Asian Journal of Research in Computer Science

Volume 17, Issue 5, Page 1-9, 2024; Article no.AJRCOS.112734
ISSN: 2581-8260

Computing the Minimum Polynomial, the Function and the Drazin Inverse of a Matrix with Matlab

Nikolaos Halidias [a*]

[a] *Department of Statistics and Actuarial - Financial Mathematics, University of the Aegean, Greece.*

Author's contribution

The sole author designed, analyzed, interpreted and prepared the manuscript.

Article Information

DOI: 10.9734/AJRCOS/2024/v17i5433

Open Peer Review History:

This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here: https://www.sdiarticle5.com/review-history/112734

Received: 13/12/2023
Accepted: 17/02/2024
Published: 23/02/2024

Original Research Article

ABSTRACT

Aims/ Objectives: In this note we will discuss the known (but not well known) problem of finding the minimum polynomial and the function of a matrix providing the simplest proofs for undergraduate students. We will try to explain with fairly simple arguments how to compute the minimum polynomial of a matrix giving also the `matlab` code for its symbolic computation. Next we will describe the (symbolic) computation of the matrix of a function via the Hermite interpolation method which seems to be the simplest method for undergraduate students. Finally we shall see how we can compute the Drazin inverse given the nth power of a matrix A.

Keywords: Minimum polynomial of a matrix; function of a matrix; `matlab` codes.

2010 Mathematics Subject Classification: 15-02, 15A09, 15A16.

**Corresponding author: E-mail: nick@aegean.gr;*

1 INTRODUCTION

A problem that arises at least at the study of Markov chains is to compute the nth power of a stochastic matrix or the exponential matrix e^{tA}. A well known approach is via the diagonalization of the matrix A but this approach needs the matrix A to be diagonalizable. Below, we will describe the Hermite interpolation method which seems to be the simplest way for undergraduate students. For more about the matrix function computation one can refer to the bibliography given at the end of this note (see for example [1], [2], [3]).

Let $A_{k\times k}$ be a matrix of real or complex numbers. If we want to compute symbolically the matrix A^n we can do the following. Find a polynomial $p(x) = x^q + c_{q-1}x^{q-1} + \cdots + c_0$ which is such that

$$p(A) = A^q + c_{q-1}A^{q-1} + \cdots + c_0\mathbb{I}_{k\times k} = 0_{k\times k}$$

Dividing theoretically the polynomial x^n, for $n \geq q$, by $p(x)$ we have

$$x^n = \pi(x)p(x) + v(x), \quad \forall x \in \mathbb{R}$$

where $v(x)$ is a polynomial with degree at most $q-1$. That means that the polynomial $\Delta(x) = x^n - \pi(x)p(x) - v(x)$ has all its coefficients equal to zero and so

$$A^n = v(A)$$

where

$$v(A) = a_{q-1}A^{q-1} + \cdots + a_1 A + a_0\mathbb{I}_{k\times k}$$

The only thing that we have to do is to find the coefficients of $v(x)$. In this problem, the roots of the polynomial $p(x)$ will play an important role. If $p_1, \cdots, p_l$ are the roots with multiplicity $j_1, \cdots, j_l$ (with $j_1 + \cdots + j_l = q$) then we construct the following system, setting $f(x) = x^n$,

$$v^{(r)}(p_i) = f^{(r)}(p_i), \quad r = 0, \cdots, j_i - 1, \quad i = 1, \cdots, l \quad (1.1)$$

where $f^{(r)}$ is the r derivative of f and $f^{(0)} = f$.

Let us look at the homogeneous version of the system (1.1) and assume that it has a nontrivial solution. According to the fundamental theorem of algebra the sum of the multiplicities of the roots of any polynomial of degree n is exactly n. The degree of the polynomial $v(\cdot)$ equal to $q-1$ therefore the sum of the multiplicities of the roots is exactly $q-1$. However, by the homogeneous version of the system (1.1) we deduce that the sum of the multiplicities of the roots of the polynomial $v(\cdot)$ equals to q and this is true only when the polynomial $v(\cdot)$ is identically zero. Therefore the homogeneous version of the system (1.1) has only the trivial solution and that means that the system (1.1) has a unique solution.

A polynomial with the property $p(A) = 0_{k\times k}$ is of course the characteristic polynomial of A. The system (1.1) suggests that the less the degree of the polynomial $p(x)$ the better. That means that we should find, if it is possible, the minimum polynomial with the property $p(A) = 0_{k\times k}$.

2 THE COMPUTATION OF THE MINIMUM POLYNOMIAL

Let $A_{k\times k}$ be a given matrix of complex numbers. Below we give the `matlab` function in order to compute the minimum polynomial of this matrix. The coefficients of the polynomial are in the vector v, i.e. the minimum polynomial is

$$m(x) = v(1)x^r + v(2)x^{r-1} + \cdots + v(r+1)$$

```
function v=minimumpoly(A) [l,n]=size(A); if l~=n
    fprintf('This is not a square matrix')
    return
end

r=1; Aj=sym(eye(l)); B=Aj(:); BB(1,1)=sym(1); while BB(r,r)==1
    Aj=Aj*A;
    B=[B Aj(:)];
    BB=rref(B);
```

```
    r=r+1;
end v=[1 transpose(flip(-BB(1:r-1,r)))]; end
```

Where this code come from? We shall explain below the mathematics beyond the above code (see for example [4], [5]).

Definition 2.1. By P_A we denote all the monic polynomials $p(x)$ which are such that $p(A) = 0_{k\times k}$.

It is well known that there exists the minimum monic polynomial (which is unique) of the matrix A that belongs to P_A and the proof can be found at any linear algebra book (see for example [6], [7], [8], [9]). We denote the minimum polynomial of A by $m_A(x)$ and its degree by q_{m_A}.

Definition 2.2. By $b_A(q)$ we denote the vector which contain the matrix A^q column by column. By $b_A(0)$ we denote the vector which contain the identity matrix column by column. By $B_A(q)$ we denote the matrix with columns $b_A(0), \cdots, b_A(q)$.

Definition 2.3. By $C_A(q)$ we denote the following set, given $q \in \mathbb{N}$,

$$C_A(q) = \{c \in \mathbb{R}^q : B_A(q-1) \cdot c = -b_A(q)\}$$

Proposition 1. If $p(x) = x^q + c_{q-1}x^{q-1} + \cdots + c_0$ and $c = \begin{pmatrix} c_0 \\ \vdots \\ c_{q-1} \end{pmatrix}$ then

$$p(x) \in P_A \iff c \in C_A(q)$$

Proof. Let $p(x) \in P_A$. That means that

$$p(A) = A^q + c_{q-1}A^{q-1} + \cdots + c_0\mathbb{I}_{k\times k} = 0_{k\times k}$$

In other words

$$c_{q-1}A^{q-1} + \cdots + c_0\mathbb{I}_{k\times k} = -A^q \tag{2.1}$$

The above set of equations can be written at the following form

$$B_A(q-1) \cdot c = -b_A(q) \tag{2.2}$$

where $c = \begin{pmatrix} c_0 \\ \vdots \\ c_{q-1} \end{pmatrix}$. That is the systems (2.1) and (2.2) are equivalent and therefore the desired conclusion is true. $\square$

Definition 2.4. We denote by Q_A the following set of integers

$$Q_A = \{q \in \mathbb{N} : C_A(q) \neq \emptyset\}$$

Theorem 1. Let $m(x)$ be the minimum polynomial of A with degree q_{m_A}. Then

$$q_{m_A} = \min Q_A$$

Proof. The set Q_A is nonempty because for $q = k$ the $C_A(q)$ contains the coefficients of the characteristic polynomial. Therefore the $q^* = \min Q_A$ is well defined and of course $q^* \in Q_A$. That means that there exists a monic polynomial $p_{q^*}(x)$ of degree q^* which is such that $p_{q^*}(A) = 0_{k\times k}$.

We know that the minimum polynomial exists and is unique therefore $q_{m_A} \in Q_A$ so $q_{m_A} \geq q^*$. This immediately mean that $q_{m_A} = q^*$ using proposition 1.

Therefore in order to compute the minimum polynomial we have to find the minimum q for which the system $B_A(q-1) \cdot c = -b_A(q)$ has a solution.

We have compared the above `matlab` code with the integrated function `minpoly` and we have deduced that in many examples is much faster.

3 THE FUNCTION AND THE DRAZIN INVERSE OF THE MATRIX A

Below we shall give two equivalent definitions of the function of a matrix. Before that we define the matrix $v(A)$ where $v(\cdot)$ is a polynomial.

Definition 3.1. Let a polynomial $v(x) = a_q x^q + \cdots + a_0$ and a given matrix $A_{k\times k}$. The matrix $v(A)$ is defined to be the

$$v(A) = a_q A^q + \cdots + a_0 \mathbb{I}_{k\times k}$$

Definition 3.2 (Hermite Interpolation)**.** Let a matrix $A_{k\times k}$ and a function $f : U \subseteq \mathbb{R} \to \mathbb{R}$. The matrix $f(A)$ is defined to be the $v(A)$ where $v(\cdot)$ is the polynomial which satisfies the system 1.1, given a polynomial $p(x)$ which is such that $p(A) = 0_{k\times k}$ and $p_1, \cdots, p_l$ are the roots with multiplicities $j_1, \cdots, j_l$ of $p(x)$.

Definition 3.3 (Taylor Expansion)**.** Let a matrix $A_{k\times k}$ and a function $f : U \subseteq \mathbb{R} \to \mathbb{R}$. The matrix $f(A)$ is defined to be the $v(A)$ where the $v(\cdot)$ is the Taylor expansion of f around zero.

It is not always defined the matrix $f(A)$ as we have discussed in [14].

Definition 3.4. Let a function $f : \mathbb{R} \to \mathbb{R}$ and a polynomial $p(x)$ with $p_1, \cdots, p_l$ are the roots of multiplicities $j_1, \cdots, j_l$ of $p(x)$. We say that the function f is well defined at the roots of $p(x)$ if $f^{(k)}(p_i)$, for $i = 1, \cdots, l$ and $k = 0, \cdots, j_i - 1$ are well defined.

Proposition 2. Let a matrix $A_{k\times k}$ and set $\Delta = \max_{i,j} |A_{ij}|$. If the function $f : \mathbb{R} \to \mathbb{R}$ can be expanded in a Taylor series around zero which is absolute convergent for any $x \in [-M, M]$ with $M > k\Delta$ then the matrix $f(A)$ is well defined by definition 3.3. Let a polynomial $p(x)$ which is such that $p(A) = 0_{k\times k}$. Then the matrix $f(A)$ is well defined by the definition 3.2 if the function f is well defined at the roots of $p(x)$.

Proof. The matrix $f(A)$ via the Taylor expansion is defined as

$$f(A) = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!} A^n$$

This matrix is well defined because the infinite series converges. Indeed, define by $\Delta = \max_{i,j} |A_{ij}|$ we have that

$$|(A^n)_{ij}| \leq k^{n-1}\Delta^n$$

We will prove it by induction. For $n = 1$ is obvious that $|(A^1)_{ij}| \leq k^0\Delta$. We assume that it holds for some n, that is $|(A^n)_{ij}| \leq k^{n-1}\Delta^n$ and we will prove that it holds for $n + 1$. We have that

$$|(A^{n+1})_{ij}| = |\sum_{l=1}^{k}(A^n)_{il}A_{lj}| \leq \sum_{l=1}^{k} k^{n-1}\Delta^n A_{lj} \leq k^n\Delta^{n+1}$$

Therefore

$$\left|\sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!}(A^n)_{ij}\right| \leq f(0) + \left|\sum_{n=1}^{\infty} \frac{f^{(n)}(0)}{n!}(A^n)_{ij}\right| \leq f(0) + \sum_{n=1}^{\infty} \frac{|f^{(n)}(0)|}{n!} k^{n-1}\Delta^n$$

That means that $\sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!}(A^n)_{ij}$ converges absolutely and therefore the matrix $f(A)$ is well defined.

On the other hand it is easy to see that the matrix $f(A)$ via definition 3.2 is well defined as soon as the right hand side of 1.1 is well defined. □

Remark 1. It is easy to see that the matrices $f(A), g(A), h(A)$ are well defined for any matrix $A_{k\times k}$ for $f(x) = e^{tx}$, $g(x) = \cos(tx)$ and $h(x) = \sin(tx)$. □

Theorem 2. Let a function f and a matrix $A_{k\times k}$. Suppose that the matrix $f(A)$ is well defined for both definitions 3.3 and 3.2. Then the two matrices are equal.

Proof. Let q the degree of the polynomial $p(x)$ which is such that $p(A) = 0_{k\times k}$ and the roots $p_1, \cdots, p_l$ with multiplicities $j_1 \cdots, j_l$. For any $m \in \mathbb{N}$ with $m \geq q$ there are $a_0(m), \cdots, a_{q-1}(m)$ such that

$$A^m = a_{q-1}(m)A^{q-1} + \cdots + a_0(m)\mathbb{I}$$

Then it is easy to see that,

$$\begin{aligned} f^{Taylor}(A) &= A^{q-1}\underbrace{\left(\sum_{n=q}^{\infty}\frac{f^{(n)}(0)}{n!}a_{q-1}(n) + \frac{f^{(q-1)}(0)}{(q-1)!}\right)}_{b_{q-1}} \\ &+ \cdots + \mathbb{I}\underbrace{\left(\sum_{n=q}^{\infty}\frac{f^{(n)}(0)}{n!}a_0(n) + \frac{f^{(0)}(0)}{0!}\right)}_{b_0} \end{aligned}$$

In order to get the above result we have rearranged the terms and this can be done since the series is absolutely convergent.

We shall see below that the same coefficients $b_0, \cdots, b_{q-1}$ will appear if we compute the matrix $f(A)$ via the Hermite interpolation. Computing A^m for $m \geq q$ the system 1.1 will contain the following equations, for $i = 1, \cdots, l$,

$$\begin{aligned} a_{q-1}(m)p_i^{q-1} + \cdots + a_0(m) &= p_i^m \\ a_{q-1}(m)(q-1)p_i^{q-2} + \cdots + a_1(m) &= mp_i^{m-1} \\ &\vdots \\ a_{q-1}(m)(q-1)\cdots(q-j_i+1)p_i^{q-j_i} + \cdots + a_{j_i-1}(m) &= \\ m\cdot(m-1)\cdots(m-j_i+2)p_i^{m-j_i+1} \end{aligned}$$

We multiply these equations by $\frac{f^{(m)}(0)}{m!}$ and then we sum from $m = q$ to ∞. Therefore the above system will contain the following equations, for $i = 1, \cdots, l$,

$$\begin{aligned} p_i^{q-1}\sum_{m=q}^{\infty}a_{q-1}(m)\frac{f^{(m)}(0)}{m!} + \cdots + \sum_{m=q}^{\infty}a_0(m)\frac{f^{(m)}(0)}{m!} &= \sum_{m=q}^{\infty}p_i^m\frac{f^{(m)}(0)}{m!} \\ (q-1)p_i^{q-1}\sum_{m=q}^{\infty}a_{q-1}(m)\frac{f^{(m)}(0)}{m!} + \cdots + \sum_{m=q}^{\infty}a_1(m)\frac{f^{(m)}(0)}{m!} &= \sum_{m=q}^{\infty}mp_i^{m-1}\frac{f^{(m)}(0)}{m!} \\ &\vdots \\ (q-1)\cdots(q-j_i+1)p_i^{q-j_i}\sum_{m=q}^{\infty}a_{q-1}(m)\frac{f^{(m)}(0)}{m!} + \cdots + \\ + \sum_{m=q}^{\infty}a_{j_i-1}(m)\frac{f^{(m)}(0)}{m!} &= \\ \sum_{m=q}^{\infty}m\cdot(m-1)\cdots(m-j_i+2)p_i^{m-j_i+1}\frac{f^{(m)}(0)}{m!} \end{aligned}$$

We add the corresponding terms in each equation so the right hand side equal to $f(p_i)$, $f^{'}(p_i)$, $\cdots$, $f^{(j_i-1)}(p_i)$. Then the unknown coefficients of the above system are the $b_0, \cdots, b_{q-1}$ and therefore we obtain the desired result.

The `matlab` function for this computation is `matfun(A,f)` and can be found at the following link https://www.mathworks.com/matlabcentral/fileexchange/132518-computing-the-minimum-polynomial-of-a-matrix. At this link one can find several auxiliary functions and examples.

The Drazin inverse of a matrix can be used at the context of differential equations, Markov chains and others (see for example [10], [11], [12], [13]).

Theorem 3. Let a given matrix $A_{k\times k}$ and let $a_{ij}(n)$ be the elements of A^n. Then the matrix B which has as its elements the numbers $a_{ij}(-1)$ is the Drazin inverse of the matrix A.

Proof. See remark 3.2 of [14]. Therefore, one way to compute the Drazin inverse of a matrix is to compute the nth power of the matrix and then set $n = -1$. Of course if the matrix is invertible then setting $n = -1$ we will get its inverse.

Remark 2. In all the above computations we can use the characteristic polynomial, the minimum polynomial or any polynomial $p(x)$ with the property $p(A) = 0_{k\times k}$. The result will be the same as we have seen in [14].

Indeed, let $v(x)$ and $\hat{v}(x)$ be the polynomials that satisfies system 1.1 using the characteristic polynomial to get $v(x)$ and a polynomial $p(x)$ (with $p(A) = 0_{k\times k}$) to get $\hat{v}(x)$. Note that the polynomial $v(x)$ is of degree $k-1$ and suppose that the polynomial $p(x)$ is of degree less or equal $k-1$. Dividing $v(x)$ by $p(x)$ we get

$$v(x) = \pi(x)p(x) + q(x)$$

Using the roots (and their multiplicities) of $p(x)$ we can compute the polynomial $q(x)$ and see that $q(x) = \hat{v}(x)$. Moreover, $\hat{v}(A) = v(A) = f(A)$.

Remark 3. Under the above point of view two things can be changed in order to increase the accuracy and decrease the time of computations. The first is to find a better way to compute the minimum polynomial and the second is to find a better way to solve the confluent Vandermonde linear system (see for example [15], [16]). □

4 EXAMPLES

We will compute the nth power of a non invertible matrix and then its Drazin inverse. Let the matrix

$$C = \begin{pmatrix} \frac{1}{5} & \frac{2}{5} & \frac{2}{5} \\ \frac{3}{10} & \frac{3}{5} & \frac{1}{10} \\ \frac{1}{10} & \frac{1}{5} & \frac{7}{10} \end{pmatrix}$$

Note that C is not invertible. Setting $f(x) = x^n$ we can use our function `matfun(C,f)` in order to compute the nth power of C. We get the following result

$$C^n = \begin{pmatrix} \frac{1}{5} & \frac{2}{5} & \frac{2}{5} \\ \frac{2^n+1}{5\,2^n} & \frac{2\,(2^n+1)}{5\,2^n} & \frac{2\,2^n-3}{5\,2^n} \\ \frac{2^n-1}{5\,2^n} & \frac{2\,(2^n-1)}{5\,2^n} & \frac{2\,2^n+3}{5\,2^n} \end{pmatrix}$$

Setting $n = -1$ we get the Drazin inverse of the matrix C which is the following

$$C_D = \begin{pmatrix} \frac{1}{5} & \frac{2}{5} & \frac{2}{5} \\ \frac{3}{5} & \frac{6}{5} & -\frac{4}{5} \\ -\frac{1}{5} & -\frac{2}{5} & \frac{8}{5} \end{pmatrix}$$

Using our function `isdrazin` we verify that the matrix C_D is indeed the Drazin inverse of C. We have taken care in our functions so that we can let $n = -1$ in order to get the Drazin inverse. This is not the case concerning the `matlab` integrated function `funm`.

Finally we will compute the nth power and the exponential of a matrix with complex eigenvalues. Let the matrix A

$$A = \begin{pmatrix} 3 & 2 \\ -1 & 3 \end{pmatrix}$$

Setting $f(x) = x^n$ and $g(x) = exp(t * x)$ and typing `matfun(A,f)` and `matfun(A,g)` we get the following results

$$A^n = \begin{pmatrix} 11^{n/2} \cos\left(n \operatorname{atan}\left(\frac{\sqrt{2}}{3}\right)\right) & \sqrt{2}\, 11^{n/2} \sin\left(n \operatorname{atan}\left(\frac{\sqrt{2}}{3}\right)\right) \\ -\frac{\sqrt{2}\, 11^{n/2} \sin\left(n \operatorname{atan}\left(\frac{\sqrt{2}}{3}\right)\right)}{2} & 11^{n/2} \cos\left(n \operatorname{atan}\left(\frac{\sqrt{2}}{3}\right)\right) \end{pmatrix}$$

and

$$e^{tA} = \begin{pmatrix} e^{3t} \cos\left(\sqrt{2}\, t\right) & \sqrt{2}\, e^{3t} \sin\left(\sqrt{2}\, t\right) \\ -\frac{\sqrt{2}\, e^{3t} \sin\left(\sqrt{2}\, t\right)}{2} & e^{3t} \cos\left(\sqrt{2}\, t\right) \end{pmatrix}$$

In our matlab functions we have transformed the complex roots of the minimum polynomial into their polar forms in order to arrive at the above results. This is not the case with the `matlab` integrated function `funm`.

Our functions `isan(an,a)` and `isexpt(at,a)` verify that the matrices `an` and `at` are indeed the nth power of `a` and the exponential matrix of `a`. For the first case we use induction while for the second case we use the fact that the matrix e^{tA} is the only matrix which satisfies the following matrix differential equation

$$\begin{aligned} (M(t))' &= AM(t) \\ M(0) &= \mathbb{I} \end{aligned}$$

The fact that the above matrix differential equation has a unique solution can be proved as follows. It is easy to see that the exponential matrix e^{tA} solves the above matrix differential equation by using the Taylor expansion and differentiating term by term since the Taylor series converges absolutely. Next, if there are two or more solutions then their difference will satisfy the following

$$\begin{aligned} (M(t))' &= AM(t) \\ M(0) &= \mathbf{0} \end{aligned}$$

But $(M(t)e^{-tA})' = M'(t)e^{-tA} - AM(t)e^{-tA} = \mathbf{0}$. Therefore $M(t)e^{-tA} = \mathbf{C}$ where $\mathbf{C}$ is a matrix independent of t. Multiplying by e^{tA}, using the fact that $e^A e^B = e^{A+B}$ for commutative matrices A, B and since $M(0) = \mathbf{0}$ it follows that $M(t) = \mathbf{0}$ for any $t \in \mathbb{R}$.

Below we compare our functions with the integrated functions of `Matlab` for large sparse matrices. We see that increasing the dimension of the matrix the ratio of times increases as well.

```
clear; syms x n t f(x)=x^n; g(x)=exp(t*x);
```

```
 K=45;
 H=zeros(K,K);
H(1,1)=1; for i=2:K H(i,i)=0.7; H(i,1)=0.3; end H(4,1)=0.1;
H(4,2)=0.1; H(4,3)=0.2; H(6,7)=0.4;

%Comparing the nth power and the exponential
tic matfun(H,f); matfun(H,g);
 t3=toc;

tic simplify(funm(H,f)); simplify(funm(H,g)); t4=toc; t4/t3

%comparing the minimum polynomial computation
tic ppH=poly2sym(minimumpoly(H),x); pH=solve(ppH==0,x); t1=toc

tic ppmH=poly2sym(minpoly(H),x); pmH=solve(ppH==0,x); t2=toc

t2/t1
```

The comparison concerning the computation of the minimum polynomial gives the result $t2/t1 = 348.17$.

The comparison concerning the computation of the nth power and the exponential matrix gives the following $t4/t3 = 1.39$ using the `matlab online`. We believe that the reason that we get better results is the use of the minimum polynomial in order to do these computations for sparse matrices.

5 CONCLUSION

In this paper we have study the problem of the computation of the minimum polynomial, the function of a matrix A and the Drazin inverse. Our first goal is to give simple but rigorous proofs for the above and the second goal is to give the `matlab` codes. We have compared our code with the integrated `matlab` code and see that in many cases our code is much faster. Moreover, in the case where the matrix A has complex roots our code give a prettier result.

COMPETING INTERESTS

Author has declared that no competing interests exist.

REFERENCES

[1] Flowe R, Harris G. A note on genelalized Vandermonde determinants. SIAM J. Matrix Anal. Appl. 1983;14(4):1146-1151.

[2] Higham N. Functions of matrices, theory and computation. SIAM; 2008.

[3] Moler C, Van Loan C. Nineteen dubious ways to compute the exponential of a matrix, twenty - five years later. SIAM Review; 2003.

[4] Bialas S, Bialas M. An algorithm for the calculation of the minimal polynomial. Bulletin Polish Academy Sciences. 2008;56.

[5] Shoup V. Efficient computation of minimal polynomials in algebraic extensions of finite fields; 1999.

[6] Berberian S. Linear algebra. Oxford Science Publications; 1992.

[7] Larson R, Falvo D. Elementary linear algebra. Houghton Mifflin Harcourt Publishing Company; 2009.

[8] Lipschutz S, Lipson M. Linear algebra. McGraw Hill; 2009.

[9] Ricardo H. A modern introduction to linear algebra. CRC Press; 2010.

[10] Cambell S, Meyer C. Generelized inverses of linear Trasformations. SIAM; 2009.

[11] Stephen L. Campbell, Carl D. Meyer, Jr., Nicholas J. Rose. Applications of the Drazin inverse to linear systems of differential equations with singular constant coefficients. SIAM Journal on Applied Mathematics. 1976;31.

[12] Israel A, Grevill T. Genelalized inverses, theory and applications. Springer; 2003.

[13] Penrose R. A generalized inverse for matrices. Mathematical Proceedings of the Cambridge Philosophical Society. 1955;51:406-413.

[14] Halidias N. On the computation of the minimum polynomial and applications. Asian Research Journal of Mathematics; 2022. Avalable:https://www.researchgate.net/publication/364617226_

[15] Respondek J. Numerical recipes for the high efficient inverse of the confluent Vandermonde matrices. App. Math. Comp. 2011;218(5):2044-2054.

[16] Respondek J. Matrix black box algorithms – a survey. Bulletin of the Polish Academy of Sciences. 2022;70(2):e140535.

Peer-review history:
The peer review history for this paper can be accessed here:
https://www.sdiarticle5.com/review-history/112734