A MULTIVARIATE POLYNOMIAL MATRIX ORDER-REDUCTION ALGORITHM FOR LINEAR FRACTIONAL TRANSFORMATION MODELLING

Andrés Marcos, Declan G. Bates, Ian Postlethwaite

Dept. of Engineering, University of Leicester,
University Road, Leicester, LE1 7RH, U.K.,
{ame12, dgb3, ixp} @le.ac.uk

Abstract:
In this paper an algorithm that provides an equivalent, but of reduced order, representation for multivariate polynomial matrices is given. It combines ideas from computational symbolic algebra and from established techniques in graph representation and polynomial/matrix algebraic manipulations. The algorithm is applied to the problem of finding minimal linear fractional transformation models.

Keywords: linear fractional transformation, symbolic multivariate polynomial matrices

1. INTRODUCTION

Matrix manipulation is one of the basic cornerstones of many fields in engineering and mathematics. For example, in the field of control the basis of state-space theory is the representation of a system as a two-by-two block matrix and its subsequent use for synthesis and analysis (Skogestad, S. and Postlethwaite, I., 1996). Indeed, modern robust control theory is built on the concept of linear fractional transformations (LFT), which allows the representation of an uncertain system in terms of nominal and uncertain parts given by matrices (Balas, G.J. *et al.*, 1998). Similarly, polynomial representations and manipulations are also ubiquitous in many areas of mathematics - note, for example, that most computer algebra systems such as *Mathematica* (Wolfram, S., 1991) and Maple (Heck, A., 1993) rely on polynomial and rational representations.

A typical objective (for example, in signal processing and control synthesis) when operating on matrices and polynomials, especially when these are multivariate, is to obtain an equivalent representation of reduced order (in terms of number of parameters and their repetitions). Ideally, the order should be minimal, but minimal representations are in general very difficult to obtain except for some simple cases. In the case of LFT modelling, it is well-known that the problem of finding a minimal order representation is equivalent to a multidimensional realization (Cockburn, J.C., 2000) which remains an open problem. Indeed, most of the available LFT algorithms search for a minimal representation by exploiting the structure in the LFT models and performing algebraic factorizations on the multivariate matrices: structured-tree decomposition (Cockburn, J.C. and Morton, B.G., 1997), numerical matrix approaches (Belcastro, C.M. and Chang, B.C., 1998), Horner factorizations (Varga, A. and Looye, G., 1999) and symbolic linearizations (Varga, A. *et al.*, 1998) amongst others.

In this paper an algorithm is proposed that builds on previous symbolic techniques to achieve a lower order representation for multivariate polynomial matrices. The main difference with previous algorithms is the use of an information management scheme which uses a set of specialized metrics to evaluate the best (in terms of achievable order reduction) column/row direction and variable ordering for the decomposition and factorization stages. It is especially useful for LFT modelling, and hence, the application of the algorithm to this problem is described in detail.

2. THEORETICAL BACKGROUND

In this section the main mathematical tools used in the proposed algorithm are introduced. The basic structure of the proposed algorithm is a combination of the structured-tree decomposition (Cockburn, J.C. and Morton, B.G., 1997) and the Horner factorization (Varga, A. and Looye, G., 1999) approaches. Addi-

⋆ This research was supported thanks to EPSRC-UK under research grant *GR*/*S*61874/01.

tional tree-height reduction techniques from symbolic algebra (Heck, A., 1993; Armita, P. and De Micheli, G., 2001) and an extension of the metrics given in (Cockburn, J.C. and Morton, B.G., 1997) are used to improve the performance of the algorithm and to decide on the 'optimal' ordering of the variables and direction (along rows or columns) for the decomposition and factorization stages. Throughout the paper, 'optimal' will always mean best achievable reduction on the total number of parameters (and their repetitions) at a given step.

2.1 Metrics

The main metrics used in the logic of the algorithm are the "presence" degree σ, the factor order fac, the reduction order red, and the "possible" reduction order red_{pos}.

The "presence" degree, $\sigma(\delta_i)$, is defined as the number of times, including powers, a variable δ_i appears in an expression (monomial, polynomial or matrix). It can be viewed as a polynomial, or matrix, extension of the relative degree of a monomial. The factor order of a variable, $fac(\delta_i)$, is the maximum power to which it can be factored out from an expression.

The reduction order for a variable, $red(\delta_i)$, is the largest reduction on the "presence" degree of an expression achievable through factorization of that variable. Assuming there are n monomials in the expression, $red(\delta_i)$ is given by $(n-1)\,fac(\delta_i)$. The last metric, $red_{pos}(\delta_i)$, is equal to the reduction order in the case all the n monomials are factorizable (i.e. each monomial contains the parameters with a minimal order equal to $fac(\delta_i)$). If there are m non-factorizable monomials in the expression, the "possible" reduction degree is given by: $(n-m-1)\,fac(\delta_i)$.

2.2 Tree Decomposition & Horner Factorization

The structured-tree decomposition is an iterative approach that decomposes an initial matrix A_0 into simpler (i.e. of reduced "presence" degree) matrices by means of two basic operations, affine factorizations (L_i, R_i) and sum decompositions (A_i):

$$A_0 = L_1 A_1 R_1 + (A_2 + A_3) \qquad (1)$$

The name comes from its graph representation nature as each reduced-order matrix is in turn sum-decomposed or affine-factorized until it can not be decomposed further, forming a tree where the nodes are one of the two basic operations and the leaves the matrices as they are reduced. A third operation, called weighted-sum decomposition, is in fact an attempt to use information similar to the above metrics (i.e. maximal factor order) when no sum decompositions or affine factorizations exist. This approach has been recently coded and made publicly available for LFT modelling in the splendid linear fractional representation (LFR) toolbox from ONERA (Magni, J.F., 2004).

The Horner factorization approach is based on the Horner form simplification for polynomials. For the evaluation of a polynomial of degree k, it requires only k multiplications and k additions which is much less expensive than the number of multiplications for the expanded form (Wolfram, S., 1991). It is also numerically more efficient and accurate. The general univariate case is given by:

$$\mathcal{P}(x) = a_n + x\cdot(a_{n-1} + \ldots x\cdot(a_1 + x\cdot a_o)\ldots)) \qquad (2)$$

For the multivariate case, an ordering of the variables must be given. This ordering is not unique and will affect the nesting and the achievable reduction in the "presence" degree.

The Horner factorization is a polynomial extension of the function 'factor' which can be considered a tree-height reduction technique as, shown in the following example:

Example 1. Compare the graph representation of the expanded polynomial $\mathcal{P} = \delta_1\delta_2\delta_3 + \delta_3\delta_2^2$ with its Horner form $\mathbf{Horner}(\mathcal{P}) = \delta_3(\delta_2(\delta_2 + \delta_1))$, see Figure 1, the reduction in tree branches and nodes is obvious.

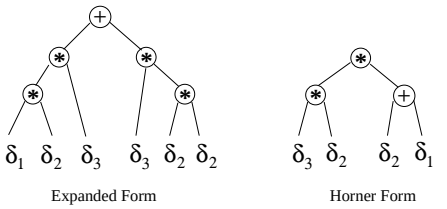

Fig. 1. Graph representation of $\mathcal{P}$ and $\mathbf{Horner}(\mathcal{P})$.

Note that the structured-tree decomposition approach does not incorporate Horner factorizations for polynomial expressions and thus will involve more operations (sums and products). Also, the use of the maximal factor order as a decision logic might prevent an optimal reduction of the "presence" degree. On the other hand, current Horner factorization algorithms follow either a pure lexicographical order or an exhaustive approach (i.e. try all combinations of the parameters) for the ordering of the variables. For large number of symbolic parameters and for the matrix case, where the optimal nesting of a polynomial might not result in optimal order reduction for the matrix, these approaches are too computationally expensive and do not typically result in a minimal realization, see following example:

Example 2. Given $M = \begin{bmatrix} \delta_1+\delta_1\delta_3^3 & \delta_3^3\delta_4 & \delta_1+\delta_1\delta_2 \\ \delta_2 & \delta_2\delta_4+\delta_5 & \delta_5 \end{bmatrix}$,
Using the standard (independent) polynomial Horner factorization on the $(1,1)$ and $(1,3)$ coefficients yields: $M = \begin{bmatrix} \delta_1(1+\delta_3^3) & \delta_3^3\delta_4 & \delta_1(1+\delta_2) \\ \delta_2 & \delta_2\delta_4+\delta_5 & \delta_5 \end{bmatrix}$.
Using the structured-tree approach, a weighted-sum decomposition (no affine or direct sum can be performed) yields:
$M = \begin{bmatrix} \delta_1(1+\delta_3^3) & 0 & \delta_1(1+\delta_2) \\ 0 & \delta_5 & \delta_5 \end{bmatrix} + \begin{bmatrix} 0 & \delta_3^3\delta_4 & 0 \\ \delta_2 & \delta_2\delta_4 & 0 \end{bmatrix}$
The left-most matrix can now be affine factorized yielding a "presence" degree of $\sigma = 6$. Similarly, an affine factorization can be applied to the right matrix to yield $\sigma = 5$, which yields a total "presence" degree of 11 (the implementation of the structured-tree in the ONERA LFR toolbox yields a σ degree of 12).
If the Horner factorization is performed matrix-wise and using the specified metrics, the ordered list for the multivariate Horner is $[\delta_3, \delta_1, \delta_2]$ which results in:
$M = \begin{bmatrix} \delta_1+\delta_1\delta_3^3 & \delta_3^3\delta_4 & \delta_1(1+\delta_2) \\ \delta_2 & \delta_2\delta_4+\delta_5 & \delta_5 \end{bmatrix}$.
This can be sum decomposed now as:
$M = \begin{bmatrix} \delta_1 & 0 & \delta_1(1+\delta_2) \\ 0 & \delta_5 & \delta_5 \end{bmatrix} + \begin{bmatrix} \delta_1\delta_3^3 & \delta_3^3\delta_4 & 0 \\ \delta_2 & \delta_2\delta_4 & 0 \end{bmatrix}$
Further affine factorizations yield a total σ of 9, which shows that multivariate Horner factorizations must be performed at the matrix-level (rather than at coefficient level).

2.3 Symbolic Algebraic Techniques

These techniques facilitate matrix manipulation and improve algorithmic performance when applied to symbolic expressions (Armita, P. and De Micheli, G., 2001). This has long been recognized and some algorithms already implement several or all of these techniques as they have also become available and optimized on all computer algebra systems. The two techniques incorporated in the algorithm proposed in this paper are expansion and substitution.

The procedure called "expand" distributes all products and positive powers over sums. Like terms are also collected and simplified, which can often result in immediate gains, e.g. $expand\{\delta_1(1+\delta_2)+\delta_1(1+\delta_1)\} = \delta_1(2+\delta_1+\delta_2)$. In the proposed algorithm, expansion is used after an iteration is completed, i.e. all possible factorizations and decompositions have been performed for the present matrix form. Its purpose is to reset the polynomial factorizations to check whether a new reduced-order matrix can be found.

Another important symbolic tool is "substitution", which replaces a sub-expression by a new symbolic variable. In its most developed form, it can be used to parallelize factorizations and reduce the complexity of an expression. In the proposed algorithm, it is used to reduce the complexity and speed up the calculation of the required metrics for the affine factorization stage.

Example 3. Given $\mathcal{P} = [\delta_1\delta_2 + \delta_1\delta_4^2 + 1 \quad \delta_1]$, it can be simplified (in terms of complexity of presentation) as $\mathcal{P} = [\delta_1(\delta_2+\delta_4^2)+1 \quad \delta_1] = [\delta_1\delta_{sub}+1 \quad \delta_1]$ so that now, the required metrics have to be calculated for only two variables: δ_1 and $\delta_{sub} = (\delta_2+\delta_4^2)$. Note as well, that any sum decomposition or affine factorization can be identified and performed faster.

3. HORNER-TREE DECOMPOSITION ALGORITHM

The proposed algorithm is called *Horner-Tree Decomposition* to emphasize its connection to the previous approaches. It generalizes them as it follows the layout of the tree decomposition approach but using the structure of the Horner factorization which has now been extended for multivariate polynomial matrices. This extension hinges on the additional symbolic tree-height reduction techniques described before and specially, on the use of a logic routine to gather and analyze the matrix decomposition at each stage.

The algorithm is divided into three main routines: information management, affine factorization and sum decomposition. The iterative combination of the last two routines yields a nested structured (i.e. Horner factorization) for multivariate polynomial matrices:

$$M = MB_1 + L_1\Big(MB_2 + L_2(\ldots(L_n\bar{MA}_nR_n)\ldots)R_2\Big)R_1 \tag{3}$$

The matrices $L_{ii}, \bar{MA}_{ii}, R_{ii}$ are obtained from the affine factorization of MA_{ii} which is obtained together with MB_{ii} from the sum decomposition of $\bar{MA}_{ii}$ (except for the initial iteration $ii = 1$ where MB_1 contains only constant numeric and constant symbolic parameters and MA_1 the rest of the symbolic coefficients). The matrices denoted as 'A' are called primary while those with 'B' are known as secondary.

3.1 Information Management

This routine condenses the logic of the algorithm and allows for a complete automated procedure for the decomposition. It is referred to as the information management routine, since its main objectives are to gather the proper information at each step of the algorithm, to analyze this information, to take a decision regarding direction and variable ordering, and to prepare the matrix for the next step.

The information gathering pertains mainly to calculating the metrics presented in Section 2.1 for all the symbolic parameters specified that are found within the matrix. For matrices with polynomial coefficients, the information manager also identifies their position and number of monomials, and prepares the matrix for the step to follow: affine factorization or sum decomposition. For the case of affine factorization, it finds the optimal 'Horner ordering' and performs a 'polynomial substitution'. For the sum decomposition, it forms a 'decomposition list' ordering the parameters.

The 'Horner ordering' of the parameters is performed based on the total (sum along rows or columns) "possible" reduction order of the parameter. This ordering might not be optimal for a particular polynomial coefficient but results in optimal matrix order reduction, see example 2. It is remarked again, that 'optimal' is defined based on the best achievable reduction in the σ degree for the specific matrix.

The 'polynomial substitution' performs a substitution of the polynomials and sub-polynomials, after the Horner factorization step, by dummy variables. The sub-polynomials are the polynomial remainders (or quotients) arising from the Horner factorization. These substitutions allow for simpler manipulation during the affine factorization subroutine as the polynomial matrix becomes a monomial matrix. It also performs substitution of 'irreducible' polynomial expressions which are those polynomials whose factorization from the matrix might imply a reduction in the σ degree, or alternatively can simplify the complexity of the expression, but that on themselves can not be factorized further, e.g. $\mathcal{P} = \delta_1(\delta_1+\delta_2) \equiv \delta_1\delta_{unred}$ but $\mathcal{P}2 = (\delta_1^2+\delta_1\delta_2) \neq \delta_{unred}$. After the affine factorization, the information manager back-substitutes the sub-polynomials and expands the matrix to prepare it for the sum decomposition step.

The 'decomposition list' contains information for the sum decomposition routine to attempt to optimize the reduction in the total σ degree for subsequent affine factorizations. Each row in the decomposition list is formed by the parameter number, its position (which row or column), the "possible" reduction order (the list is ordered in decreasing red_{pos}), and cells containing the indexes for the factorizable and non-factorizable coefficients along the specified row/column. This split of the matrix coefficients indicates the sum decomposition routine which coefficients should be assigned to the primary matrix MA_{ii} (factorizable coefficients) or to the secondary matrix MB_{ii} (the non-factorizable). Note, that once the non-factorizable coefficients are distributed to MB_{ii} the "possible" reduction order of $\bar{MA}_{ii}$ becomes the reduction order for MA_{ii}.

After the affine factorization and the sum decomposition have been performed, the information manager evaluates if the primary matrix MA_{ii} can be further decomposed or if the first of the non-decomposed $MB_{jj\leq ii}$ secondary matrices should be decomposed.

The priority is to decompose fully the primary matrix first, and subsequently to start with the secondary matrices (there might be more than one, as each time the primary matrix is passed through the decomposition scheme it will generate a secondary matrix). Note that as the secondary matrices are selected for the decomposition they become 'primary' and as such generate secondary matrices, see Figure 2.

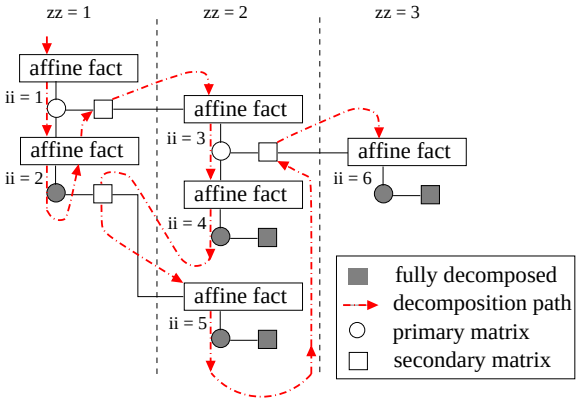

Fig. 2. Graph representation Horner-Tree algorithm.

3.2 Affine Factorization

In the affine factorization step, it is important to identify for each parameter the direction along which to perform its first factorization (i.e. left $\equiv$ rows or right $\equiv$ columns). In order to provide an optimal decision for a given parameter, the information management routine compares the total reduction and total "possible" reduction degrees along one direction with those along the other direction. There are seven possible cases:

Cases 1-3: Factorize along rows :

$$(1) red_{row} > red_{col} + red_{pos_{col}}$$
$$(2) red_{row} > red_{col} \;\&\; red_{pos_{row}} = red_{pos_{col}}$$
$$(3) red_{row} = red_{col} \neq 0 \;\&\; red_{pos_{row}} > red_{pos_{col}}$$

Cases 4-6: Factorize along columns :

$$(4) red_{col} > red_{row} + red_{pos_{row}}$$
$$(5) red_{row} < red_{col} \;\&\; red_{pos_{row}} = red_{pos_{col}}$$
$$(6) red_{row} = red_{col} \neq 0 \;\&\; red_{pos_{row}} < red_{pos_{col}}$$

The seventh case occurs when none of the previous cases is satisfied. In this case it will be very difficult and computationally expensive to ascertain along which direction to factorize. Hence, a preview of the immediate effect the factorization (along each direction) has on the decomposition is performed. The comparison is based on the two "future" matrices found by factorizing the parameter along each direction: Mf_L and Mf_R. For each of these "future" matrices, the maximum total reduction order (the sum of the σ's either along the rows or along the columns) is added to the total reduction order of the corresponding "actual" factorized matrix, M_L or M_R, and the obtained value compared to the same quantity for the other matrix. In case both achieve the same combined reduction order, an additional comparison is made based on the "possible" reduction orders of the "future" matrices and if this last comparison yields the same result, the default is to decompose along the rows (this is a design decision and the algorithm could obviously be allowed to proceed using the 'pre-viewing' capability deeper).

This approach doubles the number of calculations (as affine factorizations are performed in both directions and then only one selected) but it maximizes the total order reduction for the present and subsequent iteration.

Example 4. For the matrix, $M = \begin{bmatrix} \delta_1^2 & \delta_1 & \delta_1 \\ \delta_1^3 & \delta_1^2 & \delta_2 \end{bmatrix}$,
the reduction and "possible" reduction along the rows are: $red(\delta_1)_{rows} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$; $red_{pos}(\delta_1)_{rows} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$;
and along the columns:
$red(\delta_1)_{cols} = [2 \;\; 1 \;\; 0]$; $red_{pos}(\delta_1)_{cols} = [0 \;\; 0 \;\; 0]$;
The decision table formed by the total sum along rows and columns for each type of degree: total reduction $\sum red$ and total "possible" reduction $\sum red_{pos}$, is:

δ_1	**rows**	**cols**
$\sum red$	2	3
$\sum red_{pos}$	2	0

The above satisfies case (7), hence affine factorizations are performed along both directions:

$$M_L = L \;\; Mf_L = \begin{bmatrix} \delta_1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \delta_1 & 1 & 1 \\ \delta_1^3 & \delta_1^2 & \delta_2 \end{bmatrix}$$
$$M_R = Mf_R \;\; R = \begin{bmatrix} 1 & 1 & \delta_1 \\ \delta_1 & \delta_1 & \delta_2 \end{bmatrix} \begin{bmatrix} \delta_1^2 & 0 & 0 \\ 0 & \delta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The reduction orders of the "actual" matrices M_L and M_R are respectively 2 and 3. The corresponding decision tables for the "future" matrices Mf_L and Mf_R are:

$Mf_L(\delta_1)$	**rows**	**cols**
$\sum red$	0	1
$\sum red_{pos}$	2	0

$Mf_R(\delta_1)$	**rows**	**cols**
$\sum red$	0	0
$\sum red_{pos}$	1	0

The sum of the "actual" and "future" reduction orders for the left and right factorizations are 3 (=2+1) and 3 (=3+0) respectively. As the "future possible" reduction orders are 2 and 1 respectively, this indicates that there is a possibility to use sum decomposition on Mf_L and achieve a further reduction in the total σ when using the left factorization of the parameter first (indeed, this is the case as a further $red(\delta_1) = 2$ is obtained after the sum decomposition and an additional affine factorization -in the other direction only a further $red(\delta_1) = 1$ is achieved). Hence, this is the direction first chosen for that parameter.

If, after the affine factorization in one direction, it is possible to factorize along the other, i.e. the reduction order is non-zero, its factorization is performed before analyzing the direction for the next parameter.

This routine is applicable to both monomial and polynomial matrices, as the 'polynomial substitution' step is performed before the factorization routine is called.

3.3 Sum Decomposition

The basis of the sum decomposition is to decompose the primary matrix $\bar{MA}_{ii}$ from the affine factorization into two matrices, MA_{ii} and MB_{ii}:

$$\bar{MA}_{ii} = MA_{ii} + MB_{ii} \tag{4}$$

The sum decomposition is performed following the decomposition list obtained from the information manager. For the first row in the list, the sum decomposition assigns the factorizable coefficients for that parameter (and specified row or column) to MA_{ii}

and the rest of the coefficients along that defined row/column to MB_{ii}. Subsequently, it moves through the list from top to bottom evaluating if there is any conflict, i.e. new factorizable coefficients already placed in MB_{ii} or non-factorizable coefficients in MA_{ii}. It there is no conflict it distributes the new coefficients correspondingly and moves to the next row in the decomposition list. Otherwise, the "possible" reduction order for the parameter / position being evaluated is re-calculated after removing the conflicting coefficients. If the new red_{pos} is still better or equal than that for the next row in the decomposition list, the sum decomposition is performed with the updated coefficients, else the list is re-ordered to account for the new row and the routine proceeds to the next row in the list.

4. LFT MODELLING EXTENSION

A linear fractional transformation (LFT) is a representation of an uncertain system using two matrix operators $M = [M_{11}\ M_{12}; M_{21}\ M_{22}]$ and Δ, and a feedback interconnection, see Figure 3:

$$\mathcal{F}_U(M,\Delta) = M_{22} + M_{21}\Delta(I - M_{11}\Delta)^{-1}M_{21} \qquad (5)$$

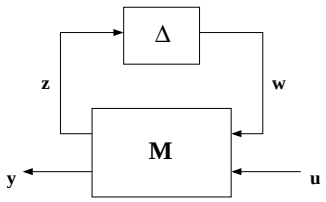

Fig. 3. Linear fractional transformation $LFT(M,\Delta)$.

The coefficient matrix M represents the nominal, known, part of the system. The uncertainty associated with the system Δ is norm-bounded, typically by 1, but otherwise unrestricted in form (structured / unstructured) or type (nonlinear / time-varying / constant). It is important to note that *unstructured uncertainty at component level becomes structured uncertainty at system level*.

The order of the LFT is said to be the number of uncertain parameters, including repetitions, that occur in Δ (e.g. $\Delta = [\delta_1 I_2 \quad 0\ ;\ 0 \quad \delta_2] => $ LFT order of 3). This is equivalent to the total matrix "presence" degree, i.e. the sum of the total σ for all the parameters. The order of an LFT is an important consideration for the control synthesis and analysis methods currently available in robust control. Indeed, in μ-analysis, which provides the main tool for performing robust stability and performance analysis, the size and type of the uncertainty matrix are key factors in determining the quality of the analysis results (Skogestad, S. and Postlethwaite, I., 1996). In particular, uncertainty matrices that contain many repeated parameters are well known to cause serious problems for the algorithms used to compute bounds on μ (Balas, G.J. *et al.*, 1998). Since many realistic robustness analysis problems can easily result in very high order LFTs, it is vital to have efficient (and automated) tools which can compute minimal, or at least close to minimal, representations of these systems.

A very important property of LFT systems is that their interconnection results in another LFT. The extension of the algorithm to LFT modelling is direct using this property and, in particular, the symbolic "object-oriented" realization of LFTs as proposed in (Magni, J.F., 2004). The "object-oriented" realization takes the point of view that for software such as MATLAB, LFTs are feedback interconnections of matrices and as such are subject to standard matrix operations: addition and multiplication of matrices are equivalent to parallel and series connection of LFTs (see the formulae given in (Lambrechts, P. *et al.*, 1993; Magni, J.F., 2004)).

More important for the application of the proposed algorithm, is the fact that the order of the uncertainty matrix for the resulting LFT is equal to the sum of the orders of the individual uncertainty matrices. Furthermore, using the results from (Magni, J.F., 2004), it is noted that the order of an LFT derived from a symbolic expression (where the symbolic variables are considered uncertain parameters) is equal to the total "presence" degree of the expression.

Hence, each of the individual matrices resulting from the proposed algorithm can be transformed into an LFT, with a diagonal uncertainty matrix of order equal to the respective total matrix "presence" degree. The resulting LFTs can be manipulated, following the defined multiplication and addition structure of the decomposition, to obtain a final LFT whose uncertainty matrix Δ is a diagonal matrix of order equal to the sum of the total "presence" degrees.

The applicability of this algorithm to dynamical systems is straight-forward recalling that an LFT is basically a generalization of the notion of state-space where the dynamic system is written as a feedback interconnection of a constant matrix and a diagonal element containing the integrator terms '$1/s$' and delays (Balas, G.J. *et al.*, 1998). This is valid as well for exact nonlinear modelling (Marcos, A. *et al.*, 2005) when the non-linearities are considered as parameters to be 'pulled out' into the Δ matrix.

5. RESULTS

Monte-Carlo tests of the software implementation are performed to evaluate the algorithm's capability and efficiency. In order to provide a baseline comparison, the command 'symtreed' provided in ONERA's LFR toolbox (Magni, J.F., 2004), which implements the structured-tree decomposition proposed in (Cockburn, J.C. and Morton, B.G., 1997), is also used.

The Monte-Carlo test is performed using 500 random polynomial matrices (200 3×3 matrices, 200 5×7 and 100 9×5). For each matrix dimension, half of the matrices are populated with 3 different symbolic parameters and the other half with 9 parameters. The random polynomial matrices are obtained by adding three independent random monomial matrices (thus, only polynomials of up to three monomials can be found in the matrix coefficients). Each random monomial matrix is obtained by creating a sparse random matrix with the chosen matrix dimensions $n \times m$ and a specified density, one matrix with density 60 and the other two with density 0.3 (so that not all the coefficients are polynomials). This density combination yields approximately $n{\cdot}m{\cdot}density$ uniformly distributed non-zero entries for each random monomial matrices. In order to fill each non-zero coefficient, k passes (where k is equal to 3 times the number of parameters, hence 9 or 27 passes) are made multiplying each time the

coefficients by δ_i^{pow} where δ_i is a randomly selected symbolic parameter from the defined list and *pow* is a random binary selection for the parameter's power (either 0 or 1, which means that each independent monomial can have a total degree of up to 9 or 27).

Table 1 shows the percentage of total runs for which one algorithm resulted in a smaller "presence" degree σ than the other. The number in parentheses indicates, respectively for each column in the table, the average total "presence" degree for the random polynomial matrix, and for the decomposed matrices using the Horner-tree and the structured-tree algorithms.

Table 1. Polynomial case: efficiency (% total runs $\sigma_A > \sigma_B$).

dimension	Horner-tree	Symtreed
3×3 (113)	82 % (62)	8.5 % (67)
5×7 (437)	99.5 % (215)	0 % (243)
9×5 (552)	100 % (266)	0 % (303)

The first important conclusion obtained looking at the average total "presence" degree is that the minimal-decomposition search step should be incorporated mainstream to the LFT modelling process (notice that almost 50 percent reduction is accomplished by both algorithms). This reduction in the number of parameters is not associated with a loss of modelling fidelity since the reduction is accomplished by matrix algebraic manipulations. Also, the current availability of software for LFT modelling and order-reduction (i.e. reference (Magni, J.F., 2004) and that stemming from this research) which simplifies its use to a simple MATLAB command supports this assertion.

Comparing the efficiency of the algorithms, it is observed that for relatively small matrices, both algorithms achieve the same reduction order 10 percent of the time (i.e. for 3×3 matrices the proposed Horner-tree algorithm resulted in better reduction 82 percent of the runs and the structured-tree algorithm in only 8.5 percent). As the complexity increases, i.e. larger σ and dimension for the random matrix, the Horner-tree algorithm improves the reduction order for almost every run.

Table 2 shows the quality of the improvement for each algorithm. The first number in each row is the average percentage improvement in the total "presence" degree between both algorithms, i.e. $pct = 100\sum[\frac{abs(\sigma(Htree)-\sigma(symtree))}{\sigma(symtree)}]/\#_{runs}$. The number in parentheses in the table indicates the average number of parameters by which one algorithm improved over the other.

Table 2. Polynomial case: performance (% average σ improvement).

dimension	Horner-tree	Symtreed
3x3	9.43 % (6)	4.72 % (2)
5x7	11.43 % (28)	0
9x5	11.81 % (36)	0

No detailed comparison on the computational time required is performed as the goal of our study is to obtain a smaller "presence" degree in the final representation. Nevertheless, it is noted that the implementation of the structured-tree decomposition is faster than the present implementation of the proposed algorithm (for 9×5 random matrices with σ of 400, typically one minute versus five minutes although for larger number of parameters the present implementation can be around 10 times slower). For the purposes of this research, the performance of the proposed algorithm is satisfactory, since for LFT modeling a reduction in the number of parameters is significantly more important than the computational time required. Future implementations of the code will reduce the current solution time.

6. CONCLUSIONS

In this paper an algorithm for the decomposition of multivariate polynomial matrices has been proposed. The resulting representation is equivalent but of reduced order, if order reduction is possible, to the original matrix. The algorithm is applicable to a wide range of fields, but particularly to linear fractional transformation modelling, one of the cornerstones in modern robust control theory. Monte-Carlo tests of the proposed algorithm applied to LFT modelling showed significant improved performance when compared with standard approaches.

REFERENCES

Armita, P. and De Micheli, G. (2001). Symbolic Algebra and Timing Driven Data-Flow Synthesis. In: *Design Automation Conference*. Las Vegas, NV. pp. 277–282.

Balas, G.J., Doyle, J.C., Glover, K., Packard, A. and Smith, R. (1998). μ-Analysis and Synthesis Toolbox.

Belcastro, C.M. and Chang, B.C. (1998). LFT Formulation for Multivariable Polynomial Problems. In: *American Control Conference*. Philadelphia, PA. pp. 1002–1007.

Cockburn, J.C. (2000). Multidimensional realizations of systems with parametric uncertainty. In: *Mathematical Theory of Networks and Systems*. Perpignan, France.

Cockburn, J.C. and Morton, B.G. (1997). Linear Fractional Representations of Uncertain Systems. *Automatica* **33**(7), 1263–1271.

Heck, A. (1993). *Introduction to Maple*. Springer-Verlag.

Lambrechts, P., Terlouw, J., Bennani, S. and Steinbuch, M. (1993). Parametric Uncertainty Modeling using LFTs. In: *American Control Conference*. San Franciso, CA. pp. 267–272.

Magni, J.F. (2004). Linear Fractional Representation Toolbox Modelling, Order Reduction, Gain Scheduling. Technical Report TR 6/08162 DCSD. ONERA, Systems Control and Flight Dynamics Department. Toulouse, France.

Marcos, A., Bates, D.G. and Postlethwaite, I. (2005). Exact Nonlinear Modeling using Symbolic Linear Fractional Transformations. In: *IFAC World Congress*. Praga, CH. Submitted.

Skogestad, S. and Postlethwaite, I. (1996). *Multivariable Feedback Control Analysis and Design*. Wiley.

Varga, A. and Looye, G. (1999). Symbolic and Numerical Software tools for LFT-based Low Order Uncertainty Modeling. In: *IEEE Symposium on Computed Aided Control System Design*. Hawai'i, USA.

Varga, A., Looye, G., Moormann, G. and Grubel, G. (1998). Automated Generation of LFT-Based Parametric Uncertainty Descriptions from Generic Aircraft Models. *Mathematical and Computer Modelling of Dynamical Systems* **4**(4), 249–274.

Wolfram, S. (1991). *Mathematica:* a System for Doing Mathematics by Computer. Addison-Wesley.