Computing Jordan Normal Forms Exactly for Commuting Matrices in Polynomial Time

Jin-yi Cai*
Department of Computer Science
SUNY at Buffalo
Buffalo, NY 14260
cai@cs.buffalo.edu

Abstract

We prove that the Jordan Normal Form of a rational matrix can be computed exactly in polynomial time. We obtain the transformation matrix and its inverse exactly, and we show how to apply the basis transformation to any commuting matrices.

1 Introduction

There are two motivations for this work on computing the Jordan Normal Form of a rational matrix exactly. The first is related to the resolution of the complexity of the A B C problem [4], and its application to the complexity problem in finitely generated commutative linear groups and semigroups in general. The second motivation is concerned with the design and analysis of uncheatable benchmarks for numerical algorithms, especially matrix multiplication [3, 1].

Our problem is the following. Given a finite set of commuting matrices over the rational numbers, $A, B, \ldots$, can we compute, in polynomial time, a basis transformation T, and the matrices under the similarity transformation $T^{-1}AT, T^{-1}BT, \ldots$, so that $T^{-1}AT$ is the Jordan Normal Form (JNF) of A? Here, computation is to be performed exactly, and not merely to be

*Research supported in part by NSF grants CCR 9057486 and CCR 9319093, and an Alfred P. Sloan Fellowship.

numerically approximated. The input size of the problem is the sum of all binary lengths of the input entries, and the complexity is measured in terms of the number of bit operations.

Of course, computing the JNF of an arbitrary rational matrix implies computing all complex roots of an arbitrary polynomial with rational coefficients. It is well known that equations of degree 5 or higher in general do not have roots expressible in radicals. Then what do we mean by computing it exactly?

What is meant by exact computation here is the following. We will deal with only algebraic numbers, and we will associate with any algebraic number an irreducible polynomial over the rationals, and a sufficiently good rational approximation, which uniquely identifies the particular root of the polynomial. Note that, given such data, an *arbitrarily* good rational approximation can be easily computed, say, by Newton's iteration. This is the approach taken by Lovász in [11], and it is consistent with Turing's notion of a computable real number [15]. In fact, in terms of computational complexity, the fact that quintic equations may not have radical expressions for their roots is largely irrelevant; it simply rules out *one* mode of expression. Of course, as we will see later, the complexity of the Galois group itself will enter the picture.

Our first motivation is concerned with commutative linear groups and semigroups. In 1980, Kannan and Lipton [8] solved the following *orbit problem*, by giving a polynomial time algorithm to it:

> Given two commuting matrices A and B over the rational numbers, does there exist a nonnegative integer i, such that $A^i = B$?

The following *generalized orbit problem*, is known as *the A B C problem*:

> Given commuting matrices A, B and C over the rational numbers, does there exist nonnegative integers i and j, such that $A^i B^j = C$?

A host of other problems are reducible to the orbit problem [8]. In [4], the complexity of the A B C problem was resolved. It was shown that the A B C problem can also be solved in polynomial time. In solving this problem, we made extensive use of the computability of the JNF of a rational matrices in polynomial time. In fact we need to use the full force of the current paper: computing the transformed matrices which commute with A.

The A B C problem is a special case of the following more general problem:

> Given commuting matrices $A_1, A_2, \ldots, A_k$ and B, over the rational numbers, does there exist nonnegative integer $i_1, i_2, \ldots, i_k$, such that $A_1^{i_1} A_2^{i_2} \cdots A_k^{i_k} = B$?

Here k is considered fixed. We hope that the techniques developed here can be generalized to solve this general case.

Our second motivation for this work is a more practical one. In [3], this author and others have initiated a study of uncheatable benchmarks.

Benchmarks have been used to test everything from the speed of a processor, to the access time, capacity, and bandwidth of a memory system. The computing community relies on them heavily to assess how well a given hardware or software system operates. They are of fundamental importance in everyday computing. Up until now, however, the study of the art of designing a good benchmark has focused on making the benchmark "realistic" in predicting how well it will perform for the intended applications; the issue of making benchmark results trustworthy has been relegated to "trusted" or third party agents, and little attention has been paid to the question of making benchmarks themselves "uncheatable". In [3] we proposed a framework based on modern cryptography and complexity theory, in which we can address questions such as how one can make benchmarks resistant to tampering and hence more trustworthy. Several concrete schemes were proposed for different benchmarks: speed of the processor, memory capacity, sorting, etc. They are "uncheatable", if certain complexity theory assumptions are true based on the hardness of factoring and discrete logarithm.

In [1], a novel idea was presented, which uses numerical instability as an alternative basis for designing uncheatable benchmarks. An uncheatable benchmark was designed for matrix multiplication based on numerical instability associated with computing the JNF. It was observed, as by virtually all numerical analysists we spoke to, that for a non-diagonalizable matrix A, it is numerically instable, and thus by implication practically impossible, to compute its Jordan Normal Form. The reason is compelling enough: Suppose $A = T^{-1}JT$ has JNF $J = \begin{pmatrix} \lambda & 1 \\ 0 & \lambda \end{pmatrix}$, where λ is any complex number. Let $\tilde{A} = T^{-1}\tilde{J}T$ be a slightly altered matrix, where $\tilde{J} = \begin{pmatrix} \lambda' & 1 \\ 0 & \lambda'' \end{pmatrix}$, and $\lambda' \neq \lambda''$, but they are close to λ. Note that, since now $\tilde{A}$ has unequal eigenvalues, the JNF for $\tilde{A}$ is not $\tilde{J}$ but $\begin{pmatrix} \lambda' & 0 \\ 0 & \lambda'' \end{pmatrix}$. In other words, the map from the space of matrices to its JNF is not a continuous map, [1] and since numerical round-off errors are unavoidable, it is hopeless to compute it.

Or is it?

In this note we show that the answer is more complicated. It is true that the map is discontinuous, and therefore in trying to compute it numerically, it is hopeless. However, that does not mean that by some other means, in particular, computing symbolically, we cannot compute the JNF of a rational matrix. On the other hand, even though we show that the JNF *can be* computed in polynomial time, the speed is still far from competitive with numerical computing, such as Q-R iteration, and therefore, our uncheatable benchmark in [1] appears to be secure.

2 Computing a basis change for the Jordan form

In this section, we show how to compute a basis change in polynomial time, such that the matrix A will have its Jordan normal form, $J_A = T^{-1}AT$. We note that, since T is computed symbolically, in the splitting field of χ_A over $\mathbf{Q}$, it is not clear how to compute T^{-1} from T in general, in polynomial time. The Galois group structure of the splitting field over $\mathbf{Q}$ is rather complicated, and in general not believed to be computable in P-time. We in fact compute $J_A = T^{-1}AT$ without actually finding T^{-1} nor performing matrix products in $T^{-1}AT$. The problem of computing T^{-1}, and computing the corresponding transformed matrix $T^{-1}BT$, for any B which commutes with A, will be discussed in later sections.

2.1 The rational reduction

Lemma 2.1 *If A and B commute, and $f(x)$ and $g(x)$ are two polynomials over a field F that are relatively prime. Then the null space* $\ker(f(A)g(A)) = \{x | f(A)g(A)x = 0\}$ *is a direct sum of* $\ker f(A)$ *and* $\ker g(A)$. *Furthermore, they are both invariant subspaces of A as well as B.*

Proof: Clearly, $\ker f(A), \ker g(A) \subseteq \ker(f(A)g(A))$. Since $f(x)$ and $g(x)$ are relatively prime, there exist polynomials $a(x)$ and $b(x)$ in $F[x]$, such that $a(x)f(x)+b(x)g(x) = 1$. Thus, for any $v \in \ker(f(A)g(A))$, $v = a(A)f(A)v+ b(A)g(A)v$, where $a(A)f(A)v \in \ker g(A)$ and $b(A)g(A)v \in \ker f(A)$. And

[1] The JNF is only unique up to permutations of the Jordan blocks. But we can carefully define a quotient space so that the map is well defined.

moreover, the sum is a direct sum, since if $v \in \ker f(A) \cap \ker g(A)$, then $v = a(A)f(A)v + b(A)g(A)v = 0$.

Since A and B commute, for any polynomial h, if $v \in \ker h(A)$, then $Bv \in \ker h(A)$, as $h(A)Bv = Bh(A)v = 0$. $\square$

We remark that over the rational numbers $\mathbf{Q}$ these computations are all in P-time. To find the polynomials $a(x), b(x) \in \mathbf{Q}[x]$, we need to carry out the Euclidean algorithm. We can also compute various null spaces and its basis over $\mathbf{Q}$.

There are quite some subtleties involved in the Euclidean algorithm, as well as linear equation solving, in P-time. We need to ensure that no coefficient gets too large, For that one has to repeatedly reduce the coefficients. See [5, 7, 6]. It is known from the work of Collins and Kannan that generalized gcd as well as linear space computations such as null space rank, basis, dimension over $\mathbf{Q}$ can all be computed in P-time in terms of bit complexity. A generalization by Kannan, Lenstra, Lovász [9] also lets us carry out these computations in P-time over an algebraic extension field $\mathbf{Q}(\lambda)$ of bounded degree, where λ is a root of an irreducible polynomial $a_d x^d + \cdots + a_0$. Here entries of $\mathbf{Q}(\lambda)$ are represented by polynomials in λ with degree $< d$, and P-time in bit complexity is measured in terms of the bit size of all rational entries, the degree d and the bit size of all the coefficients a_i. In the following we will rely on the results cited above whenever we assert certain algebraic computation is in P-time.

Now we apply the L^3–algorithm [10] and get a factorization of χ_A as a product of powers of irreducible polynomials $\chi_A = f_1^{e_1} \cdots f_k^{e_k}$, where each f_i is irreducible over $\mathbf{Q}\ [x]$ and each $e_i \geq 1$. Let $V = \mathbf{Q}^n$ be the n-dimensional space over $\mathbf{Q}$. We view A as well as any polynomial of A as linear operators on V.

Theorem 2.1 *Let $V_i = \ker_{\mathbf{Q}}(f_i(A)^{e_i})$. Then each V_i is an invariant subspace of both A and B, and V is a direct sum of these V_i:*

$$V = V_1 \oplus V_2 \oplus \cdots \oplus V_k.$$

Furthermore we can compute a basis for each V_i in P-time, such that the union of which forms a basis under which both A and B have block diagonal form corresponding to the V_i's.

The proof is a repeated application of Lemma 2.1. All computations can be done in P-time, as noted above, since we only require Euclidean algorithm over $\mathbf{Q}\ [x]$ and solving systems of linear equations over $\mathbf{Q}$.

Thus we will focus on a fixed V_i. From now on we assume that χ_A is already a power of an irreducible polynomial f^ϵ.

2.2 Powers of an irreducible polynomial

Let $\deg f = d$, let $\lambda_1, \lambda_2, \ldots, \lambda_d$ be d (distinct) roots of $f(x)$. Then $n = d\epsilon$. Note that there is a field automorphism $\sigma_i : \mathbf{Q}(\lambda_1) \to \mathbf{Q}(\lambda_i)$, which sends λ_1 to λ_i and fixes $\mathbf{Q}$. Let $F_i = \mathbf{Q}(\lambda_i)$, let F^* be the splitting field $\mathbf{Q}(\lambda_1, \ldots, \lambda_d)$. Consider the n-dimensional vector space $V = \mathbf{Q}^n$ over $\mathbf{Q}$ on which both A and B act. We may view V as a vector space over the splitting field F^* as well, and again A and B act on it. More precisely, we form the tensor product $\widehat{V} = F^* \otimes V$.

Let $\widehat{V}_i = \ker(A - \lambda_i I)^\epsilon$, then by Lemma 2.1, $\widehat{V}$ is a direct sum of the $\widehat{V}_i$'s. Formally speaking, $\widehat{V}_i$ as a subspace of $\widehat{V}$ is a vector space over F^*. But clearly we can also view $\widehat{V}_i$ as a vector space over F_i. More precisely, we can define $V_i = \ker_{F_i}(\lambda_i I - A)^\epsilon$, and $\widehat{V}_i = \ker_{F^*}(A - \lambda_i I)^\epsilon$, then $\widehat{V}_i = F^* \otimes V_i$ as a tensor product, and

$$\widehat{V} = F^* \otimes V = \bigoplus_{i=1}^{d} \widehat{V}_i.$$

The distinction of V_i and $\widehat{V}_i$ is a minor one mathematically, perhaps, but a very important one for computational purposes. We will stay within each F_i whenever possible, and stay away from F^*. The reason is that in the smaller field F_i of degree d over $\mathbf{Q}$, we can do arithmetic just as in $\mathbf{Q}$, but since we do not know the Galois group structure of $Gal(F^*, \mathbf{Q})$, in P-time, arithmetic questions involving multinomials, such as whether $\lambda_1\lambda_2 = \lambda_3\lambda_4$, are hard to answer.

We now focus on how to compute a basis in V_1 for which A has its Jordan form (i.e., all λ_1-Jordan blocks of A.) We will restrict A to V_1, and let $A_1 = A|_{V_1} - \lambda_1 I$. Define $U_j = \ker A_1^j$ for $j = 0, 1, \ldots, \epsilon, \ldots$ Clearly, $U_0 = 0$ and $V_1 = U_\epsilon = U_{\epsilon+1}$. Suppose $U_i = U_{i+1}$, then for all $j > i$, $U_i = U_j$. This is clearly seen by induction: Let $x \in U_{j+1}$ so that $A_1^{j+1}x = A_1^j A_1 x = 0$. It follows that $A_1 x \in U_j = U_i$, so $x \in U_{i+1} = U_i$.

Let $e \leq \epsilon$ be the least integer such that $U_e = U_{e+1}$, then

$$U_1 \subset \ldots \subset U_e = \ldots = U_\epsilon.$$

This e can be computed in P-time by computing the rank, over F_1, of A_1^j, or equivalently the dimension $\dim_{F_1} U_j$, as $j = 1, 2, \ldots, \epsilon$.

Let $n_1 = \dim U_1$. This is the dimension of the eigenspace of A belonging to λ_1. Since λ_1 is an eigenvalue of A, $n_1 \geq 1$. In terms of the Jordan form, n_1 is the number of Jordan λ_1-blocks of A, and if A is in its Jordan form, then the collection of unit vectors that corresponding to all the first vectors of each Jordan λ_1-block forms a basis for U_1. Similarly U_2 corresponds to all the first and second vectors of each Jordan λ_1-block, etc. Thus let

$$n_1 + n_2 + \ldots + n_i = \dim U_i,$$

for $i = 1, \ldots, e$, then

$$n_1 \geq n_2 \geq \ldots \geq n_e > 0.$$

We will inductively compute a basis for $V_1 = U_e$, $\{a_{i,j} | 1 \leq i \leq e, 1 \leq j \leq n_i\}$, for which $A|_{V_1}$ has its Jordan form. (Again, all entries of all vectors will be from F_1, and all arithmetic is done over F_1.)

First we can compute a basis for U_1, $\{b_{1,1}, \ldots, b_{1,n_1}\}$. Any basis of U_1 computable in P-time will do. Then we can extend this basis to a basis arbitrarily for U_2, $\{b_{1,1}, \ldots, b_{1,n_1}, b_{2,1}, \ldots, b_{2,n_2}\}$, etc. until we get a full basis for U_e,

$$\{b_{1,1}, \ldots, b_{1,n_1}, b_{2,1}, \ldots, b_{2,n_2}, \ldots, b_{e,1}, \ldots, b_{e,n_e}\}.$$

All of this is done in P-time. If $e = 1$, then we are finished, as $A|_{V_1}$ is a scalar matrix.

Assume $e > 1$, i.e., some λ_1-block of A has dimension at least 2.

Lemma 2.2 *For all $2 \leq i \leq e$,*

$$\{A_1 b_{i,1}, \ldots, A_1 b_{i,n_i}\} \subset U_{i-1},$$

and

$$\{A_1 b_{i,1} \bmod U_{i-2}, \ldots, A_1 b_{i,n_i} \bmod U_{i-2}\}$$

are linearly independent in the quotient space U_{i-1}/U_{i-2}.

Proof: Since for each i, j, such that $2 \leq i \leq e$, $1 \leq j \leq n_i$, $b_{ij} \in U_i$. Thus, $A_1^i b_{ij} = A_1^{i-1}(A_1 b_{ij}) = 0$, hence $A_1 b_{ij} \in U_{i-1}$.

Let $\alpha_j \in F_1$, $1 \leq j \leq n_i$, such that $\sum_{j=1}^{n_i} \alpha_j A_1 b_{ij} \in U_{i-2}$. To show linear independence, it suffices to show that all $\alpha_j = 0$. Now, $A_1(\sum_{j=1}^{n_i} \alpha_j b_{ij}) \in U_{i-2}$, hence,

$$\sum_{j=1}^{n_i} \alpha_j b_{ij} \in \ker A_1^{i-1} = U_{i-1}.$$

Thus there exist $\beta_{st} \in F_1$, $1 \leq s \leq i-1$, $1 \leq t \leq n_s$, such that

$$\sum_{j=1}^{n_i} \alpha_j b_{ij} = \sum_{s,t} \beta_{st} b_{st}.$$

Since $\{b_{st} : 1 \leq s \leq i, 1 \leq t \leq n_s\}$ forms a basis for U_i, all $\alpha_j = 0$ (as well as all $\beta_{st} = 0$.) Thus, $\{A_1 b_{i,1} \bmod U_{i-2}, \ldots, A_1 b_{i,n_i} \bmod U_{i-2}\}$ are linearly independent. □

Now we modify the basis b_{ij} as follows. Start with $i = e$, the last batch, $\{b_{e,1}, \ldots, b_{e,n_e}\}$. These are chosen, in other words, $a_{e,1} = b_{e,1}, \ldots, a_{e,n_e} = b_{e,n_e}$.

In general, suppose $\{a_{i,1}, \ldots, a_{i,n_i}\}$ have been chosen, $i \geq 2$. From the current basis $\{b_{1,1}, \ldots, b_{i-2,1}, \ldots, b_{i-2,n_{i-2}}\}$ for U_{i-2} we use the subset $\{A_1 a_{i,1}, \ldots, A_1 a_{i,n_i}\} \subset U_{i-1}$ as the first n_i vectors for extending the basis of U_{i-2} to U_{i-1}. Let $a_{i-1,1} = A_1 a_{i,1}, \ldots, a_{i-1,n_i} = A_1 a_{i,n_i}$. If $n_{i-1} = n_i$, we are done for $i-1$. If $n_{i-1} > n_i$, then extend arbitrarily in U_{i-1} until a basis for U_{i-1} is obtained. Call them $a_{i-1,n_i+1}, \ldots, a_{i-1,n_{i-1}}$. And now $\{a_{i-1,1}, \ldots, a_{i-1,n_{i-1}}\}$ are chosen. Iteratively go down with i from e to 2, we have obtained the basis for U_e.

For all i, j, such that $2 \leq i \leq e$, $1 \leq j \leq n_i$, $A|_{V_1} a_{ij} = \lambda_1 a_{ij} + a_{i-1,j}$, and $A|_{V_1} a_{1j} = \lambda_1 a_{1j}$ for all $1 \leq j \leq n_1$. Thus under this basis $A|_{V_1}$ is in its Jordan form.

To obtain a full basis under which A has its Jordan form, we apply the automorphisms σ_i, for $i = 2, \ldots, d$. Note that all the computations over F_1 are symbolic and thus extends readily to F_i *verbatim*. Thus to obtain a basis for $A|_{V_i}$ for the λ_i-Jordan blocks, we only need to replace all occurrences of λ_1 by λ_i. This is finally a basis change T, such that $T^{-1}AT$ is in Jordan form. The matrix T has a "striped form", where the first ϵ-columns are vectors over F_1, and the next ϵ-columns are vectors over F_2 under the substitution of λ_2 for λ_1, etc.

3 Transformations for commuting matrices

Before we start, we may wonder why we didn't try to compute a basis change such that both A and a commuting matrix B are simultaneously put in JNF. While it is true that if A and B commute, and if both are diagonalizable, then they can be simultaneously diagonalized. It is not true that commuting matrices can always be simultaneously put in JNF. This is seen by the following example.

3.1 An example

Consider the 4 by 4 matrices

$$X = \begin{pmatrix} \lambda I + U & 0 \\ 0 & \lambda I + U \end{pmatrix} = \begin{pmatrix} \lambda & 1 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & \lambda & 1 \\ 0 & 0 & 0 & \lambda \end{pmatrix}$$

and

$$Y = \begin{pmatrix} \lambda I & I \\ 0 & \lambda I \end{pmatrix} = \begin{pmatrix} \lambda & 0 & 1 & 0 \\ 0 & \lambda & 0 & 1 \\ 0 & 0 & \lambda & 0 \\ 0 & 0 & 0 & \lambda \end{pmatrix},$$

where

$$I = I_{2\times 2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

and

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

and λ is *any* complex number.

To verify that $XY = YX$, we can check directly, or we can go as follows: Since $X = \lambda I_4 + \begin{pmatrix} U & 0 \\ 0 & U \end{pmatrix}$ and $Y = \lambda I_4 + \begin{pmatrix} 0 & I \\ 0 & 0 \end{pmatrix}$, we only need to verify that

$$\begin{pmatrix} U & 0 \\ 0 & U \end{pmatrix} \cdot \begin{pmatrix} 0 & I \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & I \\ 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} U & 0 \\ 0 & U \end{pmatrix},$$

which is just $\begin{pmatrix} 0 & U \\ 0 & 0 \end{pmatrix}$.

Now, X is already in its Jordan form, while the (unique) Jordan form for Y is X. This can be seen by the basis change of $\{e_1, e_2, e_3, e_4\} \to \{e_1, e_3, e_2, e_4\}$.

Suppose there exists a non-singular matrix Z, such that $Z^{-1}XZ = X$ and $Z^{-1}YZ = X$. Then clearly, we must have $X = Y$, a contradiction.

In fact this failure of simultaneous Jordanization is one of the main difficulties in the A B C problem [4]. We will settle for the more modest goal of putting one of the matrices in JNF, while computing the transformed forms of the other commuting matrices exactly.

3.2 Computing $T^{-1}BT$

Now we show how to get $T^{-1}BT$. A basis change is not much good if we cannot apply to some other matrices. Note that in section 2 we did not compute T^{-1}, nor did we compute $T^{-1}AT$ by matrix product. To compute T^{-1} and $T^{-1}BT$ using standard method would involve the splitting field in general, and would not be in P-time.

As it turns out that, computing $T^{-1}BT$, using the fact that A and B commute, need not involve actually having T^{-1} computed first. We observe that, since A and B commute, $V_i = \ker(A - \lambda_i I)^{\epsilon}$ is an invariant subspace of B as well. This means that under the basis change T, $T^{-1}BT$ will have a block diagonal form, which will enable us to compute all of its entries in P-time.

More precisely, let $\epsilon = n_1 + n_2 + \ldots + n_e$ be the number of basis vectors that correspond to λ_1 (the first ϵ columns in T). Let these column vectors form an $n \times \epsilon$ matrix T_1. Let the first ϵ columns of $T^{-1}BT$ be denoted by B_1, then the last $n - \epsilon$ rows of B_1 are all 0. Let the top ϵ rows of B_1 be denoted by B_{11}. Then, $B_1 = \begin{pmatrix} B_{11} \\ 0 \end{pmatrix}$, and $BT_1 = TB_1$, which implies that $BT_1 = T_1 B_{11}$.

If we view this matrix equation column by column in B_{11}, each column gives us a system of linear equations with the entries of B_{11} as unknowns and the entries of T_1 as coefficients. Since T_1 has full column rank ϵ, we can find the appropriate rows of T_1, which gives us a square $\epsilon \times \epsilon$ system of linear equations of full rank ϵ. This gives us a unique solution for, say, the first column of B_{11}. (The system of linear equations in $BT_1 = T_1 B_{11}$ may appear *over-determined*, but our structural information has guaranteed us that there is a solution, and by the above argument a unique solution.) This can be carried out for all columns of B_{11}.

To obtain the full matrix $T^{-1}BT$ we again apply the automorphisms σ_i, for $i = 2, \ldots, d$. Thus the other blocks of $T^{-1}BT$ are obtained by substitution of λ_i for λ_1 in B_{11}.

4 Computing the inverse T^{-1}

We first prove a lemma.

Lemma 4.1 *If $A_1 = \lambda_1 I + N$ and $A_2 = \lambda_2 I + N$, where $\lambda_1 \neq \lambda_2$, and N is a nilpotent matrix. Let X be any matrix such that $A_1 X = X A_2$, then $X = 0$.*

Proof: By changing to the JNF, we can assume that N is strictly upper triangular. Suppose for a contradiction, that $X \neq 0$, and the kth column $\mathbf{x}_k$ of X is its first nonzero column, $1 \leq k \leq n$.

Consider XA_2. The kth column of XA_2 is $\lambda_2 \mathbf{x}_k$. Now consider $A_1 X$. The (n, k)-entry of $A_1 X$ is $\lambda_1 \mathbf{x}_{n,k}$, where $\mathbf{x}_{n,k}$ is the (n, k)-entry of X. Since $\lambda_1 \neq \lambda_2$, $\mathbf{x}_{n,k} = 0$.

Now the $(n-1, k)$-entry of X must also be zero, since the $(n-1, k)$-entry of XA_2 is $\lambda_2 \mathbf{x}_{n-1,k}$, while the same entry in $A_1 X$ is $\lambda_1 \mathbf{x}_{n-1,k}$, due to the fact that $\mathbf{x}_{n,k} = 0$.

An easy induction proves that in fact $\mathbf{x}_k = 0$, and thus $X = 0$. $\square$

Theorem 4.1 *The inverse T^{-1} can be computed in polynomial time.*

Proof: We have computed T, and J, the JNF of A. If A is invertible, then we can also compute A^{-1} and J^{-1} easily, since A is a rational matrix, and J^{-1} has a closed form formula. If A is singular, then we can set $A' = dI + A$, for a sufficiently large d, then both A' and its JNF $J' = dI + J$ are invertible. Thus it is without loss of generality that we assume A is invertible.

Now we can compute an invertible S, such that $SA^{-1} = J^{-1}S$. This can be accomplished by a "row" vector version of the procedure similar to the computation of T. The details are given in Appendix 1.

Note that S has a "striped" form, where the first ϵ-rows are over $\mathbf{Q}(\lambda_1)$, and then an equal number of rows over $\mathbf{Q}(\lambda_2)$ follows, etc.

Since $SAS^{-1} = J = T^{-1}AT$, we get that $S^{-1}JS = A = TJT^{-1}$, which implies that ST commutes with J.

Write $J = diag\{J_1, J_2, \ldots, J_k\}$, where each $J_i = \lambda_i + N$, N is nilpotent and the same for all i. By Lemma 4.1, $X = ST$ has block diagonal form as well $diag\{X_1, X_2, \ldots, X_k\}$. Each $X_i = S_i \cdot T_i$ is over $\mathbf{Q}(\lambda_i)$, thus can be computed. In fact, by applying the automorphism σ_i, all we need to compute is the first block. It follows that we can also compute the inverse X^{-1}.

Now $T^{-1} = X^{-1}S = diag\{X_1^{-1}, X_2^{-1}, \ldots, X_k^{-1}\}S$. By the "striped" form of S, this product can be computed. $\square$

Appendix: Computing S

We first consider the closed form formula for the inverse of a Jordan block. Assume $\lambda \neq 0$.

Let $J = \lambda I_k + U$, a Jordan block of size k, where

$$U = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix},$$

with 1's on the upper off-diagonal, and 0's everywhere else. Then J^{-1} has the closed formula

$$J^{-1} = \mu \left[I + (-\mu U) + (-\mu U)^2 + \cdots + (-\mu U)^{k-1} \right],$$

where $\mu = 1/\lambda$. In fact,

$$\begin{aligned} (I + aU)^n &= \sum_{i=0}^{k-1} \binom{n}{i} (aU)^i \\ &= I + \binom{n}{1} aU + \binom{n}{2} (aU)^2 + \cdots + \binom{n}{k-1} (aU)^{k-1}, \end{aligned}$$

for all $n \in \mathbf{Z}$. For negative $-n$, this specializes to

$$\begin{aligned} (I + aU)^{-n} &= \sum_{i=0}^{k-1} (-1)^i \binom{n+i-1}{i} (aU)^i \\ &= I - \binom{n}{1} aU + \binom{n+1}{2} (aU)^2 + \cdots + (-1)^{k-1} \binom{n+k-2}{k-1} (aU)^{k-1}. \end{aligned}$$

Thus,

$$J^{-n} = \mu^n \left[I - \binom{n}{1} \mu U + \binom{n+1}{2} (\mu U)^2 + \cdots + (-1)^{k-1} \binom{n+k-2}{k-1} (\mu U)^{k-1} \right],$$

for all $n \geq 0$.

Assume V_λ is the subspace $\ker(A - \lambda I)^\epsilon$, where all the λ-blocks of A are located. Let $M = (A|_{V_\lambda})^{-1} - \mu I$, then, in each Jordan block of size k, M has the form

$$\begin{aligned} & \sum_{s=2}^{k} (-1)^{s-1} \mu^s U^{s-1} \\ = \; & -\mu^2 U + \mu^3 U^2 + \cdots + (-1)^{k-1} \mu^k U^{k-1}. \end{aligned}$$

We now show how to find a basis of row vectors S, such that SMS^{-1} has this form.

Recall the definition of $U_i = \ker A_1^i$, where $A_1 = A|_{V_{\lambda_1}} - \lambda_1 I$, and

$$n_1 + n_2 + \ldots + n_i = \dim U_i,$$

for $i = 1, \ldots, e$.

The desired basis of row vectors $\{b_{1,1}^T, \ldots, b_{1,n_1}^T, \ldots, b_{e,1}^T, \ldots, b_{e,n_e}^T\}$ should satisfy

$$b_{i,j}^T M = \sum_{s=2}^{i} (-1)^{s-1} \mu^s b_{i-(s-1),j}^T,$$

for $1 \le i \le e, 1 \le j \le n_i$. (The sum is 0, if $i = 1$.)

This basis is computed by a double induction.

We start with any basis $\{a_{1,1}^T, \ldots, a_{1,n_1}^T, \ldots, a_{e,1}^T, \ldots, a_{e,n_e}^T\}$, such that $\{a_{1,1}^T, \ldots, a_{1,n_1}^T, \ldots, a_{i,1}^T, \ldots, a_{i,n_i}^T\}$ is a basis for U_i, for $1 \le i \le e$.

If $e = 1$, then we are done: In this case, $\{a_{1,1}^T, \ldots, a_{1,n_1}^T\}$ are all row-eigenvectors of A, therefore that of A^{-1} as well, so that M is identically 0. All blocks are 1 by 1.

Suppose $e > 1$. Form the quotient space U_e/U_1. Inductively, from $\{a_{2,1}^T, \ldots, a_{2,n_2}^T, \ldots, a_{e,1}^T, \ldots, a_{e,n_e}^T \pmod{U_1}\}$ we can form a basis

$$\{\tilde{b}_{2,1}^T, \ldots, \tilde{b}_{2,n_2}^T, \ldots, \tilde{b}_{e,1}^T, \ldots, \tilde{b}_{e,n_e}^T \pmod{U_1}\}$$

such that

$$\tilde{b}_{i,j}^T M = \sum_{s=2}^{i-1} (-1)^{s-1} \mu^s \tilde{b}_{i-(s-1),j}^T \pmod{U_1},$$

for $2 \le i \le e, 1 \le j \le n_i$. (The sum is 0, if $i = 2$.)

Now we will modify the $\tilde{b}_{i,j}^T$'s to finish the proof. The final $b_{i,j}^T$'s will be $\equiv \tilde{b}_{i,j}^T \pmod{U_1}$.

Now $\tilde{b}_{2,j}^T M \in U_1$, for $1 \le j \le n_2$. By Lemma 2.2, they are linearly independent. We let the first n_2 basis vectors for U_1 be chosen as, $b_{1,j}^T = \tilde{b}_{2,j}^T M$, for $1 \le j \le n_2$. If $n_1 > n_2$, then extend arbitrarily to a basis $\{b_{1,1}^T, \ldots, b_{1,n_1}^T\}$ for the eigenspace U_1. For $i \ge 2$, inductively assume we have chosen $\{b_{1,1}^T, \ldots, b_{1,n_1}^T, \ldots, b_{i-1,1}^T, \ldots, b_{i-1,n_{i-1}}^T\}$ such that

$$b_{\ell,j}^T M = \sum_{s=2}^{\ell} (-1)^{s-1} \mu^s b_{\ell-(s-1),j}^T,$$

for $2 \leq \ell \leq i-1$, $1 \leq j \leq n_\ell$, and

$$\tilde{b}_{i,j}^T M = \sum_{s=2}^{i-1} (-1)^{s-1} \mu^s b_{i-(s-1),j}^T + \xi_j b_{1,j}^T,$$

for some number $\xi_j \in \mathbf{Q}$ and $1 \leq j \leq n_i$.

Now $\tilde{b}_{i,j}^T$, for $1 \leq j \leq n_i$, are to be modified by adding a suitable multiple of $b_{1,j}^T$ such that $\tilde{b}_{i,j}^T \equiv b_{i,j}^T \pmod{U_1}$ and

$$b_{i,j}^T M = \sum_{s=2}^{i} (-1)^{s-1} \mu^s b_{i-(s-1),j}^T,$$

for $1 \leq j \leq n_i$. This completes the proof.

Acknowledgement

I would like to thank Pat Eberlein, Susan Landau, Ravi Kannan, Steve Schanuel and Zeke Zalcstein for helpful discussions.

References

[1] S. Ar, and J. Cai, Reliable Benchmarks Using Numerical Instability, in the Proceedings of *SIAM-ACM Annual Symposium on Discrete Algorithms*, (SODA), 1994.

[2] M. Beaudry, Membership testing in commutative transformation semigroups, Information and Computation, Vol 79 (1988), 84–93.

[3] J. Cai, R. J. Lipton, R. Sedgewick and A. Yao, Towards Uncheatable Benchmarks, In the Proceedings of *The Structure in Complexity Theory Conference*, (1993) 2–11.

[4] J. Cai, Richard J. Lipton, and Yechezkel Zalcstein, The Complexity of the A B C Problem Resolved, manuscript.

[5] G. E. Collins, Subresultants and reduced polynomial remainder sequences, *J. ACM*, 14, 1 (1967). 128–142.

[6] D.E. Knuth, *The Art of Computer Programming*, vol 2, Addison-Wesley, 2nd Edition, 1981.

[7] R. Kannan, The size of numbers in the analysis of certain algorithms, PH. D. Dissertation, Operations Research Dept., Cornell University, Ithaca, NY. 1980.

[8] R. Kannan and R. Lipton, The orbit problem is decidable, STOC 1980, 252-261. See also "Polynomial-time algorithms for the orbit problem", JACM, Vol 33, No. 4, 1986, 808–821.

[9] R. Kannan, A. K. Lenstra, L. Lovász, "Polynomial factorization and non-randomness of bits of algebraic numbers and certain transcendental numbers", Math. of Comp., Vol. 50, No. 181, Jan. 1988, 235–250.

[10] A. K. Lenstra, H. W. Lenstra, L. Lovász, Factoring polynomials with rational coefficients, Math. Ann. 261:515-534, 1982.

[11] L. Lovász, *An Algorithmic Theory of Numbers, Graphs and Convexity*, CBMS 50, SIAM, 1986.

[12] S. Landau, personal communications.

[13] K. A. Mihailova, The occurrence problem for a direct product of groups, Doklady Akads. Nauk (USSR), 119 (1958), 1103-1105.

[14] M. Paterson, Unsolvability in 3 by 3 matrices, J. of Math. and Physics, Vol 49 (1970), 105–107.

[15] A. M. Turing, On computable real numbers, with an application to the Entscheidung problems, Proc. London Math. Soc., 42, 230–265.