INVERSES OF MULTIVARIABLE POLYNOMIAL MATRICES BY DISCRETE FOURIER TRANSFORMS

S. Vologiannidis, N. Karampetakis

email: (svol , karampet)@math.auth.gr
Department of Mathematics,
Aristotle University of Thessaloniki,
Thessaloniki 54006, Greece

Abstract

Two discrete Fourier transform based algorithms are proposed for the computation of the Moore-Penrose and Drazin inverse of a multivariable polynomial matrix.

1 Introduction

Let $\mathbb{R}$ be the set of real numbers, $\mathbb{R}^{p\times m}$ be the set of $p \times m$ real matrices, $\mathbb{R}[z_1, z_2, ..., z_n]$ (resp. $\mathbb{R}(z_1, z_2, ..., z_n)$) denotes the polynomials (resp. rational expressions) with real coefficients in the n indeterminates $z_1, z_2, ..., z_n$. The $p\times m$ matrices with elements in $\mathbb{R}[z_1, z_2, ..., z_n]$ (resp. $\mathbb{R}(z_1, z_2, ..., z_n)$) are denoted by $\mathbb{R}[z_1, z_2, ..., z_n]^{p\times m}$ (resp. $\mathbb{R}(z_1, z_2, ..., z_n)^{p\times m}$). By I_p we denote the identity matrix of order p, and by $0_{p,m}$ the $p \times m$ null matrix. By $A(z_1, z_2, ..., z_n)^D$ (resp. $A(z_1, z_2, ..., z_n)^+$) we denote the Drazin and Moore-Penrose inverse of $A(z_1, z_2, ..., z_n)$.

The Moore-Penrose inverse has originally been defined by Penrose [16], while later Decell [2] proposed a Leverrier-Faddeev algorithm for its computation. An extension of this algorithm to the one and two-variable polynomial matrices has been proposed by [9], [12], [13], [14]. A Leverrier-Faddeev algorithm has also been proposed by Grevile [6] for the computation of the Drazin inverse of square constant matrices with extensions to the one-variable polynomial matrices by [11], [18].

The Leverrier algorithms have the advantage that are easily implemented in symbolic programming languages like Mathematica, Maple etc. However, their main disadvantage, is that are not stable if they are implemented in other high level programming languages such as C++, Fortran etc. In order to overcome these difficulties we may use other techniques such as interpolation methods. Schuster and Hippe [17] for example, use interpolation techniques in order to find the inverse of a polynomial matrix. However, if we need to increase the speed and robustness of our algorithms we may interested in finding algorithms based on Discrete Fourier Transforms (DFT) or better Fast Fourier Transforms (FFT). The main advantages of the DFT based algorithms are:

1. There are very efficient algorithms available both in software and hardware.
2. Parallel environment (through symmetric multiprocessing or other techniques) greatly benefits their speed.

Actually during the past two decades there has been extensive use of DFT - based algorithms, due to their computational speed and accuracy. Some remarkable examples, but not the only ones of the use of DFT in linear algebra problems, are the calculation of the determinantal polynomial by [15], the computation of the transfer function of generalized n-dimensional systems by [10] and the solutions of polynomial matrix Diophantine equations by [8].

The main reason for the interest in these two specific inverses are due to their applications in inverse systems, solution of AutoRegressive Moving Average representations [7], solution of Diophantine equations which gives rise to numerous applications to the field of control system synthesis (see for example [12] and its references) and in the study of multidimensional filters which find numerous applications in image processing, electrical networks with variable elements etc. Note that in case of square and nonsingular matrices, both inverses coincide with the known inverse of the matrix. Therefore the computation of these special inverses gives rise also to applications where the usual inverse of a matrix is required such as the computation of the transfer function of a matrix ([1], [10]).

The main purpose of this work is to present a DFT-algorithm for the evaluation of the generalized inverse and the Drazin inverse of a multivariable polynomial matrix. More specifically in section 2 we introduce the n-dimensional discrete Fourier transform, while later in section 3 and 4 we propose two new DFT algorithms for the evaluation of the generalized and Drazin inverse respectively of a polynomial matrix in n indeterminates. The whole theory is illustrated via an illustrative example coming from the field of control system synthesis.

2 Multidimensional Discrete Fourier Transform

Consider the finite sequence $X(k_1, \ldots, k_n)$ and $\tilde{X}(r_1, \ldots, r_n)$, $k_i, r_i = 0, 1, ..., M_i$. In order for the sequence $X(k_1, \ldots, k_n)$ and $\tilde{X}(r_1, \ldots, r_n)$ to constitute an DFT pair the following relations should hold [4] :

$$\tilde{X}(r_1, .., r_n) = \\ = \sum_{k_1=0}^{M_1} \cdots \sum_{k_n=0}^{M_n} X(k_1, .., k_n) W_1^{-k_1 r_1} \cdots W_1^{-k_n r_n} \quad (1)$$

$$X(k_1, .., k_n) = \\ = \frac{1}{R} \sum_{r_1=0}^{M_1} \cdots \sum_{r_n=0}^{M_n} \tilde{X}(r_1, .., r_n) W_1^{k_1 r_1} \cdots W_1^{k_n r_n} \quad (2)$$

where

$$W_i = e^{\frac{2\pi j}{M_i+1}} \ \forall i = 1, 2, 3, ..., n \tag{3}$$

$$R = \prod_{i=1}^{n} (M_i + 1) \tag{4}$$

and $X, \tilde{X}$ are discrete argument matrix-valued functions, with dimensions $p \times m$. The computation of the multidimensional DFT and its inverse can be also accomplished using fast Fourier transform techniques. A very fast free implementation of FFT can be found in [5].

3 Generalized Inverse of a Multivariable Polynomial Matrix

The generalized inverse of a constant matrix was defined by Penrose in [16].

Definition 1 *[16] For every matrix $A \in \mathbb{R}^{p\times m}$, a unique matrix $A^+ \in \mathbb{R}^{m\times p}$, which is called generalized inverse, exists satisfying*

(i) $AA^+A = A$

(ii) $A^+AA^+ = A^+$

(iii) $(AA^+)^T = AA^+$

(iv) $(A^+A)^T = A^+A$

where A^T denotes the transpose of A. In the special case that the matrix A is square nonsingular matrix, the generalized inverse of A is simply its inverse i.e. $A^+ = A^{-1}$.

Consider the polynomial matrix with real coefficients in the n indeterminates $z_1, z_2, ..., z_n$ (called nD polynomial matrix)

$$A(z_1, \ldots, z_n) = \sum_{k_1=0}^{M_1} \cdots \sum_{k_n=0}^{M_n} A_{k_1 \ldots k_n} z_1^{k_1} \ldots z_n^{k_n} \tag{5}$$

where $A(z_1, \ldots, z_n) \in \mathbb{R}[z_1, \ldots, z_n]^{p\times m}$, with $A_{k_1\ldots k_n} \in \mathbb{R}^{p\times m}$, and p not necessarily equal to m. In an analogous way we define the generalized inverse $A(z_1, \ldots, z_n)^+ \in \mathbb{R}(z_1, \ldots, z_n)^{m\times p}$ of the polynomial matrix $A(z_1, \ldots, z_n) \in \mathbb{R}[z_1, \ldots, z_n]^{p\times m}$ defined in (5) as the matrix which satisfies the properties (i)-(iv) of Definition (1). We will denote

$$\bar{z} = (z_1, \ldots, z_n) \in \mathbb{R}^n$$

and with a slight abuse of notation

$$a(\bar{z}) \equiv a(z_1, \ldots, z_n) \in \mathbb{R}[z_1, \ldots, z_n]$$

will denote a nD polynomial. Following the steps of [13] we have

Theorem 2 *Let $A(\bar{z}) = A(z_1, \ldots, z_n) \in \mathbb{R}[z_1, \ldots, z_n]^{p\times m}$ as in (5) and*

$$\begin{aligned} a(s, z_1, \ldots, z_n) &= \det\left[sI_p - A(\bar{z})A(\bar{z})^T\right] \\ &= (a_0(\bar{z})s^p + \cdots + a_{p-1}(\bar{z})s + a_p(\bar{z})) \end{aligned} \tag{6}$$

$a_0(\bar{z}) = 1$, be the characteristic polynomial of $A(\bar{z}) \times A(\bar{z})^T$. Let k such that $a_p(\bar{z}) \equiv 0, ..., a_{k+1}(\bar{z}) \equiv 0$ while $a_k(\bar{z}) \neq 0$, and define $\Lambda := \{(\bar{z}) \in \mathbb{C}^n : a_k(\bar{z}) = 0\}$. Then the generalized inverse $A(\bar{z})^+$ of $A(\bar{z})$ for $\bar{z} \in \mathbb{C}^n - \Lambda$ is given by

$$A(\bar{z})^+ = -\frac{1}{a_k(\bar{z})} A(\bar{z})^T B_{k-1}(\bar{z}) \tag{7}$$

$$\begin{aligned} &B_{k-1}(\bar{z}) = \\ &= a_0(\bar{z})\left[A(\bar{z})A(\bar{z})^T\right]^{k-1} + \cdots + a_{k-1}(\bar{z})I_p \end{aligned} \tag{8}$$

If $k = 0$ is the largest integer such that $a_k(\bar{z}) \neq 0$, then $A(\bar{z})^+ = 0$. For those $\bar{z} \in \Lambda$ we can use the same algorithm again.

Proof. The theorem is easily proved using the logic in the proof of the corresponding theorem about constant matrices in [2]. ■

Remark 3 *The algorithm described in (2) is efficient using symbolic programming languages. Its main advantages are*
1) It consists of simple recursions.
2) No matrix inversion is required.
3)In the case that $p > m$ we can compute the transpose $A(\bar{z})^T$ and compute $A(\bar{z})^+ = [A(\bar{z})^+]^T$. The algorithm will be completed faster since it will need m rather than p steps.

In the following we will propose a new algorithm for the calculation of the generalized inverse which combines the above advantages with numerical stability and robustness by using interpolation and discrete Fourier transforms.

Evaluation of the generalized inverse of $A(z_1, \ldots, z_n)$
Step 1.
It is easily seen from (6), that the greatest powers of the $(n+1)$ variables in $a(s, z_1, \ldots, z_n)$ are

$$\begin{aligned} &\deg_s(a(s, z_1, \ldots, z_n)) = p := b_0 \\ &\deg_{z_1}(a(s, z_1, \ldots, z_n)) \leq 2pM_1 := b_1 \\ &\vdots \\ &\deg_{z_n}(a(s, z_1, \ldots, z_n)) \leq 2pM_n := b_n \end{aligned}$$

Thus, the polynomial $a(s, z_1, \ldots, z_n)$ can be written as

$$a(s, z_1, \ldots, z_n) = \sum_{k_0=0}^{b_0} \cdots \sum_{k_n=0}^{b_n} a_{k_0\ldots k_n} s^{k_0} z_1^{k_1} \ldots z_n^{k_n} \tag{9}$$

and can be numerically computed via interpolation using the following R points

$$\begin{aligned} u_i(r_j) = W_i^{-r_j}; i = 0, \ldots, n \text{ and } r_j = 0, 1, ..., b_i \\ W_i = e^{\frac{2\pi j}{b_i+1}} \end{aligned} \tag{10}$$

where

$$R = \prod_{i=0}^{n} (b_i + 1)$$

In order, to evaluate the coefficients $a_{k_0k_1\ldots k_n}$ define

$$\begin{aligned} \tilde{a}_{r_0r_1\ldots r_n} = \det[&u_0(r_0)I_p - A(u_1(r_1), \ldots, u_n(r_n)) \\ &\times [A(u_1(r_1), \ldots, u_n(r_n))]^T] \end{aligned} \tag{11}$$

From (9), (10), (11) we get

$$\tilde{a}_{r_0 r_1 \dots r_n} = \sum_{l_0=0}^{b_0} \cdots \sum_{l_n=0}^{b_n} (a_{l_0 \dots l_n}) \left(W_0^{-r_0 l_0} \dots W_n^{-r_n l_n} \right) \tag{12}$$

Notice that $[a_{l_0 l_1 \dots l_n}]$ and $[\tilde{a}_{r_0 r_1 \dots r_n}]$ form a DFT pair and thus using (2) we have

$$a_{l_0 l_1 \dots l_n} = \frac{1}{R} \sum_{r_0=0}^{b_0} \cdots \sum_{r_n=0}^{b_n} \tilde{a}_{r_0 \dots r_n} W_0^{r_0 l_0} \dots W_n^{r_n l_n}$$

where $l_i = 0, \dots, b_i$.

Step 2. (Evaluate $a_k(\bar{z})$)

Find $k : a_{k+1}(\bar{z}) = a_{k+2}(\bar{z}) = \cdots = a_p(\bar{z}) = 0$ and $a_k(\bar{z}) \neq 0$

Step 3. (Evaluate $C(\bar{z}) = A(\bar{z})^T B_{k-1}(\bar{z})$)

The greatest powers of z_i in

$$\begin{gathered} C(\bar{z}) = A(\bar{z})^T B_{k-1}(\bar{z}) = \\ = a_0(\bar{z}) \left[A(\bar{z}) A(\bar{z})^T \right]^{k-1} + \cdots + a_{k-1}(\bar{z}) I_p \end{gathered} \tag{13}$$

is

$$n_i = \max \{2(k-1)M_i + M_i, k = 1, \dots, p\} = (2p-1)M_i$$

Using the previous observation $C(\bar{z})$ can be written as

$$C(\bar{z}) = \sum_{l_1=0}^{n_1} \cdots \sum_{l_n=0}^{n_n} C_{l_0 \cdots l_n} \left(z_1^{l_1} \cdots z_n^{l_n} \right) \tag{14}$$

We compute $C(\bar{z})$ via interpolation using the following R points

$$\begin{gathered} u_i(r_j) = W_i^{-r_j}; i = 1, \dots, n \text{ and } r_j = 0, 1, ..., n_i \\ W_i = e^{\frac{2\pi j}{n_i + 1}} \end{gathered} \tag{15}$$

where

$$R = \prod_{i=1}^{n} \{(2p-1)M_i + 1\} \tag{16}$$

To evaluate the coefficients $C_{l_0 \cdots l_n}$ define

$$\tilde{C}_{r_1 \cdots r_n} = C(u_1(r_1), \dots u_n(r_n)) \tag{17}$$

Using (15), (16), (17) becomes

$$\tilde{C}_{r_1 \cdots r_n} = \sum_{l_1=0}^{n_1} \cdots \sum_{l_n=0}^{n_n} C_{l_0 \cdots l_n} W_1^{-r_1 l_1} \cdots W_n^{-r_n l_n}$$

which through (2)

$$C_{l_0 \cdots l_n} = \frac{1}{R} \sum_{l_1=0}^{n_1} \cdots \sum_{l_n=0}^{n_n} \tilde{C}_{r_1 \cdots r_n} W_1^{r_1 l_1} \cdots W_n^{r_n l_n}$$

where $l_i = 0, \dots, n_i$.

Step 4. (Evaluation of the generalized inverse)

$$A(\bar{z})^+ = -\frac{1}{a_k(\bar{z})} C(\bar{z})$$

The complexity of the Moore-Penrose generalized inverse algorithm is bounded by $\mathcal{O}\left(m^4 R L\right)$ where R is the maximum R occuring in the steps of the algorithm described above and $L = \max \left\{ \sum_{i=0}^{n} \log(b_i + 1), \sum_{i=1}^{n} \log(n_i + 1) \right\}$.

4 Drazin Inverse of a Multivariable Polynomial Matrix

The Drazin inverse of a constant matrix was defined by Drazin in [3].

Definition 4 *For every matrix $A \in \mathbb{R}^{m \times m}$, there exists a unique matrix $A^D \in \mathbb{R}^{m \times m}$, which is called Drazin inverse, satisfying*

(i) $A^D A^{k+1} = A^k$ for $k = ind(A) = \min(k \in N : rank\left(A^k\right) = rank\left(A^{k+1}\right))$

(ii) $A^D A A^D = A^D$

(iii) $A A^D = A^D A$

In the special case that the matrix A is square and nonsingular matrix, the Drazin inverse of A is simply its inverse i.e. $A^D = A^{-1}$.

In an analogous way we define the Drazin inverse of polynomial matrix $A(z_1, \dots, z_n) \in \mathbb{R}[z_1, \dots, z_n]^{m \times m}$ defined in (5) as the matrix which satisfies the properties of Definition (4). The following theorem proposes a new algorithm for the computation of the Drazin inverse of a nD polynomial matrix, which generalizes the results in [18].

Theorem 5 *Consider a nonregular nD polynomial matrix $A(\bar{z})$. Assume that*

$$\begin{gathered} a(s, z_1, \dots, z_n) = \det \left[s I_m - A(\bar{z}) \right] \\ = (a_0(\bar{z}) s^m + \cdots + a_{m-1}(\bar{z}) s + a_m(\bar{z})) \end{gathered} \tag{18}$$

where

$$a_0(\bar{z}) \equiv 1, z \in \mathbb{C}$$

is the characteristic polynomial of $A(\bar{z})$. Also, consider the following sequence of $m \times m$ polynomial matrices

$$\begin{gathered} B_j(\bar{z}) = a_0(\bar{z}) A(\bar{z})^j + \cdots a_{j-1}(\bar{z}) A(\bar{z}) + a_j(\bar{z}) I_m, \\ a_0(\bar{z}) = 1, j = 0, \dots, m \end{gathered} \tag{19}$$

Let

$$a_m(\bar{z}) \equiv 0, \dots, a_{t+1}(\bar{z}) \equiv 0, a_t(\bar{z}) \neq 0. \tag{20}$$

Define the following set:

$$\Lambda = \{\bar{z}_i \in \mathbb{C}^n : a_t(\bar{z}_i) = 0\}$$

Also, assume that

$$B_m(\bar{z}),...,B_r(\bar{z}) = 0, B_{r-1}(\bar{z}) \neq 0$$

and $k = r - t$. In the case $\bar{z} \in \mathbb{C}^n - \Lambda$ and $k > 0$, the Drazin inverse of $A(\bar{z})$ is given by

$$A(\bar{z})^D = \frac{A(\bar{z})^k B_{t-1}(\bar{z})^{k+1}}{a_t(\bar{z})^{k+1}} \tag{21}$$

$$B_{t-1}(\bar{z}) = a_0(\bar{z}) A(\bar{z})^{t-1} + \cdots + a_{t-2}(\bar{z}) A(s) + a_{t-1}(\bar{z}) I_m$$

In the case $\bar{z} \in \mathbb{C}^n - \Lambda$ and $k = 0$, we get $A(\bar{z})^D = O$.
For $\bar{z}_i \in \Lambda$ we can use the same algorithm again.

Proof. The proof uses the same logic as the one in [18]. For the sake of brevity the interested reader is advised to read [18]. ■

In order to show that a multivariable polynomial matrix is zero we need the following lemma.

Lemma 6 *A polynomial matrix* $B(z_1,\ldots,z_n) \in \mathbb{R}[z_1,\ldots,z_n]^{m\times m}$ *of degree* q_i *in respect with variables* z_i *is the zero polynomial matrix iff its value at* R *distinct points is the zero matrix where*

$$R=\prod_{i=0}^{n}(q_i+1)$$

In the following we suggest a computationally attractive algorithm for the calculation of Drazin inverses based on interpolation techniques and DFT.

(Evaluation of the Drazin inverse of a nD polynomial matrix)
Step 1
It is easily seen from (18), that the greatest powers of the $(n+1)$ variables in $a(s,z_1,\ldots,z_n)$ are

$$\begin{aligned}&\deg_s\left(a(s,z_1,\ldots,z_n)\right)=m:=b_0\\&\deg_{z_1}\left(a(s,z_1,\ldots,z_n)\right)\le mM_1:=b_1\\&\vdots\\&\deg_{z_n}\left(a(s,z_1,\ldots,z_n)\right)\le mM_n:=b_n\end{aligned}$$

So the polynomial $a(s,z_1,\ldots,z_n)$ can be written as

$$\begin{gathered}a(s,z_1,\ldots,z_n)=\\=\sum_{k_0=0}^{b_0}\sum_{k_1=0}^{b_1}\cdots\sum_{k_n=0}^{b_n}\left(a_{k_0k_1\ldots k_n}\right)\left(s^{k_0}z_1^{k_1}\ldots z_n^{k_n}\right)\end{gathered}\tag{22}$$

and can be numerically computed via interpolation using the following R points

$$\begin{gathered}u_i(r_j)=W_i^{-r_j};i=0,\ldots,n \text{ and } r_j=0,1,...,b_i\\W_i=e^{\frac{2\pi j}{b_i+1}}\end{gathered}\tag{23}$$

where

$$R=\prod_{i=0}^{n}(b_i+1)$$

To evaluate the coefficients $a_{k_0k_1\ldots k_n}$ define

$$\tilde{a}_{r_0r_1\ldots r_n}=\det\left[u_0(r_0)I_p-A(u_1(r_1),\ldots,u_n(r_n))\right]\tag{24}$$

From (22), (23), (24) we get

$$\begin{gathered}\tilde{a}_{r_0r_1\ldots r_n}=\\=\sum_{l_0=0}^{b_0}\sum_{l_1=0}^{b_1}\cdots\sum_{l_n=0}^{b_n}\left(a_{l_0l_1\ldots l_n}\right)\left(W_0^{-r_0l_0}\ldots W_n^{-r_nl_n}\right)\end{gathered}\tag{25}$$

Notice that $[a_{l_0l_1\ldots l_n}]$ and $[\tilde{a}_{r_0r_1\ldots r_n}]$ form a DFT pair and thus using the above equation and (2) we have

$$a_{l_0l_1\ldots l_n}=\frac{1}{R}\sum_{r_0=0}^{b_0}\sum_{r_1=0}^{b_1}\cdots\sum_{r_n=0}^{b_n}\tilde{a}_{r_0r_1\ldots r_n}W_0^{r_0l_0}\ldots W_n^{r_nl_n}$$

where $l_i=0,\ldots,b_i$.
Step 2. *(Evaluate* $a_t(\bar{z})$ as in (20)*)*
Find $k: a_{t+1}(\bar{z})=a_{t+2}(\bar{z})=\cdots=a_m(\bar{z})=0$ and $a_t(\bar{z})\neq 0$
Step 3.*(Evaluate* $r\ge t: B_m(\bar{z})\equiv 0$,...,$B_r(\bar{z})\equiv 0, B_{r-1}(\bar{z})\neq 0$*)*
Consider the polynomial matrix $B_i(\bar{z})$. To check whether $B_i(\bar{z})$ is the zero matrix using lemma (6),

$$R_i=\prod_{i=0}^{n}(iM_i+1)$$

interpolation points are needed. In order now to determine the value of $r\ge t$ which satisfy the property : $B_m(\bar{z})\equiv 0$,...,$B_r(\bar{z})\equiv 0, B_{r-1}(\bar{z})\neq 0$, R_i we use the following short algorithm
Do WHILE $(B_i(\bar{z})=0\,\forall u(r))$
i=i-1
Check through lemma (6) whether $B_i(\bar{z})=0$
END DO
$r=i$
Step 4. *(Evaluation of* $C(\bar{z})=A(\bar{z})^kB_{t-1}(\bar{z})^{k+1}$*)*
The greatest powers of z_i in

$$C(\bar{z})=A(\bar{z})^kB_{t-1}(\bar{z})^{k+1}$$

where

$$B_{t-1}(\bar{z})=a_0(\bar{z})A(\bar{z})^{t-1}+\cdots+a_{t-2}(\bar{z})A(s)+a_{t-1}(\bar{z})I_m$$

is

$$n_i=(t-1)(k+1)M_i\tag{26}$$

Using (26), $C(\bar{z})$ can be written as

$$C(\bar{z})=\sum_{l_1=0}^{n_1}\cdots\sum_{l_n=0}^{n_n}C_{l_0\cdots l_n}\left(z_1^{l_1}\cdots z_n^{l_n}\right)$$

We compute $C(\bar{z})$ via interpolation using the following R points

$$\begin{gathered}u_i(r_j)=W_i^{-r_j};i=1,\ldots,n \text{ and } r_j=0,1,...,n_i\\W_i=e^{\frac{2\pi j}{n_i+1}}\end{gathered}\tag{27}$$

where

$$R=\prod_{i=1}^{n}\{(n_i+1\}\tag{28}$$

To evaluate the coefficients $C_{l_0\cdots l_n}$ define

$$\tilde{C}_{r_1\cdots r_n}=C(u_1(r_1),\ldots u_n(r_n))\tag{29}$$

Using (26), (27), (29) becomes

$$\tilde{C}_{r_1\cdots r_n}=\sum_{l_1=0}^{n_1}\cdots\sum_{l_n=0}^{n_n}C_{l_0\cdots l_n}W_1^{-r_1l_1}\cdots W_n^{-r_nl_n}$$

which through (2)

$$C_{l_0\cdots l_n}=\frac{1}{R}\sum_{l_1=0}^{n_1}\cdots\sum_{l_n=0}^{n_n}\tilde{C}_{r_1\cdots r_n}W_1^{r_1l_1}\cdots W_n^{r_nl_n}$$

where $l_i = 0, \ldots, n_i$.
Step 5.*(Evaluation of $c(\bar{z}) = a_t(\bar{z})^{k+1}$)*
The greatest power of z_i appearing in $a_t(\bar{z})^{k+1}$ are

$$b_i = tM_i(k+1)$$

so $c(\bar{z})$ can be written as

$$c(\bar{z}) = \sum_{k_1=0}^{b_1} \cdots \sum_{k_n=0}^{b_n} (c_{k_1 \ldots k_n}) \left(z_1^{k_1} \ldots z_n^{k_n} \right)$$

and can be numerically computed via interpolation using the following R points

$$u_i(r_j) = W_i^{-r_j}; i = 0, \ldots, n \text{ and } r_j = 0, 1, ..., b_i \qquad W_i = e^{\frac{2\pi j}{b_i+1}} \tag{30}$$

where

$$R = \prod_{i=0}^{n} (b_i + 1)$$

Define

$$\tilde{c}_{r_1 \ldots r_n} = \sum_{l_1=0}^{b_1} \cdots \sum_{l_n=0}^{b_n} (c_{k_1 \ldots k_n}) \left(W_1^{-r_1 l_1} \ldots W_n^{-r_n l_n} \right) \tag{31}$$

Using the above equation and (2) we have

$$c_{l_1 \ldots l_n} = \frac{1}{R} \sum_{r_1=0}^{b_1} \cdots \sum_{r_n=0}^{b_n} \tilde{c}_{r_1 \ldots r_n} W_1^{r_1 l_1} \ldots W_n^{r_n l_n}$$

where $l_i = 0, \ldots, b_i$.
Step 6. *(Evaluation of the Drazin inverse)*

$$A(\bar{z})^D = \frac{A(\bar{z})^k B_{t-1}(\bar{z})^{k+1}}{a_t(\bar{z})^{k+1}} = \frac{C(\bar{z})}{c(\bar{z})}$$

The complexity of the Drazin inverse algorithm is bounded by $\mathcal{O}\left(m^4 RL\right)$ where R is the maximum of the R and R_i arising in the steps of the algorithm as described above and $L = \max\left\{L_1, L_2, L_3, \hat{L}_{r-1}, ..., \hat{L}_m\right\}$ where

$$L_1 = \sum_{i=0}^{n} \log(b_i + 1)$$

$$L_2 = \sum_{i=1}^{n} \log(n_i + 1)$$

$$L_3 = \sum_{i=1}^{n} \log(d_i + 1)$$

$$\hat{L}_i = \sum_{i=0}^{n} \log(m_{ij} + 1), i = r-1, ..., m$$

5 Implementation

In this section some experimental results about the efficiency of the algorithm on the computation of the Moore-Penrose generalized inverse are presented. Similar results hold for the computation of the Drazin inverse. We used random $2D$ polynomial matrices up to dimensions 5×5 and of polynomial degrees up to 7. Note that the polynomials that appeared as elements in the matrix had almost all their coefficients nonzero. The algorithms were implemented using Mathematica 4.1 on a Pentium III 700Mhz with 128Mb of RAM. The default floating point accuracy of Mathematica was used. The following figure presents a comparison of the FFT based method (blue line) and the direct computation of the Moore-Penrose generalized inverse using theorem (2) (red line). Note that the horizontal axis represents different random polynomial matrices as their degrees and dimensions increase, while the vertical axis the CPU time in seconds returned by Mathematica's function *Timing[]*.

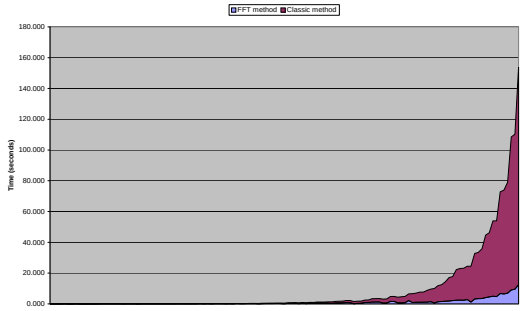

Figure 1. Timing results.

The benefits of the FFT based algorithm are obvious as the degrees and the dimension of the matrices increase. In order to highlight the complicacy of the Moore-Penrose inverse, note the simple fact that the denominator polynomial of the Moore-Penrose inverse of a general 3×4 $2D$ polynomial matrix with polynomial degrees 3 is of degree 18 having 361 terms, while each of the elements of the numerator is of degree 15 having 256 terms. The above procedure was applied for the algorithm concerning the evaluation of the Drazin inverse with similar results.

6 Conclusions

In this paper two algorithms have been presented for determining the generalized and Drazin inverse of nD polynomial matrices. The algorithms are based on the discrete Fourier transform and therefore have the main advantages of speed and robustness in contrast to other known algorithms. The theoretical work is accompanied by an example that tackles the problem of model matching. Other applications of the theory introduced, include the solution of multivariable Diophantine equations and its application to control system synthesis problems, the computation of the transfer function matrix of multidimensional systems, the solution of multidimensional AutoRegressive representations etc. The above mentioned algorithms may be easily extended in order to determine other kind of inverses

such as $\{2\}$, $\{1,2\}$, $\{1,2,3\}$ and $\{1,2,4\}$ inverses of multivariable polynomial matrices by using the Leverrier-Faddeev algorithms presented in [19].

References

[1] Antoniou, G. E., G. O. A. Glentis, S. J. Varoufakis, and D. A. Karras: 1989, 'Transfer function determination of singular systems using the DFT'. *IEEE Trans. Circuits and Systems* **36**(8), 1140–1142.

[2] Decell, Jr., H. P.: 1965, 'An application of the Cayley-Hamilton theorem to generalized matrix inversion'. *SIAM Rev.* **7**, 526–528. (extended in [20]).

[3] Drazin, M. P.: 1958, 'Pseudo inverses in associative rings and semigroups'. *Amer. Math. Monthly* **65**, 506–514.

[4] Dudgeon, D. and R. Mersereau: 1984, *Multidimensional Digital Signal Processing*. Prentice Hall.

[5] Frigo, M. and S. Johnson: 1998, 'FFTW: An Adaptive Software Architecture for the FFT'. In: *ICASSP conference proceedings (vol. 3, pp. 1381-1384)*.

[6] Greville, T. N. E.: 1973, 'The Souriau-Frame algorithm and the Drazin pseudoinverse'. *Linear Algebra and Appl.* **6**, 205–208.

[7] Güyer, T., O. Kı ymaz, G. Bilgici, and Ş. Mirasyedioğlu: 2001, 'A new method for computing the solutions of differential equation systems using generalized inverse via Maple'. *Appl. Math. Comput.* **121**(2-3), 291–299.

[8] Hromcik, M. and M. Sebek: 2001, 'Fast Fourier transform and linear polynomial matrix equations'. In: *Proceedings of the 1rst IFAC Workshop on Systems Structure and Control, Prague, Czech Republic*.

[9] Jones, J., N. P. Karampetakis, and A. C. Pugh: 1998, 'The computation and application of the generalized inverse via Maple'. *J. Symbolic Comput.* **25**(1), 99–124.

[10] Kaczorek, T.: 2001, 'Transfer function computation for generalized n-dimensional systems'. *Journal of the Franklin Institute* **338**, 83–90.

[11] Karampetakis, N. and P. Stanimirovic: 2001, 'On the computation of the Drazin inverse of a polynomial matrix'. In: *1rst IFAC Symposium on Systems Structure and Control, Prague, Czech Republic*.

[12] Karampetakis, N. P.: 1997a, 'Computation of the generalized inverse of a polynomial matrix and applications'. *Linear Algebra Appl.* **252**, 35–60.

[13] Karampetakis, N. P.: 1997b, 'Generalized inverses of two-variable polynomial matrices and applications'. *Circuits Systems Signal Process.* **16**(4), 439–453.

[14] Karampetakis, N. P. and P. Tzekis: 2001, 'On the computation of the generalized inverse of a polynomial matrix'. *IMA J. Math. Control Inform.* **18**(1), 83–97.

[15] Paccagnella, L. E. and G. L. Pierobon: 1976, 'FFT calculation of a determinantal polynomial'. *IEEE Trans. on Automatic Control* pp. 401–402.

[16] Penrose, R.: 1955, 'A Generalized Inverse for Matrices'. *Proceedings of the Cambridge Philosophical Society* **51**, 406–413.

[17] Schuster, A. and P. Hippe: 1992, 'Inversion of polynomial matrices by interpolation'. *IEEE Trans. Automat. Control* **37**(3), 363–365.

[18] Stanimirovic, P. and N. Karampetakis: 2000, 'Symbolic implementation of Leverrier-Faddeev algorithm and applications.'. In: *8th IEEE Medit. Conference on Control and Automation, Patras, Greece.*

[19] Stanimirovic, P. S.: 2002, 'A finite algorithm for generalized inverses of polynomial and rational matrices'. *Applied Mathematics and Computation*.

[20] Wang, G.: 1987, 'A finite algorithm for computing the weighted Moore-Penrose inverse A^+_{MN}'. *Appl. Math. Comput.* **23**(4), 277–289.