lceil \log_2 |S| \rceil$ *random bits.*

Proof. From the definition of $\mathcal{C}_F$ we know that $v_H = \mathcal{C}_F v_G$ corresponds to the partial product operation $\mathsf{PP}_s(F,G,i)$. There, multiplying both side of the equation gives $\vec{\alpha} \cdot v_H = H(\alpha) = (\vec{\alpha}\mathcal{C}_F) \cdot v_G$. Since $\sum_{k=0}^{n/s-1} y_k$ corresponds, in our algorithm, exactly to $(\vec{\alpha}\mathcal{C}_F) \cdot v_G$, this proves the correctness of our algorithm. Assuming $H \neq \mathsf{PP}_s(F,G,i)$, the value $H(\alpha) - (\vec{\alpha}\mathcal{C}_F) \cdot v_G$ is a non zero polynomial of $\mathbb{K}[\alpha]$ of degree less than s. Hence, such polynomial can be zero only for s values of $\alpha \in S \subset \mathbb{K}$ which gives the expected probability. Finally, the complexity of our algorithm is dominated by step 2. Each loop costs exactly $O(s)$ operations in $\mathbb{K}$ by using Corollary 4.3. Since the size of the loop is n/s, the final complexity is $O(n)$ as promised. □

For some specific cases, one is able to reduce the complexity of $\mathsf{PP}_s(F,G,i)$. Indeed, depending on the value of i some Toeplitz matrices $T_{\bar{F}_k}$ will be zero. Using the structure of $\mathcal{C}_F$, one can prove that the number of non zero matrices is given by $\lceil \frac{i}{s} \rceil$ if $i < n$ and $\lceil \frac{m-i}{s} \rceil$ if $i > m$. For such cases, the complexity drops down to $O(s\lceil \frac{i}{s} \rceil)$ and $O(s\lceil \frac{m-i}{s} \rceil)$ which are below $O(n)$.

References

[1] T. Mulders, On short multiplications and divisions, Applicable Algebra in Engineering, Communication and Computing 11 (2000) 69–88.

[2] G. Hanrot, M. Quercia, P. Zimmermann, The middle product algorithm i, Applicable Algebra in Engineering, Communication and Computing 14 (2004) 415–38.

[3] D. Harvey, Faster algorithms for the square root and reciprocal of power series, Mathematics of Computation 80 (2011) 387–94.

[4] J. van der Hoeven, Newton's method and fft trading, Journal of Symbolic Computation 45 (2010) 857 –78.

[5] A. Bostan, G. Lecerf, E. Schost, Tellegen's principle into practice, in: Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation, ACM, 2003, pp. 37–44.

[6] A. Karatsuba, Y. Ofman, Multiplication of Multidigit Numbers on Automata, Soviet Physics-Doklady 7 (1963) 595–6.

[7] J. v. z. Gathen, J. Gerhard, Modern Computer Algebra, 3rd ed., Cambridge University Press, New York, NY, USA, 2013.

[8] D. Harvey, J. V. D. Hoeven, G. Lecerf, Faster polynomial multiplication over finite fields, J. ACM 63 (2017) 52:1–52:23.

[9] R. Zippel, Probabilistic algorithms for sparse polynomials, in: Symbolic and Algebraic Computation. In: Lecture Notes in Comput. Sci., vol. 72, Springer-Verlag, 1979, pp. 216–26.

[10] J. T. Schwartz, Fast probabilistic algorithms for verification of polynomial identities, J. ACM 27 (1980) 701–17.

[11] R. A. Demillo, R. J. Lipton, A probabilistic remark on algebraic program testing, Information Processing Letters 7 (1978) 193 –5.

[12] P. Giorgi, R. Lebreton, Online order basis algorithm and its impact on the block Wiedemann algorithm, in: Proceedings of the 2014 International Symposium on Symbolic and Algebraic Computation, ACM, 2014, pp. 202–9.

[13] M. Roşca, A. Sakzad, D. Stehlé, R. Steinfeld, Middle-product learning with errors, in: Advances in Cryptology – CRYPTO 2017, 2017, pp. 283–97.

[14] T. Kimbrel, R. K. Sinha, A probabilistic algorithm for verifying matrix products using o(n2) time and log2n + o(1) random bits, Information Processing Letters 45 (1993) 107 –10.

[15] G. H. Golub, C. F. Van Loan, Matrix Computations (3rd Ed.), Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[16] G. Hanrot, P. Zimmermann, A long note on mulders short product, Journal of Symbolic Computation 37 (2004) 391 – 401.