
ELSEVIER Linear Algebra and its Applications 357 (2002) 259–271

LINEAR ALGEBRA AND ITS APPLICATIONS

www.elsevier.com/locate/laa

# Inversion of polynomial matrices via state-space

J.C. Basilio [1]

*Departamento de Eletrotécnica, Universidade Federal do Rio de Janeiro, Escola de Engenharia, Cidade Universitária, Ilha do Fundão, 21.945-970, Rio de Janeiro, Brazil*

Received 30 August 2001; accepted 30 April 2002

Submitted by P. Lancester

**Abstract**

In this paper, the problem of computing inverses of polynomial matrices has been revisited and algorithms of easy implementation are proposed to deal with either column or non-column reduced matrices. Other contributions of the paper are algorithms to perform column reduction and determinant of polynomial matrices.


*Keywords:* Matrix polynomials; State-space methods; Polynomial matrix inversion; Matrix fraction description

## 1. Introduction

Polynomial matrices play an important role in mathematics [1,2] and also in control engineering [3–5]. Among the topics related to polynomial matrices, the computation of inverses has recently been of great interest [5–9]. Inouye [6] proposes an algorithm, which requires the polynomial matrix to be column-proper. [2] This deficiency has been removed by Buslowicz [7] without ensuring that the inverse numerator matrix is irreducible and the denominator polynomial is minimal. This constraint has been partially solved by Zhang [8], who addressed the case of column reduced polynomial matrices. The algorithms proposed by Inouye,

*E-mail address:* basilio@coep.ufrj.br (J.C. Basilio).

[1] This work has been supported in part by the Brazilian Research Council (CNPq) under Grant No. 520190/96.

[2] A matrix is said to be column-proper if the highest power coefficient matrix is full rank.

0024-3795/02/$ - see front matter 
PII: S0024-3795(02)00418-4

Buslowicz and Zhang have, in common, the fact that they are all based on recursive formulae, being suitable only for matrices of low dimensions. In a different way, Schuster and Hippe [9] compute polynomial matrix inverses by interpolation, but the efficiency of the proposed algorithm is very dependent on the interpolating points which are to be chosen. A more general algorithm for the computation of polynomial matrix inverse has been given by Stefanidis et al. [5], which is based on the inversion of a real matrix obtained from an appropriate Sylvester resultant matrix. Sylvester resultant matrices are generally of high order and this makes the algorithm proposed by Stefanidis et al. also suitable only for polynomial matrices of low dimensions.

In this paper we propose a general algorithm valid for both column and non-column reduced matrices. When the matrix is column reduced, a minimal state-space realization for the inverse can be obtained directly. Otherwise, an algorithm, which is a modification of a previous one [10], provides the means to obtain a column reduced matrix by post-multiplication of the original one by unimodular matrices. In that case, inversion is carried out by finding the inverse of the resulting reduced matrix and then by pre-multiplying the inverse by the unimodular matrix constructed in the reduction algorithm. In both cases, Leverrier's algorithm can be deployed to get a transfer function representation of the inverse. The whole algorithm is very simple to implement and this has been possible because well-known facts of the theory of polynomial matrix and state-space realizations have been put together. To the author's knowledge, this had not been done before.

This paper is structured as follows. Section 2 presents the problem of inverting a polynomial matrix and proposes three algorithms: Algorithm 1 proposes to find a state-space representation of the inverse of a column reduced matrix; Algorithm 2 proposes to carry out column reduction and Algorithm 3 proposes to find inverses of non-column reduced matrices. The results of the paper are illustrated by means of two numerical examples in Section 3. Finally, conclusions are drawn in Section 4.

## 2. The algorithm

### 2.1. Problem formulation

Let $\mathbb{R}^{m\times m}[s]$ and $\mathbb{R}^{m\times m}(s)$ denote, respectively, the rings of polynomial and rational matrices (not necessarily proper) of order $m$ and let $D(s) \in \mathbb{R}^{m\times m}[s]$. The problem of computing the inverse of $D(s)$ can be stated as follows: find a matrix $G(s) \in \mathbb{R}^{m\times m}(s)$ such that $G(s)D(s) = I_m$, where $I_m$ denotes the identity matrix of order $m$. Assuming that $D^{-1}(s)$ exists, $G(s)$ can be written as

$$G(s) = I_m D^{-1}(s). \tag{1}$$

When $D(s)$ is not a unimodular matrix, then the right-hand side of Eq. (1) represents a matrix fraction description (MFD)[3] of $G(s)$. Moreover, this MFD is irreducible, i.e., all the greatest common right divisors of $I$ and $D(s)$ are unimodular, as stated in the following proposition.

**Proposition 1.** *$G(s) = I_m D^{-1}(s)$ is an irreducible MFD of $G(s)$.*

**Proof.** It suffices to find two matrices $\tilde{X}(s), \tilde{Y}(s) \in \mathbb{R}^{m\times m}[s]$ satisfying the Bezout identity [4, p. 379]:

$$\tilde{X}(s)D(s) + \tilde{Y}(s)I = I. \tag{2}$$

It is immediate to see that $\tilde{X}(s) = I$ and $\tilde{Y}(s) = I - D(s)$ satisfy Eq. (2). □

**Remark A.** Although Proposition 1 seems to be very straightforward, the fact that $G(s) = I D^{-1}(s)$ is a right-coprime MFD has not appeared in the literature. Indeed, the derivation of the algorithm proposed by Stefanidis et al. [5] starts from the same manner as above, i.e., by writing the inverse of a polynomial matrix $D(s)$ as $G(s) = I D^{-1}(s)$, a right MFD. The authors, however, have not realized the coprime nature of this MFD. In the sequel, Stefanidis et al. write $G(s)$ as $G(s) = [d(s)I_n]^{-1}\text{Adj}[D(s)]$ (a left MFD), where $d(s)$ is the determinant of $D(s)$, and by inverting a real matrix obtained from an appropriate resultant matrix, $\text{Adj}[D(s)]$ is obtained. It is important to remark that the real matrix to be inverted in [5] has generally high dimensions, which makes the computation of $D^{-1}(s)$ numerically expensive.

With Proposition 1 in mind, a minimal order realization for $G(s)$ can be obtained according to the following result.

**Proposition 2.** *Any realization of order equal to the degree of the determinant of the denominator matrix of an MFD will be minimal if and only if the MFD is irreducible.*

**Proof.** See Kailath [4, p. 439]. □

Therefore, the problem of finding the inverse of a polynomial matrix $D(s)$ turns out to be the one of finding a minimal state-space realization for the irreducible MFD given in (1). When $D(s)$ is column reduced, such a realization can be obtained directly [4, p. 403], as shown in the following section.

[3] A MFD of a matrix $G(s) \in \mathbb{R}^{m\times m}(s)$ is a "ratio" of two matrices $N(s), M(s) \in \mathbb{R}^{m\times m}[s]$ such that $G(s) = N(s)M^{-1}(s)$.

### 2.2. A minimal realization for $G(s) = I_m D^{-1}(s)$ when $D(s)$ is column reduced

It is well known [4, p. 384] that any polynomial matrix $D(s)$ can be written as follows:

$$D(s) = D_{\text{hc}} \operatorname{diag}\{s^{\nu_i}, i = 1, 2, \ldots, m\} + D_{\text{lc}} \Psi(s), \tag{3}$$

where $\nu_i,\ i = 1, 2, \ldots, m$ are the column degrees of $D(s)$ (the degrees of the polynomials of largest degrees of each column of $D(s)$), $D_{\text{hc}}$ is the leading coefficient matrix (a matrix whose $i$th column is formed with the coefficients of $s^{\nu_i}$), $D_{\text{lc}}$ is a matrix formed with the coefficients of the remaining lower degree terms and $\Psi(s)$ has the following form:

$$\Psi^{\text{T}}(s) = \operatorname{diag}\left\{[s^{\nu_i - 1} \ \cdots \ s \ 1],\ i = 1, 2, \ldots, m\right\}. \tag{4}$$

It is not hard to check that when $D(s)$ is column reduced then $D_{\text{hc}}$ is non-singular and therefore the degree of the determinant polynomial of $D(s)$ will be given by:

$$\deg \det[D(s)] = \sum_{i=1}^{m} \nu_i. \tag{5}$$

This implies that, any state-space realization of (1) of order $n = \sum_{i=1}^{m} \nu_i$ will, according to Proposition 2, be minimal. A straightforward realization for (1) is the so-called controller realization [4, p. 403], which requires the following conditions to be met: (i) $D(s)$ column reduced, and (ii) $G(s)$ strictly proper. At this point it will be assumed that condition (i) is satisfied (if this is not the case, an algorithm to form a column reduced matrix equivalent to $D(s)$ will be proposed in Section 2.4). As far as condition (ii) is concerned, notice that if one or more columns of $D(s)$ have column degree equal to zero, then $G(s) = I D^{-1}(s)$ will not be strictly proper. When this happens, it is necessary first to post-multiply $D(s)$ by a diagonal matrix $\Delta(s)$ whose diagonal elements are either $s$ or 1, if the corresponding column degrees of $D(s)$ are 0 or greater than 0. For example, let $D(s) \in \mathbb{R}^{2\times 2}[s]$ and assume that its column degrees are 1 and 0. In this case $\Delta(s) = \operatorname{diag}\{1, s\}$.

The post-multiplication of $D(s)$ by a diagonal matrix $\Delta(s)$, defined as above, implies that one is now computing the inverse of the following polynomial matrix:

$$\tilde{D}(s) = D(s)\Delta(s). \tag{6}$$

From the computational point of view this does not represent any problem since the multiplication of a polynomial matrix by the diagonal matrix defined above is straightforward. It is also important to notice that:

(i) $\tilde{D}(s)$ and $D(s)$ have the same high coefficient matrices;
(ii) $\det[\tilde{D}(s)] = \det[D(s)]s^{n_0}$, where $n_0$ is the number of columns of $D(s)$ with degree equal zero;
(iii) $D^{-1}(s) = \Delta(s)\tilde{D}^{-1}(s)$.

With this in mind, a minimal order state-space realization for $\tilde{D}^{-1}(s)$ can be obtained according to the following algorithm.

**Algorithm 1.**
Step 1. For a given column reduced polynomial matrix

$$D(s) = \left[\underline{d}_1(s)\ \underline{d}_2(s)\ \cdots\ \underline{d}_m(s)\right]$$

compute $\tilde{D}(s) = D(s)\Delta(s)$, where $\Delta(s) = \text{diag}\{\delta_i(s),\ i = 1, 2, \ldots, m\}$ with $\delta_i(s) = 1$, if $\deg[\underline{d}_i(s)] \neq 0$, or $\delta_i(s) = s$, if $\deg[\underline{d}_i(s)] = 0$.

Step 2. Find constant matrices $D_{\text{hc}}$ and $D_{\text{lc}}$ such that $\tilde{D}(s) = D_{\text{hc}}\ \text{diag}\{s^{\nu_i},\ i = 1, 2, \ldots, m\} + D_{\text{lc}}\Psi(s)$ as given in (3).

Step 3. Form the matrices $A_{\text{c}_0}$, $B_{\text{c}_0}$ and $C_{\text{c}_0}$, where

$$A_{\text{c}_0} = \text{block diag}\left\{\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix},\ \nu_i \times \nu_i,\ i = 1, 2, \ldots, m\right\},$$

$$B_{\text{c}_0} = \text{block diag}\left\{\begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^{\text{T}},\ \nu_i \times 1,\ i = 1, 2, \ldots, m\right\}$$

and compute the matrices $A_{\text{c}}$ and $B_{\text{c}}$ as follows:

$$A_{\text{c}} = A_{\text{c}_0} - B_{\text{c}_0} D_{\text{hc}}^{-1} D_{\text{lc}},$$
$$B_{\text{c}} = B_{\text{c}_0} D_{\text{hc}}^{-1}.$$

Step 4. Write $I_m$, the numerator matrix of $\tilde{G}(s) = I_m \tilde{D}^{-1}(s)$, as $I_m = N_{\text{lc}}\Psi(s)$, where

$$N_{\text{lc}} = \text{block diag}\left\{\begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix},\ 1 \times \nu_i,\ i = 1, 2, \ldots, m\right\}$$

and obtain $C_{\text{c}} = N_{\text{lc}}$.

Step 5. $[A_{\text{c}}, B_{\text{c}}, C_{\text{c}}]$ is a minimal order realization of $\tilde{G}(s) = I_m \tilde{D}^{-1}(s)$.

**Remark B.**
1. The algorithm above provides a simple and effective way to form a minimal order state-space realization of the inverse of a column reduced polynomial matrix $\tilde{D}(s)$. Notice that since $\tilde{D}^{-1}(s) = C_{\text{c}}(sI - A_{\text{c}})^{-1}B_{\text{c}}$, then a transfer function representation of $\tilde{D}^{-1}(s)$ can be obtained directly via Leverrier's algorithm. [4] The computation of $D^{-1}(s)$ will then be immediate since $D^{-1}(s) = \Delta(s)\tilde{D}^{-1}(s)$.
2. According to [4, p. 409], $\det[\tilde{D}(s)] = \det(D_{\text{hc}})\ \det(sI - A_{\text{c}})$, and once again, Leverrier's algorithm can be deployed to compute $\det[\tilde{D}(s)]$ in a straightforward

[4] A concise and helpful presentation of Leverrier's algorithm can be found in [4, p. 657].

way. In addition, since the realization is minimal, then deg det$[\tilde{D}(s)] = n$, which implies that the denominator polynomial of $D^{-1}(s)$ is minimal.

### 2.3. Reduction of non-column reduced matrices

When the matrix $D(s)$ is not column reduced, it is necessary, before applying Algorithm 1, to find a column reduced matrix $\bar{D}(s)$, Smith equivalent to $D(s)$. This is done by post-multiplying $D(s)$ by a unimodular matrix $U(s)$, which is generally formed by the product of unimodular matrices.

Column reduction of polynomial matrices can be done in a variety of ways [3,4]. The methods are generally based on elementary column operations, which are known to be very sensitive. Here we propose a more robust algorithm which is based on singular value decomposition. This algorithm is a modification of a previous one [10] that was used to find a right-coprime matrix fraction description of a transfer matrix. The idea is, at each iteration, to post-multiply the matrix by an elementary one (formed by a suitable permutation of the identity matrix) with the view to reordering (in descending order) the column degrees and, in the sequel, to post-multiply by a unimodular matrix (formed by replacing some columns of the identity matrix by polynomial vectors) in order to reduce appropriately the column degrees. The algorithm below gives not only the column reduced matrix $\bar{D}(s)$, Smith equivalent to $D(s)$, but also the unimodular matrix $U(s)$. The reader may find it useful to refer to Example 2 while following the steps of the algorithm.

**Algorithm 2.**
Step 1. Define $\bar{D}(s) = D(s)$ and $U(s) = I_m$.

Step 2. Find a constant matrix $E$ such that $\hat{D}(s) = \bar{D}(s)E$ has column degrees $\hat{\nu}_1 \geqslant \hat{\nu}_2 \geqslant \cdots \geqslant \hat{\nu}_m$. The matrix $E$ can be obtained by an appropriate permutation of the columns of the identity matrix of order $m$. Compute $\hat{U}(s) = U(s)E$.

Step 3. Write $\hat{D}(s) = \hat{D}_{\mathrm{hc}}\hat{S}(s) + \hat{D}_{\mathrm{lc}}\hat{\Psi}(s)$, where $\hat{S}(s) = \mathrm{diag}\{s^{\hat{\nu}_i},\ i = 1, 2, \ldots, m\}$ and $\hat{D}_{\mathrm{hc}}$, $\hat{D}_{\mathrm{lc}}$ and $\hat{\Psi}(s)$ are formed as in (3) and (4).

Step 4. Perform a singular value decomposition of $\hat{D}_{\mathrm{hc}}$ and write $\hat{D}_{\mathrm{hc}} = \hat{X}\hat{\Sigma}\hat{Y}^{\star}$. Let $p$ denote the number of zero singular values of $\hat{D}_{\mathrm{hc}}$. If $p = 0$, then make $\bar{D}(s) = \hat{D}(s)$, $U(s) = \hat{U}(s)$ and stop. Otherwise, collect the columns $\hat{y}_{m-p+j}$, $j = 1, \ldots, p$ of $\hat{Y}$ associated with the $p$ zero singular values and form the set

$$\hat{\mathscr{Y}} = \{\hat{y}_{m-p+j},\ j = 1, \ldots, p\}.$$

Step 5. Let $\hat{\kappa}_j$ denote the position of the first non-zero element of $\hat{y}_{m-p+j}$ and notice that, because of the way singular value decompositions are performed, $\hat{\kappa}_{j+1} \geqslant \hat{\kappa}_j$. Form the set

$$\bar{\mathscr{Y}} = \{\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_{\bar{p}}\},$$

by keeping only the elements of $\hat{\mathscr{Y}}$ with distinct positions of the first non-zero elements. Let the position of the first non-zero element of $\bar{y}_j$ be $\bar{\kappa}_j$.

Step 6. Form a unimodular matrix $\bar{U}(s)$ by replacing columns $\bar{\kappa}_j$, $j = 1, \ldots, \bar{p}$, of the identity matrix of order $m$ by polynomial vectors $\bar{u}_j(s)$ whose elements $\bar{u}_{k,j}(s)$, $k = 1, \ldots, m$ are given by:

$$\bar{u}_{k,j}(s) = \begin{cases} 0, & k < \bar{\kappa}_j, \\ \bar{y}_{k,j} s^{\hat{\nu}_{\bar{\kappa}_j} - \hat{\nu}_k}, & k \geqslant \bar{\kappa}_j. \end{cases}$$

Step 7. Compute $\bar{D}(s) = \hat{D}(s)\bar{U}(s)$, $U(s) = \hat{U}(s)\bar{U}(s)$ and go back to step 2.

**Remark C.** The reader might have realized that the matrices $\hat{D}(s)$ and $\hat{U}(s)$ are re-defined at each iteration. This is so because the matrices actually needed are the column reduced matrix $\bar{D}(s)$ and the unimodular matrix $U(s)$ which carries out the column reduction of $D(s)$.

### *2.4. Inverse of non-column reduced matrices*

Algorithm 2 provides the means to obtain column reduced and unimodular matrices ($\bar{D}(s)$ and $U(s)$, respectively) such that for a given non-column reduced matrix $D(s)$, we have:

$$\bar{D}(s) = D(s)U(s). \tag{7}$$

Let $\bar{G}(s) = \bar{D}^{-1}(s)$ and notice that since $\bar{D}(s)$ is column reduced. Then Algorithm 1 can be deployed to compute $\bar{G}(s)$. In addition, from Eq. (7), it can be checked that:

$$D^{-1}(s) = U(s)\bar{D}^{-1}(s), \tag{8}$$

which implies that

$$G(s) = U(s)\bar{G}(s). \tag{9}$$

The computation of the inverse of non-column reduced polynomial matrices can be summarized in the following algorithm.

**Algorithm 3.** For a given $D(s)$, non-column reduced, then $G(s) = D^{-1}(s)$ is obtained as follows:
Step 1. Use Algorithm 2 to compute $\bar{D}(s)$ and $U(s)$ such that $\bar{D}(s) = D(s)U(s)$, where $\bar{D}(s)$ is column reduced and $U(s)$ unimodular.

Step 2. Use Algorithm 1 to compute $\bar{G}(s) = \bar{D}^{-1}(s)$.

Step 3. Compute $G(s) = U(s)\bar{G}(s)$.

## 3. Examples

In this section the methodology proposed in this paper will be illustrated by means of two numerical examples.

### 3.1. Example 1

Consider, initially, the inversion of the following column reduced polynomial matrix:

$$D(s) = \begin{bmatrix} 5.4848s^2 + 1.7321s - 0.2887 \\ -0.8660s^2 - 3.4641s - 0.2887 \\ 3.4641s^2 + 2.3094s - 1.4434 \end{bmatrix} \begin{bmatrix} -1.4142s^2 + 4.2426s + 2.1213 & -s^2 - 3s + 1 \\ -1.4142s - 2.1213 & -2 \\ 3.5355s + 2.8284 & -s^2 \end{bmatrix}. \tag{10}$$

A minimal order state-space realization for $G(s) = D^{-1}(s)$ can be obtained in a straightforward way providing $D(s)$ is written as

$$D(s) = \begin{bmatrix} 5.4848 & -1.4142 & -1 \\ -0.8660 & 0 & 0 \\ 3.4641 & 0 & -1 \end{bmatrix} S(s) + \begin{bmatrix} 1.7321 & -0.2887 & 4.2426 & 2.1213 & -3 & 1 \\ -3.4641 & -0.2887 & -1.4142 & -2.1213 & 0 & -2 \\ 2.3094 & -1.4434 & 3.5355 & 2.8284 & 0 & 0 \end{bmatrix} \Psi(s),$$

where

$$S(s) = \operatorname{diag}\{s^2, s^2, s^2\} \quad \text{and} \quad \Psi^{\mathrm{T}}(s) = \begin{bmatrix} s & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & s & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & s & 1 \end{bmatrix}.$$

Thus, according to steps 2 and 3 of Algorithm 1, $G(s) = C_{\mathrm{c}}(sI - A_{\mathrm{c}})^{-1}B_{\mathrm{c}}$, where

$$A_{\mathrm{c}} = \begin{bmatrix} -4 & -0.3333 & -1.6330 & -2.4495 & 0 & -2.3094 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ -6.1237 & 0.3402 & -1.8333 & -4 & -2.1213 & -2.5927 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ -11.5470 & -2.5981 & -2.1213 & -5.6569 & 0 & -8 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$$B_c = \begin{bmatrix} 0 & -1.1547 & 0 \\ 0 & 0 & 0 \\ -0.7071 & -1.6499 & 0.7071 \\ 0 & 0 & 0 \\ 0 & -4 & -1 \\ 0 & 0 & 0 \end{bmatrix},$$

$$C_c = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Finally, the use of Leverrier's algorithm leads to the following transfer function representation for $G(s)$:

$$\begin{aligned} G(s) = \frac{1}{d(s)} \Bigg( & \begin{bmatrix} 0 & -1.1547 & 0 \\ -0.7071 & -1.6499 & 0.7071 \\ 0 & -4 & -1 \end{bmatrix} s^4 \\ & + \begin{bmatrix} 1.1547 & 0.5774 & -1.1547 \\ -2.8284 & 8.9567 & 4.9497 \\ 1.5 & -6.5 & -7.3333 \end{bmatrix} s^3 \\ & + \begin{bmatrix} 1.7321 & -9.2376 & -2.8868 \\ -5.8926 & 1.8856 & 16.9706 \\ -3.3333 & -2 & 1.6667 \end{bmatrix} s^2 \\ & + \begin{bmatrix} 5.7735 & -4.0415 & -10.9697 \\ -3.7712 & -5.4212 & 0.7071 \\ -6.5 & -4.1667 & 4.3333 \end{bmatrix} s \\ & + \begin{bmatrix} 4.6188 & 2.3094 & -1.7321 \\ 2.3570 & 1.1785 & -0.7071 \\ -3.1667 & -1.8333 & 1 \end{bmatrix} \Bigg), \end{aligned} \tag{11}$$

where

$$\begin{aligned} d(s) = {} & s^6 + 5.8333s^5 + 5.1667s^4 + 26.6667s^3 \\ & + 22.3333s^2 + 11.3333s + 0.5. \end{aligned}$$

### 3.2. Example 2

Suppose it is necessary to find an inverse of the following $3 \times 3$ non-column reduced polynomial matrix:

$$D(s) = \begin{bmatrix} s^3 + s^2 + 5s + 3 & -s^2 - 3s + 1 & 2s^4 + s^3 + 2s + 1 \\ -3 & -2 & s^2 + 5s + 1 \\ s^3 + 5s + 4 & -s^2 & 2s^4 + s^3 + 3s^2 + 4s + 5 \end{bmatrix}.$$

According to Algorithm 3 the first step is to compute a column reduced $\bar{D}(s)$ and a unimodular matrix $U(s)$ such that $\bar{D}(s) = D(s)U(s)$. This can be done by following the steps of Algorithm 2.

Initially, let us define $\bar{D}(s) = D(s)$ and $U(s) = I_3$. Notice that the column degrees of $D(s)$ are 3, 2 and 4, respectively. This implies that $\bar{D}(s)$ should be multiplied by the following elementary matrix:

$$E = \begin{bmatrix} e_3 & e_1 & e_2 \end{bmatrix},$$

where $e_i$ denotes the $i$th column of the identity matrix, in order to obtain a matrix $\hat{D}(s) = \bar{D}(s)E$ whose columns degrees are in descending order. Therefore, at this stage, $\hat{U}(s) = E$. In addition, writing $\hat{D}(s)$ as $\hat{D}(s) = \hat{D}_{\mathrm{hc}}\hat{S}(s) + \hat{D}_{\mathrm{lc}}(s)$, one obtains:

$$\hat{D}_{\mathrm{hc}} = \begin{bmatrix} 2 & 1 & -1 \\ 0 & 0 & 0 \\ 2 & 1 & -1 \end{bmatrix} \quad \text{and} \quad S(s) = \operatorname{diag}\{s^4, s^3, s^2\}.$$

Notice that $\hat{D}_{\mathrm{hc}}$ has clearly rank equal 1. Performing now a singular value decomposition of $\hat{D}_{\mathrm{hc}}$ and taking, according to step 4 of Algorithm 2, the columns of $\hat{Y}$ corresponding to the two zero singular values, the following set can be formed:

$$\hat{\mathscr{Y}} = \left\{ \begin{bmatrix} -0.5774 \\ 0.5774 \\ -0.5774 \end{bmatrix}, \begin{bmatrix} 0 \\ 0.7071 \\ 0.7071 \end{bmatrix} \right\}.$$

It is important to note that the first nonzero elements of the columns of $\hat{\mathscr{Y}}$ are in different positions, which allows us to construct, in accordance with step 6 of Algorithm 2, the matrix $\bar{U}(s)$ directly, as follows:

$$\bar{U}(s) = \begin{bmatrix} -0.5774 & 0 & 0 \\ 0.5774s & 0.7071 & 0 \\ -0.5774s^2 & 0.7071s & 1 \end{bmatrix}.$$

Notice that $\bar{U}(s)$ is constructed in order to be a lower triangular matrix, being, therefore, unimodular.

Following step 7 of Algorithm 2, it is possible now to compute $\bar{D}(s)$ and $U(s)$, which are given as:

$$\bar{D}(s) = \begin{bmatrix} 1.7321s^3 + 2.3094s^2 + 0.5774s - 0.5774 \\ 0.5774s^2 - 4.6188s - 0.5774 \\ -0.5774s^3 + 1.1547s^2 - 2.8868 \end{bmatrix}$$

$$\begin{bmatrix} -1.4142s^2 + 4.2426s + 2.1213 & -s^2 - 3s + 1 \\ -1.4142s - 2.1213 & -2 \\ 3.5355s + 2.8284 & -s^2 \end{bmatrix}$$

and

$$U(s) = E\bar{U}(s) = \begin{bmatrix} 0.5774s & 0.7071 & 0 \\ -0.5774s^2 & 0.7071s & 1 \\ -0.5774 & 0 & 0 \end{bmatrix}.$$

Going back to step 2, it is necessary now to find a new matrix $E$. It is clear that $E = I_3$, since the column degrees of $\bar{D}(s)$ are already in descending order. Thus we can make $\hat{D}(s) = \bar{D}(s)$ and $\hat{U}(s) = U(s)$. As pointed out in Remark B, since what one is actually interested in is the resulting column reduced matrix $\bar{D}(s)$ and the unimodular matrix $U(s)$, the matrices $\hat{D}(s)$ and $\hat{U}(s)$ may be discarded after $\bar{D}(s)$ and $U(s)$ having been computed.

As in the previous iteration, the high leading coefficient matrix $\hat{D}_{\text{hc}}$ and the matrix $\hat{S}(s)$ of column degrees for $\hat{D}(s)$ should be obtained. These matrices are given by:

$$\hat{D}_{\text{hc}} = \begin{bmatrix} 1.7321 & -1.4142 & -1 \\ 0 & 0 & 0 \\ -0.5774 & 0 & -1 \end{bmatrix} \quad \text{and} \quad \hat{S}(s) = \operatorname{diag}\{s^3, s^2, s^2\}.$$

By performing singular value decomposition, it can be seen that $\hat{D}_{\text{hc}}$ has one zero singular value, which shows that $\hat{D}(s)$ is still non-column reduced. Consequently $\hat{\mathscr{Y}}$ is a single-element set, as follows:

$$\hat{\mathscr{Y}} = \left\{ \begin{bmatrix} 0.5 & 0.8165 & -0.2887 \end{bmatrix}^{\text{t}} \right\}.$$

This implies that

$$\bar{U}(s) = \begin{bmatrix} 0.5 & & \\ 0.8165s & \underline{e}_2 & \underline{e}_3 \\ -0.2887s & & \end{bmatrix}.$$

The new matrices $\bar{D}(s)$ and $U(s)$ will be given by:

$$\bar{D}(s) = \begin{bmatrix} 5.4848s^2 + 1.7321s - 0.2887 \\ -0.8660s^2 - 3.4641s - 0.2887 \\ 3.4641s^2 + 2.3094s - 1.4434 \end{bmatrix}$$
$$\begin{bmatrix} -1.4142s^2 + 4.2426s + 2.1213 & -s^2 - 3s + 1 \\ -1.4142s - 2.1213 & -2 \\ 3.5355s + 2.8284 & -s^2 \end{bmatrix}$$

and

$$U(s) = \hat{U}(s)\bar{U}(s) = \begin{bmatrix} 0.8660s & 0.7071 & 0 \\ 0.2887s^2 - 0.2887s & 0.7071s & 1 \\ -0.2887 & 0 & 0 \end{bmatrix}.$$

Since the matrix $\bar{D}(s)$ above is column reduced, Algorithm 2 reaches its end. The next step of Algorithm 3 is to use Algorithm 1 to find $\bar{D}^{-1}(s)$. Notice that this has already been done since $\bar{D}(s)$ given above is equal to the matrix given in (10), whose inverse is shown in (11). Therefore a transfer function representation for $G(s) = D^{-1}(s) = U(s)\bar{G}(s)$ is given by:

$$
\begin{aligned}
G(s) = \frac{1}{d(s)} \Bigg( & \begin{bmatrix} 0 & 0 & 0 \\ 0 & -0.3333 & 0 \\ 0 & 0 & 0 \end{bmatrix} s^6 \\
& + \begin{bmatrix} 0 & -1 & 0 \\ -0.1667 & -0.6667 & 0.1667 \\ 0 & 0 & 0 \end{bmatrix} s^5 \\
& + \begin{bmatrix} 0.5 & -0.6667 & -0.5 \\ -1.8333 & -0.5 & 2 \\ 0 & 0.3333 & 0 \end{bmatrix} s^4 \\
& + \begin{bmatrix} -0.5 & -1.6667 & 1 \\ -1.5 & -3.6667 & 2.3333 \\ -0.3333 & -0.1667 & 0.3333 \end{bmatrix} s^3 \\
& + \begin{bmatrix} 0.8333 & -2.1667 & 2.5 \\ -6.3333 & -4 & 4.8333 \\ -0.5 & 2.6667 & 0.8333 \end{bmatrix} s^2 \\
& + \begin{bmatrix} 1.3333 & -1.8333 & -1 \\ -6.1667 & -4 & 4.3333 \\ -1.6667 & 1.1667 & 3.1667 \end{bmatrix} s \\
& + \begin{bmatrix} 1.6667 & 0.8333 & -0.5000 \\ -3.1667 & -1.8333 & 1 \\ -1.3333 & -0.6667 & 0.5000 \end{bmatrix} \Bigg),
\end{aligned}
$$

where, as in Example 1,

$$
\begin{aligned}
d(s) = & s^6 + 5.8333s^5 + 5.1667s^4 + 26.6667s^3 \\
& + 22.3333s^2 + 11.3333s + 0.5.
\end{aligned}
$$

## 4. Conclusions

In this paper, algorithms of easy implementation to compute inverses of either column or non-column reduced polynomial matrices have been proposed. With the help of Leverrier's algorithm, it has also been suggested an efficient way to compute determinant of polynomial matrices. Finally, numerical examples are given to illustrate the efficiency and simplicity of the algorithms.

## Acknowledgement

The author is grateful to Prof. S. P. Bhattacharyya (Texas A&M University) for the encouragement to write and submit this paper for publication.

## References

[1] F.R. Gantmacher, The Theory of Matrices, vols. I and II, Chelsea Publishing Company, New York, 1959.
[2] I. Gohbergb, P. Lancaster, L. Rodman, Matrix Polynomials, Academic Press, New York, 1982.
[3] W.A. Wolovich, Linear Multivariable Systems, Springer, New York, 1974.
[4] T. Kailath, Linear Systems, Prentice-Hall, Englewood Cliffs, NJ, 1980.
[5] P. Stefanidis, A.P. Paplinski, M.J. Gibbard, Numerical Operations with Polynomial Matrices, Lecture Notes in Control and Information Sciences, vol. 171, Springer, Berlin, 1992.
[6] Y. Inouye, An algortihm for inverting polynomial matrices, Int. J. Control 30 (1979) 989–999.
[7] M. Buslowicz, Inversion of polynomial matrices, Int. J. Control 33 (1980) 977–984.
[8] S. Zhang, Inversion of polynomial matrices, Int. J. Control 46 (1987) 33–37.
[9] A. Schuster, P. Hippe, Inversion of polynomial matrices by interpolation, IEEE Trans. Autom. Control 37 (1992) 363–365.
[10] J.C. Basilio, B. Kouvaritakis, An algorithm for coprime matrix fraction description using Sylvester matrices, Linear Algebra Appl. 266 (1997) 107–125.