Control and Cybernetics

vol. **49** (2020) No. 1

On the block decomposition and spectral factors of λ-matrices*

by

Belkacem Bekhiti[1], Bachir Nail[2*], Abdelhakim Dahimene[3], Kamel Hariche[3] and George F. Fragulis[4]

[1] Institute of Aeronautics and Space Studies,
Saad Dahlab University, Blida, Algeria
[2] Mechanical Engineering, Materials and Structures Laboratory, Institute of Science and Technology, Tissemsilt University Center, Tissemsilt, Algeria
[3] Department of Control Systems, Electronics and Electrotechnics Institute, University of Boumerdes (Ex:INELEC), Algeria
[4] Western Macedonia University of Applied Sciences, Kozani, Greece
*corresponding author: nail_bachir@yahoo.fr

Abstract: In this paper we have factorized matrix polynomials into a complete set of spectral factors using a new design algorithm, and with some systematic procedures a complete set of block roots (solvents) have been obtained. The newly developed procedure is just an extension of the (scalar) Horner method to its block form for use in the computation of the block roots of matrix polynomial, the block-Horner method bringing a local iterative nature, faster convergence, nested programmable scheme, needless of any prior knowledge of the matrix polynomial, with the only one inconvenience, which is the strong dependence on the initial guess. In order to avoid this trap, we proposed a combination of two computational procedures, for which the complete program starts with the right block-$Q.D.$ algorithm. It is then followed by a refinement of the right factor by block-Horner's algorithm. This results in the global nature of the program, which is faster in execution, has well defined initial conditions, and good convergence in much less time.

Keywords: block roots, solvents, spectral factors, block-$Q.D.$ algorithm, block-Horner's algorithm, matrix polynomial

1. Introduction

In the early days of control and system theory, frequency domain techniques were the principal tools of analysis, modeling and design for linear systems. However, it is the dynamic systems that can be modeled by a scalar m^{th} order linear differential (difference) equation with constant coefficients that are amenable to

*Submitted: May 2020; Accepted: June 2020

this type of analysis, see DiStefano, Stubberud and Williams (1967) and Gohberg, Lancaster and Rodman (1982). These systems have a single input and a single output (SISO). In this case, the transfer function is a ratio of two scalar polynomials. The dynamic properties of the system (time response, stability, etc.) depend on the roots of the denominator, or, in other words, on the solution of the underlying homogeneous differential equation (difference equation), see Yaici and Hariche (2014). The denominator of such a system is a scalar polynomial and its spectral characteristics depend on the location of its roots in the s-plane, hence the factorization (root finding) of scalar polynomials is an important tool of analysis and design for linear systems, see Dahimene (2009). But for the systems that have multiple inputs and multiple outputs (MIMO), the dynamics can be modeled by high-degree coupled differential equations or l^{th} degree m^{th} order vector linear differential (difference) equation with matrix of constant coefficients, which can be considered as an extension to the scalar case, and then resulting in matrix transfer function. When one studies high order MIMO systems, the size of the matrices involved becomes prohibitive. This is why there is a reappearance nowadays of transfer function (which become rational matrices) description, see Chen (1984), Kailath (1980) or Kucera (1979). In this context, the dynamic properties of the system under study are determined by the latent roots and /or the spectral factors of a matrix polynomial. This is why we find quite a lot of publications, associated with those matrices in system and control journals, as this is exemplified by the studies od Ahn (1982), Resende and Kaskurewicz (1989), Shieh and Solak (1987).

The algebraic theory of matrix polynomials has been investigated by Dennis, Traub and Weber (1976), Denman (1977), Denman and Beavers (1976), Gohberg, Kaashoek and Rodman (1978), Shieh and Chahin (1981), Shieh and Tsay (1981), and Tsai, Shieh and Shen (1988). Various computational algorithms (see Dennis, Traub and Weber, 1976; Denman and Beavers, 1976; Gohberg, Kaashoek and Rodman, 1978; Shieh and Chahin, 1981, Shieh and Tsay, 1984) are available for finding the solvents and spectral factors of a matrix polynomial. A very well-known method and approach (Dennis, Traub and Weber, 1976, 1978; Shieh, Chang and McInnis, 1986) is the use of the eigenvalues and eigenvectors of the block companion form matrix that can be constructed from the matrix polynomial (the λ-matrices) to construct the solvents of the matrix polynomial based upon the definition of solvents.

Still, it is often inefficient to explicitly determine the eigenvalues and eigenvectors of a matrix, which can be ill conditioned and either non-defective or defective. On the other hand, yet without prior knowledge of the eigenvalues and eigenvectors of the matrix, the Newton-Raphson method (Shieh and Chahin, 1981; Shieh, Tsay and Coleman, 1981) has been successfully utilized for finding the solvents. Also, the block-power method has been developed by Tsai, Shieh and Shen (1988) for finding the solvents and spectral factors of a general nonsingular polynomial matrix, which may be monic and/or comonic.

However, the matrix polynomial of interest must have distinct block solvents, and the convergence rate of the power method depends strongly on the ratio of the two block eigenvalues of largest magnitude, see Shieh and Tsay (1982). Moreover, there are quite some numerical methods for computing the block roots of matrix polynomials without any prior knowledge of the eigenvalues and eigenvectors of the matrix polynomial. Of such sophisticated algorithms, the most efficient and satisfactorily stable one that gives the complete set of solvents at the same time is the $Q.D.$ algorithm. The use of the $Q.D.$ algorithm for such purpose has been suggested by K. Hariche (1987), and has been briefly detailed, studied and extended to the matrix polynomials by Dahimene (2009).

The purpose of this paper is, first, to briefly illustrate the so called block quotient-difference ($Q.D.$) algorithm as developed by Dahimene (2009) and, secondly, to extend the (scalar) Horner method to its block form for use in the computation of the block roots of matrix polynomial and the determination of the complete set of solvents and spectral factors of a monic polynomial. Without any prior knowledge of the eigenvalues and eigenvectors of the matrix, to mention the scalar case, one can refer to A. Pathan and T. Collyer (2003), who present an excellent survey on Horner's method and its application in solving polynomial equations by determining the location of roots.

The objectives of this paper can be described as follows:

- Illustration and finalization of the block quotient-difference ($Q.D.$) algorithm for the purpose of spectral decomposition and matrix polynomial factorization.
- Construction of a new block-Horner array and block-Horner algorithm for extracting the complete set of spectral factors of matrix polynomials.
- Proposing a combined algorithm for the purpose of fast convergence, high stability and for avoiding the initial guess.
- Finally, we have commented on and discussed the obtained results with some perspectives and suggestions, which are stated for the completeness of the work, and we finish the paper by conclusion.

2. Preliminaries

For completeness of derivations presented in the latter part of this paper, we review pertinent definitions and theorems below.

2.1. Survey on matrix polynomials

Here we are going to define and explore some algebraic theory of matrix polynomials, solvents, latent structure, spectral factors and the transformation between solvents and spectral factors.

Definition 1 *Given the set of $m \times m$ complex matrices $A_0, A_1, ..., A_l$, the following matrix valued function of the complex variable λ is called a matrix polynomial of degree l and order m:*

$$A(\lambda) = A_0\lambda^l + A_1\lambda^{l-1} + ... + A_{l-1}\lambda + A_l. \tag{1}$$

Definition 2 *The matrix polynomial $A(\lambda)$ is called:*
i. Monic if A_0 is the identity matrix.
ii. Comonic if A_l is the identity matrix.
iii. Regular if $det(A(\lambda)) \neq 0$.
iv. Singular if $det(A(\lambda))$ is identically zero.
v. Unimodular if $det(A(\lambda))$ is nonzero constant.

Definition 3 *The complex number λ_i is called a latent root of the matrix polynomial $A(\lambda)$ if it is a solution of the scalar polynomial equation $det(A(\lambda)) = 0$. The nontrivial vector p, solution of $A(\lambda_i)p = 0_m$, is called a primary right latent vector associated with λ_i. Similarly, the nontrivial vector q, solution of $q^T A(\lambda_i) = 0_m$, is called a primary left latent vector associated with λ_i.*

Remark 1 *In this work we suppose that the eigenvalues of the matrix polynomial $A(\lambda)$ are distinct, and that $A(\lambda)$ is a monic matrix polynomial.*

If $A(\lambda)$ has a singular leading matrix coefficient (A_l), then $A(\lambda)$ has latent roots at infinity. From the definition we can see that the latent problem of a matrix polynomial is a generalization of the concept of eigenproblem for square matrices. Indeed, we can consider the classical eigenvalues/vector problem as finding the latent root/vector of a linear matrix polynomial ($\lambda I - A$). We can also define the spectrum of a matrix polynomial $A(\lambda)$ as being the set of all its latent roots (notation $\sigma(\lambda)$). It is essentially the same definition as the one of the spectrum of a square matrix.

Definition 4 *A right block root is also called solvent of monic λ-matrix $A(\lambda)$ and is an $m \times m$ real matrix R such that:*

$$\begin{aligned} &R^l + A_1R^{l-1} + ... + A_{l-1}R + A_l = O_m \\ &\Leftrightarrow A_R(R) = \sum_{i=0}^{l} A_iR^{l-i} = O_m, \end{aligned} \tag{2}$$

while a left solvent is an $m \times m$ real matrix L such that:

$$\begin{aligned} &L^l + L^{l-1}A_1 + ... + LA_{l-1} + A_l = O_m \\ &\Leftrightarrow A_L(L) = \sum_{i=0}^{l} L^{l-i}A_i = O_m. \end{aligned} \tag{3}$$

The following are important facts on solvents (Leang et al., 1986):

- Solvents of a matrix polynomial do not always exist.

- Generalized right (left) eigenvectors of a right (left) solvent are the generalized latent vectors of the corresponding matrix polynomial.

DEFINITION 5 *A matrix R (respectively: L) is called a right (respectively: left) solvent of the matrix polynomial if and only if the binomial $(\lambda I - R)$ (respectively: $(\lambda I - L)$) divides exactly $A(\lambda)$ on the right (respectively: left).*

THEOREM 1 *(Hariche and Denman, 1989) Given a matrix polynomial*

$$A(\lambda) = A_0\lambda^l + A_1\lambda^{l-1} + ... + A_{l-1}\lambda + A_l \tag{4}$$

a) The remainder of the division of $A(\lambda)$ on the right by the binomial $(\lambda I - X)$ is $A_R(X)$
b) The remainder of the division of $A(\lambda)$ on the left by the binomial $(\lambda I - X)$ is $A_L(X)$.

This means that there exist matrix polynomials $Q(\lambda)$ and $S(\lambda)$ such that:

$$\begin{array}{rcl} A(\lambda) & = & Q(\lambda)(\lambda I - X) + A_R(X) \\ & = & (\lambda I - X)S(\lambda) + A_L(X). \end{array} \tag{5}$$

COROLLARY 1 *Hariche and Denman (1989) also give the fundamental relation that exists between the right solvent (respectively: left solvent) and the right (respectively: left) linear factor:*

$$\begin{array}{rcl} A_R(X) & = & 0 \text{ iff } A(\lambda) = Q(\lambda)(\lambda I - X) \\ A_L(X) & = & 0 \text{ iff } A(\lambda) = (\lambda I - X)S(\lambda). \end{array} \tag{6}$$

DEFINITION 6 *A set of solvents $\{R_1, R_2, ..., R_l\}$ is a complete set if $\cup\sigma(R_i) = \sigma(A_c)$, where σ denotes the spectrum of a matrix.*

THEOREM 2 *(Dennis, Traub and Weber, 1976) A set of solvents $\{R_1, R_2, ..., R_l\}$ is a complete set if and only if $det(V_R(R_1, R_2, ..., R_l)) \neq 0$, where V_R is a Vandermonde matrix, corresponding to $\{R_1, R_2, ..., R_l\}$, given as*

$$V_R(R_1, R_2, ..., R_l) = \begin{pmatrix} I_m & I_m & ... & I_m \\ R_1 & R_2 & ... & R_l \\ \vdots & \vdots & ... & \vdots \\ R_1^{l-1} & R_2^{l-1} & ... & R_l^{l-1} \end{pmatrix}. \tag{7}$$

REMARK 2 *We can define a set of left solvents in the same way as in the previous theorem. The relationship between latent roots, latent vectors, and the solvents can be stated as follows:*

THEOREM 3 *(Tsai, Chen and Shieh, 1992) If $A(\lambda)$ has n linearly independent right latent vectors $p_1, p_2, ..., p_n$ (left latent vectors $q_1, q_2, ..., q_n$) corresponding to latent roots $\lambda_1, \lambda_2, ..., \lambda_n$, then $P\Lambda P^{-1}, (Q\Lambda Q^{-1})$ is a right (left) solvent, where: $P = [p_1, p_2, ..., p_n], (Q = [q_1, q_2, ..., q_n]^T)$ and $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$.*

Proof See Tsai, Chen and Shieh (1992).

Theorem 4 *(Tsai, Chen and Shieh, 1992) If $A(\lambda)$ has n latent roots $\lambda_1, \lambda_2, ..., \lambda_n$ and the corresponding right latent vectors $p_1, p_2, ..., p_n$, as well as the left latent vectors $q_1, q_2, ..., q_n$ are both linearly independent, then the associated right solvent R and left solvent L are related by: $R = WLW^{-1}$, where: $W = PQ$ and $P = [p_1, p_2, ..., p_n]$, $(Q = [q_1, q_2, ..., q_n]^T)$ and "T" stands for transpose.*

For design and analysis of large-scale multivariable systems, it is necessary to determine a complete set of solvents of the matrix polynomial. Given the matrix polynomial $A(\lambda)$, if a right solvent R is obtained, the left solvent L of $A(\lambda)$, associated with R, can be determined by using the following algorithmic relationship (Tsai, Chen and Shieh, 1992):

$$L = Q^{-1}RQ \qquad \text{rank}(Q) = m, \tag{8}$$

where Q is the solution of the following linear matrix equation (see Tsai, Chen and Shieh, 1992):

$$\sum_{i=0}^{l-1} R^{l-1-i} Q B_i = I_m, \tag{9}$$

or, in the vector form, using the Kronecker product, we have

$$Vec(Q) = \left(\sum_{i=0}^{l-1} {B_i}^T \bigotimes \left(\ R^{l-1-i} \ \right) \right)^{-1} Vec(I_m), \tag{10}$$

where: $\bigotimes$ designates the Kronecker product, and B_i are the matrix coefficients of $B(\lambda)$ with $\lambda I_m - R$ factored out from $A(\lambda)$, i.e.,

$$B(\lambda) = A(\lambda) \left(\ \lambda I_m - R \ \right)^{-1} = \sum_{i=0}^{l-1} B_i \lambda^{l-1-i} \tag{11}$$

$$B(\lambda) = B_0 \lambda^{l-1} + B_1 \lambda^{l-2} + ... + B_{l-1}. \tag{12}$$

We can compute the coefficients B_i, using the algorithm of synthetic division:

$$\begin{aligned} ccB_0 &= I_m \\ B_1 &= B_0 A_1 + B_0 R \\ &\cdots \\ B_k &= B_0 A_k + B_{k-1} R \quad k = 1, 2, ..., l-1 \\ O_m &= B_0 A_l + B_{l-1} R. \end{aligned}$$

Theorem 5 *(Tsai, Chen and Shieh, 1992) If the elementary divisors of $A(\lambda)$ are linear, then $A(\lambda)$ can be factored into the product of l-linear monic λ-matrices called a complete set of spectral factors,*

$$A(\lambda) = (\lambda I_m - Q_l)(\lambda I_m - Q_{l-1})...(\lambda I_m - Q_1), \tag{13}$$

where $(\lambda I_m - Q_i),\ i = 1, ..., l$ *are referred to as a complete set of linear spectral factors. The* $m \times m$ *complex matrices* $Q_i, i = 1, ..., l$ *are called the spectral factors of the* λ*-matrix* $A(\lambda)$*.*

The most right spectral factor Q_1 is a right solvent of $A(\lambda)$ and the most left spectral factor Q_l is a left solvent of $A(\lambda)$, whereas the remaining spectral factors may or may not be solvents of $A(\lambda)$. The relationship between solvents and spectral factors was explored in Tsay, Chen and Shieh(1992), and various transformations have been developed.

2.2. Transformation of solvents to spectral factors

Since the diagonal forms of a complete set of solvents and those of a complete set of spectral factors are identical, then they are related by similarity transformation.

THEOREM 6 *(Shieh and Tsay, 1981) Consider a complete set of right solvents* $\{R_1, R_2, ..., R_l\}$ *of monic* λ*-matrix* $A(\lambda)$*; then* $A(\lambda)$ *can be factored as:*

$$A(\lambda) = N_l(\lambda) = (\lambda I_m - Q_l)(\lambda I_m - Q_{l-1})...(\lambda I_m - Q_1)$$

by using the following recursive scheme

$$Q_k = \left(N_{(k-1)R}(R_k)\right) R_k \left(N_{(k-1)R}(R_k)\right)^{-1}, \qquad k = 1, ..., l \tag{14}$$

where:

$$N_k(\lambda) = (\lambda I_m - Q_k) N_{k-1}(\lambda) \qquad k = 1, ..., l \tag{15}$$

and for any j *we write*

$$N_{kR}(R_j) = N_{(k-1)R}(R_j) R_j - Q_k N_{(k-1)R}(R_j), \qquad k = 1, ..., l$$

with: $N_0(\lambda) = I_m$ $N_{0R}(R_j) = I_m$ *for any* j *and rank*$(N_{(k-1)R}(R_k)) = m, \quad k = 1, ..., l$*.*

PROOF See Shieh and Tsay (1981).

In a similar manner, the spectral factors can be obtained from the known L_i of $A(\lambda)$ as follow:

$$Q_k = Q_{l+1-k} \tag{16}$$

$$Q_k = \left(\ M_{(k-1)L}(L_k)\ \right)^{-1} L_k \left(\ M_{(k-1)L}(L_k)\ \right), \qquad k = 1, ..., l \tag{17}$$

where

$$M_k(\lambda) = M_{k-1}(\lambda)(\lambda I_m - Q_k) \qquad k = 1, ..., l \tag{18}$$

and for any j we write

$$M_{kL}(L_j) = L_j M_{(k-1)L}(L_j) - M_{(k-1)L}(L_j)Q_k, \qquad k = 1, ..., l$$

with:

$$M_0(\lambda) = I_m \quad M_{0L}(L_j) = I_m \quad \text{for any } j$$

$$\text{rank}(M_{(k-1)L}(L_k)) = m \qquad k = 1, ..., l.$$

$M_{(k-1)L}(L_i)$ is a left matrix polynomial of $M_{(k-1)}(\lambda)$ having λ replaced by a left solvent L_j, so that the spectral factorization of $A(\lambda)$ becomes:

$$A(\lambda) = M_l(\lambda) = (\lambda I_m - Q_1)(\lambda I_m - Q_2)...(\lambda I_m - Q_l).$$

2.3. Transformation of spectral factors to solvents

Given a complete set of spectral factors of a λ-matrix $A(\lambda)$, then a corresponding complete set of right (left) solvents can be obtained. The transformation of spectral factors to right (left) solvents of a λ-matrix can be derived as follows:

Theorem 7 *Given a monic λ-matrix with all elementary divisors being linear*

$$A(\lambda) = (\lambda I_m - Q_l)(\lambda I_m - Q_{l-1})...(\lambda I_m - Q_1)$$

where Q_i ($\triangleq Q_{l+1-i}$) $i = 1, ..., l$ are a complete set of spectral factors of a λ-matrix $A(\lambda)$, and $Q_i \bigcap Q_j = \varnothing$ define λ-matrices $N_i(\lambda)$ $i = 1, ..., l$ as follows:

$$N_i(\lambda) = (\lambda I - Q_i)^{-1} N_{i-1}(\lambda) \tag{19}$$

$$N_i(\lambda) = I_m \lambda^{l-i} + A_{1i}\lambda^{l-i-1} + ... + A_{(l-i-1)i}\lambda + A_{(l-i)i} \tag{20}$$

with $N_0 = A(\lambda)$, then the transformation matrix P_i, which transforms the spectral factor Q_i ($\triangleq Q_{l+1-i}$) to the right solvent R_i ($\triangleq R_{l+1-i}$) of $A(\lambda)$ can be constructed from the new algorithm as follows ($rank(P_i) = m$):

$$R_i \triangleq R_{l+1-i} = P_i Q_i {P_i}^{-1} i = 1, ..., l \tag{21}$$

where the $m \times m$ matrix P_i can be solved from the following matrix equation, $i = 1, ..., m$:

$$Vec(P_i) = (G_{Ni}(Q_i))^{-1} Vec(I_m) \tag{22}$$

in which $G_{Ni}(Qi)$ ($rank(G_{N_i}(Q_i)) = m^2$) is defined by:

$$G_{Ni}(Q_i) \triangleq (Q_i^{l-i})^T \otimes I_m + (Q_i^{l-i-1})^T \otimes A_{1i} + ... + {Q_i}^T \otimes A_{(l-i)i} + I_m \otimes A_{(l-i)i}.$$

Proof See Shieh and Tsay (1981).

In the same fashion the complete set of spectral factors $Q_i,\ i = 1, 2, ..., l$ can be converted into the left solvents $L_i,\ i = 1, 2, ..., l$ using the following algorithm:

$$M_i(\lambda) = M_{i-1}(\lambda)(\lambda I_m - Q_i)^{-1},\ \ i = 1, ..., l \tag{23}$$

$$M_i(\lambda) = I_m\lambda^{l-i} + A_{1i}\lambda^{l-i-1} + ... + A_{(l-i-1)i}\lambda + A_{(l-i)i} \tag{24}$$

$$H_{Mi}(Q_i) \triangleq I_m \otimes Q_i^{l-i} + A_{1i}^T \otimes Q_i^{l-i-1} + ... + A_{(l-i-1)i}^T \otimes Q_i + A_{(l-i)i}^T \otimes I_m$$

$$Vec(S_i) = (H_{Mi}(Q_i))^{-1} Vec(I_m), \qquad \text{rank}(H_{M_i}(Q_i)) = m^2$$

$$L_i = S_i^{-1} Q_i S_i \qquad i = 1, ..., l. \tag{25}$$

2.4. Block companion forms

In analogy with scalar polynomials, a useful tool for the analysis of matrix polynomials is the block companion form matrix. Given a λ-matrix as in eq.(1) where $A_i \in C^{m\times m}$ and $\lambda \in C$, the associated block right and left companion form matrices are:

$$A_R = \begin{pmatrix} O_m & I_m & \cdots & O_m \\ O_m & O_m & \cdots & O_m \\ \vdots & \vdots & \ldots & O_m \\ O_m & O_m & \vdots & I_m \\ -A_l & -A_{l-1} & \cdots & -A_1 \end{pmatrix}, \quad A_L = \begin{pmatrix} O_m & \cdots & O_m & -A_l \\ I_m & \cdots & O_m & -A_{l-1} \\ \vdots & \vdots & \vdots & \vdots \\ O_m & \cdots & O_m & -A_2 \\ O_m & \cdots & I_m & -A_1 \end{pmatrix}. \tag{26}$$

Note that A_L is the block transpose of A_R. If the matrix polynomial $A(\lambda)$ has a complete set of solvents, these companion matrices can be respectively block diagonalised via the right (left) block Vandermonde matrix, defined by:

$$V_R = \begin{pmatrix} I_m & I_m & ... & I_m \\ R_1 & R_2 & ... & R_l \\ \vdots & \vdots & ... & \vdots \\ R_1^{l-1} & R_2^{l-1} & ... & R_l^{l-1} \end{pmatrix}, \quad V_L = \begin{pmatrix} I_m & L_1 \cdots {L_1}^{l-1} \\ I_m & L_2 \cdots {L_2}^{l-1} \\ \vdots & \vdots \ \ddots \ \vdots \\ I_m & L_l \cdots {L_l}^{l-1} \end{pmatrix} \tag{27}$$

where $R_1, R_2, ..., R_l$ and/or $L_1, L_2, ..., L_l$ represent the complete set of right (left) solvents. Since the block Vandermonde matrices are nonsingular, see Yaici and Hariche (2014a,b), and Yaici, Hariche and Tim (2014), we can write

$${V_R}^{-1} A_R V_R = \text{Blockdiag}(R_1, R_2, ..., R_l) \tag{28}$$

$${V_L}^{-1} A_L V_L = \text{Blockdiag}(L_1, L_2, ..., L_l). \tag{29}$$

These similarity transformations do a block decoupling of the spectrum of $A(\lambda)$, which is very useful in the analysis and design of high order control systems.

3. Spectral factorization algorithms

In this section we are going to present some of the existing algorithms that can factorize a linear term from a given matrix polynomial. Firstly, we consider the generalized quotient difference algorithm, and next we give a new extended algorithm, based on the Horner scheme. The matrix quotient-difference ($Q.D.$) algorithm is a generalization of the scalar one (Henrici, 1958). The use of the $Q.D.$ algorithm for such purpose has been suggested firstly in Zabot and Hariche (1997). The scalar $Q.D.$ algorithm is just one of the many global methods that are commonly used for finding the roots of a scalar polynomial. The Quotient-Difference scheme for matrix polynomials can be defined just like the scalar one (Dahimene, 2009) by a set of recurrence equations. The algorithm consists in building a table that we call the $Q.D.$ tableau.

3.1. The right block matrix $Q.D.$ algorithm

Given a matrix polynomial with nonsingular coefficients as in eq. (4), the objective is to find the spectral factors of $A(\lambda)$ that will allow us to write $A(\lambda)$ as a product of n linear factors as in eq. (14). Writing in block left companion form, we have:

$$C_3 = \begin{pmatrix} -A_1 & I_m & \cdots & O_m \\ -A_2 & O_m & \cdots & O_m \\ \vdots & \vdots & \vdots & \vdots \\ -A_{l-1} & \cdots & \cdots & I_m \\ -A_l & \cdots & \cdots & O_m \end{pmatrix}. \tag{30}$$

The required transformation is a sequence of LR decomposition such that:

$$C_3 = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} I_m & O_m \\ X_m & I_m \end{pmatrix} \begin{pmatrix} A & B \\ C & D \end{pmatrix} \tag{31}$$

where:

$$C_{11} = \begin{pmatrix} -A_1 & I_m & \cdots & O_m \\ -A_2 & O_m & \cdots & O_m \\ \vdots & \vdots & \vdots & \vdots \\ -A_{l-2} & \cdots & \cdots & I_m \\ -A_{l-1} & \cdots & \cdots & O_m \end{pmatrix}, \qquad C_{12} = \begin{pmatrix} O_m \\ O_m \\ \vdots \\ O_m \\ I_m \end{pmatrix}$$

$$C_{21} = [-A_l \;\; O_m \;\; O_m, ..., O_m], \qquad C_{22} = [O_m].$$

It is required to have $C = 0$, then let

$$X = [-X_1 \;\; X_2 \;\; X_3, ..., X_{l-1}]. \tag{32}$$

We obtain the following set of equations:

$$X_1 A_1 + X_2 A_2 + ... + X_{l-1} A_{l-1} = A_l$$
$$X_1 = X_2 = ... = X_{l-2} = 0$$
$$X_{l-1} + D = 0.$$

The above leads to the following decomposition of C_3:

$$C_3 = \begin{pmatrix} I_m & \cdots & O_m & O_m \\ O_m & \cdots & O_m & O_m \\ \vdots & \vdots & \vdots & \vdots \\ O_m & \cdots & I_m & O_m \\ O_m & \cdots & A_l A_{l-1}^{-1} & I_m \end{pmatrix} \begin{pmatrix} -A_1 & I_m & \cdots & O_m \\ -A_2 & O_m & \cdots & O_m \\ \vdots & \vdots & \vdots & \vdots \\ -A_{l-1} & O_m & \cdots & I_m \\ O_m & O_m & \cdots & -A_l A_{l-1}^{-1} \end{pmatrix}.$$

Hence, C_3 can be written as:

$$C_3 = L_{-(l-2)} R_{-(l-2)}. \tag{33}$$

We continue this process for the block C_{11} up till C_3 is equivalent to a matrix R_0

$$C_3 = L_{-(l-2)} L_{-(l-3)} ... L_0 R_0 \tag{34}$$

$$R_0 = \begin{pmatrix} -A_1 & I_m & \cdots & O_m & O_m \\ O_m & -A_2 A_1^{-1} & \cdots & O_m & O_m \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ O_m & O_m & \cdots & -A_{l-1} A_{l-2}^{-1} & I_m \\ O_m & O_m & \cdots & O_m & -A_l A_{l-1}^{-1} \end{pmatrix}. \tag{35}$$

It is clear that if the matrices $L_0, L_{-1}, ..., L_{l-2}$ are equal to identity matrices, then the block companion matrix C_3 will be similar to the following matrix:

$$M = \begin{pmatrix} Q_1 & I_m & \cdots & O_m & O_m \\ O_m & Q_2 & \cdots & O_m & O_m \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ O_m & O_m & \cdots & Q_{l-1} & I_m \\ O_m & O_m & \cdots & O_m & Q_l \end{pmatrix}. \tag{36}$$

The following theorem shows that under certain conditions, the sequence of $L_0, L_{-1}, ..., L_{l-2}$ converges to identities.

THEOREM 8 *Let $M = X\Lambda X^{-1}$, where*

$$\Lambda = \begin{pmatrix} R_1 & O_m & \cdots & O_m \\ O_m & R_2 & \cdots & O_m \\ \vdots & \vdots & \ddots & \vdots \\ O_m & O_m & \cdots & R_l \end{pmatrix}. \tag{37}$$

If the following conditions are satisfied:

a) dominance relation exists between R_k: $R_1 > R_2 > ....R_l$
b) $X^{-1} = Y$ has a block LR factorization $L_y R_y$
c) X has a block LR factorization $L_x R_x$

then the block LR algorithm just defined converges (i.e. $L_k \to I$).

Proof The proof is similar to the one in Parlett (1967), see also Dahimene (2009).

It means that we can start the $Q.D.$ algorithm by considering:

$$E_1{}^{(0)} = -A_2 A_1{}^{-1}, \quad E_2{}^{(-1)} = -A_3 A_2{}^{-1},$$
$$E_3{}^{(-2)} = -A_4 A_3{}^{-1}, ..., E_{l-1}{}^{-(l-2)} = -A_l A_{l-1}{}^{-1},$$
$$Q_1{}^{(0)} = -A_1, Q_2{}^{(-1)} = O_m, ..., Q_{l-1}{}^{-(l-2)} = O_m.$$

Those last two equations provide us with the first two rows of the $Q.D.$ tableau (one row of $Q's$ and one row of $E's$). Hence, we can solve the rhombus rules for the bottom element (called the south element by Henrici, 1958). We obtain the row generation of the $Q.D.$ algorithm:

$$\begin{cases} Q_i{}^{(j+1)} = Q_i{}^{(j)} + E_i{}^{(j)} - E_i{}^{(j+1)} \\ E_i{}^{(j+1)} = Q_{i+1}{}^{(j)} E_i{}^{(j)} \left[\ Q_i{}^{(j+1)} \ \right]^{-1} \end{cases} . \tag{38}$$

Writing this in tabular form yields

Table 1. The extended $Q.D$ scheme

	$-A_1$		O_m		O_m		$\cdots$
O_m		$-A_2 A_1{}^{-1}$		$-A_3 A_2{}^{-1}$		$-A_4 A_3{}^{-1}$	$\cdots$
	$Q_1{}^{(1)}$		$Q_2{}^{(0)}$		$Q_3{}^{(-1)}$		$\cdots$
O_m		$E_1{}^{(1)}$		$E_2{}^{(0)}$		$E_3{}^{(-1)}$	$\cdots$
	$Q_1{}^{(2)}$		$Q_2{}^{(1)}$		$Q_3{}^{(0)}$		$\cdots$
O_m		$E_1{}^{(2)}$		$E_2{}^{(1)}$		$E_3{}^{(0)}$	$\cdots$
	$Q_1{}^{(3)}$		$Q_2{}^{(2)}$		$Q_3{}^{(1)}$		$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

where the $Q_i{}^{(j)}$ are the spectral factors of $A(\lambda)$. In addition, note that the $Q.D.$ algorithm gives all spectral factors simultaneously and in the dominance order. We have chosen, in the above scheme, the row generation algorithm because it is more stable numerically. For further information about the row generation algorithm and the column generation algorithm we may refer the reader to see Dahimene (2009).

EXAMPLE 1 Consider a matrix polynomial of 2^{nd} order and 3^{rd} degree with the following matrix coefficients.

$$A_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad A_1 = \begin{pmatrix} -27.152538 & 0.8166050 \\ -179.782629 & 38.152538 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} 116.387033 & 84.978971 \\ 1043.444653 & 836.739866 \end{pmatrix}, \qquad A_3 = \begin{pmatrix} 126.928789 & 335.502350 \\ 1038.682417 & 2947.561338 \end{pmatrix}.$$

We apply now the generalized row generation $Q.D.$ algorithm to find the complete set of spectral factors and then we use the similarity transformations given by Shieh to obtain the complete set of solvents both right and left.

Step 1 initialization of the program to start:
Enter the degree and the order $m = 2, l = 3$
Enter the number of iterations $N = 35$
Enter the matrix polynomial coefficients A_i

Step 2 Construct Q_1 and E_1, the first row of $Q's$ and the first row of $E's$
$Q_1 = [-A_1A_0^{-1} \;\; O_2 \;\; O_2], \;\; E_1 = [O_2 \;\; A_2A_1^{-1} \;\; A_3A_2^{-1} \;\; O_2]$

Step 3 Building or generating the rest of rows using the rhombus rules

For $n = 1 : N$
$E_2 = 0; Q_2 = 0;$
  For $k = 1 : 2 : m \star l$
$q_2 = (E_1((:; k + 2 : k + 3) - E_1(:; k : k + 1)) + Q_1(:; k : k + 1);$
    $Q_2 = [Q_2; q_2];$
  End
$Q_2;$
  For $k = 1 : 2 : m \star l - 2$
    $e_2 = (Q_2(:; k + 2 : k + 3))(E_1(:; k : k + 1))(Q_2(:; (k : k + 1)))^{-1};$
    $E_2 = [E_2; e_2];$
  End
$E_2 = [O_2; E_2; O_2];$
$Q_1 = Q_2;$
$E_1 = E_2;$
End
$Q_1;$
$S_1 = Q_1(:; 1 : 2) S_2 = Q_1(:; 3 : 4) S_3 = Q_1(:; 5 : 6)$

When we run the above scheme, then we obtain the following complete set of spectral factors S_i:

$$Q_1 = \begin{pmatrix} 3.0000 & 2.0000 & -8.2908 & 0.7118 & 32.4434 & -3.5284 \\ -90.000 & -15.000 & -16.8400 & 8.1248 & 286.6226 & -31.2773 \end{pmatrix}$$

$$S_1 = \begin{pmatrix} 3.0 & 2.0 \\ -90.0 & -15.0 \end{pmatrix}, S_2 = \begin{pmatrix} -8.2908 & 0.7118 \\ -16.8400 & 8.1248 \end{pmatrix},$$

$$S_3 = \begin{pmatrix} 32.4434 & -3.5284 \\ 286.6226 & -31.2773 \end{pmatrix}.$$

Now, we should extract a complete set of right solvents from those block spectra using the algorithmic similarity transformations in equations from (21) to (24).

Step 1 Reversing the orientation of spectral factors (spectral factors are noted by S_i)

$$U_1 = S_3; \quad U_2 = S_2; \quad U_3 = S_1.$$

Step 2 Determination of coefficients using the synthetic long division and then finding the corresponding transformation matrix as in Theorem 7.

$$N_{11} = A_1 + U_1;$$
$$N_{12} = A_2 + U_1 \star N_{11};$$
$$G_1 = (U_1^2)^T \bigotimes I_2 + (U_1)^T \bigotimes N_{11} + I_2 \bigotimes N_{12};$$
$$vecp_1 = G_1^{-1} \star [1; 0; 0; 1]$$
$$p_1 = [vecp_1(1:2), vecp_1(3:4)];$$
$$R_1 = p_1 \star U_1 \star (p_1)^{-1}.$$

One can verify the first solvent using:
rightzero1= $A_0 \star (R_1)^3 + A_1 \star (R_1)^2 + A_2 \star R_1 + A_3$.

Step 3 Redo the same process for the next right solvents

$$N_{21} = N_{11} + U_2;$$
$$G_2 = (U_2)^T \bigotimes I_2 + I_2 \bigotimes N_{21};$$
$$vec\, p_2 = G_2^{-1} \star [1; 0; 0; 1]$$
$$p_2 = [vec\, p_2(1:2), vec\, p_2(3:4)];$$
$$R_2 = p_2 \star U_2 \star (p_2)^{-1}.$$

For verification one can also use:
rightzero 2= $A_0 \star (R_2)^3 + A_1 \star (R_2)^2 + A_2 \star R_2 + A_3$.

Step 4 Regarding the last solvents we obtain them directly from the most left spectral factor: $R_3 = S_1$ or we use the defined transformation:

$$G_3 = (I_2)^T \bigotimes I_2;$$
$$vecp_3 = G_3^{-1} \star [1; 0; 0; 1]$$
$$p_3 = [vecp_3(1:2), vecp_3(3:4)];$$
$$R_3 = p_3 \star U_3 \star p_3^{-1} = U_3.$$

The result of this procedure so is as follows:

$$R_1 = \begin{pmatrix} 0.36366 & -4.5495 \\ -0.81832 & 0.80238 \end{pmatrix}, \quad R_2 = \begin{pmatrix} 7.2354 & 1.4024 \\ 1.2995 & -7.4015 \end{pmatrix},$$

$$R_3 = \begin{pmatrix} 3.0000 & 2.0000 \\ -90.000 & -15.000 \end{pmatrix}.$$

Finally, we can also obtain the corresponding complete set of left solvents using the algorithmic similarity transformation, described in equations from (10) to (12).

Step 1 Determination of coefficients using the synthetic long division

$$B_{0i} = I_2;$$
$$B_{1i} = A_1 + R_i;$$
$$B_{2i} = A_2 + B_{1i} \star R_i.$$

Step 2 Finding the corresponding similarity transformation matrix as in equations (10) to (12).

For $i = 1 : l$
$vecQ_i = (B_{0i}^T \otimes (R_i)^2 + (B_{1i})^T \otimes R_i + (B_{2i})^T \otimes I_2)^{-1} \star [1; 0; 0; 1];$
$Q_i = [vecQ_i(1:2), vecQ_i(3; 4];$
$l_i = (Q_i)^{-1} \star R_i \star Q_i.$

One can verify the left solvents using:
Leftzero $= L_i^3 \star A_0 + L_i^2 \star A_1 + L_i \star A_2 + A_3$
End

The left solvents are now obtained:

$$L_1 = \begin{pmatrix} 32.443 & -3.5284 \\ 286.622 & -31.2773 \end{pmatrix}, \qquad L_2 = \begin{pmatrix} 25.1323 & -2.8370 \\ 204.5931 & -25.2983 \end{pmatrix},$$

$$L_3 = \begin{pmatrix} 21.0123 & -4.6531 \\ 178.0910 & -33.0123 \end{pmatrix}.$$

3.2. Extended Horner algorithm

Horner's method is a technique for evaluating a polynomials quickly. It requires l multiplications and l additions to evaluate a polynomial equation. It is also a nested algorithmic program that can decompose a polynomial into a multiplication of l linear factors, this scheme (Horner's method) being based on the Euclidean synthetic long division.

As a division algorithm, Horner's method is a nesting technique, requiring only l multiplications and l additions to evaluate an arbitrary l^{th}-degree polynomial, which can be surveyed by Horner's theorem (Burden and Faires, 2005).

Theorem 9 *Let the function $P(x)$ be the polynomial of degree l, defined on the real field $P : R \to R$, where: a_i are constant coefficients and x is real variable.*

$$P(x) = a_0 x^l + a_1 x^{l-1} + ... + a_{l-1} x + a_l. \tag{39}$$

If $b_0 = a_0$ and $b_k = a_k + b_{k-1}\alpha, \quad k = l, ..., 2, 1,$
then $b_l = P(\alpha)$ and $P(x)$ can be written as:

$$P(x) = (x - \alpha)Q(x) + b_l, \tag{40}$$

where:

$$Q(x) = b_0 x^{l-1} + b_1 x^{l-2} + ... + b_{l-2} x + b_{l-1}. \tag{41}$$

Proof The theorem can be proven using a direct calculation.

$$P(x) = a_0 x^l + a_1 x^{l-1} + ... + a_{l-1} x + a_l$$
$$P(x) = (x - \alpha)(b_0 x^{l-1} + b_1 x^{l-2} + ... + b_{l-2} x + b_{l-1}) + b_l.$$

Identifying the coefficients of x with different powers we get:

$$b_0 = a_0$$
$$b_1 = a_1 + b_0\alpha$$
$$\vdots$$
$$b_k = a_k + b_{k-1}\alpha \text{ where } k = l, l-1, ..., 2.$$

Now, if α is a root of the polynomial $P(x)$, then b_l should be zero, and $a_l + b_{l-1}\alpha = 0$.

Hence, we may write

$$\alpha = -\left(\frac{a_l}{b_{l-1}} \right) \quad \text{or} \quad x_{k+1} = -\left(\frac{a_l}{b_{l-1,k}} \right) \quad k = 0, 1, ...$$

The algorithm of the Horner method in its recursive formula is then:

$$b_{i,k} = a_k + b_{i,k-1} x_k; \quad i = 1, ..., l \quad \text{and} \quad b_{0,k} = a_0.$$

Now, let us generalize this nested algorithm to matrix polynomials, consider the monic λ-matrix $A(\lambda)$ and according to Theorem 1 the matrix $A(\lambda)$ can be factored as:

$$A(\lambda) = Q(\lambda)(\lambda I - X) + A_R(X). \tag{42}$$

Here,

$$A(\lambda) = A_0\lambda^l + A_1\lambda^{l-1} + ... + A_l = \sum_{i=0}^{l} A_i\lambda^{l-i}$$

$$Q(\lambda) = B_0\lambda^{l-1} + B_1\lambda^{l-2} + ... + B_{l-1} = \sum_{i=0}^{l-1} B_i\lambda^{l-i-1}$$

$$A_R(X) = \text{constant}.$$

By using the algorithm of synthetic long division for matrices we get:

$$B_0 = A_0 = I_m$$
$$B_1 = B_0A_1 + B_0X$$
$$\cdots$$
$$B_k = B_0A_k + B_{k-1}X \quad k = 1, 2, ..., l-1$$
$$O_m = B_0A_l + B_{l-1}X.$$

On the basis of the last two equations we can iterate the process to get the recursive algorithm as follows:

Enter the number of iterations N
For $k = 0 : N$
Enter the degree and the order $m; l$
Enter the matrix polynomial coefficients A_i
$X_0 =$ initial guess;
    For $i = 1 : l$
        $B_{i,k} = B_0A_i + B_{i-1,k}X_k$;
    End
$X_{k+1} = -(B_{(l-1),k})^{-1}B_0A_l$;
$X_k = X_{k+1}$;
End

When you get the first spectral factor, repeat the process until you get the complete set.

Example 2 Consider a matrix polynomial of 2nd order and 3rd degree with the following matrix coefficients.

$$A(\lambda) = A_0\lambda^3 + A_1\lambda^2 + A_2\lambda + A_3$$

with

$$A_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad A_1 = \begin{pmatrix} 11.0000 & -1.0000 \\ 6.7196 & 17.0000 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} 30.0000 & -11.0000 \\ 70.9107 & 82.5304 \end{pmatrix}, \quad A_3 = \begin{pmatrix} -0.0000 & -30.0000 \\ 182.0000 & 89.8393 \end{pmatrix}.$$

We apply now the extended Horner's method via its algorithmic version to find the complete set of spectral factors and then we use the similarity transformations, given by Shieh and Tsay (1981) to obtain the complete set of solvents, both right and left.

To make the procedure clear and more understandable, let us first give the block Horner scheme in its algorithmic version:

Assume $X_0 = O_2$ as an initial guess
Initial starting value of iterations $k = 0$ and $(X_0 = O_2)$
η : tolerance error
Given ε where $\varepsilon \geq \eta$
While $\varepsilon \geq \eta$

$$A_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$A_1 = \begin{pmatrix} 11.0000 & -1.0000 \\ 6.7196 & 17.0000 \end{pmatrix} \qquad B_0 = A_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} 30.0000 & -11.0000 \\ 70.9107 & 82.5304 \end{pmatrix}, \quad \begin{cases} B_1(k) = B_0A_1 + B_0X_k \\ B_1(0) = \begin{pmatrix} 11.0000 & -1.0000 \\ 6.7196 & 17.0000 \end{pmatrix} \end{cases}$$

$$A_3 = \begin{pmatrix} -0.0000 & -30.0000 \\ 182.0000 & 89.8393 \end{pmatrix}, \quad \begin{cases} B_2(k) = B_0A_2 + B_1(k)X_k \\ B_2(0) = \begin{pmatrix} 30.0000 & -11.0000 \\ 70.9107 & 82.5304 \end{pmatrix} \end{cases}$$

$$X_{k+1} = -(B_2(k))^{-1}B_0A_3$$

$$\varepsilon = 100.\frac{||X_{k+1} - X_k||}{||X - k||}$$

$$X_k = X_{k+1}$$

$$k = k + 1$$

End

Start the procedure and let it running, and in effect we obtain the following complete set of spectral factors:

$$S_1 = \begin{pmatrix} 0.00 & 1.00 \\ -3.25 & 2.00 \end{pmatrix}, \qquad S_2 = \begin{pmatrix} -5.000 & 0.000 \\ -1.6042 & -7.000 \end{pmatrix}, \qquad S_3 = \begin{pmatrix} -6.000 & -0.000 \\ -1.8655 & 8.000 \end{pmatrix}.$$

Finally, when we apply the similarity transformation algorithm, as in equations from (21) to (24), to right (or left) solvent form, we get:

$$R_1 = \begin{pmatrix} -5.9574 & 0.2553 \\ -0.3404 & -8.0426 \end{pmatrix}, \qquad R_2 = \begin{pmatrix} -4.9412 & 0.2941 \\ -0.4118 & -7.0588 \end{pmatrix}, R_3 = \begin{pmatrix} 0.000 & 1 \\ -3.25 & -2 \end{pmatrix}.$$

3.2.1. Reformulation of the block Horner method

Now, we will introduce a new perspective on the preceding procedure, which constitutes an efficient algorithm for the convergence study. Thus, with some reconfigurations and algebraic manipulation, we have:

$$
\begin{aligned}
&B_0(k) = A_0 = I_m \\
&B_1(k) = B_0(k)A_1 + B_0(k)X(k) \\
&... \\
&B_{l-1}(k) = B_0(k)A_{l-1} + B_{l-2}(k)X(k) \\
&O_m = B_0(k)A_l + B_{l-1}(k)X(k).
\end{aligned}
$$

After back substitution we get:

$$
\begin{aligned}
&B_{l-1}(k) = A_{l-1} + ... + A_1X^{l-1}(k) + B_0X^l(k) \\
&\Rightarrow B_{l-1}(k) = [A_R(X_k) - A_l]X_k^{-1} \\
&\Rightarrow (B_{l-1}(k))^{-1} = X_k[A_R(X_k) - A_l]^{-1}.
\end{aligned}
$$

Finally, we obtain the following iterative formula ($k = 0, 1, ...$):

$$
X_{k+1} = -(B_{l-1}(k))^{-1}A_l = X_k[A_l - A_R(X_k)]^{-1}A_l. \tag{43}
$$

Algorithm:

Enter the degree and the order, m, l
Enter the matrix polynomial coefficients $A_i \in R^{m\times m}$
$X_0 \in R^{m\times m}$ = initial guess;
Give some small η and (δ =initial start)$> \eta$
$k = 0$

While $\delta \geq \eta$

$$
\begin{aligned}
&X_{k+1} = X_k[A_l - A_RX_k]^{-1}A_l; \\
&\delta = 100.\frac{\|X_{k+1} - X_k\|}{\|X_k\|}; \\
&X_k \leftarrow X_{k+1}; \\
&k = k + 1;
\end{aligned}
$$

End

Convergence condition

Using the last iterated equation (43) that we have obtained and with the help of some norm properties we can establish under what conditions the algorithm will converge to the needed solution.

1. *Upper bound*

$$\text{eq.}(45) \Leftrightarrow X_{k+1} - X_k = X_{k+1}A_l^{-1}A_R(X_k)$$
$$\text{eq.}(45) \Leftrightarrow \|X_{k+1} - X_k\| = \|X_{k+1}A_l^{-1}A_R(X_k)\|$$
$$\text{eq.}(45) \Leftrightarrow \|X_{k+1} - X_k\| \leq \|X_{k+1}\|.\|A_l^{-1}\|.\|A_R X_k\|$$
$$\text{eq.}(45) \Leftrightarrow \frac{\|X_{k+1} - X_k\|}{\left(\|X_{k+1}\|.\|A_l^{-1}\| \right)} \leq \|A_R(X_k)\|$$

Now, if X_k tends to a constant matrix $\|X_k\| \to M$ as $k \to \infty$ and $\|X_k^{-1}\| \to N$ with $\|A_l^{-1}\| = \gamma$ and $\|A_l\| = \delta$, then:

$$\lim_{k\to\infty} \|A_R(X_k)\| \geq \frac{\|X_(k+1) - X_k\|}{\left(\|X_(k+1)\|.\|A_l^{-1}\| \right)}$$
$$\Rightarrow \lim_{k\to\infty} \|A_R(X_k)\| \geq \frac{\xi_k}{\gamma.M}. \tag{44}$$

2. *Lower bound*

$$A_R(X_k) = A_l[I - X_{k+1}^{-1}X_k] = A_l(X_{k+1})^{-1}[X_{k+1} - X_k]$$
$$\Rightarrow \|A_R(X_k)\| \leq \|A_l\|.\|(X_{k+1})^{-1}\|.\|X_{k+1} - X_k\|$$
$$\Rightarrow \lim_{k\to\infty} \|A_R(X_k)\| \leq \delta.N\xi_k. \tag{45}$$

From the above results (44) and (45) we can deduce that:

$$\frac{\xi_k}{\gamma.M} \leq \lim_{k\to\infty} \|A_R(X_k)\| \leq \delta.N\xi_k \tag{46}$$

Finally, if the matrix X_k tends to a constant matrix $X_k \to S$ and $(A_l - A_R(X_k))$ is a nonsingular matrix, then S is a solvent of the matrix polynomial $A_R(S) = O_m$.

Convergence type

In order to deduce the convergence type we should get a ratio relationship between any two successive differences

$$X_{k+1} - S = X_k([A_l - A_R(X_k)]^{-1}A_l - I) + X_k - S. \tag{47}$$

Let us define $F(X_k) = ([A_l - A_R(X_k)]^{-1}A_l - I)$, then we have:

$$\|X_k - S\| - \|X_k F(X_k)\| \leq \|X_{k+1} - S\| \leq \|X_k - S\| + \|X_k F(X_k)\|. \tag{48}$$

We know that:

$$\lim_{k\to\infty} \|X_k F(X_k)\| = \lim_{k\to\infty} \Delta_k = \xi, \tag{49}$$

and from equations (50) and (51) we deduce that:

$$1 - \frac{\Delta_k}{\|X_k - S\|} \leq \frac{\|X_{k+1} - S\|}{\|X_k - S\|} \leq 1 + \frac{\Delta_k}{\|X_k - S\|}.$$

Finally:

$$\lim_{k\to\infty} \frac{\|X_{k+1} - S\|}{\|X_k - S\|} = 1. \tag{50}$$

The block Horner method linearly converges or, more precisely, it is worse than linear, but, on the other hand, it does not depend so much on the initial conditions (knowledge of the starting point).

EXAMPLE 3 *Consider the following matrix polynomial with repeated spectral factor:*

$$A(\lambda) = \left[\lambda I - \begin{pmatrix} -7.1230 & -6.3246 \\ 5.9279 & 5.1230 \end{pmatrix} \right]^2 = A_0\lambda^2 + A_1\lambda + A_2,$$

with

$$A_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad A_1 = \begin{pmatrix} 14.2461 & 12.6493 \\ -11.8557 & -10.2461 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} 13.2461 & 12.6493 \\ -11.8557 & -11.2461 \end{pmatrix}.$$

Find X such that $A_R(X) = O_2$

$$A_R(X) = A_0X^2 + A_1X + A_2.$$

If we apply the block Horner algorithm, we find

$$X_1 = \begin{pmatrix} -2.7323 & -1.8068 \\ 1.6798 & 0.7521 \end{pmatrix}, \qquad X_2 = \begin{pmatrix} -11.5138 & -10.8424 \\ 10.1759 & 9.4939 \end{pmatrix}.$$

REMARK 3 *The proposed Horner algorithm finds the whole set of spectral factors if it exists, and does this even if there is no dominance in between them.*

3.2.2. Crossbred Newton–Horner method

In order to accelerate the block Horner method we make a crossbred (hybrid) with generalized Newton algorithm, which is very fast due to its restricted local nature (i.e. quadratic convergence). Hence, the obtained algorithm will benefit from the advantages of both of them, being faster in execution and featuring wide–range starting point.

Horner iteration	**Newton iteration**
$X_{k+1} - X_k = X_{k+1}A_l^{-1}A_R(X_k),$	$X_{k+1} - X_k = -J^{-1}(X_k)A_R(X_k)$

By combining them, we get:

$$X_{(k+1} = X_k + (X_k - J^{-1}(X_k)A_R(X_k))A_l{}^{-1}A_R(X_k), \tag{51}$$

where $J(X_k)$ is the Frechet differential.

Definition 7 *Let B_1 and B_2 be Banach spaces and A_R a nonlinear operator from B_1 to B_2. If there exists a linear operator L from B_1 to B_2 such that:*

$$B_1 \to B_2$$
$$H \to L(X+H)$$

where:

$$\|A_R(X+H) - A_R(X) - L(X+H)\| = O(\|H\|)$$
$$X, H \in B_1 \qquad A_R(X), L(X+H) \in B_2$$

then $L(X+H)$ is called the Frechet derivative of A_R at X and sometimes is written $dA_R(X,H)$. It is also read the Frechet derivative of A_R at X in the direction H, and $J(X_k).H = L(X+H)$.

Algorithm:

Enter the degree and the order, m, l
Enter the matrix polynomial coefficients $A_i \in R^{m\times m}$
$X_0 \in R^{m\times m}$ = initial guess;
Give initial $J(X_0) \in R^{m\times m}$
Give some small η and (δ = initial start) $> \eta$
$k = 0$
While $\delta \geq \eta$

$$X_{k+1} = X_k(I_m + A_l^{-1}A_r(X_k)) - J^{-1}(X_k)A_R(X_K)A_l^{-1}A_R(X_k);$$
$$H_k = X_{k+1} - X_k;$$
$$J(X_k) = (A_R(X_{k+1}) - A_R(X_k))H_k^{-1};$$
$$\delta = 100.\frac{||X_{k+1} - X_k||}{||X_k||};$$
$$X_k \leftarrow X_{k+1};$$
$$k = k+1;$$

End

3.2.3. Two stage block Horner algorithm

To accelerate the block Horner algorithm we use now a two stage and/or the Newton like iteration. Now, by using Theorem 1 we get the following equation:

$$A_R(X) = (X-\Theta)(X^{l-1} + B_1X^{l-2} + ... + B_{l-1}) + B_l$$
$$\Rightarrow A_R(X) = (X-\Theta)Q(X) + B_l$$
$$\Rightarrow X - \Theta = (A_R(X) - B_l)Q^{-1}(X),$$

where $Q(X) = X^{l-1} + B_1X^{l-2} + ... + B_{l-1}$ and $A_R(\Theta) = B_l$. Now, if Θ is a solvent, then $B_l = O_m$. If we now assume that $\Theta = X_{k+1}$ is a solvent to the matrix

polynomial A_R, then $A_R(X_{k+1}) = O_m$ and $X_{k+1} = X_k - A_R(X_k)Q^{-1}(X_{k+1})$. Set also $Q(X) = (X - \Theta)(X^{l-1} + C_1X^{l-2} + ... + C_{l-2}) + C_{l-1}$. On the basis of the Horner scheme we can evaluate both B_i and C_i recursively:

$B_0 = I_m$	$C_0 = I_m$
$B_1 = A_1 + B_0X$	$C_1 = B_1 + C_0X$
$B_2 = A_2 + B_1X$	$C_2 = B_2 + C_1X$
...	...
$B_{l-1} = A_{l-1} + B_{l-2}X$	...
$O_m = B_l = A_l + B_{l-1}X = A_R(X)$	$C_{l-1} = B_{l-1} + C_{l-2}X = Q(X)$
After iterating the last equation we get:	**After iterating the last equation we get**:
$B_l(k) = A_l + B_{l-1}(k)X_k$	$C_{l-1}(k) = B_{l-1}(k) + C_{l-2}(k)X_k$
$B_l(k) = A_R(X_k)$	$C_{l-1}(k) = Q(X_k)$

Algorithm:

X_0 =initial guess, $(B_0 = C_0 = I) \in R^{m\times m}$
Enter small enough number η (tolerance error)
$(\delta =$ initial start $) > \eta$
Enter the set of $m \times m$ $(A_0; A_1, ..., A_l)$ matrices
$k = 0$
While $\delta > \eta$
  For i $= 1 : 1 : l$
    $B_i(k) = A_i + B_{i-1}(k)X_k$
  End
  For $i = 1 : 1 : l - 1$
    $C_i(k) = B_i(k) + C_{i-1}(k)X_k$
  End
  $X_{k+1} = X_k - B_l(k)(C_{l-1}(k))^{-1}$
  $\delta = 100.\frac{||X_{k+1}-X_k||}{||X_k||}$;
  $X_k \leftarrow X_{k+1}$;
  $k = k + 1$;
End

Remark 4 *This two–stage algorithm makes jointly use of the two advantages, offered by the Horner scheme and Newton algorithm, because it is nested by its very nature, largely independent of the initial conditions and faster in execution, due to the similarity or the conformity to Newton method.*

Example 4 *We are given the following matrix polynomial*

$$A_R(X) = A_0X^3 + A_1X^2 + A_2X + A_3,$$

where:

$$A_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad A_1 = \begin{pmatrix} 12.8793 & -0.4881 \\ -2.0989 & 15.1207 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} 56.5645 & -8.7887 \\ 10.2686 & 55.9659 \end{pmatrix}, \quad A_3 = \begin{pmatrix} 95.9331 & -37.5549 \\ 160.9539 & -6.0938 \end{pmatrix}.$$

We apply the two stage Horner algorithm, and after 15 iterations we get:

$$X_0 = \begin{pmatrix} 5.2114 & 4.8890 \\ 2.3159 & 6.2406 \end{pmatrix}, \quad X_1 = \begin{pmatrix} -3.0729 & 1.4058 \\ -4.6569 & 1.0730 \end{pmatrix},$$

with

$$A_R(X_{15}) = \begin{pmatrix} -0.0081 & 0.0106 \\ 0.0265 & 0.0145 \end{pmatrix}.$$

3.2.4. Reformulation of the two stage block Horner method

After back substitution of the nested programmed scheme and accumulation we obtain:

$$B_l(k) = A_R(X_k) \quad \text{and}$$
$$C_{l-1}(k) = l{X_k}^{l-1} + l - 1A_1{X_k}^{l-2} + ... + A_{l-1} = \Delta(X_k)\ .$$

A variant of the two–stage block Horner algorithm can be obtained when we use the compact forms of the matrices $B_l(k)$ and $C_{l-1}(k)$ in terms of A_R coefficients, which leads to Newton–like iterative process.

Algorithm:

$X_0 =$ initial guess
Enter small enough number η (tolerance error) and ($\delta =$ initial start) $> \eta$
Enter the set of $m \times m(A_0; A_1, ..., A_l)$ matrices
$k = 0$
For $i = 0 : 1 : l - 1$
    $\Delta_i = (l - i)A_i$
End
While $\delta > \eta$

$$A_R(X_k) = A_0X_k^l + A_1X_k^{l-1} + ... + A_l$$
$$\Delta(X_k) = \Delta_0X_k^{l-1} + \Delta_1X_k^{l-2} + ... + \Delta_{l-1}$$
$$X_{k+1} = X_k - A_R(X_k)(\Delta(X_k))^{-1}$$
$$\delta = 100.\frac{||X_{k+1} - X_k||_2}{||X_k||_2}$$
$$X_k \leftarrow X_{k+1}$$
$$k \leftarrow k + 1$$

End

3.3. Combined algorithm and refinement

The proposed hybrid or two–stage block-Horner's algorithm converges rapidly and performs a recursive iteration, which is more efficient for digital software programing implementation. On the other hand, without any prior knowledge of the eigenvalues and eigenvectors of the matrix polynomial, the Horner's algorithm has been successfully utilized for finding the solvents, but this method depends upon the initial guess to a limited degree (the initial value of X_k is randomly chosen). Hence, in some cases it turns out to be very hard or impossible to accommodate or somehow restrict the initial gauss. For this reason, in order to avoid the strong dependence on the initial guess we proposed to initiate the whole program by the $Q.D.$ algorithm, which is numerically more stable and its initial starting values are well defined and/or exactly determined.

The complete procedure starts with the $Q.D.$ algorithm. It is then followed by a refinement of the right factor by the Horner's algorithm. After deflation of the $Q.D.$ program, Horner's algorithm is again applied using the next Q output from the $Q.D.$ algorithm and the process is repeated until we are left with a linear term. Of course, this process can be applied only to polynomial matrices that satisfy the conditions of the theorem (i.e. complete right and left factorization and complete dominance relation between solvents).

REMARK 5 *The same results are obtained as those that we have seen with the preceding algorithms, but the mixed refined algorithm goes faster in execution, with well defined initial conditions, and good convergence in much less time.*

3.4. Comments

- For the $Q.D.$ algorithm we have made the implicit assumption that an LR factorization exists at each step. If such factorization cannot be performed, this will lead to a breakdown of the algorithm.

- A numerical method for solving a given problem is said to be local if it is based on a local (simpler) model of the problem around the solution. From this definition, we can see that in order to use a local method, one has to provide an initial approximation of the solution. This initial approximation can be provided by a global method. As shown in Dahimene (2009), local methods are converging fast, while the global ones are quite slow. This implies that a good strategy is to start solving the problem by using a global method and then refine the solution by a local method.

- Of the advantages of the $Q.D.$ algorithm the essential ones are: the outputting of the complete set of spectral factors at a time, well defined starting point, determination of the complete set of spectral factors without the need to know the spectrum of matrix polynomial and the global nature. There is only one inconvenience, which is quite slow convergence.

- The Horner's algorithm brings in the local iterative nature, faster convergence, nested programmable scheme, without the need of any prior knowledge of the matrix polynomial, with only one inconvenience, which is the strong dependence on the initial guess.

- Many research studies have been done on the spectral decomposition for matrix polynomials to achieve complete factorization and reconstruction of the block roots using a large variety of algebraic and geometric numerical approaches, but (to our knowledge) nothing has been done for block-Horner's algorithm and/or block-$Q.D.$ algorithm.

4. Application in control system design

4.1. Decoupling controller design

The dynamic modeling of physical linear time invariant multi–input–multi–output systems results in high degree coupled vector differential equations with matrix constant coefficients or a matrix transfer function, where, in this case, the relationship between the input and output is a ratio of two matrix polynomials, expressed as a right (or left) matrix fraction description (RMFD or LMFD):

$$\begin{cases} H(\lambda) = N_R(\lambda){D_R}^{-1}(\lambda) \\ \qquad\quad = {D_L}^{-1}(\lambda)N_L(\lambda). \end{cases} \tag{52}$$

where: N_R, D_R, N_L and D_L are matrix polynomials and λ stands for the $\left(\dfrac{d}{dt}\right)$ operator. This fact has led to an active research effort in matrix polynomials theory, see also Yaici and Hariche (2014a,b), Bekhiti et al. (2015).

Idea: our objective here is to decouple the MIMO dynamic systems. Let us first factorize the numerator of the matrix polynomial $N(\lambda)$ into a complete set of spectral factors using one of the very well-known algorithms, then we place those found block zeros by forcing the denominator to have exactly those ones via state feedback control. Hence, the decoupling objectives are achieved.

Consider the square matrix transfer function:

$$\begin{aligned} H(\lambda) &= N(\lambda)D^{-1}(\lambda) = \left(\sum_{i=0}^{k} N_i\lambda^i \right) \left(\sum_{i=0}^{l} D_i\lambda^i \right)^{-1} \\ &= (N_k\lambda^k + ... + N_1\lambda + N_0)(D_l\lambda^l + ... + D_1\lambda + D_0)^{-1} \end{aligned}$$

where:
$D_l = I$ is an $m \times m$ identity matrix and
$N_i \in R^{m\times m}$, $(i = 0, 1, ..., k)$

$D_i \in R^{m\times m}$, $(i = 0, 1, ..., l)$, $l > k$.

Assume that $N(\lambda)$ can be factorized into k block zeros and $D(\lambda)$ can be factorized into l block roots (using one of the proposed algorithms):

$$N(\lambda) = N_k(\lambda I - Z_1)...(\lambda I - Z_k) \tag{53}$$

and

$$D(\lambda) = (\lambda I - Q_1)...(\lambda I - Q_l). \tag{54}$$

We also know that the matrix transfer function can be written in terms of state space matrices: $H(\lambda) = C(\lambda I - A)^{-1}B$. Now, via the use of state feedback the control law becomes state dependent and can be rewritten as

$$u(t) = -K.X(t) + F.r(t).$$

Hence, we obtain the following closed loop system:

$$(H(\lambda))_{closed} = C(\lambda I - A + BK)^{-1}BF = N(\lambda)D_d^{-1}(\lambda)F,$$

where: $D_d(\lambda) = (\lambda I - Q_{d1})...(\lambda I - Q_{dl})$ and Q_{di} are the desired spectral factors to be placed,

$$\begin{aligned} H(\lambda)_{closed} &= N(\lambda)D_d^{-1}(\lambda)F \\ H(\lambda)_{closed} &= N_k(\lambda I - Z_1)...(\lambda I - Z_k)(\lambda I - Q_{dl})^{-1}...(\lambda I - Q_{d1})^{-1}F \end{aligned} \tag{55}$$

where we choose:

$Q_{d1} = N_k J_1 N_k^{-1}, ..., Q_{d(l-k)} = N_k J_{(l-k)} N_k^{-1}$
$Q_{d(l-k+1)} = Z_1, ..., Q_{dl} = Z_k$
$J_i = diag(\lambda_{i1}, ..., \lambda_{im}), \quad F = (N_k)^{-1}$.

Now, by assigning those block roots, the system becomes decoupled and the closed loop matrix transfer function becomes:

$$H(\lambda)_{closed} = (\lambda I - J_1)^{-1}...(\lambda I - J_{l-k})^{-1}. \tag{56}$$

Let us summarize the preceding procedure in the following algorithmic version, meant to be more understandable and efficient for the use in linear multi-variable control systems.

Algorithm:

- Assume that all states are available and measurable.
- Check the Block Observability and Block Controllability of a given state space model of square dynamic system.

- Construct the right numerator and right denominator matrix polynomials using algorithms presented in Yaici and Hariche (2014).
- Decompose or factorize the numerator matrix polynomial into a complete set of block spectral factors using the proposed method.
- Choose the k spectral factors of numerator as block roots for the denominator and design the remaining ones in a diagonal form.
- Construct the desired matrix polynomial from those obtained block spectral data, see Yaici and Hariche (2014b).
- Design the state feedback gain matrix in the controller form and then transform it to the original base, see Shieh, Chang and McInnis (1986). Here at this point we are ready to design SISO tracking regulators for each of the input-output pairs, because the system is perfectly decoupled.

Example 5 *Find the decoupling gain matrix via block structure assignment for a turbogenerator system given by its state space representation ($n = 6, m = 2, p = 2$):*

$$\begin{cases} \dfrac{dX}{dt} = AX + Bu \\ Y = CX + Du \end{cases}$$

with

$$A = \begin{pmatrix} 8.2906 & 8.7757 & 5.8109 & 5.2892 & 4.4454 & 3.454 \\ 6.2659 & 0.1436 & 6.3715 & 6.9435 & 0.8540 & 9.4682 \\ 5.3875 & 2.9430 & 6.5127 & 2.1240 & 0.5734 & 5.2019 \\ 6.5051 & 1.7991 & 8.6462 & 5.4328 & 6.2945 & 9.5381 \\ 7.2663 & 9.2629 & 0.5595 & 7.0252 & 7.9618 & 0.7360 \\ 0.9449 & 0.6818 & 8.1686 & 9.5643 & 6.9119 & 2.0703 \end{pmatrix}, B = \begin{pmatrix} 3.8751 & 3.9243 \\ 4.5709 & 1.3542 \\ 3.9128 & 1.1391 \\ 1.4777 & 1.6051 \\ 0.7592 & 4.1478 \\ 4.2396 & 4.1109 \end{pmatrix}$$

$$C = \begin{pmatrix} 1.7120 & 0.8581 & 2.3888 & 1.3386 & 0.8371 & 2.7110 \\ 1.7155 & 2.0974 & 1.3248 & 1.3970 & 2.0261 & 2.7256 \end{pmatrix}, \qquad D = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$

$A \in R^{n\times n}$, $B \in R^{n\times m}$, $C \in R^{p\times n}$, $D \in R^{m\times m}$, $X \in R^{n\times 1}$ is the state vector, $u \in R^{m\times 1}$ is the input vector, and $Y \in R^{p\times 1}$ is the output vector.

The resulting dynamic model is a high degree coupled vector differential equation with matrix constant coefficients or a matrix transfer function (λ-description), given by:

$$H(\lambda) = C(\lambda I - A)^{-1}B + D = N(\lambda)D(\lambda)^{-1} = \begin{pmatrix} H_{11}(\lambda) & H_{12}(\lambda) \\ H_{21}(\lambda) & H_{22}(\lambda) \end{pmatrix},$$

where

$$
\begin{aligned}
H_{11}(\lambda) &= \left(\frac{34.01\lambda^5 - 116.9\lambda^4 - 221.2\lambda^3 - 102.5\lambda^2 - 1.955\times10^4\lambda - 2077}{\lambda^6 - 30.41\lambda^5 - 45.33\lambda^4 + 515.1\lambda^3 - 1343\lambda^2 + 1.805\times10^4\lambda + 9102}\right)\\
H_{21}(\lambda) &= \left(\frac{36.58\lambda^5 - 42.7\lambda^4 + 696\lambda^3 + 3295\lambda^2 - 208.4\lambda + 1.979\times10^4}{\lambda^6 - 30.41\lambda^5 - 45.33\lambda^4 + 515.1\lambda^3 - 1343\lambda^2 + 1.805\times10^4\lambda + 9102}\right)\\
H_{12}(\lambda) &= \left(\frac{27.37\lambda^5 - 53.9\lambda^4 + 12.19\lambda^3 - 767.4\lambda^2 - 7918\lambda - 8267}{\lambda^6 - 30.41\lambda^5 - 45.33\lambda^4 + 515.1\lambda^3 - 1343\lambda^2 + 1.805\times10^4\lambda + 9102}\right)\\
H_{22}(\lambda) &= \left(\frac{32.93\lambda^5 - 66.88\lambda^4 + 1.5\lambda^3 - 1474\lambda^2 - 8675\lambda - 1.675\times10^4}{\lambda^6 - 30.41\lambda^5 - 45.33\lambda^4 + 515.1\lambda^3 - 1343\lambda^2 + 1.805\times10^4\lambda + 9102}\right).
\end{aligned}
$$

In order to design the decoupling control of the turbogenerator system we should firstly ***decompose*** the numerator and denominator matrix polynomials $N(\lambda)$ and $D(\lambda)$ into a complete set of spectral factors and then check the governability and estimability of the plant. The Block Controllability and Block Observability matrices are defined by (see Yaici and Hariche, 2014a,b):

$$
W_c = \left(\ B, AB, ..., A^{l-1}B\ \right) \ \text{ and } \ W_o = \left(\ C^T \quad A^TC^T \quad \cdots \quad A^{q-1^T}C^T\ \right)^T
$$

$rank(W_c) = rank([B, \quad AB, \quad A^2B]) = 6 \quad \text{and} \quad l = \dfrac{n}{m} = 3 \Rightarrow$ block controllable system.

$rank(W_o) = rank([C^T, \quad C^TA^T, \quad C^TA^{2^T}]^T) = 6 \quad \text{and} \quad q = \dfrac{n}{p} = 3 \Rightarrow$ block observable system.

W_c and W_o are both of full rank and so the dynamic system is Block Controllable and Block Observable.

Now, we should construct the numerator and denominator matrix polynomials from the state space data as follows (see Kailath and Li, 1980; Kucera, 1979, and Solak, 1987):

$$
\left\{\begin{array}{lcll} D(\lambda) &=& D_3\lambda^3 + D_2\lambda^2 + D_1\lambda + D_0, & D_3 = I_2\\ N(\lambda) &=& N_3\lambda^3 + N_2\lambda^2 + N_1\lambda + N_0, & N_3 = O_2, \end{array}\right.
$$

where:

$$
\begin{pmatrix} D_0\\ D_1\\ D_2 \end{pmatrix} = -\left[\ B, AB, A^2B\ \right]^{-1} A^3B = \begin{pmatrix} -37.0170 & 28.2888\\ -223.8750 & -74.7887\\ 34.8029 & 20.9798\\ -280.2609 & -216.8345\\ -14.0378 & -7.5183\\ -12.3898 & -16.3740 \end{pmatrix}
$$

$$\begin{pmatrix} N_0^T \\ N_1^T \\ N_2^T \end{pmatrix}^T = \begin{bmatrix} CB, CAB, CA^2B \end{bmatrix} \begin{pmatrix} D_1 & D_2 & D_3 \\ D_2 & D_3 & O_2 \\ D_3 & O_2 & O_2 \end{pmatrix}$$
$$= \begin{pmatrix} 211.7886 & 61.4727 & 100.8960 & 74.4572 & 34.0105 & 27.3669 \\ 331.6250 & 199.1758 & 148.1818 & 120.4265 & 36.5764 & 32.9324 \end{pmatrix}.$$

Let us decompose the numerator and denominator matrix polynomials and reconstruct their block roots (using the proposed decomposition algorithms):

$$N(\lambda) = N_2(\lambda I - Z_1)(\lambda I - Z_2) \quad \text{and} \quad D(\lambda) = (\lambda I - Q_1)(\lambda I - Q_2)(\lambda I - Q_3).$$

The block spectral factors are approximately calculated with a residual normed tolerance error given by:

$$\varepsilon_i = \frac{\|{Z_i}^*\| - \|Z_i\|}{\|{Z_i}^*\|} \quad i = 1, 2, \quad \text{and} \quad \xi_i = \frac{\|{Q_i}^*\| - \|Q_i\|}{\|{Q_i}^*\|} \quad i = 1, 2, 3.$$

Remark 6 *The last block pole Q_3 can be constructed using the synthetic long division. The diagrams of Fig. 1 illustrate a comparison study between the proposed algorithms in term of the convergence speed and residual normed tolerance error.*

The numerator block zeros are computed using the very well-known numerical methods that can factorize matrix polynomial into a complete set of block roots. Using the refined block Horner's method, as illustrated in this paper, we get:

$$N(Z_i) = O_2 \Rightarrow Z_1 = \begin{pmatrix} 24.7235 & 23.1394 \\ -27.4494 & -24.9281 \end{pmatrix}, \quad Z_2 = \begin{pmatrix} -18.5711 & -16.0841 \\ 16.1166 & 13.4353 \end{pmatrix}.$$

The desired denominator is of third order, and is written down in the form:

$$D_d(\lambda) = D_{d3}\lambda^3 + D_{d2}\lambda^2 + D_{d1}\lambda + D_{d0}.$$

Using the prescribed decoupling algorithm we obtain:

$$F = {N_2}^{-1}, \quad J_1 = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}, \quad Q_{d1} = {N_2}^{-1} J_1 {N_2}^{-1}, \quad Q_{d2} = Z_1, \quad Q_{d3} = Z_2$$

$$D_d(\lambda) = (\lambda I - Q_{d1})(\lambda I - Q_{d2})(\lambda I - Q_{d3}) = I\lambda^3 + D_{d2}\lambda^2 + D_{d1}\lambda + D_{d0},$$

where:

$$D_{d2} = -(Q_{d1} + Q_{d2} + Q_{d3}) = \begin{pmatrix} -13.5596 & -14.6249 \\ 21.7809 & 21.8999 \end{pmatrix}$$

$$D_{d1} = (Q_{d1}Q_{d2} + Q_{d1}Q_{d3} + Q_{d2}Q_{d3}) = \begin{pmatrix} -126.4282 & -121.5061 \\ 161.6710 & 152.4741 \end{pmatrix}$$

$$D_{d0} = -(Q_{d1}Q_{d2}Q_{d3}) = \begin{pmatrix} -178.9732 & -164.0512 \\ 223.2851 & 202.6227 \end{pmatrix}.$$

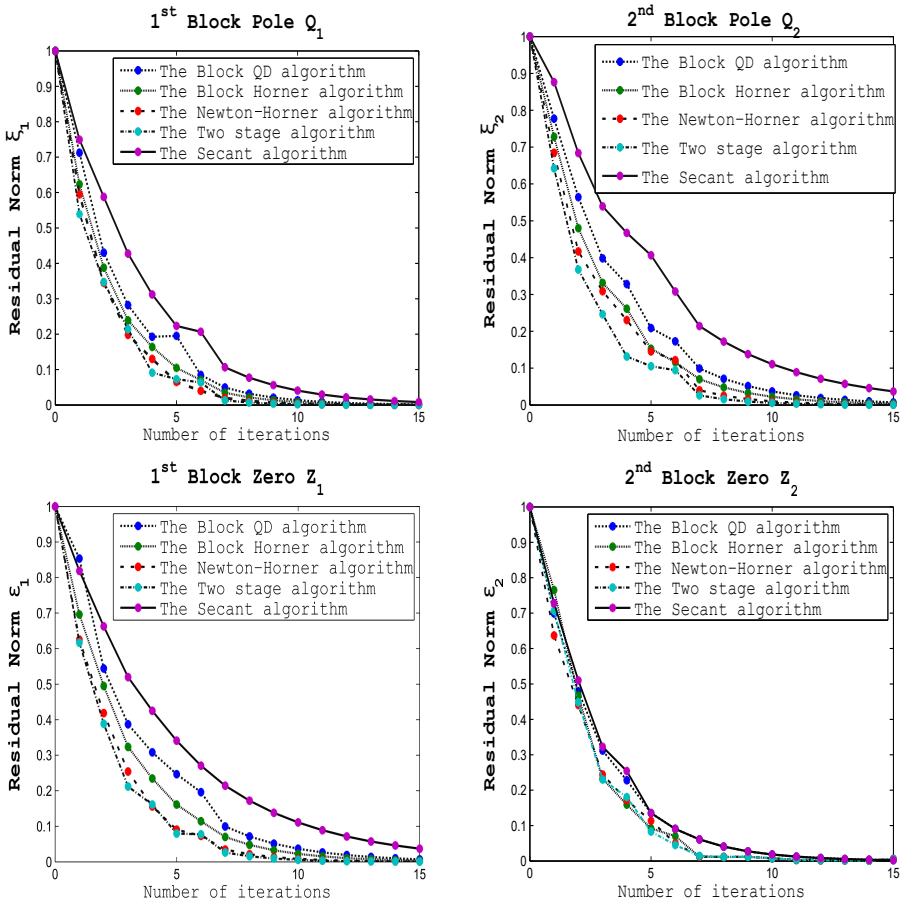

Figure 1. The residual error norm comparison study

The state feedback gain matrix of the block controller form is obtained by, see Tsay and Shieh (1983) and Bekhiti et al. (2015):

$$K_{ci} = D_{di} - D_i \quad \text{With} \quad i = 0, 1, 2 \qquad \text{and} \quad K_c = [K_{c0}, K_{c1}, K_{c2}].$$

Now, let us go back to the original base by the following similarity transformation:

$$K = K_c T_c \text{ and } T_c = \left(\begin{bmatrix} B, & AB, & A^2B \end{bmatrix} \begin{pmatrix} D_1 & D_2 & D_3 \\ D_2 & D_3 & O_2 \\ D_3 & O_2 & O_2 \end{pmatrix} \right)^{-1}$$

$$K = \begin{pmatrix} -0.7616 & -3.2690 & 3.5737 & -0.0716 & -2.3462 & 1.4801 \\ 3.0439 & 5.4047 & -2.3853 & 2.4117 & 4.2560 & 0.0494 \end{pmatrix}.$$

The new state space representation of the decoupled system after applying the state feedback is:

$$A_d = (A - BK), \quad B_d = BF, \quad \text{and} \quad C_d = C$$

$$\begin{aligned} H(\lambda)_{closed} &= C(\lambda I - A + BK)^{-1}BF \\ &= N(\lambda)D_d^{-1}(\lambda)F = \begin{pmatrix} \dfrac{1}{\lambda+1} & 0 \\ 0 & \dfrac{1}{\lambda+2} \end{pmatrix}. \end{aligned}$$

Now we can design a SISO PID controller for each input–output pair, using the known tuning methods, for example the Ziegler–Nichols method, or any other one.

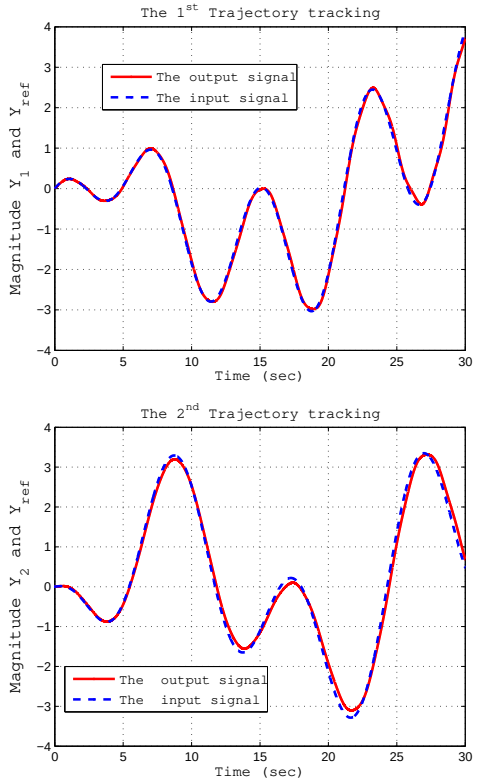

Figure 2. The trajectory tracking control of the decoupled system

From the obtained simulation results, as shown in Fig. 2, we see that the controlled plant tracks its reference trajectory with very small error, no overshooting, and no static error appears at both transient and steady state regimes, meaning that both tracking and regulation objectives are being attained by the

procedure. Finally, from the error signal, the BIBO stability is guaranteed, while the internal stability is not, and this is not surprising or new to us due to the cancellation phenomenon in the designed controller. Hence, from the simulation results we deduce that the block zeros are well computed and the numerator matrix polynomial $N(\lambda)$ is perfectly decomposed using the proposed procedure.

4.2. Suggestions for further research

The results obtained during this research work generated many questions and problems, whose potential solutions are to be explored:

- Devising other globalization techniques for the block-Horner's algorithm to avoid the local restriction and the problem of initial guess, arriving at a very fast global nested program. This involves also exploring and extending other scalar numerical methods to factorize matrix polynomials.

- Both of the block-Horner's algorithm, and the block-$Q.D.$ algorithm, as used in our work, converge to factors of a matrix polynomial. By using the defined similarity transformations, we can derive the solvents. However, it would be convenient to have a global algorithm that converges rapidly and directly to all solvents.

- If some E column in the $Q.D.$ tableau converges, this implies that there exists a factorization of the matrix polynomial that splits the spectrum into a dominant set and a dominated one. If the system under consideration is a discrete–time system, we know that the largest modulus latent roots have the preponderant effect on the dynamic properties of the system. In such case, the $Q.D.$ algorithm can become a tool for system reduction (using the dominant mode concept).

- The computational procedure for finding the solvents of a matrix polynomial with repeated block roots (solvents) and/or spectral factors needs to be investigated further.

5. Conclusion

In this paper we have introduced new numerical approaches for determining the complete sets of spectral factors and solvents of a monic matrix polynomial. For avoiding the initial guess we have proposed a systematic method for the block-Horner's algorithm via a refinement of the block-$Q.D.$ algorithm. At least three advantages are offered by the proposed technique: (i) an algorithm with global nature is obtained; hence there is no initial-guess problem during the whole procedure, (ii) high speed convergence to each solution is obtained and only a few iterations are required; (iii) via the help of refinement and direct cascading, the algorithms are easily coupled together and the whole scheme is suitable

for digital software processor and can easily be computerized. The obtained solvents can be considered as a useful tool for carrying out the block partial fraction expansion for the inverse of a matrix polynomial. Those partial fractions are, of course, matrix transfer functions of reduced order linear systems, such that the realization of them leads to block diagonal (block-decoupling) or parallel decomposed multivariable linear time invariant systems. For such systems analysis and design can be easily performed, as the dynamic properties of MIMO systems depend on their block pole of its characteristic matrix polynomial. Therefore, they can be used as tools for block-pole placement, block-system identification and block-model order reduction. In addition, the proposed method can be employed to carry out the block spectral factorization of a matrix polynomial for problems in optimal control, filtering and estimation.

Finally, for the purpose of relocating block roots of matrix polynomials via latent structure assignment, a new algorithm has been developed and the results will be presented very soon. Furthermore, another algorithm for block order reduction based on solvent and block moment matching is under realization.

References

Ahn, S. M. (1982) Stability of a Matrix Polynomial in Discrete Systems. *IEEE Trans. on Auto. Contr.*, **AC-27**, 1122-1124.

Barnett, S. (1971) *Matrices in Control Theory*, Van Nostrand Reinhold, New York.

Bekhiti, B., Dahimene, A., Nail, B., Hariche, K. and Hamadouche, A. (2015) On Block Roots of Matrix Polynomials Based MIMO Control System Design. 4^{th} *International Conference on Electrical Engineering ICEE Boumerdes.*

Burden, R. L. and Faires, J. D. (2005) *Numerical Analysis*, 8th Edition. Thomson Brooks/Cole, Belmont, CA, USA.

Chen, C. T. (1984) *Linear System Theory and Design.* Holt, Reinhart and Winston.

Dahimene, A. (2009) Incomplete matrix partial fraction expansion. *Control and Cybernetics*, **38**, 3.

DiStefano, J. J., Stubberud, A. R. and Williams, I.J. (1967) *Theory and Problems of Feedback and Control Systems.* Mc Graw Hill.

Denman, E. D. (1977) Matrix polynomials, roots, and spectral factors. *A &. Math. Comput.* **3**:359-368.

Denman, E. D. and Beavers, A. N. (1976) The matrix sign function and computations in systems. *Appl. Math. Comput.* **2**:63-94.

Dennis, J. E., Traub, J. F. and Weber, R. P. (1976) The algebraic theory of matrix polynomials. *J. Numer. Anal.* **13**:831-845.

Dennis, J. E., Traub, J. F. and Weber, R. P. (1978) Algorithms for solvents of matrix polynomials. *J. Numer. Anal.* **15**:523-533.

Gohberg, I., Kaashoek, M. A. and Rodman, L. (1978) Spectral analysis

of operator polynomial and a generalized Vandermonde matrix, 1. The finite-dimensional case. In: *Topics in Functional Analysis*. Academic Press, 91-128.

Gohberg, I., Lancaster P. and Rodman, L. (1982) *Matrix Polynomials*. Academic Press.

Hariche, K. (1987) Interpolation Theory in the Structural Analysis of λ-matrices. Chapter 3, Ph. D. Dissertation, University of Houston.

Hariche, K. and Denman, E. D. (1988) On Solvents and Lagrange Interpolating λ-Matrices. *Applied Mathematics and Computation* 25321–332.

Hariche, K. and Denman, E. D. (1989) Interpolation Theory and λ-Matrices. *Journal of Mathematical Analysis and Applications* **143**, 53 and 547.

Henrici, P. (1958) The Quotient-Difference Algorithm. *Nat. Bur. Standards Appl. Math. Series*, **49**, 23-46.

Higham, N. J. (2008) *Functions of Matrices: Theory and Computation*. SIAM.

Kailath, T. and Li, W. (1980) *Linear Systems*. Prentice Hall.

Kucera, V. (1979) *Discrete Linear Control: The Polynomial Equation Approach*. John Wiley.

Meyer, C. D. (2000) *Matrix Analysis and Applied Linear Algebra*. SIAM.

Parlett, B. N. (1967) The LU and QR Algorithms. In: A. Ralston and H. S. Wilf, *Mathematical Methods for Digital Computers*, **2**. John Wiley.

Pereira, E. (2003) On solvents of matrix polynomials. *Appl. Numer. Math.*, **47**, 197–208,

Pereira, E. (2003) Block eigenvalues and solution of differential matrix equation. *Mathematical Notes*, Miskolc, **4**, 1, 45–51.

Pathan, A. and Collyer, T. (2003) The wonder of Horner's method. *Mathematical Gazette*, **87**, 509, 230-242.

Resende, P. and Kaskurewicz, E. (1989) A Sufficient Condition for the Stability of Matrix Polynomials. *IEEE Trans. on. Auto. Contr.*, **AC-34**, 539-541.

Shieh, L. S. and Sacheti, S. (1978) A Matrix in the Block Schwarz Form and the Stability of Matrix Polynomials. *Int. J. Control*, 27, 245-259.

Shieh, L. S. and Chahin, N. (1981) A computer-aided method for the factorization of matrix polynomials. *Internat. Systems Sci.*, **12**:1303-1316.

Shieh, L. S., Tsay, Y. T. and Coleman, N. I. (1981) Algorithms for solvents and spectral factors of matrix polynomials. *Internat. J. Control*, **12**:1303-1316 .

Shieh, L. S. and Tsay, Y. T. (1981) Transformation of solvent and spectral factors of matrix polynomial, and their applications. *Internat. J. Control*, **34**:813-823.

Shieh, L. S. and Tsay, Y. T. (1982a) Transformation of a class of multivariable control systems to block companion forms. *IEEE Trans. Automat. Control*, **27**:199-203.

Shieh, L. S. and Tsay, Y. T. (1982b) Block modal matrices and their applications to multivariable control systems. *IEE Proc. D Control Theory Appl.*, **2**:41-48.

Shieh, L. S. and Tsay, Y. T. (1984) Algebraic-geometric approach for the modal reduction of large-scale multivariable systems. *IEE Proc. D Control Theory Appl.* **131**(1):23-26.

Shieh, L. S., Chang, F. R. and McInnis, B. C. (1986) The block partial fraction expansion of a matrix fraction description with repeated block poles. *IEEE Trans. Automut. Control.* **31**:23-36.

Shieh, L. S., Chang, F. R. and McInnis, B. C. (1986) The block partial fraction expansion of a matrix fraction description with repeated blocl poles. *IEEE Trans. Auto. Cont.* **AC-31**, 236-239.

Solak, M. K. (1987) Divisors of Polynomial Matrices: Theory and Applications. *IEEE Trans. on Auto. Contr.*, **AC-32**, 916-919.

Tsai, J. S. H., Shieh, I. S. and Shen, T. T. C. (1988) Block power method for computing solvents and spectral factors of matrix polynomials. *Internut. Computers and Math. Appl.*, **16**:683-699.

Tsay, Y. T. and Shieh, L. S. (1983) Block decompositions and block modal controls of multivariable control systems. *Automatica*, **19**, 1, 29-40.

Tsai, J. S. H., Chen C. M. and Shieh, L. S. (1992) A Computer-Aided Method for Solvents and Spectral Factors of Matrix Polynomials. *Applied Mathematics and Computation*, **47**:211-235.

Yaici, M., Hariche, K. and Clark, T. (2014) Contribution to the Polynomial Eigenvalue Problem. *International Journal of Mathematics, Computational, Natural and Physical Engineering*, **8**(10).

Yaici, M. and Hariche, K. (2014a) On eigenstructure assignment using block poles placement. *European Journal of Control*, **20**(5), 217–226.

Yaici, M. and Hariche, K. (2014b) On Solvents of Matrix Polynomials. *International Journal of Modeling and Optimization*, **4**, 4.

Zabot, H and Hariche, K. (1997) On solvents-based model reduction of MIMO systems. *International Journal of Systems Science*, **28**, 5, 499-505.