ON COMPUTING DETERMINANTS OF LARGE SYLVESTER TYPE MATRICES

Petr Kujan ***,*,1 **Martin Hromčík** **,2
Michael Šebek ****,*,3

* *Department of Control Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague*
** *Centre for Applied Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Karlovo nam. 13, 12135 Prague 2, Czech Republic*

*** *Centre for Applied Cybernetics, Institute of Information Theory and Automation*
**** *Institute of Information Theory and Automation, Pod vodarenskou vezi 4, 18208 Praha 8, Czech Republic*

Abstract: This work is devoted to computation of large n-D polynomial determinants with a special structure. Applications involve n-D systems theory (e.g. coprimeness test for two n-D polynomials or the theory of algebraic equations. More specifically, these determinants were exploited in (Chiasson *et al.*, 2003*b*; Chiasson *et al.*, 2003*a*) to solve the practical problem of multilevel converter by a special computational procedure. To tackle the concerned problem it is essential to solve a system of polynomial equations with many unknowns. An algorithm was chosen based on elimination theory using resultants leading to the fundamental problem of computing determinants of large Sylvester type matrices with n-D polynomial entries. The aim of this work is to propose and test new numerical algorithms that would make it possible to solve the concerned problems more effectively. *Copyright ©2005 IFAC*

Keywords: Polynomial methods, Numerical methods, Fast Fourier Transform

1. INTRODUCTION

The problem considered in this paper is as follows:
Input: square (large) n-D polynomial matrix with special structure (the matrix of Sylvester type) (1) where a_i and b_i are multivariable scalar polynomials in variables $x = x_1, x_2, \dots, x_n$.

Output: n-D scalar polynomial $s(x) = \det S(x)$.

Possible applications are briefly introduced below.

- Common factor test (Barnett, 1983). In the n-D systems theory problems it is often essential to decide whether two n-D polynomials are coprime, or whether they have a common factor. An efficient test can be based on the observation that two polynomials

$$a(x,y) = a_0(x) + a_1(x)\,y + \cdots + a_{d_a}(x)\,y^{d_a}$$
$$b(x,y) = b_0(x) + b_1(x)\,y + \cdots + b_{d_b}(x)\,y^{d_b}$$

  have a common factor in the variable y if and only if $\det S(x) = 0$. Such a way, the original problem is reduced to the computation of determinant with the concerned structure.
- Solving systems of polynomial equations with more unknowns. To solve the set of n alge-

[1] The work of Petr Kujan was supported by the grant 102/03/H116 of the Grant Agency of the Czech Republic and by the Ministry of Education of the Czech Republic under the contract No. 1M6840770004.
[2] The work of Martin Hromčík was supported by the Ministry of Education of the Czech Republic under the contract No. 1M6840770004.
[3] The work of Michael Šebek was supported by the Ministry of Education of the Czech Republic under contract MSMT LA 065.

braic equations in n unknowns, $p_i(x_1, \ldots, x_n) = 0,\ i = 1, 2, \ldots, n$, where p_i are polynomials in n variables, the elimination approach can be called leading to successive evaluation of resultants - determinants of related Sylvester-type polynomial matrices, see (Cox *et al.*, 1996).

The latter case is closely related to the papers (Chiasson *et al.*, 2003*b*; Chiasson *et al.*, 2003*a*) where the problem of a multilever DC/AC converter design is studied. This problem is briefly introduced in the next section and serves as a motivation for computational procedures suggested in the subsequent parts of this report.

$$S(x) = \left[\begin{array}{c|c} A_{11} & B_{12} \\ \hline A_{21} & B_{22} \end{array}\right] = \tag{1}$$

$$\left[\begin{array}{cccc|ccc}
a_0 & & & & b_0 & & 0\\
a_1 & a_0 & 0 & & b_1 & \ddots & \\
\vdots & \vdots & \ddots & & \vdots & \ddots & b_0\\
a_{d_a} & a_{d_a-1} & a_0 & & & & b_1\\
 & a_{d_a} & a_1 & a_0 & b_{d_b-1} & & \vdots\\ \hline
 & \ddots & \vdots & \vdots & b_{d_b} & \ddots & \\
 & 0 & a_{d_a} & a_{d_a-1} & & \ddots & b_{d_b-1}\\
 & & & a_{d_a} & 0 & & b_{d_b}
\end{array}\right]$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{\deg b = d_b}\quad\underbrace{\qquad\qquad\qquad\qquad}_{\deg a = d_a}$$

2. MULTILEVEL CONVERTER

The multilevel converter is a power electronics device built to synthesize a desired AC voltage from several levels of DC voltage, see (Chiasson *et al.*, 2003*b*). Most of the material presented in this section is adopted from (Chiasson *et al.*, 2003*a*), including the symbolic elimination procedure for the three- and five-sources cases. We basically just confront the symbolic implementation of the elimination procedure as presented in (Chiasson *et al.*, 2003*a*) with our numerical implementation.

A key issue in the fundamental switching scheme is to determine the switching angles such that just the fundamental voltage is generated and undesirable higher order harmonics are suppressed. The Fourier series expansion of the (staircase) output voltage waveform of the multilevel converter is

$$V(\omega t) = \sum_{n=1,3,5,\ldots}^{\infty} \frac{4V_{dc}}{n\pi} \cdot \Big(\cos(n\theta_1\omega) + \cdots + \\ + \cos(n\theta_s\omega)\Big) \cdot \sin(n\omega t) \tag{2}$$

where s is the number of DC sources. Ideally, given a desired fundamental voltage V_1, one wants to determine the switching angles $\theta_1, \theta_2, \ldots, \theta_s$ so that (2) becomes $V(\omega t) = V_1 \sin(\omega t)$. The goal here is to choose the switching angles $0 \leq \theta_1 \leq \theta_2 \leq \cdots \leq \theta_s \leq \pi/2$ so as to make the first harmonic equal to the desired fundamental voltage V_1 and specific lower dominant harmonics of $V(\omega t)$ equal to zero. As the application of interest here is a three-phase system, the triplen harmonics in each phase need not be cancelled as they automatically cancel in the line-to-line voltages. Specifically, in case of $s = 5$ DC sources, the desire is to cancel the 5th, 7th, 11th, 13th order harmonics as they dominate the total harmonic distortion. The higher residual frequencies remove the output filter. The mathematical statement of these conditions leads to a set of transcendental equations and, after some trigonometric substitutions $\cos(n\theta\omega) = T_n(\cos\theta\omega)$ [4], to a set of algebraic equations:

$$p_1(x) \triangleq x_1 + x_2 + x_3 + x_4 + x_5 - m = 0 \tag{3}$$
$$p_5(x) \triangleq \sum_{i=1}^{5}(5\,x_i - 20\,x_i^3 + 16\,x_i^5) = 0$$
$$p_7(x) \triangleq \sum_{i=1}^{5}(-7\,x_i + 56\,x_i^3 - 112\,x_i^5 + 64\,x_i^7) = 0$$
$$p_{11}(x) \triangleq \sum_{i=1}^{5}(-11\,x_i + 220\,x_i^3 + \cdots + 1024\,x_i^{11}) = 0$$
$$p_{13}(x) \triangleq \sum_{i=1}^{5}(13\,x_i - 364\,x_i^3 + \cdots + 4096\,x_i^{13}) = 0$$

where $p_i(x)$ is polynomials in variables $x = (x_1, x_2, x_3, x_4, x_5)$ and $m \triangleq V_1/(4V_{dc}/\pi)$. The modulation index is $m_a = m/s = V_1/(4sV_{dc}/\pi)$. This is a set of five equations in the five unknowns x_1, x_2, x_3, x_4, x_5. Further, the solution must satisfy $0 \leq x_1 < x_2 < x_3 < x_4 < x_5 \leq 1$.

This set of equations is possible to solve via numerical Newton iterative procedure, or symbolic Gröbner basis, or elimination method using resultants (Cox *et al.*, 1996). We concentrate on the elimination theory in the following.

2.1 Elimination method using resultants

The elimination approach leads to successive elimination of variables (Cox *et al.*, 1996). In each step the number of equations and variables is decreased by one. From the computational point of view, the k-th step consists of the evaluation a polynomial determinant in $n-k$ variables where n is the number of equations (and, at the same time, of variables). After n steps we arrive at a single variable polynomial the roots of which can be evaluated

[4] $T_n(\cos\theta)$ is Chebyshev polynomial of the first kind.

using any standard procedure (Higham, 1996). A backward substitution procedure than follows and the n-tuples solving the initial set are separated.

To illustrate the problem, the three source multilevel inverter will bee used to illustrate the approach. The conditions are only the first four equations $p_1(x), p_5(x), p_7(x)$ from (3). Eliminating x_1 by substituting $x_1 = m - (x_2 + x_3)$ into $p_5(x), p_7(x)$ gives

$$\begin{aligned} p_5(x_2, x_3) &= 5(m - x_2 - x_3) - \dots \\ &\quad \dots +16x_2^5 + 5x_3 - 20x_3^3 + 16x_3^5 \quad (4) \\ p_7(x_2, x_3) &= -7(m - x_2 - x_3) + \dots \\ &\quad \dots +56x_3^3 - 112x_3^5 + 64x_3^7 \end{aligned}$$

where

$$\begin{aligned} \deg_{x_2}\{p_5(x_2,x_3)\} = 5\,, \deg_{x_3}\{p_5(x_2,x_3)\} = 5 \\ \deg_{x_2}\{p_7(x_2,x_3)\} = 7\,, \deg_{x_3}\{p_7(x_2,x_3)\} = 7 \end{aligned}$$

Now it is our goal to find zeros of polynomial system (4). The first step is elimination of unknown x_2 by constructing appropriate Sylvester matrix. Let us consider $p_5(x_2, x_3)$ and $p_7(x_2, x_3)$ as polynomials
$p_5(x_2, x_3) = p_{5_0}(x_3) + p_{5_1}(x_3)\cdot x_2 + \dots + p_{5_5}(x_3)\cdot x_2^5$,
$p_7(x_2, x_3) = p_{7_0}(x_3) + p_{7_1}(x_3)\cdot x_2 + \dots + p_{7_7}(x_3)\cdot x_2^7$
in x_2 whose coefficients are polynomials in x_3. Then the $n \times n$ Sylvester matrix, where $n = deg_{x_2}\{p_5\} + \deg_{x_2}\{p_7\} = 5 + 7 = 12$, is defined by

$$S_{p_5,p_7}(x_3) = \left[\begin{array}{cccc} p_{5_0} & 0 & p_{7_0} & 0 \\ \vdots & \ddots & p_{7_1} & \ddots \\ p_{5_5} & & \vdots & \ddots \\ 0 & \ddots & p_{7_7} & \\ \vdots & & 0 & \ddots \end{array}\right] . \qquad (5)$$
$$\underbrace{\qquad}_{7-times} \quad \underbrace{\qquad}_{5-times}$$

In the second step the determinat of the Sylvestr matrix (5) is evaluated. Then the resultant polynomial is defined by

$$\begin{aligned} r_{11}(x_3) &= \text{Res}\,(p_5(x_2,x_3), p_7(x_2,x_3), x_2) \\ &\triangleq \det(S_{p_5,p_7}(x_3)), \end{aligned} \qquad (6)$$

Further zeros $\boldsymbol{r_{11}} = \text{Roots}\,(r_{11}(x_3))$ of the scalar polynomial $r_{11}(x_3)$ are computed. The set of zeros $\boldsymbol{r_{11}}$ is substituted to previous polynomials $p_5(x_2, x_3)$ and $p_7(x_2, x_3)$ in the variable x_3. Thus we get two sets of polynomials that are only in one variable x_2. Once again the zeros in both sets are computed. Finally identical zeros for both sets are selected.

2.2 Symmetric polynomials

In the last example (3 DC sources), computation of the resultant polynomial (6) was required. This involved setting up a 12×12 Sylvester matrix (5) and then computing its determinant to obtain the resultant polynomial (6) whose degree was 30. However, as one adds more DC sources, the size and degrees of Sylvester matrices grow rapidly. This is very complicated because time-consumption and memory usage get very high. To get around this computational difficulty to some extent, the symmetry of the involved polynomials can be exploited.

The polynomials (3) are symmetric, that is,

$$p_i(x_1, x_2, \dots, x_n) = p_i(x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)}),$$
$$\text{for all } i = 1, 2, \dots, n \text{ and any permutation } \pi(\cdot).$$

Take only first three equation (3) for 3 DC sources (for more DC sources its similar) and define the elementary symmetric functions (polynomials)

$$\begin{aligned} &s_1 \triangleq x_1 + x_2 + x_3, \quad s_2 \triangleq x_1x_2 + x_1x_3 + x_2x_3 \\ &s_3 \triangleq x_1x_2x_3 \end{aligned} \qquad (7)$$

The key point here is that the degrees of polynomials after substituting s_1, s_2, s_3 are lower than the degrees of $p_5(x_2, x_3), p_7(x_2, x_3)$ in x_2, x_3 (see (4)). Resulting Sylvester matrices have thus lower size and degrees and computations become faster. In the case of three DC sources it is required to compute the determinants of 3×3 Sylvester matrix instead 12×12.

2.3 Determinant evaluation

Solving the scheduled task by elimination of variables using resultants, the biggest problem lies in the time consumption and memory demands of the calculation of the Sylvester matrix determinant. The matrix is of high size and degree when a great number of sources are considered.

The symbolic way was has been chosen in the mentioned papers (Chiasson *et al.*, 2003*b*; Chiasson *et al.*, 2003*a*). However, results have been received for small numbers of DC sources only, namely, for three in the basic formulations of the computational procedure (Chiasson *et al.*, 2003*b*), or for five if the symmetry of the equations is properly employed (Chiasson *et al.*, 2003*a*).

A symbolic method of the calculation results from standard symbolic functions of the system MATHEMATICA (Wolfram Research, 2004 [online]). The resultants are computed using a standard command `Resultant`. The symbolic method of the calculation, in comparison with FFT based numeric calculations (described below), has an advantage in the ability of computation with (exact) integers. Then during the finding of the roots of a polynomial equation there are no problems with roundoff errors. On the other hand, the computational complexity of symbolic procedures is very high in general and in this case prevents the problem with more than five DC sources to be processed.

3. ALTERNATIVE APPROACHES TO THE COMPUTATION OF RESULTANTS

As explained in the preceding sections, there is a demand for designing dedicated and efficient numerical solvers for the polynomial Sylvester-type n-D determinant problem. We now propose our contribution to this objective.

3.1 FFT based numerical method

An efficient numerical algorithm for the polynomial determinant was proposed in algorithm 1. The method is based on FFT and is the most efficient computational method for this task up to the authors' knowledge, see (Hromcik and Sebek, 2000). We present a modification of this routine for the multivariate case in the sequel.

Algorithm 1. Determinant of Polynomial Matrix with n -variables (FFT method)

Input: Square polynomial matrix with n variables $P(s_1, s_2, \ldots, s_n)$ of size m and degrees $\{d_1, d_2, \ldots, d_n\}$.

Output: Scalar polynomial $p(s_1, s_2, \ldots, s_n)$ with n variables - the determinant of $P(s_1, s_2, \ldots, s_n)$.

Step 1: Compute the upper bound $\mathcal{D} = \{D_j | j = 1, 2, \ldots, n\}$ for the degrees of the determinant:

$$D_j = \min\{\sum_{i=1}^{m} \deg_{c_i}(P(s_j)), \sum_{i=1}^{m} \deg_{r_i}(P(s_j))\}$$

where $\deg_{c_i}(P(s_j))$ and $\deg_{r_i}(P(s_j))$ are the i-th column and row degrees of $P(s_j)$ respectively.

Step 2: Using *FFT* algorithm, perform direct n-D DFT at points $(D_1 + 1) \times (D_2 + 1) \times \cdots \times (D_n + 1)$ on the set $\mathcal{P} = \{P_{i_1,\ldots,i_n}\}$ of coefficient matrices of $P(s_1, \ldots, s_n)$ and obtain the set $\mathcal{Y} = \{Y_{j_1,\ldots,j_n} | j_1 = 0, \ldots, D_1; j_2 = 0, \ldots, D_2; \ldots; j_n = 0, \ldots, D_n\}$.

Step 3: Compute the set $\boldsymbol{z} = \{z_{j_1,\ldots,j_n} | j_1 = 1, \ldots, D_1; j_2 = 1, \ldots, D_2; \ldots; j_n = 1, \ldots, D_n\}$, where $z_{j_1,\ldots,j_n} = \det(Y_{j_1,\ldots,j_n})$

Step 4: Perform inverse n-D DFT on $\boldsymbol{z}$ using the *FFT* algorithm to obtain the coefficient set $\boldsymbol{p} = \{p_{i_1,\ldots,i_n} | i_1 = 0, \ldots, D_1; i_2 = 0, \ldots, D_2; \ldots; i_n = 0, \ldots, D_n\}$ of the determinant

$$p(s_1, \ldots, s_n) = \sum_{i_1=0}^{D_1} \cdots \sum_{i_n=0}^{D_n} p_{i_1,\ldots,i_n} \cdot s_1^{i_1} \ldots s_n^{i_n} \quad \square$$

One would expect that substituting the generic symbolic determinant routines with dedicated numerical polynomial determinant solvers should increase the performance of the elimination procedure considerably and that the problem could be resolved for a much larger number of DC sources. Surprisingly, the situation gets rather complicated. The problem is that the successive evaluation of polynomial determinants leads to accumulation of rounding errors. The more equations are to be resolved, the more severe these problems get. In addition, as the algorithm is quite general, it does not exploit the high structurality of the matrix involved to decrease computational time and improve accuracy.

3.2 Numerical method based on FFT and Schur's complement

Performance of the efficient FFT based routine described above (algorithm 1) can be further increased for the Sylvester-type matrices considered in this report by exploiting the structure of the resultant matrices.

Theorem 1. (Schur's complement). If A_{11} and B_{22} are square matrices, then

$$\left| \left[\begin{array}{c|c} A_{11} & B_{12} \\ \hline A_{21} & B_{22} \end{array} \right] \right| = \begin{cases} |B_{22}| \cdot |A_{11} - B_{12} B_{22}^{-1} A_{21}| \\ |A_{11}| \cdot |B_{22} - A_{21} A_{11}^{-1} B_{12}| \end{cases} \tag{8}$$

When B_{22}^{-1} or A_{11}^{-1} exist.
(Brackets $|\cdot|$ denote function $\det(\cdot)$). $\square$

The Schur's complement shall be applied now to the constant matrices - DFT samples of the initial polynomial matrix - in the step 3 of the algorithm 1. Considering the structure (1) of the considered matrices, the matrices $A_{11}, A_{21}, B_{12}, B_{22}$ in (8) are triangular.

This modification of the Step 3 consists of the following steps:

- The expression $X = B_{22}^{-1} A_{21}$ is calculated easily solving linear system $A_{21} = B_{22} X$ which is in a special upper-triangular form.
- Consequently, numerical matrix multiplication $Y = B_{12} X$ and subtraction $Z = A_{11} - Y$ is performed.
- Computation of the determinant $\det Z$ is then performed with an approximately half-scale constant matrix. Matrix B_{22} is upper-triangular with equal elements in diagonal

and the determinant $\det B_{22}$ is therefore only a power the diagonal element.

This way seems to be considerably faster because determinants of scaled-down matrices are computed. Unfortunately, large numerical errors arise typically from the calculation of the difference of two matrices $A_{11} - B_{12} \cdot X$ since the coefficients of the matrix A_{11} are far lower compared to matrix $B_{12} \cdot X$ and they are not handled properly when standard IEEE double precision floating-point numbers are used. Therefore it is necessary to start to work with real numbers of an extended precision and to keep them during the whole computation which is possible and straightforward in the used *MATHEMATICA* software environment. The determination of a sufficient precision leading to correct results is then an issue. During experimental calculations, a higher precision of the floats was set empirically, leading to correct results (see the section Experimental testing).

Another disadvantage of this method consists in a large memory demands when an input polynomial matrix reaches considerable size and degrees, see Step 2 in algorithm 1.

3.3 Schur's complement applied to polynomial entries

In the previous paragraphs, the Schur's complement was applied to constant matrices received as a result of DFT applied on the polynomial resultant matrix. However, the theorem 1 remains valid for general matrices, including polynomial, and can be addressed directly. Effective functions for n-D polynomial multiplication and addition, as applied for instance in (PolynomialPackage, 2004 [online]), can be employed along with a selected polynomial determinant routine. In general case, the Sylvester structure causes problems since the term $B_{22}^{-1}A_{21}$ in Theorem 1 is rational and, in addition to polynomial calculations, a rational matrix determinant solver is required. Nevertheless, for the special case of multilevel converter, this approach is applicable as the degree of scalar polynomial b_{d_b} is always equal to 0 (it is a constant nonzero number).

3.4 Permutation method

This approach is based directly on the definition of determinant.

Definition 1. (Determinant). For an $m \times m$ matrix $A = [a_{ij}]$, the *determinant* of A is defined to be the scalar

$$\det(A) = \sum_p \sigma(p)\, a_{1p_1} a_{2p_2} \dots a_{mp_m} \ ,$$

where the sum is taken over the $m!$ permutations $p = (p_1, p_2, \dots, p_m)$ of $(1, 2, \dots, m)$.

Here the sign $\sigma(p)$ of a permutation p is defined as

$$\sigma(p) = \begin{cases} +1 \text{ if } p \text{ can be restored to natural order} \\ \quad \text{by an } \textit{even} \text{ number of interchanges,} \\ -1 \text{ if } p \text{ can be restored to natural order} \\ \quad \text{by an } \textit{odd} \text{ number of interchanges } \square \end{cases}$$

The definition is known to be usable only for small-to-medium size problems in general since the number of terms "explodes" exponentially with the size of input matrix. However, considering the special Sylvester-type structure of resultant matrices, the situation does not become so dramatic and the number of nonzero terms is typically by several orders lower for large scale matrices than the general matrix estimate would suggest. Some figures are in the table 1.

Table 1. Comparison numbers of nonzero terms.

Input matrix			**Number of nonzero terms**	
m	deg a	deg b	general matrix	resultant matrix
6	4	2	6! = 720	33
6	3	3	6! = 720	56
7	4	3	7! = 5040	155
10	5	5	10! = 3628800	9440

The problem now lies in determining the set of valid permutations yielding nonzero terms. A solution is suggested in the iterative algorithm 2 to follow.

Algorithm 2. Determinant of Polynomial Matrix with n -variables (Permutations method)

Input and output: Same as in algorithm 1.

```
Step 1: Make the set $\mathcal{D} = \{d_i \,|\, i = 1, 2, \dots m\}$
  of indexes, where $d_i = \{j \,|\, j = 1, 2, \dots, m \wedge P_{[i,j]} \neq 0\}$
Step 2: Initialization list of possible nonzero permutations: $\boldsymbol{p} = \{\{d_{1,1}\}, \{d_{1,2}\}\}$
Step 3: Compute in iterative loop all possible nonzero permutations:
  FOR $k := 2$ TO $m$ DO
      $\boldsymbol{p} = \{p_l \,|\, l = 1, 2, \dots, \text{Length}[\boldsymbol{p}] * \text{Length}[d_k]\}$,
  where
  $p_l = \{\text{Append}[p_i, d_{k,j}] \,|\, (i = 1, 2, \dots, \text{Length}[\boldsymbol{p}]; j = 1, 2, \dots, \text{Length}[d_k]) \wedge d_{k,j} \notin p_i\}$
Step 4: Computation of determinant
```

$$p(s_1, s_2, \dots, s_n) = \sum_{i=1}^{\text{Length}[\boldsymbol{p}]} \prod_{j=1}^{m} P_{[j, \boldsymbol{p}_{i,j}]} \qquad \square$$

Step 3 in the algorithm 2 is unfortunately rather time-consuming and unfortunately prevents this method to be applied for large problems. The routine however remains useful for medium-size Sylvester type matrices.

4. EXPERIMENTAL TESTING

Numerical test have been carried out to prove performance of the new algorithms. Our algorithms have been implemented as *Mathematica* packages and connected to the already existing program Polynomial package (PolynomialPackage, 2004 [online]). All the computations were performed in *Mathematica* 5.0 for Linux A-64 (Wolfram Research, 2004 [online]) on SGI ALTIX 3700 with 16 CPU Itanium 2 and 20 GB RAM (Computing and information centre of CTU, 2004 [online]).

Table 2 contains execution times in seconds for the following algorithms proposed in this paper:

A1 - DFT based method (modified Algoritm 1 for special type of Sylvester matrices),
A2 - DFT based method for universal Sylvester matrices (Algoritm 1),
A3 - Schur's complement based method (Theorem 1),
A4 - standard symbolic computation performed by function `Det`, see (Chiasson *et al.*, 2003*a*).

Test square Sylvester matrices are of size m with 3 variables and degree $d = (d_1, d_2, d_3)$ where $\deg a = \deg b$, $\deg a_{d_a} = \deg b_{d_b} = 0$ (see (1)). The coefficients are random integers with absolute value less or equal 1000. A star (*) means that the execution time exceeds to 1000 seconds.

Table 2. Comparative execution times in second.

Input matrix		**Algorithm**			
m	d	A4	A2	A1	A3
4	$(4,4,4)$	4.17	0.64	0.59	1.17
6	$(5,5,5)$	42.1	6.32	6.25	7.56
6	$(6,6,6)$	*	12.72	11.7	117.06

Comments table 2 :

- Algorithm **A1** is slightly faster than **A2** because DFT is not computed in zero points and theorem 1 is used for faster computation of constant determinants.
- Symbolic computations are accurate, but long executions times make this approach practically unusable.

5. CONCLUSION

The problem of large determinants of polynomial matrices with special structure was considered in this report. The practical motivation standing behind is the design of a multilevel DC/AC inverter. Standard polynomial determinant routines are not efficient enough to tackle this complex task as has been evidenced by references to literature. Based on this founding, we studied possible ways to exploit the structure of the involved polynomial matrices and proposed algorithms that address this specific problem.

Namely, three algorithms were suggested, based respectively on FFT, the Schur's complement, and the determinant definition. All these approaches have been implemented in the *Mathematica* computational environment and exposed to experimental testing. The methods perform considerably better than standard symbolic routines that have been used for the multilevel inverter problem so far.

On the other hand, the multilevel inverter task has proved to be computationally demanding and some non-standard procedures have to be incorporated that complicate the routines. For instance, the precision increased beyond the IEEE-double standard in order to accommodate accumulating rounding errors makes the calculations much longer and more involving.

Obviously, some further effort will be needed to solve the multilevel inverter problem analytically for considerably greater number of DC sources. Although the proposed routines perform well in comparison with the symbolic approach, they do not perform well enough to defeat the rapidly increasing computational demands as the number of DC levels increases.

REFERENCES

Barnett, S. (1983). *Polynomial and Linear Control Systems.* Pure and Applied Mathematics.

Chiasson, J.N., K.J. McKenzie and Du Zhong (2003*a*). Elimination of harmonics in a multilevel converter using the theory of symmetric polynomials and resultants. *Proc. of the 42nd IEEE, Conf. on Decision and Control.*

Chiasson, J.N., L.M. Tolbert, K.J. McKenzie and Du Zhong (2003*b*). Control of multilevel converter using resultant theory. *IEEE Transactions on Control Systes Technology.*

Computing and information centre of CTU (2004 [online]). `http://www.civ.cvut.cz`.

Cox, D., J. Little and D. O'Shea (1996). *Varieties, and Algorithms an Introduction to Computational Algebraic Geometry and Commutative Algebra.* 2nd ed.. Springer-Verlag. New-York.

Higham, N.J. (1996). *Varieties, and Algorithms an Introduction to Computational Algebraic Geometry and Commutative Algebra.* SIAM.

Hromcik, M. and M. Sebek (2000). Fast fourier transform and robustness analysis with respect to parametric uncertainties. *Preprints of the IFAC Robust Control Design Conf.*

PolynomialPackage (2004 [online]). `http://puma.feld.cvut.cz`.

Wolfram Research, Inc. (2004 [online]). `http://www.mathematica.com`.