Drazin inverse of one-variable polynomial matrices

Predrag S. Stanimirović and Milan B. Tasić

Abstract

There is proposed[1] a representation of the Drazin inverse of a given polynomial square matrix, based on the extension of the Leverrier-Faddeev algorithm. Also, an algorithm for symbolic computation of the Drazin inverse of polynomial matrix is established. This algorithm represents an extension of the papers [5], [7] and a continuation of the papers [8], [9], [10]. The implementation is developed in the symbolic package MATHEMATICA.

1 Introduction

Let $\mathbf{C}$ be the set of complex numbers and $\mathbf{C}^{m\times n}$ be the set of $m\times n$ complex matrices. As usual, $\mathbf{C}[s]$ denotes the polynomials with complex coefficients in the indeterminate s. The $m\times n$ matrices with elements in $\mathbf{C}[s]$ are denoted by $\mathbf{C}[s]^{m\times n}$. By I_r we denote the identity matrix of the order r, and by $\mathbf{O}$ is denoted an appropriate null matrix. Also, by $\mathrm{Tr}(A)$ is denoted the trace of A.

Various methods are used for computing the usual inverse of a constant regular matrix $A\in\mathbf{C}^{m\times n}$ by means of the Leverrier-Faddeev algorithm (also called Souriau-Frame algorithm) are presented in [4], [3], [1] and [11]. A more general algorithm for computing the Moore-Penrose generalized inverses of a given rectangular or singular constant matrix $A\in\mathbf{C}^{m\times n}$, based on the Leverrier-Faddeev algorithm, is originated in [2].

Also, in [1], a new derivation of the Leverrier-Faddeev algorithm is utilized to produce corresponding computational scheme for the inverse of the polynomial matrix $s^2I_n - sA_1 - A_2$ of power 2. In [11] it is introduced a new extension of Leverrier's algorithm for computing the inverse of a matrix polynomial of arbitrary degree. In [9] it is derived a representation and an algorithm for computation of the Moore-Penrose inverse of a nonregular polynomial matrix of an arbitrary degree. In [8] it is described an algorithm for computing the Moore-Penrose inverse of a singular rational matrix and its implementation in the symbolic computational language MAPLE.

Let us mention that in the literature it is known a number of applications of the Moore-Penrose inverse of polynomial and rational matrices [8], [10].

[1]Presented at the IMC "Filomat 2001", Niš, August 26–30, 2001

2000 Mathematics Subject Classification: 15A09, 68Q40
Keywords: Polynomial matrix, Leverrier-Faddeev algorithm, characteristic polynomial, Generalized inverses.

A modification of the Leverrier-Faddeev algorithm for computation of the Drazin inverse is discovered in [5] by Grevile. Hartwig in [6] continues investigation of this algorithm. An alternative finite algorithm for computation of the Drazin inverse is introduced in [7].

Hence, we observe the absence of algorithms for computation of the Drazin inverse of polynomial and rational matrices. In the second section we restate well-known modification of the Leverrier-Faddeev algorithm for computation of the Drazin inverse, introduced in [5] and alternatively verified in [7]. In the third section it is presented an extension of the Grevile's modification of the Leverrier-Faddeev algorithm, to the set of one-variable nonregular rational matrices. We also propose an algorithm for computation of the Drazin inverse for one-variable polynomial matrix in terms of its coefficient matrices. There proposed results can be considered as a continuation of the papers [8], [9], [10] and a generalization of the results from [5] and [7]. In the fourth section we describe implementation details of the algorithm and give an illustrative example.

2 Preliminaries

In this section we consider a constant singular square complex matrix $A \in \mathbf{C}^{n\times n}$. In [5] is introduced the following representation of the Drazin inverse.

Theorem 2.1 *[5], [7]. Consider a singular square matrix $A \in \mathbf{C}^{n\times n}$. Assume that*

$$\begin{aligned} a(z) &= \det[zI_n - A] \\ &= a_0 z^n + a_1 z^{n-1} + \cdots + a_{n-1}z + a_n, \quad a_0 = 1, \quad z \in \mathbf{C} \end{aligned}$$

is the characteristic polynomial of A. Also, consider the following sequence of $n\times n$ constant matrices defined by coefficients a_i and powers of A:

$$B_j = a_0 A^j + a_1 A^{j-1} + \cdots + a_{j-1}A + a_j I_n, \quad a_0 = 1, \quad j = 0, \dots, n$$

Let r denote the smallest integer such that $B_r = \mathbf{O}$, let t denote the largest integer satisfying $a_t \neq 0$, and let $k = r - t$. Then, the Drazin pseudoinverse of A is given by

$$A^D = (-1)^{k+1} a_t^{-k-1} A^k B_{t-1}^{k+1}. \tag{1}$$

Also, in [5] is proposed the following algorithm for computation of the Drazin inverse of A, which avoids explicit matrix powering.

Algorithm 2.1 *Consider $A \in \mathbf{C}^{n\times n}$.*

S1. Construct the sequence of complex numbers $\{a_0, a_1, \dots, a_n\}$ and the sequence of $n\times n$ matrices $\{B_0, B_1, \dots, B_n\}$ in the following way:

$$\begin{array}{lll} A_0 = \mathbf{O}, & a_0 = 1, & B_0 = I_n \\ A_1 = AB_0, & a_1 = -\frac{Tr(A_1)}{1}, & B_1 = A_1 + a_1 I_n \\ \dots & \dots & \dots \\ A_n = AB_{n-1}, & a_n = -\frac{Tr(A_n)}{n}, & B_n = A_n + a_n I_n \end{array} \tag{2}$$

S2. Let $t = \max\{l : a_l \neq 0\}$, $r = \min\{l : B_l = \mathbf{O}\}$, $k = r - t$. *Then the Drazin inverse* A^D *is given by (2.1).*

3 Drazin inverse of a polynomial matrix

Now, we assume that $A(s) \in \mathbf{C}[s]^{n\times n}$ is a polynomial matrix of the form

$$A(s)=\{a_{ij}(s)\}=A_q s^q + A_{q-1}s^{q-1} + \cdots A_1 s + A_0 \in \mathbf{C}[s]^{n\times n}, \tag{3}$$

where $a_{ij}(s) \in \mathbf{C}[s]$, $A_i \in \mathbf{C}^{n\times n}$, $i = 0,\ldots,q$ are constant complex matrices, and s is an unknown variable. The proof of the following statement is similar with the corresponding one from [5]. This statement is valid also for rational matrices.

Lemma 3.1 *Consider a nonregular one-variable polynomial matrix* $A(s)$ *of the form* (3). *Assume that*

$$\begin{aligned} a(z,s) &= \det[zI_n - A(s)] = a_0(s)z^n + a_1(s)z^{n-1} + \cdots a_{n-1}(s)z + a_n(s), \\ & \qquad a_0(s) \equiv 1,\ a_i(s) \in \mathbf{C}[s], \quad z \in \mathbf{C} \end{aligned}$$

is the characteristic polynomial of $A(s)$*. Also, consider the following sequence of* $n \times n$ *polynomial matrices*

$$\begin{aligned} B_j(s) &= a_0(s)A(s)^j + a_1(s)A(s)^{j-1} + \cdots a_{j-1}(s)A(s) + a_j(s)I_n, \\ & \qquad a_0(s) = 1, \quad j = 0,\ldots,n \end{aligned} \tag{4}$$

Let

$$\begin{aligned} a_n(s) &\equiv 0,\ldots,a_{t+1}(s) \equiv 0, \quad a_t(s) \neq 0, \\ {}_n(s) &\equiv \mathbf{O},\ldots,B_r(s) \equiv \mathbf{O}, \quad B_{r-1}(s) \neq \mathbf{O}. \end{aligned}$$

The Drazin inverse of $A(s)$ *is given by*

$$A(s)^D = (-1)^{k+1} a_t(s)^{-k-1} A(s)^k B_{t-1}(s)^{k+1}.$$

In view of the results of Lemma 3.1 we present the following algorithm for computation of the Drazin inverse. This algorithm is a generalization of *Algorithm 2.1* and permits simultaneous computation of the polynomials $a_i(s)$ and polynomial matrices $B_i(s)$.

Algorithm 3.1 *Input is a given polynomial matrix* $A(s) \in \mathbf{C}[s]^{n\times n}$.

S1. Construct the sequence of polynomials $\{a_0(s),\ldots,a_n(s)\}$ *and the sequence of polynomial matrices* $\{B_0(s),\ldots,B_n(s)\}$ *as in the following:*

$$\begin{array}{lll} A_0(s)=\mathbf{O}, & a_0(s)=1, & B_0(s)=I_n \\ A_1(s)=A(s)B_0(s), & a_1(s)=-\frac{Tr(A_1(s))}{1}, & B_1(s)=A_1(s)+a_1(s)I_n \\ \cdots & \cdots & \cdots \\ A_n(s)=A(s)B_{n-1}(s), & a_n(s)=-\frac{Tr(A_n(s))}{n}, & B_n(s)=A_n(s)+a_n(s)I_n. \end{array} \tag{5}$$

S2. Let

$$t = \max\{l : a_l(s) \neq 0\}, \quad r = \min\{l : B_l(s) = \mathbf{O}\}, \quad k = r - t.$$

The Drazin inverse $A(s)^D$ is given by

$$A(s)^D = a_t(s)^{-k-1} A(s)^k B_{t-1}(s)^{k+1} \tag{6}$$

Algorithm 3.1 is quite related to Lemma 3.1. Its implementation is practical only in symbolic computational packages, because it deals with polynomials and polynomial matrices. Our intent is to derive a representation of the Drazin inverse which is independent with respect to Lemma 3.1. This is way to look for coefficients in the polynomials $a_i(s)$ and constant coefficient matrices in the polynomial matrices $B_i(s)$. This approach gives us a representation of the Drazin inverse in terms of coefficient matrices A_i, and an analogous algorithm for computation of the Drazin inverse.

Theorem 3.1 *Let the polynomial matrix $A(s)$ is of the form* (3)*. Consider the polynomials $\{a_0(s), \ldots, a_n(s)\}$ and the polynomial matrices $\{B_0(s), \ldots, B_n(s)\}$ defined as in Algorithm 3.1. Rewrite $a_i(s)$ and $B_i(s)$ as*

$$a_i(s) = \sum_{j=0}^{iq} a_{i,j} s^j, \quad i = 1, \ldots, n, \tag{7}$$

$$B_i(s) = \sum_{j=0}^{iq} B_{i,j} s^j, \quad i = 1, \ldots, n, \tag{8}$$

where $a_{i,j}$, $i = 1, \ldots, n$, $j = 0, \ldots, iq$ are scalars, and $B_{i,j}$, $i = 1, \ldots, n$, $j = 1, \ldots, iq$ are constant coefficient matrices corresponding to powers s^j. Then we have

$$a_{i+1,j} = -\frac{1}{i+1} Tr\Big(\sum_{l=0}^{j} A_{j-l} B_{i,l}\Big) \tag{9}$$

$$B_{i+1,j} = \sum_{l=0}^{j} A_{j-l} B_{i,l} + a_{i+1,j} I_n, \tag{10}$$

for $i = 0, \ldots, n-1$, $\quad j = 0, \ldots, (i+1)q$, where in (9) *and* (10) *it is assumed $A_k = \mathbf{O}$, $k \geq q+1$, $B_{i,k} = \mathbf{O}$, $k \geq iq+1$.*

Proof. It is not difficult to verify from (5) that the greatest powers of $A_i(s)$ (and thus of $B_i(s)$) are equal to iq, $i = 0, \ldots, n-1$. Also, the degree of the polynomial quantities $a_i(s)$, $i = 1, \ldots, n$ is at most equal to iq. Hence $a_i(s)$ and $B_i(s)$ can be written in the form (7) and (8), respectively. Also, from (3), (5) and (8), we get the following

$$A_{i+1}(s) = A(s)B_i(s) = \left(\sum_{j=0}^{q} A_j s^j\right)\left(\sum_{l=0}^{iq} B_{i,l} s^l\right) = \sum_{j=0}^{(i+1)q}\left(\sum_{l=0}^{j} A_{j-l} B_{i,l}\right) s^j. \tag{11}$$

Now, an application of (5) and (11) leads to

$$a_{i+1}(s) = -\frac{1}{i+1}\mathrm{Tr}(A_{i+1}(s)) = \sum_{j=0}^{(i+1)q}\left[-\frac{1}{i+1}\mathrm{Tr}(\left(\sum_{l=0}^{j} A_{j-l}B_{i,l}\right))\right] s^j. \quad (12)$$

The identity (9) follows from (7) and (12).

On the other hand, using (5), (7) and (11), we get

$$\begin{aligned} B_{i+1}(s) &= A_{i+1}(s) + a_{i+1}(s)I_n \\ &= \sum_{j=0}^{(i+1)q}\left(\sum_{l=0}^{j} A_{j-l}B_{i,l}\right)s^j + \sum_{j=0}^{(i+1)q} a_{i+1,j}I_n s^j \qquad (13) \\ &= \sum_{j=0}^{(i+1)q}\left(\sum_{l=0}^{j} A_{j-l}B_{i,l} + a_{i+1,j}I_n\right)s^j \end{aligned}$$

The identity (10) follows from (8) and (13). ∎

Now we are in a position to state the following algorithm for computation of the Drazin inverse $A(s)^D$ in terms of coefficient matrices A_i.

Algorithm 3.2

Initial conditions: $B_{0,0} = I_n, \quad A_j = 0,\ j = q+1, \dots, nq.$

Boundary conditions:

$$\begin{aligned} B_{0,j} &= \mathbf{O} \quad \forall j \in \mathbf{N} \\ B_{i,j} &= \mathbf{O}, \quad i=0,\dots,n-1,\ j=iq+1,\dots,(n-1)q. \end{aligned}$$

Recursive relations for $a_i(s)$ *are defined in* (9).

Recursive relations for $B_i(s)$ *are defined in* (10).

Termination criteria: Compute the first t *and* r *satisfying*

$$\begin{aligned} a_{t+1,j} &= a_{t+2,j} = \cdots = a_{n,j} = 0 \quad \forall j \in \mathbf{N}, \\ B_{r,j} &= \mathbf{O}, \quad j = 0, 1, \dots, rq. \end{aligned}$$

Output: Compute $k = r - t$ *and return*

$$A(s)^D = (-1)^{k+1}\left(\sum_{j=0}^{tq} a_{t,j}s^j\right)^{-k-1}\left(\sum_{i=0}^{q} A_i s^i\right)^k \left(\sum_{l=0}^{(t-1)q} B_{t-1,l}s^l\right)^{k+1}.$$

4 Implementation

It is well-known that all Cayley-Hamilton-theorem-based algorithms are numerically unstable because it is implicitly taking powers of matrices [10]. Therefore, algorithms 3.1-3.2 work for rational and polynomial matrices, respectively, but for poorly conditioned data results can be significantly incorrect [10]. This fact is a motivation for the usage of symbolic programming language MATHEMATICA, version 4.1, in the implementation. About the package MATHEMATICA see, for example, [12] and [13].

Essentially, main function *DrzPoly[]* deals with a polynomial matrix represented by the corresponding three-dimensional list of the form $\{l_1, \dots, l_{q+1}\}$. The coefficient matrices A_i are indicated by the global array A, which is defined by

$$A[[i]] = l_{i-1},\ \ i = 0, \dots, q, \quad A[[i]] = \mathbf{O},\ \ i = q+1, \dots, nq.$$

Recursive relations (9) and (10) for computing the real numbers $a_{i,j}$ and the constant matrices $B_{i,j}$, respectively, are implemented in the following two functions, appropriate also for procedural programming languages. The result of the standard function *Tr[a_]* is the trace of matrix a.

```
Fa[i1_,j1_]:=-1/i1*Tr[Sum[A[[j1-l+1]].FB[i1-1,l],{l,0,j1}]];
FB[u_,v_]:= Block[{w={},nul},
    nul=Table[0,{n},{n}];    (* The zero matrix *)
    If[u==0 && v==0,  w=IdentityMatrix[n],  (* Then *)
      If[u==0,  (* Else if*)
        w=nul,    (* Then *)
        If[(v>=u q+1)&&(v<=(n-1) q), (* Else if *)
          w=nul,
          w=Sum[A[[v-k+1]].FB[u-1,k],{k,0,v}]+Fa[u,v]*IdentityMatrix[n]
    ] ] ];
    Return[w]
 ];
```

However, three-dimensional list representation is inconvenient for the user. It is much more operative to present each element of a polynomial matrix as the corresponding polynomial $a_{ij}(s) \in \mathbf{C}[s]$. By means of the function *FrmA[A_List,var_]* we construct the list of coefficient matrices corresponding to matrix $A(var) = \{a_{ij}(var)\}$. In the function *FrmA[A_List,var_]* we use symbolic computations, unsuitable for procedural languages. By the expression *q=Max[Exponent[a,var,List]]* we pick the greatest power q in (3). Also, the coefficient matrix A_i corresponding to var^i is computed by *koef = Coefficient[a,var^i]*.

```
FrmA[a_List,var_]:= Block[{L={},L1,nul,q,n,m,i},
   {n,m}=Dimensions[a]; nul=Table[0,{i,n},{j,m}];
   L1=nul;    q=Max[Exponent[a,var,List]];
   For[i=1, i<=q, i++,
      koef=Coefficient[a,var^i];  L=Append[L,koef];    L1=L1+koef*var^i
   ];
   If[L==={}, Return[{a-L1}] ];
   L1={a-L1};  For[i=1,i<=Length[L],i++, AppendTo[L1,L[[i]]] ];
   Return[L1]
 ]
```

The method for computation of the Drazin inverse which is described in *Algorithm 3.2*, is implemented in the following function *DrzPoly[mat_List]*. A major problem in the implementation of this function in procedural programming languages is the simplification of polynomials corresponding to elements of the Drazin inverse. This problem in MATHEMATICA is easily solved by the standard function *Simplify*, which is applicable to rational expressions with real coefficients.

```
DrzPoly[mat_List]:=
 Block[{A,i,j,nul,k,q,r=t=0,log1=log2=True,var=Variables[mat]},
   If[var =={}, A={mat},A=FrmA[mat,First[var]]];
   q=Length[A]-1; {n,n}=Dimensions[A[[1]]];  nul=Table[0,{n},{n}];
   For[i=q+1,i<=n q,i++,A=Append[A,nul]];
   For[i=0,i<=n-1,i++,
      For[j=0,j<=(i+1) q,j++,
        If[Fa[i+1,j]==0 ,log1=log1 && True,log1=log1 && False];
        If[FB[i+1,j]===nul ,log2=log2 &&True,log2=log2 &&False];
      ];
      If[!log1,t=i+1;log1=True];     If[!log2,r=i+1];
   ];
   k=r-t;
   rez1=(-1)^(k+1)*Sum[Fa[t,j]*First[var]^j,{j,0,t q}]^(-k-1);
   rez2=MatrixPower[Sum[A[[i+1]] First[var]^i,{i,0,q}],k];
   rez3=MatrixPower[Sum[FB[t-1,l] First[var]^l,{l,0,(t-1) q}],k+1];
   rez=rez1*rez2.rez3;
   Return[Simplify[rez]]
 ]
```

Example 4.1 The polynomial matrix

$$A(s)=\begin{bmatrix} 1+s & s & 1+s \\ s^2 & -1+s & s \\ 1+s & s & 1+s \end{bmatrix}=\begin{bmatrix} 1 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 1 \end{bmatrix}+\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}s+\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}s^2.$$

is represented by the following three-dimensional list:
`{{{1,0,1},{0,-1,0},{1,0,1}},{{1,1,1},{0,1,1},{1,1,1}},{{0,0,0},{1,0,0},{0,0,0}}}.`
generated by the expression **FrmA[a,s]** Applying the program $DrzPoly[mat]$, we get $t=2$, $r=3$ and the following Drazin inverse of A:

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

5 Conclusion

We develop two algorithms for symbolic computation of the Drazin inverse of a given square one-variable polynomial matrix of the form (3). These algorithms are a continuation of the papers [8], [9], [10] and extension of the papers [5], [6], [7]. Also, the implementation in the package MATHEMATICA of the algorithm is described.

References

[1] S. Barnett, *Leverrier's algorithm: a new proof and extensions*, SIAM J. Matrix Anal. Appl. **10** (1989), 551–556.

[2] H.P. Decell, *An application of the Cayley-Hamilton theorem to generalized matrix inversion*, SIAM Review **7** No 4 (1965), 526–528.

[3] D.K. Faddeev and V.N. Faddeeva, *Computational Methods of Linear Algebra*, Freeman, San Francisko, 1963.

[4] J.S. Frame, *A simple recursion formula for inverting a matrix*, Bull. Amer. Math. Soc. **55** (1949), 19–45.

[5] T.N.E. Grevile, *The Souriau-Frame algorithm and the Drazin pseudoinverse*, Linear Algebra Appl. **6** (1973), 205–208.

[6] R.E. Hartwig, *More on the Souriau-Frame algorithm and the Drazin inverse*, SIAM J. Appl. Math. **31** No 1 (1976), 42–46.

[7] J. Ji, *An alternative limit expression of Drazin inverse and its applications*, Appl. Math. Comput. **61** (1994), 151–156.

[8] J. Jones, N.P. Karampetakis and A.C. Pugh, *The computation and application of the generalized inverse vai Maple*, J. Symbolic Computation **25** (1998), 99–124.

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

[10] N.P. Karampetakis, *Generalized inverses of two-variable polynomial matrices and applications*, Circuits Systems Signal Processing **16** (1997), 439–453.

[11] G. Wang and Y. Lin, *A new extension of the Leverrier's algorithm* Linear Algebra Appl. **180** (1993), 227–238.

[12] S. Wolfram, *Mathematica: a System for Doing Mathematics by Computer*, Addison-Wesley Publishing Co, Redwood City, California, 1991.

[13] S. Wolfram, *Mathematica Book, Version 3.0* Addison-Wesley Publishing Co, Redwood City, California, 1996.

University of Niš, Faculty of Science and Mathematics,
Višegradsa 33, 18000 Niš, Yugoslavia.
pecko@pmf.pmf.ni.ac.yu,
milan12t@ptt.yu