Bull. Math. Soc. Sci. Math. Roumanie
Tome 56(104) No. 2, 2013, 217–228

Computing Minimal Polynomial of Matrices over Algebraic Extension Fields

by
Amir Hashemi and Benyamin M.-Alizadeh

Abstract

In this paper, we present a new and efficient algorithm for computing minimal polynomial of matrices over algebraic extension fields using the Gröbner bases technique. We have implemented our algorithm in Maple and we evaluate its performance and compare it to the performance of the function `MinimalPolynomial` of Maple 15 and also of the Białas algorithm as a new algorithm to compute minimal polynomial of matrices.

Key Words: Minimal Polynomial, Gröbner Bases, Algebraic Extension Fields.
2010 Mathematics Subject Classification: Primary 15A15,
Secondary 13P10.

Introduction

It is well known from the Cayley Hamilton theorem that any matrix over real numbers satisfies its characteristic equation, i.e., for any matrix $A_{n\times n}$ over real numbers, if $f(s) = \det(sI_r - A) = s^n + a_{n-1}s^{n-1} + \cdots + a_0$, then $f(A) = 0$. There is another polynomial known as the *minimal polynomial*, say $m(s)$, such that $m(A) = 0$. This is the least degree monic polynomial which satisfies the equation $m(A) = 0$. The reason behind the interest in computing the minimal polynomial of a matrix is its applications in solving a system of polynomial equations, polynomial factorization, cryptography, effective Galois theory and so on.

A classical approach to compute the minimal polynomial of a matrix $A_{n\times n}$ is to determine the first matrix A^k for which $\{I, A, A^2, \ldots, A^k\}$ is linearly dependent. Let k be the smallest positive integer such that $A^k = \sum_{i=0}^{k-1} \alpha_i A^i$, then the minimal polynomial of A is $m(s) = s^k - \alpha_{k-1}s^{k-1} - \cdots - \alpha_1 s - \alpha_0$. The Gram-Schmidt orthogonalization procedure with the standard inner product is the perfect theoretical tool for determining k and the α_i's (see [10], page 643). For other algorithms to compute the minimal polynomial of a constant matrix, see [2, 3] for example.

In this paper, we present a new and efficient algorithm for computing the minimal polynomial of a matrix $A_{n\times n}$ over a finite algebraic extension field F (of field of rational numbers or finite fields) by using the *Gröbner bases* technique. Indeed, the reason for which we employ Gröbner bases is that the extension field F may be defined by a polynomial ring modulo a maximal ideal I, and so, Gröbner bases technique can be utilized to perform the computations in this polynomial ring. In order to explain our method, let $m(s) = a_n s^n + a_{n-1}s^{n-1} + \cdots + a_0$ be the minimal polynomial of A where the a_i's are to be found (note that some of them may be zero). From $m(A) = 0$ we derive n^2 equations. Adding these equations to I and computing the Gröbner bases of this new ideal, we find a simple representation for the equations. These new relations allows us to compute efficiently the a_i's and therefore the minimal polynomial of A.

The structure of this paper is as follows. Section 1 is devoted to a short description of the algorithm stated in [3] which computes the minimal polynomial of matrices. Throughout this paper, we refer to this algorithm as *Białas algorithm*. In Section 2, we state our main results for computing the minimal polynomial of a matrix over an algebraic extension field. Section 3 is devoted to the description of our new algorithm, and illustration of its behaviour with an example. In Section 4, we compare our algorithm with the function `MinimalPolynomial` of Maple 15 and also with the Białas algorithm via some examples. Finally, Section 5 presents an evaluation of our algorithm in the special case when its input is a matrix over the field of rational numbers.

1 Białas algorithm

In this section, we review briefly the Białas algorithm [3] to compute the minimal polynomial of matrices. This algorithm employs only elementary row operations to compute the coefficients of the minimal polynomial of a matrix. To be more precisely, let F be a field and $M_{n\times n}$ denote the set of all $n \times n$ matrices over F where n is a natural number. Furthermore, the notions $0_{n\times n}$ and $I_{n\times n}$ denote zero and identity matrices, respectively. Now let $A \in M_{n\times n}$, and assume that $m(x) = x^k + \lambda_{k-1}x^{k-1} + \cdots + \lambda_0$ is the minimal polynomial of A where the λ_i's are to be computed. According to the definition of the minimal polynomial, we have $m(A) = 0_{n\times n}$. Thus, $\Lambda = (\lambda_0, \dots, \lambda_{k-1}, 1, 0, \dots, 0) \in F^{n+1}$ is a zero of the equation

$$x_0A^0 + \cdots + x_nA^n = 0$$

in the x_i's if we set x_i equals to the i-th element of Λ. Hence, the λ_i's (and therefore the minimal polynomial of A) may be calculated if one solves the above matrix equation. In doing so, for $\ell = 0, \dots, n$, let $A^\ell = (a^\ell_{i,j})_{n\times n}$ where $1 \le i, j \le n$. Expanding the above equation, we obtain the following n^2 equations:

$$\begin{cases} a^0_{1,1}x_0 + \cdots + a^n_{1,1}x_n & = & 0 \\ a^0_{1,2}x_0 + \cdots + a^n_{1,2}x_n & = & 0 \\ \vdots & \vdots & \vdots \\ a^0_{n,n}x_0 + \cdots + a^n_{n,n}x_n & = & 0. \end{cases}$$

This linear system is homogeneous for which the matrix coefficient is

$$\begin{bmatrix} a^0_{1,1} & \cdots & a^n_{1,1} \\ a^0_{1,2} & \cdots & a^n_{1,2} \\ \vdots & \ddots & \vdots \\ a^0_{n,n} & \cdots & a^n_{n,n} \end{bmatrix}_{n^2\times(n+1)}$$

where the i-th column corresponds to A^i. Therefore, to compute the minimal polynomial of A, it is enough to construct the above matrix and perform the Gaussian elimination on it. Then, the minimal polynomial of A is obtained by solving the system associated to the resulting matrix and by setting the free variables to be zero.

Remark 1. *The Białas algorithm works when the base field is an algebraic extended field, since the Gaussian elimination could be performed over algebraic extension fields, as well.*

2 Statement of the main result

In this section, we state our main result for computing the minimal polynomial of a matrix which is based on the use of Gröbner bases technique. We recall first the basic definition of Gröbner bases. The notion of Gröbner bases was introduced by B. Buchberger, who gave the first algorithm to compute it (see [4, 5, 6]). This algorithm has been implemented in most general computer algebra systems like Maple, Mathematica, Singular, Macaulay2 and CoCoA.

Let $R = K[x_1, \dots, x_n]$ be a polynomial ring where K is an arbitrary field. Let $f_1, \dots, f_k \in R$ be a sequence of k polynomials and let $I = \langle f_1, \dots, f_k \rangle$ be the ideal of R generated by the f_i's. We need also a monomial ordering on R. We recall here the definition of *lexicographical ordering* (lex), denoted by $\prec_{lex}$, which is a special monomial ordering having some interesting properties. For this we denote by $\deg_i(m)$ the degree in x_i of a monomial m. If m and m' are monomials, then $m \prec m'$ if and only if the first non-zero entry of the vector $(\deg_1(m') - \deg_1(m), \dots, \deg_n(m') - \deg_n(m))$ is positive (see [7]).

Let us fix a monomial ordering $\prec$ on R. The *leading monomial* of a polynomial $f \in R$ is the greatest monomial (w.r.t. $\prec$) which appears in f, and we denote it by $\mathrm{LM}(f)$. The *leading coefficient* of f, written $\mathrm{LC}(f)$, is the coefficient of $\mathrm{LM}(f)$ in f. The *leading term* of f is $\mathrm{LT}(f) = \mathrm{LC}(f)\mathrm{LM}(f)$. The *leading term ideal* of I is defined as

$$\mathrm{LT}(I) = \langle \mathrm{LT}(f) \mid f \in I \rangle.$$

Definition 1. *A finite set $\{g_1, \dots, g_s\} \subset I$ is a Gröbner basis of I w.r.t. $\prec$ if* $\mathrm{LT}(I) = \langle \mathrm{LT}(g_1), \dots, \mathrm{LT}(g_s) \rangle$.

A Gröbner basis $G \subset I$ is called *reduced* if the leading coefficient of each element of G is 1 and no monomial in any element of G is in $\mathrm{LT}(I)$. It is worth noting that every non-zero ideal I has a unique reduced Gröbner basis, see [7] page 92.

Since Maple has implementation of the Faugère's F_4 algorithm (see `Groebner` package and [8]), and this algorithm has a good performance over rational field $\mathbb{Q}$ and finite fields, we present then our method for finite algebraic extensions of these fields. For simplification of notations, in the sequel we consider $\mathbb{Q}$ as the base field, and our results hold for finite fields. By a finite algebraic extension field F of $\mathbb{Q}$, we mean the field $\mathbb{Q}(\alpha_1, \dots, \alpha_k)$ where the α_i's are algebraic over $\mathbb{Q}$. Thus, we can associate to each α_i a monic polynomial f_i with the coefficient in $\mathbb{Q}$ such that $f_i(\alpha_i) = 0$. We call f_i the minimal polynomial of α_i over $\mathbb{Q}$. According to Kronecker's construction, we have the $\mathbb{Q}$-algebra homomorphism

$$\phi : \mathbb{Q}[x_1, \dots, x_k] \to \mathbb{Q}(\alpha_1, \dots, \alpha_k)$$

defined by $x_i \mapsto \alpha_i$ where $\mathrm{Ker}(\phi) = \langle f_1, \ldots, f_k \rangle$. We denote this ideal by I which is a maximal ideal. Thus, $\mathbb{Q}(\alpha_1, \ldots, \alpha_k) = \mathbb{Q}[x_1, \ldots, x_k]/I$. We will assume that the ideal I is represented by its Gröbner basis w.r.t. a monomial ordering which is denoted by $\prec_I$. For more details on the relation of the Gröbner bases to the algebraic extension fields, we refer to [1].

Let A be an $n \times n$ matrix over F and $m(s) = a_n s^n + a_{n-1}s^{n-1} + \cdots + a_0$ be its minimal polynomial where the a_i's are to be computed. From $m(A) = 0$, we can derive n^2 algebraic relations between the a_i's. Computing the Gröbner bases of the ideal constructed by adding these equations to I, we can simplify these equations and compute then the minimal polynomial of A. Now, we state our main result of this paper in which $|X|$ denotes the size of a set X.

Theorem 1. *Using the above notations, let G be the reduced Gröbner basis of the ideal in $\mathbb{Q}[x_1, \ldots, x_k, a_0, \ldots, a_n]$ generated by $f_1, \ldots, f_k$ and the n^2 polynomials obtained from $m(A) = 0$ w.r.t. $\prec$ such that $a_0 \prec_{lex} \cdots \prec_{lex} a_n$, $x_i \prec_{lex} a_j$ for any i and j, and $\prec$ over the x_i's is equivalent to $\prec_I$. Let $d = |G| - k$ and r be the remainder of the division of $m_1(s) = m(s)|_{a_{d+1}=\cdots=a_n=0}$ by $G_1 = G|_{a_{d+1}=\cdots=a_n=0}$. Dividing r by its leading coefficient yields the minimal polynomial of A.*

Proof: Note that the equations come from $m(A) = 0$ are linear polynomials in the a_i's. Therefore, using the homomorphism ϕ, we can associate a matrix S over F to this system. So computing the Gröbner basis of the ideal generated by $f_1, \ldots, f_k$ and the n^2 polynomials obtained from $m(A) = 0$, w.r.t. $\prec$ is equivalent to performing a Gaussian elimination over F on S. On the other hand, since $f_1, \ldots, f_k$ is a Gröbner basis, G contains $f_1, \ldots, f_k$ (by the definition of $\prec$). These follow that the rank of S is d, and the degree of the minimal polynomial of A is also d. Indeed, for each i, if the coefficient of s^i in the minimal polynomial of A is zero then a_i appears in G. In the rest of the proof, without loss of generality, suppose that a_0 is the first non-zero coefficient.

We use now the fact that A has a unique minimal polynomial. Therefore, we can impose the conditions on the a_i's such that $m(s)$ is unique. For this, we substitute $a_{d+1}, \ldots, a_n$ by 0 in $m(s)$ and G. We can see easily that G_1 remains a Gröbner basis. This is followed from the definition of $\prec$ and the fact that $f_1, \ldots, f_k$ is a Gröbner basis w.r.t. $\prec_I$. This implies that r is unique (see [7], Proposition 1, page 82). Since we use G_1 to compute the normal form of $m_1(s)$ by G_1, we can suppose that G_1 is reduced. So, we can conclude that each polynomial in G_1 is a binomial (in a_i and a_0) and all coefficients of r are divisible by a_0. Therefore, by the existence and uniqueness of the minimal polynomial of A, if we divide r by its leading coefficient we obtain the minimal polynomial of A. □

3 Description of the new algorithms

In this section, we describe our new algorithm (based on Theorem 1) for computing the minimal polynomial of a matrix over an algebraic extension of $\mathbb{Q}$. This section includes also an example which illustrate the behaviour of this algorithm.

As we have shown in the proof of Theorem 1, to compute the minimal polynomial of A, we need to calculate the multiplicative inverse of an algebraic number. So, our first objective is to do this using Gröbner bases technique. The following simple lemma may be seen as a corollary of [1], Theorem 2.6.3.

Lemma 1. *Let $F = \mathbb{Q}(\alpha_1, \ldots, \alpha_k) = \mathbb{Q}[x_1, \ldots, x_k]/I$ be an algebraic extension field of $\mathbb{Q}$ where $I = \langle f_1, \ldots, f_k \rangle$ is a maximal ideal. Let $\sigma = f(\alpha_1, \ldots, \alpha_k) \in F$ for some $f \in \mathbb{Q}[x_1, \ldots, x_k]$ and G be the Gröbner basis of the ideal $J = \langle f_1, \ldots, f_k, yf(x_1, \ldots, x_k) - 1 \rangle$ w.r.t. $\prec$ where y is a new variable and $x_1 \prec_{lex} \cdots \prec_{lex} x_n \prec_{lex} y$. Then there exist a polynomial $g \in \mathbb{Q}[x_1, \ldots, x_k]$ such that $y - g \in G$ and $g(\alpha_1, \ldots, \alpha_k)$ is the inverse of σ.*

The above lemma, provides a novel algorithm to compute the inverse of an algebraic number, as follow.

Algorithm 1 AlgebraicInverse

Require: G_0 a Gröbner basis for the maximal ideal I and $f(\alpha_1, \ldots, \alpha_k)$ an algebraic number
Ensure: The algebraic inverse of $f(\alpha_1, \ldots, \alpha_k)$
$J := \langle G_0, yf(x_1, \ldots, x_k) - 1 \rangle$
$G :=$ The Gröbner basis for J w.r.t. the lexicographical ordering where y is the greatest variable
Find $g(x_1, \ldots, x_k)$ such that $y - g(x_1, \ldots, x_k) \in G$
Return $g(\alpha_1, \ldots, \alpha_k)$

Example 1. *In this example we compute the inverse of $-\alpha_2^3 + \alpha_2^2 + \alpha_2 + 1 \in \mathbb{Z}_5(\alpha_1, \alpha_2) = \mathbb{Z}_5[t_1, t_2]/\langle t_1^2+1, t_2^2+t_1 \rangle$, using* AlgebraicInverse *algorithm. For this, we compute a Gröbner basis for the ideal $\langle t_1^2 + 1, t_2^2 + t_1, y(-t_2^3 + t_2^2 + t_2 + 1) - 1 \rangle$ w.r.t. the monomial ordering $t_1 \prec_{lex} t_2 \prec_{lex} y$, which is equal to*

$$\{t_1^2 + 1, t_2^2 + t_1, 2y - t_1 - t_2\}.$$

Therefore, the inverse of $-\alpha_2^3 + \alpha_2^2 + \alpha_2 + 1$ is $\frac{1}{2}(\alpha_1 + \alpha_2)$.

Noro in [11], has presented another method for computing the inverse of an algebraic number and it seems (empirically) that his method need more computations than ours. To explain this method, let $\mathrm{NF}_G(f)$ be the remainder of the division of f by G. Under the assumption of Lemma 1, let $B = \{v_1, \ldots, v_t\}$ be a basis for the $\mathbb{Q}$-vector space $\mathbb{Q}[x_1, \ldots, x_k]/I$. Then the inverse of σ is $\sum_{i=1}^t c_i v_i$, where $c_i \in \mathbb{Q}$ satisfying $\sum_{i=1}^t c_i \mathrm{NF}_G(f v_i) = 1$. The following tables compare the efficiency of Algorithm 1 with Noro's algorithm over a field of characteristic p where p is the first prime number greater than 2^{32}. In doing so, we computed the algebraic inverse of $\sum_{i=1}^n \alpha_i$ in the field $\mathbb{Z}_p(\alpha_1, \ldots, \alpha_n) = \mathbb{Z}_p[t_1, \ldots, t_n]/\langle t_1^2 - p_1, \ldots, t_n^2 - p_n \rangle$ where n is a natural number and p_i is the i-th prime number for $i = 1, \ldots, n$. The results are shown in the following tables where the timings (both here and in the other sections) are conducted on a personal computer with 3.2GHz, Intel(R)-Core(TM), i7 CPU, 6 GB RAM and 64 bits under the windows 7 operating system. The time (resp. memory) column shows the CPU time in seconds consumed (resp. amount of gigabytes of memory used) by the corresponding algorithm.

$n=4$	time	memory
AlgebraicInverse	0.00	0.00
Noro	0.03	0.00

$n=6$	time	memory
AlgebraicInverse	0.00	0.00
Noro	0.61	0.03

$n=8$	time	memory
AlgebraicInverse	0.00	0.00
Noro	22.17	0.73

$n=10$	time	memory
AlgebraicInverse	357.56	32.93
Noro	1279.05	17.84

As this tables show Algorithm 1 needs less time and memory than Noro's algorithm. Based on our practical experiences we therefore decided to use Algorithm 1 for the next calculations. We present now our algorithm (based on Theorem 1) for computing the minimal polynomial of a matrix over a finite algebraic extension of $\mathbb{Q}$.

Algorithm 2 MinPoly

Require: $A_{n\times n}$ a matrix, and G_0 a Gröbner basis for the maximal ideal I
Ensure: Minimal polynomial $m(s)$ of A
$J := \langle \sum_{k=0}^{n} a_k A^k[i,j] \mid i,j = 1,\dots,n\rangle$
$G :=$ The Gröbner basis for $\langle G_0 \cup J\rangle$ w.r.t. the lexicographical ordering
$d := \mid G \mid - \mid G_0 \mid$
$G_1 := G|_{a_{d+1}=\dots=a_n=0}$
$m :=$ Remainder$(\sum_{i=0}^{d} a_i s^i, G_1)$
$\sigma :=$ AlgebraicInverse(LC(m))
$m := \sigma m$
Return m

The following example shows the computation of the minimal polynomial of a matrix using the above algorithms.

Example 2. *We would like to compute the minimal polynomial of the following matrix over the field $\mathbb{Z}_5(\alpha_1,\alpha_2) = \mathbb{Z}_5[t_1,t_2]/\langle t_1^2+1, t_2^2+t_1\rangle$.*

$$A = \begin{bmatrix} \alpha_1 & 1 & 0 \\ \alpha_1+\alpha_2 & 2 & 1 \\ 1 & 3 & \alpha_1\alpha_2+1 \end{bmatrix}$$

Let $m(s) = a_3s^3 + a_2s^2 + a_1s + a_0$ be a polynomial vanishing on A. After computing the matrices A^2 and A^3, we have the following polynomials from $m(A) = 0$.

$$
\begin{array}{lll}
f_1 & := & a_0+\alpha_1 a_1+(\alpha_1+\alpha_2-1)a_2+(\alpha_1+2\alpha_2+2\alpha_1\alpha_2-1)a_3 \\
f_2 & := & a_1+(\alpha_1+2)a_2+(6+3\alpha_1+\alpha_2)a_3 \\
f_3 & := & a_2+(\alpha_1\alpha_2+\alpha_1+3)a_3 \\
f_4 & := & (\alpha_2+\alpha_1)a_1+(\alpha_1\alpha_2+2\alpha_2+2\alpha_1)a_2+(5\alpha_1\alpha_2+6\alpha_2+6\alpha_1)a_3 \\
f_5 & := & a_0+2a_1+(\alpha_1+\alpha_2+7)a_2+(4\alpha_1+4\alpha_2+4\alpha_1\alpha_2+23)a_3 \\
f_6 & := & a_1+(\alpha_1\alpha_2+3)a_2+(2\alpha_1+\alpha_2+4\alpha_1\alpha_2+10)a_3 \\
f_7 & := & a_1+(4\alpha_1+3\alpha_2+\alpha_1\alpha_2+1)a_2+(12\alpha_1+6\alpha_2+5\alpha_1\alpha_2+3)a_3 \\
f_8 & := & 3a_1+(3\alpha_1\alpha_2+10)a_2+(7\alpha_1+3\alpha_2+13\alpha_1\alpha_2+33)a_3 \\
f_9 & := & a_0+(\alpha_1\alpha_2+1)a_1+(\alpha_1+2\alpha_1\alpha_2+4)a_2+(3\alpha_1-\alpha_2+9\alpha_1\alpha_2+14)a_3.
\end{array}
$$

Let G be the Gröbner basis of $\langle f_1,\dots,f_9,t_1^2+1,t_2^2+t_1\rangle$ w.r.t. $t_1 \prec_{lex} t_2 \prec_{lex} a_0 \prec_{lex} \cdots \prec_{lex} a_3$. Since $|G|=5$, then we keep all the coefficient of $m(s)$. The remainder of the division of $m(s)$ by G, after dividing it by a_0, is equal to

$$m_1(s)=(-\alpha_2^3+\alpha_2^2+\alpha_2+1)s^3+(-\alpha_2^2-3\alpha_2)s^2-(2\alpha_2^2+3\alpha_2-1)s+1.$$

Now, As mentioned in Example 1 the inverse of $-\alpha_2^3+\alpha_2^2+\alpha_2+1$ is $\frac{1}{2}(\alpha_1+\alpha_2)$ and by multiplying $m_1(s)$ by $\frac{1}{2}(\alpha_1+\alpha_2)$ the minimal polynomial of A is equal to

$$s^3-(\alpha_1\alpha_2+\alpha_1+3)s^2+(2\alpha_1\alpha_2+2\alpha_1-2\alpha_2-1)s+2\alpha_1+2\alpha_2.$$

Remark 2. *It should be noted that since we use Gröbner bases technique, and its computation has double exponential worst-case complexity (by the well-known example due to Mayr and Meyer [9]), then our algorithm has the same complexity to compute the minimal polynomial of a matrix over an algebraic extension field.*

4 Experiments and results

We have implemented the MinPoly and Białas algorithms [1] in Maple 15 to compare with the function `MinimalPolynomial` from `LinearAlgebra` package of Maple 15. The results are shown in the following tables. These tables show the result of running the MinPoly and Białas algorithms and `MinimalPolynomial` of Maple 15 for six random matrices $A_{n\times n}$ with $n\in\{10,12,14,16,18,20\}$ and $A[1,1]=\sqrt[2]{2}/3$ and $A[2,2]=\sqrt[3]{3}/2$ and other entries of A are random integers.

[1]The Maple code of our programs are available at `http://amirhashemi.iut.ac.ir/software.html`

10 × 10	time	memory
MinPoly	4.59	0.33
Maple	7.31	0.63
Białas	49.17	3.10

12 × 12	time	memory
MinPoly	8.03	0.51
Maple	21.87	2.00
Białas	474.80	27.36

14 × 14	time	memory
MinPoly	17.24	0.80
Maple	56.80	5.31
Białas	2236.55	123.12

16 × 16	time	memory
MinPoly	55.58	1.24
Maple	125.25	11.74
Białas	> 8 h	∞

18 × 18	time	memory
MinPoly	244.27	2.07
Maple	270.93	23.29
Białas	> 8 h	∞

20 × 20	time	memory
MinPoly	1219.00	3.29
Maple	519.36	41.00
Białas	> 8 h	∞

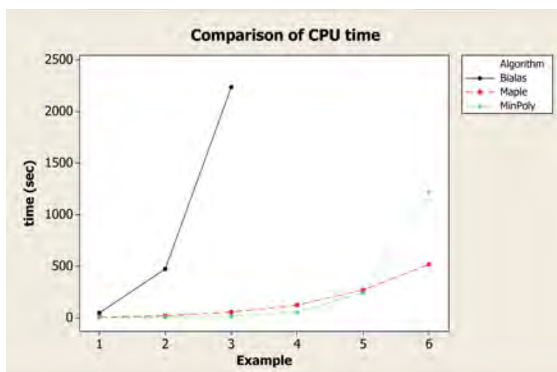

Figure 1: The time comparison of MinPoly, Maple and Białas

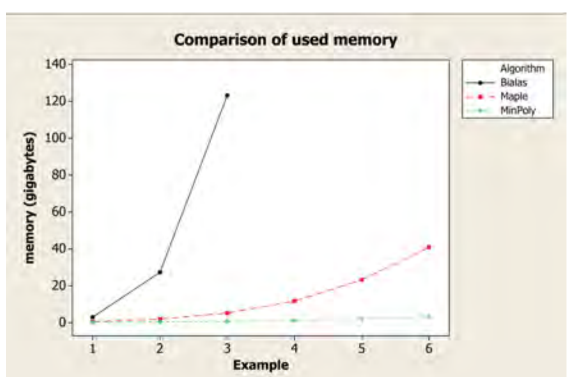

Figure 2: The memory comparison of MinPoly, Maple and Białas

The experiments we made seem to show that this first implementation is already very efficient. The efficiency of our algorithm comes from Theorem 1 where we have used Gröbner bases technique. This technique allows us to do not create a new object, such as matrix, which is very time consuming in MAPLE structure and also to use the efficiency of linear algebra methods for computing the Gröbner bases (see [8]). It is worth noting that, since the function `Basis` from `Groebner` package of MAPLE can compute the Gröbner bases in a polynomial ring over finite fields, then our algorithm can compute the minimal polynomial of a matrix over such a field, which seems not to be the case for the function `MinimalPolynomial` of MAPLE. Although our algorithm needs more time and memory for the last example than MAPLE 15, however we found few examples of this occurring, and that shows the performance of our algorithm versus two others.

5 Appendix

In this section, we discuss MINPOLY algorithm in the special case when its input is matrix over the field of rational numbers. Let A be an $n \times n$ matrix over the field $\mathbb{Q}$, and $m(s) = a_n s^n + a_{n-1}s^{n-1} + \cdots + a_0 \in \mathbb{Q}[a_0, \ldots, a_n, s]$ be a polynomial vanishing on A. Let also G be the reduced Gröbner basis of the ideal generated by the n^2 linear polynomials obtained from $m(A) = 0$, w.r.t. $a_0 \prec_{lex} \cdots \prec_{lex} a_n$. In the case that A is a matrix over $\mathbb{Q}$, Gröbner bases computation solves only a linear system which may be not interesting. Now, similar to Theorem 1, we can state the following corollary for this special case.

Corollary 1. *Using the above notations, let r be the remainder of the division of $m_1(s) = m(s)|_{a_{d+1}=\cdots=a_n=0}$ by $G_1 = G|_{a_{d+1}=\cdots=a_n=0}$ where $d = |G|$. Dividing r by its leading coefficient yields the minimal polynomial of A.*

In the following example, we show the computation of the minimal polynomial of a matrix over $\mathbb{Q}$.

Example 3. *We are willing to compute the minimal polynomial of the matrix*

$$A = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -2 & 4 \end{bmatrix}.$$

For this, we calculate first the following powers of A:

$$A^2 = \begin{bmatrix} 4 & 4 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & -1 & 5 \\ 0 & 0 & -10 & 14 \end{bmatrix}, A^3 = \begin{bmatrix} 8 & 12 & 0 & 0 \\ 0 & 8 & 0 & 0 \\ 0 & 0 & -11 & 19 \\ 0 & 0 & -38 & 46 \end{bmatrix}, A^4 = \begin{bmatrix} 16 & 32 & 0 & 0 \\ 0 & 16 & 0 & 0 \\ 0 & 0 & -49 & 65 \\ 0 & 0 & -130 & 146 \end{bmatrix}$$

Let $m(s) = a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0$ be a polynomial such that $m(A) = 0$. From this, we *obtain the following non-zero linear polynomials.*

$$\begin{array}{lcl} f_1 & := & 16a_4 + 8a_3 + 4a_2 + 2a_1 + a_0 \\ f_2 & := & 32a_4 + 12a_3 + 4a_2 + a_1 \\ f_3 & := & 16a_4 + 8a_3 + 4a_2 + 2a_1 + a_0 \\ f_4 & := & -49a_4 + -11a_3 - a_2 + a_1 + a_0 \\ f_5 & := & 65a_4 + 19a_3 + 5a_2 + a_1 \\ f_6 & := & -130a_4 - 38a_3 - 10a_2 - 2a_1 \\ f_7 & := & 146a_4 + 46a_3 + 14a_2 + 4a_1 + a_0. \end{array}$$

The reduced Gröbner basis of the ideal $\langle f_1, \ldots, f_7 \rangle$ *w.r.t. the ordering* $a_0 \prec_{plex} a_1 \prec_{plex} a_2 \prec_{plex} a_3 \prec_{plex} a_4$ *is equal to* $G = \{48a_1 + 36a_2 + 43a_0, -21a_1 - 25a_0 + 36a_3, 3a_1 + 4a_0 + 36a_4\}$ *which has three polynomials. Therefore, the minimal polynomial of* A *has degree three, and we can put* $a_4 = 0$ *in* G *and* $m(s)$*. Then* $G_1 = \{48a_1 + 36a_2 + 43a_0, -21a_1 - 25a_0 + 36a_3, 3a_1 + 4a_0\}$ *and its reduced form is equal to* $\{3a_1 + 4a_0, 12a_2 - 7a_0, 12a_3 + a_0\}$*. By computing the normal form of* $m_1(s)$ *by* G_1 *and dividing this new polynomial by its leading coefficient, we have the minimal polynomial of* A *is equal to* $s^3 - 7s^2 + 16s - 12$*.*

The following tables show the result of running the MinPoly algorithm, Białas algorithm and `MinimalPolynomial` of Maple 15 for six random $n \times n$ integer matrices with $n \in \{40, 50, 60, 70, 80, 90\}$.

40×40	time	memory
MinPoly	16.32	1.21
Maple	133.05	9.22
Białas	145.06	10.02

50×50	time	memory
MinPoly	50.00	3.00
Maple	637.06	34.67
Białas	686.44	37.13

60×60	time	memory
MinPoly	107.20	6.37
Maple	2495.14	107.60
Białas	2719.30	113.13

70×70	time	memory
MinPoly	192.80	12.41
Maple	7583.54	279.77
Białas	8664.64	291.72

80×80	time	memory
MinPoly	401.31	21.75
Maple	21822.71	637.17
Białas	22414.54	658.50

90×90	time	memory
MinPoly	689.51	36.31
Maple	> 8 h	∞
Białas	> 8 h	∞

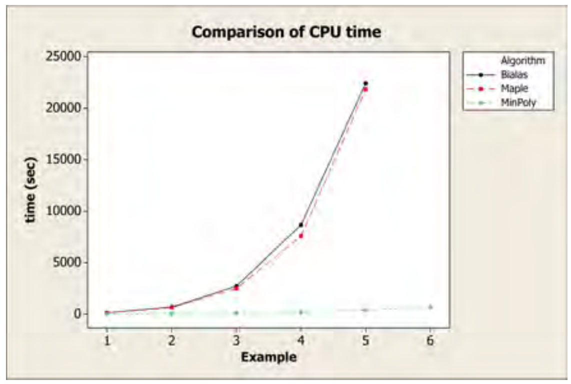

Figure 3: The time comparison of MINPOLY, MAPLE and BIAŁAS

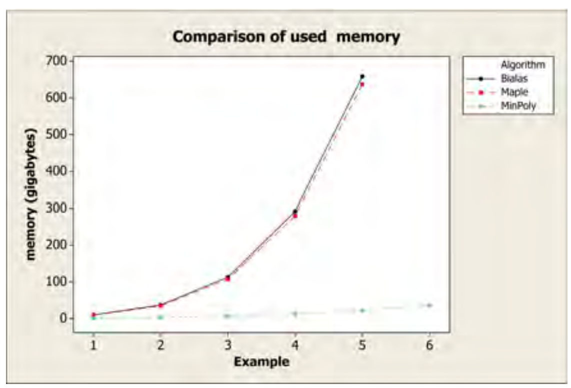

Figure 4: The memory comparison of MINPOLY, MAPLE and BIAŁAS

As the above tables and diagrams show MINPOLY is faster than two other algorithm for computing the minimal polynomial of matrices over field of rational numbers.

Acknowledgement. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the paper.

References

[1] W. W. ADAMS and P. LOUSTAUNAU, *An introduction to Gröbner bases*, American Mathematical Society, 1994.

[2] D. AUGOT and P. CAMION, On the computation of minimal polynomials, cyclic vectors, and frobenius forms, *Lin. Alg. App.*, **260** (1997), pp.61–pp.94.

[3] S. Białas and M. Białas, An algorithm for the calculation of the minimal polynomial, *Bulletin of the Polish academy of sciences: technical sciences*, **56(4)** (2008), pp.391–pp.393.

[4] B. Buchberger, *Ein algorithmus zum auffinden der basiselemente des restklassenringes nach einem nuildimensionalen polynomideal*, PhD thesis, Universität Innsbruck, 1965.

[5] B. Buchberger, An algorithm for finding the basis elements in the residue class ring modulo a zero dimensional polynomial ideal, *Journal of Symbolic Computation, Special Issue on Logic, Mathematics, and Computer Science: Interactions*, **41(3-4)** (2006), pp.475–pp.511.

[6] B. Buchberger, Gröbner-bases: An algorithmic method in polynomial ideal theory, *Chapter 6 in: N. K. Bose (ed.), Multidimensional Systems Theory - Progress, Directions and Open Problems in Multidimensional Systems Theory, Reidel Publishing Company, Dodrecht - Boston - Lancaster* (1985), pp.184–pp.232.

[7] D. Cox, J. Little, and D. O'Shea, *Ideals, varieties, and algorithms*, Undergraduate Texts in Mathematics. Springer-Verlag, New York, third edition, 2007.

[8] J.-C. Faugère, A new efficient algorithm for computing Gröbner bases (F_4), *J. Pure Appl. Algebra*, **139(1-3)** (1999), pp.61–pp.88.

[9] E. W. Mayr and A. R. Meyer, The complexity of the word problems for commutative semigroups and polynomial ideals, *Adv. in Math.*, **46(3)** (1982), pp.305–pp.329.

[10] C. D. Meyer, *Matrix analysis and applied linear algebra*, Society for Industrial and Applied Mathematics, 2000.

[11] M. Noro, An efficient implementation for computing Gröbner bases over algebraic number fields, *Lecture Notes in Computer Science*, **4151** (2006), pp.99–pp.109.

Received: 09.12.2011,
Revised: 08.01.2013,
Accepted: 12.01.13.

Department of Mathematical Sciences, Isfahan University of Technology, Isfahan, 84156-83111, Iran
E-mails: `Amir.Hashemi@cc.iut.ac.ir`,
`B.Alizadeh@math.iut.ac.ir`