NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

ISBN 0-315-56037-1

On p-adic Computation of the Rational Form of a Matrix

Marie-Hélène Mathieu

A Thesis

in

The Department

of

Computer Science

Presented in Partial Fulfilment of the Requirements
for the Degree of Master of Computer Science
Concordia University
Montréal, Québec, Canada

February 1990

ABSTRACT

On p-adic Computation of
the Rational Form of a Matrix

Marie-Hélène Mathieu

We consider the problem of bringing a given matrix into "cyclic form", from which the rational form can be computed easily. Two new algorithms are proposed (exact and modular), described, implemented and compared to a third algorithm.

For the exact algorithm, matrices are taken to have rational integers and computations are done in the ring $\mathbb{Z}$.

The modular algorithm takes matrices as having p-adic integer entries and computations are done with rational integer approximation to p-adic integers. We give bounds on the precision necessary to ensure that the resulting cyclic form is indeed similar to the original matrix. We also give a criterion for deciding whether the cyclic form is correct.

These results have been accepted for publication in the Journal of Symbolic Computation [MF].

ACKNOWLEDGEMENTS

My deepest gratitude goes to Dr David Ford for his great help and support throughout this research. I want to thank him for sharing his knowledge and enthusiasm of the subject of Computational Number Theory with me, for giving me the opportunity to meet the best researchers in this field, and for financial support.

I also thank the staff from the Computer Science department for helping me with technical problems related to this research and the writing of this thesis.

I dedicate this thesis to Scott and my family whose confidence in me and constant encouragement brought me to the end of this project and the beginning of new projects.

CONTENTS

List of Symbols

$\mathbb{N}$: ring of natural numbers

$\mathbb{Z}$: ring of rational integers

$\mathbb{Q}$: field of rational numbers

$\mathbb{R}$: field of real numbers

$\mathbb{C}$: field of complex numbers

$\mathbb{Z}_p$: ring of p-adic integers

$\mathbb{Q}_p$: field of p-adic numbers

$\cdot \equiv \cdot \pmod{p^k}$: congruence modulo p^k

$\mathrm{ORD}_p(\cdot)$: order of a p-adic integer

$|\cdot|_p$: p-adic absolute value

$val(\cdot)$: valuation of a p-adic integer

$\mathbb{Z}_p^{n \times n}$: ring of $n \times n$ matrices having p-adic integer entries

I : identity matrix

T^* : adjoint of matrix T

$\mathrm{row}_j(\cdot)$: j^{th} row of a matrix

$\mathrm{col}_j(\cdot)$: j^{th} column of a matrix

$a_{i,j}$: matrix entry in row i and column j

$\mathrm{diag}(\cdot)$: diagonal matrix

$\det(\cdot)$: determinant of a matrix

$\|\cdot\|$: vector or matrix norm

$\sim$: matrix similarity

$F[x]$: polynomial in x with coefficients from field F

$\deg(\cdot)$: degree of a polynomial

$\dim(\cdot)$: dimension of a vector space

$\oplus$: direct sum of subspaces

$< \cdot >$: cyclic subspace generated by a vector

$\{\cdot\}$: set notation

$\cap$: intersection

$\in$: belongs to

$\subseteq$: included

$[\cdot, \cdot]$: closed interval

$(\cdot, \cdot)$: open interval

$\forall$: for all elements

∞ : positive infinity

$-\infty$: negative infinity

$\rightarrow$: mapping

$\int_a^b$: definite integral

P(< expression >) : probability of < expression >

$\mathcal{O}(\ )$: big O notation

$\binom{\cdot}{\cdot}$: number of combinations

$a \leftarrow b$: replacing a by b

$\max(a, b)$: maximum of a, b

$\min(a, b)$: minimum of a, b

$\lceil \cdot \rceil$: ceiling function

$\lfloor \cdot \rfloor$: floor function

$|\cdot|$: (real or complex) absolute value

$a|b$: a divides b

$a \nmid b$: a does not divide b

List of Tables

CHAPTER 1

INTRODUCTION

The two most important canonical forms for matrices are the Jordan form and the rational normal form.

These canonical forms enable us to compute characteristic roots easily and to determine if matrices are similar. The advantage of the Jordan form is that it displays the characteristic roots explicitly (but these values may lie in an extension of the original field). The advantage of the rational normal form is that its entries all lie in the ring generated by the entries of the original matrix; it can be constructed over any Euclidean ring (see [**Her, ch. 6,7**]).

Furthermore, the rational normal form gives us the invariant factors (and from them the characteristic roots) for any given linear transformation.

We say that two $n \times n$ matrices, A and B, are *similar*, denoted by $A \sim B$, if and only if there exists a nonsingular matrix T, such that

$$AT = TB. \tag{1}$$

Note that (1) is always true for $T = 0$; that's why it is indispensable that $\det(T) \neq 0$. To solve (1), we would have to solve an $n^2 \times n^2$ system of equations which would give a family of solutions for T. In general it would be extremely

difficult to determine if that family contains an invertible matrix T.

Theorem : *Two matrices A and B are similar if and only if they represent the same linear transformation with respect to (possibly) different bases.*

This is easily seen from the definition of a matrix in **[Her, ch. 6]**. □

It follows that matrix similarity is an equivalence relation.

We are concerned with the computation of the rational normal form of a matrix. We have found that it is sufficient to have

$$AT \equiv TB \pmod{p^\alpha}$$

$$TU \equiv p^\gamma I \pmod{p^\alpha}, \qquad \gamma < \alpha$$

and β (defined in §3.2) determined from A and B with $\beta < \alpha - \gamma$.

The first chapter introduces basic concepts and definitions. The second chapter presents p-adic integers. In chapter 3, we introduce the necessary background to understand the two new algorithms for computing the cyclic form of a matrix :

1) an exact algorithm (ch. 4)

2) a modular algorithm (ch. 5)

Chapter 6 describes the procedures used to generate the data used in testing the algorithms. The results are presented in chapter 7.

Finally, chapter 8 gives an example for which our modular algorithm performs particularly well. It also studies the behavior of the other algorithms with respect to that example.

1.1 Notation

We let V be a vector space over the field F and T a linear transformation on V. $V = V_1 \oplus V_2$ means that V is the direct sum of the subspace V_1 and V_2; i.e., every vector of V is the sum of a vector in V_1 and a vector in V_2 and $V_1 \cap V_2 = \mathbf{0}$, the zero vector. We denote by $f\mathbf{v}$, $f(T)$ applied to $\mathbf{v}$ where $f(x)$ is a polynomial. For any vector $\mathbf{v} \in V, < \mathbf{v} > = \{f\mathbf{v} \mid f \in F[x]\}$ is the cyclic subspace generated by the vector $\mathbf{v}$.

1.2 Definitions

Every $k \times k$ matrix $A = (a_{i,j})$ with entries in F has a unique monic polynomial, $m(x)$ in $F[x]$, of least degree for which A is a zero; i.e.

$$m(A) = m_0 I + m_1 A + m_2 A^2 + \cdots + m_r A^r = 0$$

(with $r \leq k$ and $m_r = 1$). It is called the "minimum polynomial" of the matrix.

Theorem : *The degree of the minimum polynomial of a matrix is the dimension of a maximal cyclic subspace.*

[HK, ch. 7] gives a similar theorem which proves this one. □

The "characteristic polynomial" of a matrix A is given by

$$c(x) = \det(xI - A) = c_k x^k + c_{k-1} x^{k-1} + \cdots + c_1 x + c_0 .$$

The roots of $c(x)$ are called the "characteristic roots" of the linear transformation represented by the matrix A. For any matrix, we have $m(x)|c(x)$.

The $k \times k$ matrix $B = (b_{i,j})$ is called a "companion matrix" if, for $1 \leq j \leq k-1, b_{i,j} = 1$ when $i = j+1$ and $b_{i,j} = 0$ otherwise; i.e.

$$B = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & b_{1,k} \\ 1 & 0 & 0 & \dots & 0 & b_{2,k} \\ 0 & 1 & 0 & \dots & 0 & b_{3,k} \\ 0 & 0 & 1 & \dots & 0 & b_{4,k} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & b_{k,k} \end{pmatrix} .$$

The minimum polynomial of a companion matrix coincides with its characteristic polynomial, which is

$$b(x) = x^k - b_{k,k}x^{k-1} - \cdots - b_{2,k}x - b_{1,k} .$$

We say a matrix is in "cyclic form" if it is a direct sum of companion matrices; i.e.

$$C = \begin{pmatrix} C_1 & 0 & \dots & 0 \\ 0 & C_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & C_r \end{pmatrix}$$

where C_i is a companion matrix having characteristic polynomial $p_i(x)$ and 0 is the zero matrix. A matrix is in "rational normal form" (RNF) if it is in cyclic form, and the characteristic polynomial of each of the companion matrices divides the characteristic polynomial of the next; i.e.

$$p_1 | p_2 | p_3 | \cdots | p_r$$

for the matrix above. These polynomials are called the "invariant factors" of the matrix. The rational normal form of a matrix gives both the minimum polynomial and the characteristic polynomial of that matrix; e.g. for the matrix above, its characteristic polynomial is given by

$$c(x) = p_1(x)p_2(x)\cdots p_r(x) ;$$

its minimum polynomial is given by

$$m(x) = p_r(x) \ .$$

Every matrix is similar to a unique matrix in rational normal form; so two matrices A and B are similar if and only if RNF(A) = RNF(B).

1.3 Cyclic form vs RNF

We now show that the invariant factors, and hence the rational normal form, can be easily computed by polynomial *gcd* and *lcm* operations on the characteristic polynomials of the components of any matrix in cyclic form.

If $V = V_1 \oplus V_2$, it follows that each vector $\mathbf{v} \in V$ is the sum of *unique* vectors $\mathbf{v}_1 \in V_1$ and $\mathbf{v}_2 \in V_2$:

$$\begin{aligned} \mathbf{v}_1 + \mathbf{v}_2 = \mathbf{w}_1 + \mathbf{w}_2 \ &\Rightarrow \ \mathbf{v}_1 - \mathbf{w}_1 = \mathbf{w}_2 - \mathbf{v}_2 \quad \in V_1 \cap V_2 \\ &\Rightarrow \ \mathbf{v}_1 - \mathbf{w}_1 = \mathbf{w}_2 - \mathbf{v}_2 = \mathbf{0} \\ &\Rightarrow \ \mathbf{w}_1 = \mathbf{v}_1, \ \mathbf{w}_2 = \mathbf{v}_2 \qquad \square \end{aligned}$$

We assume :

$V = < \mathbf{v}_1 > \oplus < \mathbf{v}_2 >$

$f_1 = \text{minpol}(\mathbf{v}_1)$ = the minimum polynomial of $\mathbf{v}_1$

$f_2 = \text{minpol}(\mathbf{v}_2)$

Let $g_1 = lcm(f_1, f_2)$, $g_2 = gcd(f_1, f_2)$, $h_1 = f_1/g_2$, $h_2 = f_2/g_2$. Compute a_1, a_2 so that $a_1 f_1 + a_2 f_2 = g_2$ (and so $a_1 h_1 + a_2 h_2 = 1$). Let $\mathbf{w}_1 = \mathbf{v}_1 - \mathbf{v}_2$, $\mathbf{w}_2 = (a_1 h_1)\mathbf{v}_1 + (a_2 h_2)\mathbf{v}_2$.

Claim 1 : $g_1 = minpol(\mathbf{w}_1)$.

PROOF :

1) $g_1\mathbf{w}_1 = g_1\mathbf{v}_1 - g_1\mathbf{v}_2$

$= f_1 h_2 \mathbf{v}_1 - f_2 h_1 \mathbf{v}_2$

$= \mathbf{0} - 0$

$= \mathbf{0}$

2) $p\mathbf{w}_1 = 0 \Rightarrow p\mathbf{v}_1 - p\mathbf{v}_2 = 0$

$\Rightarrow p\mathbf{v}_1 = \mathbf{0}, p\mathbf{v}_2 = \mathbf{0}$

$\Rightarrow f_1 | p,\ f_2 | p$

$\Rightarrow g_1 | p$ □

Claim 2 : $g_2 = minpol(\mathbf{w}_2)$.

PROOF :

1) $g_2\mathbf{w}_2 = a_1 h_1 g_2 \mathbf{v}_1 + a_2 h_2 g_2 \mathbf{v}_2$

$= a_1 f_1 \mathbf{v}_1 + a_2 f_2 \mathbf{v}_2$

$= a_1 \cdot \mathbf{0} + a_2 \cdot \mathbf{0}$

$= \mathbf{0}$

2) $p\mathbf{w}_2 = \mathbf{0} \Rightarrow pa_1 h_1 \mathbf{v}_1 + pa_2 h_2 \mathbf{v}_2 = \mathbf{0}$

$\Rightarrow pa_1 h_1 \mathbf{v}_1 = \mathbf{0},\ pa_2 h_2 \mathbf{v}_2 = \mathbf{0}$

$\Rightarrow f_1 | pa_1 h_1,\ f_2 | pa_2 h_2$

$\Rightarrow g_2 | pa_1 h_1,\ g_2 | pa_2 h_2$

$\Rightarrow g_2 | p(a_1 h_1 + a_2 h_2)$

$\Rightarrow g_2 | p \cdot 1$

$\Rightarrow g_2 | p$ □

Claim 3 : $< \mathbf{w_1} > \cap < \mathbf{w_2} > = \mathbf{0}$.

PROOF : Suppose $\mathbf{w} \in < \mathbf{w_1} > \cap < \mathbf{w_2} >$. Then $\mathbf{w} = p\mathbf{w}_1 = q\mathbf{w}_2$ for some polynomials p and q.

$$
\begin{aligned}
p\mathbf{w}_1 = q\mathbf{w}_2 &\Rightarrow p(\mathbf{v}_1 - \mathbf{v}_2) = q(a_1h_1\mathbf{v}_1 + a_2h_2\mathbf{v}_2) \\
&\Rightarrow p\mathbf{v}_1 - p\mathbf{v}_2 = qa_1h_1\mathbf{v}_1 + qa_2h_2\mathbf{v}_2 \\
&\Rightarrow (p - qa_1h_1)\mathbf{v}_1 = (p + qa_2h_2)\mathbf{v}_2 \\
&\Rightarrow (p - qa_1h_1)\mathbf{v}_1 = \mathbf{0},\ (p + qa_2h_2)\mathbf{v}_2 = \mathbf{0} \\
&\Rightarrow f_1 | p - qa_1h_1,\ f_2 | p + qa_2h_2 \\
&\Rightarrow g_2 | p - qa_1h_1,\ g_2 | p + qa_2h_2 \\
&\Rightarrow g_2 | (p + qa_2h_2) - (p - qa_1h_1) \\
&\Rightarrow g_2 | q(a_2h_2 + a_1h_1) \\
&\Rightarrow g_2 | q \cdot 1 \\
&\Rightarrow g_2 | q \\
&\Rightarrow q\mathbf{w}_2 = \mathbf{0} \\
&\Rightarrow \mathbf{w} = \mathbf{0} \qquad \square
\end{aligned}
$$

Claim 4 : $V = < \mathbf{w_1} > + < \mathbf{w_2} >$.

PROOF : Because $< \mathbf{w_1} > \cap < \mathbf{w_2} > = \mathbf{0}$, we have

$$\begin{aligned}
\dim(< \mathbf{w_1} > + < \mathbf{w_2} >) &= \dim(< \mathbf{w_1} >) + \dim(< \mathbf{w_2} >) \\
&= \deg(g_1) + \deg(g_2) \\
&= \deg(lcm(f_1, f_2)) + \deg(gcd(f_1, f_2)) \\
&= \deg(f_1) + \deg(f_2) \\
&= \dim(< \mathbf{v_1} >) + \dim(< \mathbf{v_2} >) \\
&= \dim(V) \qquad \square
\end{aligned}$$

The generalization for matrices having more than two components is straightforward.

So the problem of bringing a given matrix into rational normal form reduces to that of bringing the matrix into cyclic form. Therefore, from this point on, we restrict our attention to the problem of bringing a given matrix into cyclic form.

The two algorithms that we propose will solve that problem. For comparison purposes, we implemented the second algorithm of Ozello ([**Ozel**]), which computes the rational normal form of a given matrix with integer entries. So far as we know, Ozello has the best algorithm using rational integer computations.

Lüneberg's recent book ([**Lün**]) contains a bibliography and a survey of algorithms for computing the rational normal form. He mentions that no one has found an effective algorithm based on modular arithmetic to compute the rational normal form of a matrix. He cites difficulties arising from the existence of "bad

primes". A paper by Howell ([**How**]) describes an algorithm based on the Chinese Remainder Theorem which, as she mentions herself in the paper, fails in general to produce the rational normal form.

In the next chapter, we will introduce p-adic integers and their properties. We will see in chapters 3 and 5 how these properties can be applied to overcome the difficulties mentioned by Lüneberg.

CHAPTER 2

p-ADIC NUMBERS

We now introduce "p-adic integers" ($\mathbb{Z}_p$) and "p-adic numbers" ($\mathbb{Q}_p$) . To avoid confusion between members of $\mathbb{Z}_p$ and members of $\mathbb{Z}$, the latter will be referred as "rational integers".

For the present thesis, our interest lies mostly with p-adic integers; so this chapter is mostly devoted to p-adic integers with a few words about p-adic numbers. An accessible treatment of p-adic numbers is in [**Mah**].

2.1 Definitions

For a fixed prime number p, the field $\mathbb{Q}_p$ is defined as the set of all sums of the form

$$\psi = \sum_{j=f}^{\infty} r_j p^j \ ,$$

with $0 \leq r_j \leq p-1$ for all j, and $f \in \mathbb{Z}$.

Similarly, we define the ring $\mathbb{Z}_p$ to be the set of all sums of the form

$$\alpha = \sum_{j=0}^{\infty} r_j p^j \ , \tag{2.1.1}$$

with $0 \leq r_j \leq p-1$ for all j. The coefficients r_j are called the "p-adic digits" of α.

Alternatively, we will use

$$\cdots r_j r_{j-1} \cdots r_2 r_1 r_0{}_p$$

to denote

$$r_0 + r_1 p + r_2 p^2 + \cdots + r_{j-1} p^{j-1} + r_j p^j + \cdots$$

where p is called the base.

For α as in (2.1.1), we define

$$\begin{aligned} \alpha_k &= \sum_{j=0}^{k-1} r_j p^j \ , \\ \bar{\alpha} &= \sum_{j=0}^{\infty} (p-1-r_j) p^j \ . \end{aligned} \tag{2.1.2}$$

$\bar{\alpha}$ is called the "complement" of α.

Obviously here

$$\begin{aligned} &\alpha_k \in \mathbb{Z} \ , \\ &\alpha_k = \sum_{j=0}^{k-1} r_j p^j + \sum_{j=k}^{\infty} 0 p^j \in \mathbb{Z}_p \ . \end{aligned} \tag{2.1.3}$$

We say that two p-adic integers, α and β, are equal if their p-adic digits coincide. If only the first k digits of α and β coincide, i.e. $\alpha_k = \beta_k$, we write $\alpha \equiv \beta \pmod{p^k}$. In particular we have $\alpha \equiv \alpha_k \pmod{p^k}$.

If $\alpha_k = 0$ but $\alpha_{k+1} \neq 0$ (i.e., $r_k \neq 0$ but $r_j = 0$ for $0 \leq j < k$) then we write $ORD_p(\alpha) = k$ (by convention, $ORD_p(0) = \infty$). In more general terms, k is the highest power of p which divides α.

For any rational integer r, we define the p-adic absolute value of r as

$$|r|_p = p^{-ORD(r)} \ ; \qquad |0|_p = 0 \ .$$

Relative to the p-adic absolute value, every r and s in $\mathbb{Z}$ has the following properties :

i) $|r|_p = 1 \Leftrightarrow p \not| r$

ii) $|r|_p = p^f \Leftrightarrow |p^f r|_p = 1$

iii) for every integer n, $|p^n r|_p = p^{-n}|r|_p$

iv) $|r \pm s|_p \leq \max(|r|_p, |s|_p)$

v) $|rs|_p = |r|_p |s|_p$.

These properties will prove very useful later on. The p-adic absolute value is also defined for $x \in \mathbb{Q}$ and similar properties result (see [**Mah, ch.1**]). Details of the arithmetic operations in $\mathbb{Z}_p$ are given below in §2.3.

2.2 Units in $\mathbb{Z}_p$

In a ring R, a "unit" is an element $\alpha \in R$ for which there exists an element $\beta \in R$ such that $\alpha\beta = 1$; we write $\beta = \alpha^{-1}$.

Examples :

in $\mathbb{Z}$, the units are $1, -1$

in $\mathbb{Q}$, the units are $\mathbb{Q} - \{0\}$

Proposition : *$\alpha \in \mathbb{Z}_p$ is a unit if and only if $\alpha_1 \neq 0$.*

Proof : Suppose $\alpha_1 \neq 0$. Let β_1 be the solution of

$$1 \equiv \alpha_1 \beta_1 \pmod{p} .$$

For $k \geq 1$, suppose α_k and β_k have been found so that

$$1 \equiv \alpha_k \beta_k \pmod{p^k} .$$

Since $p^k \mid (1 - \alpha_k\beta_k)$ and $\alpha_{2k} \equiv \alpha_k \pmod{p^k}$ we have

$$p^{2k} \mid (1 - \alpha_{2k}\beta_k)^2 = 1 - \alpha_{2k}\beta_k(2 - \alpha_{2k}\beta_k) .$$

Therefore, if we take

$$\beta_{2k} \equiv \beta_k(2 - \alpha_{2k}\beta_k) \pmod{p^{2k}}$$

we will have

$$\beta_{2k} \equiv \beta_k \pmod{p^k}$$

and

$$1 \equiv \alpha_{2k}\beta_{2k} \pmod{p^{2k}} .$$

In this fashion we determine the p-adic digits of β, so that $\alpha\beta = 1$.

Conversely, suppose α is a unit in $\mathbb{Z}_p$. Let $\beta = \alpha^{-1}$. Then

$$1 \equiv \alpha_1\beta_1 \pmod{p} ,$$

and therefore $\alpha_1 \neq 0$. □

The proof gives us a method to find the inverse of any unit in $\mathbb{Z}_p$. Here is an example.

Example : Let $p = 7, \alpha = 5_7 \in \mathbb{Z}_7$. Find β such that $\alpha\beta = 1$.

Note that $\alpha = 5 + 0 \cdot p + 0 \cdot p^2 + 0 \cdot p^3 + \cdots$.

1) solving $\alpha_1\beta_1 \equiv 1 \pmod{p}$, we find $\beta_1 = 3$.

2) $k = 1$

$$\beta_{2k} = \beta_2 \equiv \beta_1(2 - \alpha_2\beta_1) \pmod{p^2}$$

$$\equiv 3 \cdot (2 - 5 \cdot 3) \pmod{7^2}$$

$$\equiv 3 \cdot (-13) \pmod{49}$$

$$\equiv -39 \pmod{49}$$

$$\beta_2 = 10 = 3 + 1 \cdot 7$$

so far, $\beta = 13_7$.

3) $k = 2$

$$\beta_{2k} = \beta_4 \equiv \beta_2(2 - \alpha_4\beta_2) \pmod{p^4}$$

$$\equiv 10 \cdot (2 - 5 \cdot 10) \pmod{7^4}$$

$$\equiv 10 \cdot (-48) \pmod{2401}$$

$$\equiv -480 \pmod{2401}$$

$$\beta_4 = 1921 = 3 + 1 \cdot 7 + 4 \cdot 7^2 + 5 \cdot 7^3$$

and so on .

In $\mathbb{Z}_7$, the p-adic digits of 5^{-1} are

$$\cdots 5413_7 .$$

Proposition : *For any $\alpha \in \mathbb{Z}_p$, $\alpha \in \mathbb{Q}$ if and only if the p-adic digits of α are eventually periodic.*

Proof : The proof is in two parts.

($\Leftarrow$) If α has periodic p-adic digits then

$$\alpha = \beta + \gamma$$

where β is the non-periodic finite part of α and γ is the periodic part.

Let δ be the k periodic p-adic digits so

$$\gamma = \delta + \gamma \cdot p^k$$

and

$$\gamma = \frac{\delta}{1 - p^k} \cdot$$

Since $\delta \in \mathbb{Z}$ and $1 - p^k \in \mathbb{Z}$ then $\gamma \in \mathbb{Q}$ and thus $\beta + \gamma \in \mathbb{Q}$ (since $\beta \in \mathbb{Z}$).

Therefore $\alpha \in \mathbb{Q}$.

($\Rightarrow$) If $\alpha \in \mathbb{Q}$ then $\alpha = \frac{r}{s}$ where $r,\ s \in \mathbb{Z},\ s \geq 1,\ (p, s) = 1$ (if $p^k \mid s$ then $\alpha = \frac{rp^k}{s'}$ where $s' = \frac{s}{p^k}$ so $(p, s') = 1$).

By the Euclidean algorithm, we can find integers $x,\ y$ such that $sx + py = 1$. Then

$$s(rx) + p(ry) = r\ ,$$

$$s(rx - kp) + p(ry + ks) = r$$

with integer k such that $0 \leq rx - kp \leq p - 1$.

Let $a_0 = rx - kp$ and $r_1 = ry + ks$ then

$$\frac{r}{s} = a_0 + p\frac{r_1}{s}\ .$$

Similarly,

$$\begin{aligned} \frac{r_1}{s} &= a_1 + p\frac{r_2}{s} \\ \frac{r_2}{s} &= a_2 + p\frac{r_3}{s} \end{aligned} \qquad (2.2.1)$$

$$\cdots$$

so that

$$\frac{r}{s} = a_0 + a_1 p + a_2 p^2 + \cdots + a_{n-1} p^{n-1} + p^n \frac{r_n}{s}$$

with $0 \leq a_j \leq p-1$ for $0 \leq j \leq n-1$.

Let $A_n = a_0 + a_1 p + a_2 p^2 + \cdots + a_{n-1} p^{n-1}$ then

$$\frac{r}{s} = A_n + p^n \frac{r_n}{s}\,, \qquad 0 \leq A_n \leq p^n - 1$$

and

$$r_n = \frac{r - A_n s}{p^n}\,, \qquad \frac{r-(p^n-1)s}{p^n} \leq r_n \leq \frac{r}{p^n}\,.$$

Therefore, for $n \to \infty$

$$-s \leq r_n \leq 0 \tag{2.2.2}$$

and r_n has only finitely many possibilities.

From (2.2.1), we see that

$$\frac{r_n}{s} = a_n + p \frac{r_{n+1}}{s}$$

so that a_n and r_{n+1} are uniquely determined from r_n. Thus, from (2.2.2), the sequences $r_1,\ r_2,\ r_3, \cdots$ and $a_0,\ a_1,\ a_2, \cdots$ must be periodic from some suffix onward. □

The periodic digits in a p-adic integer are denoted by a dot above the first and last digits; e.g. $\cdots 2541254125413_7$ will be written as $\dot{2}54\dot{1}3_7$.

From the proof, we see that given a periodic p-adic integer, we can compute which rational number it represents.

Example : What is the 7-adic integer $\dot{2}54\dot{1}3_7$?

Let $\beta = \dot{2}54\dot{1}3_7$, $\gamma = \dot{2}54\dot{1}_7$ so $\beta = 3 + \gamma \cdot 7$.

We have

$$\gamma = 2541_7 + \gamma \cdot 7^4 \qquad \text{or}$$

$$\gamma = 960 + \gamma \cdot 2401 \ .$$

Therefore

$$\gamma = -\frac{2}{5}$$

and

$$\beta = 3 - \frac{2}{5} \cdot 7 = \frac{1}{5}.$$

We thus find $\beta = 5^{-1}$.

2.3 Arithmetic in $\mathbb{Z}_p$

Arithmetic operations, addition, subtraction, multiplication and division are easily computed in $\mathbb{Z}_p$. They are defined as follows for any α and $\beta \in \mathbb{Z}_p$ where

$$\alpha = a_0 + a_1 p + \cdots + a_{n-1} p^{n-1} + \cdots$$

$$\beta = b_0 + b_1 p + \cdots + b_{n-1} p^{n-1} + \cdots$$

Addition : If $0 \leq a_j + b_j \leq p-1$ for all j then

$$\chi = \alpha + \beta = (a_0 + b_0) + (a_1 + b_1)p + \cdots + (a_{n-1} + b_{n-1})p^{n-1} + \cdots$$

$$= c_0 + c_1 p + \cdots + c_{n-1} p^{n-1} + \cdots$$

On the other hand, if $0 \leq a_j + b_j \leq p-1$ for $0 \leq j \leq k-1$ and $a_k + b_k > p$ then

$$c_0 = a_0 + b_0, \cdots, c_{k-1} = a_{k-1} + b_{k-1}$$

$$a_k + b_k = c_k + p$$

$$(a_k + b_k)p^k = (c_k + p)p^k = c_k p^k + 1 \cdot p^{k+1} \ .$$

Now add 1 to the next term, i.e. $c_{k+1} = a_{k+1} + b_{k+1} + 1$, and proceed to get χ. If the sum is finite ($\alpha = \alpha_n, \beta = \beta_n$) and $k = n-1$ then

$$\chi = c_0 + c_1 p + \cdots + c_{n-1} p^{n-1} + 1 \cdot p^n \ .$$

We see that addition is defined so that for all $k \geq 0$

$$(\alpha + \beta)_k \equiv \alpha_k + \beta_k \pmod{p^k} .$$

Subtraction :

From equation (2.1.2), we observe that

$$0 = 1 + \sum_{j=0}^{\infty}(p-1)p^j = 1 + \bar{0} ,$$

so that

$$-1 = \bar{0} .$$

In general, we have

$$\begin{aligned} \alpha + \bar{\alpha} &= \sum_{j=0}^{\infty} r_j p^j + \sum_{j=0}^{\infty}(p-1-r_j)p^j \\ &= \sum_{j=0}^{\infty}(p-1)p^j \\ &= \bar{0} = -1 , \end{aligned}$$

so that

$$-\alpha = 1 + \bar{\alpha} . \qquad (2.3.1)$$

Therefore,

$$\begin{aligned} \alpha - \beta &= \alpha + (-\beta) \\ &= \alpha + (1 + \bar{\beta}) \end{aligned}$$

i.e., we compute $\bar{\beta}$ and then proceed with the addition.

Multiplication :

For clarity's sake, we will now assume that

$$\alpha = \alpha_{n+1} = a_0 + a_1 p + \cdots + a_n p^n \ ,$$

$$\beta = \beta_{n+1} = b_0 + b_1 p + \cdots + b_n p^n \ .$$

Extension to infinite series is straightforward.

Let $\chi = \alpha\beta$, we proceed in a manner similar to polynomial multiplication so

$$\chi = (a_0 b_0) + (a_0 b_1 + a_1 b_0)p + (a_0 b_2 + a_1 b_1 + a_2 b_0)p^2 + \cdots + (a_n b_n)p^{2n} \ .$$

We find the p-adic digits of χ in the following manner :

$$c_0 + d_1 \cdot p = a_0 b_0 \qquad 0 \leq c_0 \leq p-1$$

$$c_1 + d_2 \cdot p = a_0 b_1 + a_1 b_0 + d_1 \qquad 0 \leq c_1 \leq p-1$$

$$c_2 + d_3 \cdot p = a_0 b_2 + a_1 b_1 + a_2 b_0 + d_2 \qquad 0 \leq c_2 \leq p-1$$

$$\cdots$$

and so on, so that

$$\chi = c_0 + c_1 p + \cdots + c_{2n} p^{2n} \ .$$

Multiplication is thus defined so that for all $k \geq 0$

$$(\alpha\beta)_k \equiv \alpha_k \beta_k \pmod{p^k} \ .$$

Division :

For any two p-adic integers α and β (with $\beta_1 \neq 0$), $\dfrac{\alpha}{\beta}$ can be rewritten as $\alpha\beta^{-1}$. Since β is a p-adic unit, we can find β^{-1} as shown in §2.2 and then use multiplication.

Here are some examples.

Let

$$\alpha = 13_7 \in \mathbb{Z}_7$$

and

$$\beta = 5_7 \in \mathbb{Z}_7 \ .$$

Addition :

$$\alpha + \beta = (3+5) + (1+0)\cdot 7$$

$$= (1 + 1\cdot 7) + 1\cdot 7$$

$$= 1 + 2\cdot 7$$

$$= 21_7 \in \mathbb{Z}_7$$

Subtraction :

$$\bar{\beta} = 1 + 6\cdot 7 + 6\cdot 7^2 + \cdots = \dot{6}1_7$$

$$-\beta = 1 + 1 + 6\cdot 7 + 6\cdot 7^2 + \cdots$$

$$= 2 + 6\cdot 7 + 6\cdot 7^2 + \cdots = \dot{6}2_7$$

$$\alpha - \beta = (3+2) + (1+6)\cdot 7 + 6\cdot 7^2 + 6\cdot 7^3 + \cdots$$

$$= 5 + 0\cdot 7 + 0\cdot 7^2 + 0\cdot 7^3 + \cdots$$

$$= 5_7 \in \mathbb{Z}_7$$

Multiplication :

$$\alpha\beta = (3 + 1\cdot 7) \times (5 + 0\cdot 7)$$

$$= 15 + (5+0)\cdot 7 + 0\cdot 7^2$$

$$15 = 1 + 2\cdot 7$$

$$5 + 2 = 0 + 1\cdot 7$$

$$1 + 0 = 1 + 0\cdot 7$$

so $\alpha\beta = 101_7 \in \mathbb{Z}_7$.

Division :

$$\beta^{-1} = \dot{2}54\dot{1}3_7$$

$$\alpha\beta^{-1} = (3 + 1 \cdot 7) \times (3 + 1 \cdot 7 + 4 \cdot 7^2 + 5 \cdot 7^3 + 2 \cdot 7^4 + \cdots)$$

$$= 9 + (3 + 3) \cdot 7 + (12 + 1) \cdot 7^2 + (15 + 4) \cdot 7^3 + (6 + 5) \cdot 7^4$$

$$+ (3 + 2) \cdot 7^5 + (12 + 1) \cdot 7^6 + \cdots$$

$$9 = 2 + 1 \cdot 7$$

$$3 + 3 + 1 = 0 + 1 \cdot 7$$

$$12 + 1 + 1 = 0 + 2 \cdot 7$$

$$15 + 4 + 2 = 0 + 3 \cdot 7$$

$$6 + 5 + 3 = 0 + 2 \cdot 7$$

$$3 + 2 + 2 = 0 + 1 \cdot 7$$

$$12 + 1 + 1 = 0 + 2 \cdot 7$$

from this point on, it is periodic with a period of 4 starting with the third term in the list above. So

$$\frac{\alpha}{\beta} = \bar{0}2_7$$

$$= 2_7 \in \mathbb{Z}_7 \ .$$

The results are easily verified in this case since

$$\alpha = 10 \quad \in \mathbb{Z}$$

$$\beta = 5 \quad \in \mathbb{Z}$$

so

$$\alpha+\beta = 1+2\cdot 7 = 15 \quad \in \mathbb{Z}$$

$$\alpha-\beta = 5 \quad \in \mathbb{Z}$$

$$\alpha\beta = 8+6\cdot 7 = 50 \quad \in \mathbb{Z}$$

$$\frac{\alpha}{\beta} = 2 \quad \in \mathbb{Z}\ .$$

2.4 Pertinent results

Here are some results arising directly from the definitions given in the first two section, the most important of which is that $\mathbb{Z} \subseteq \mathbb{Z}_p$.

This is easily seen since a positive rational integer α can be written (base p) in the form (2.1.1) with only finitely many non-zero digits (as shown in (2.1.3)). From (2.3.1), it follows that the negative rational integers are also present in $\mathbb{Z}_p$. From now on, we should think of $\mathbb{Z}$ as special p-adic integers that satisfy either

1) all but finitely many of its digits equal to zero ($\alpha \geq 0$)

2) all but finitely many of its digits equal to $p-1$ ($\alpha < 0$).

With respect to the p-adic absolute value, we find that $\mathbb{Z}$ is dense in $\mathbb{Z}_p$ (equivalently, $\mathbb{Z}_p$ is the *completion* of $\mathbb{Z}$).

Similarly, we can deduce that $\mathbb{Q} \subseteq \mathbb{Q}_p$ and regard $\mathbb{Q}_p$ as the completion of $\mathbb{Q}$ with respect to the p-adic absolute value.

For the second result we require the following definition :

Given K, a commutative ring with identity 1, and w, a mapping from K to the set of non-negative real numbers. If

1) $w(0) = 0$; $w(a) > 0$ if $a \in K - \{0\}$

2) $w(a \pm b) \leq w(a) + w(b)$, $\forall a, b \in K$

3) $w(ab) = w(a)w(b)$, $\forall a, b \in K$

then w is called a valuation.

In [**Mah, ch.2**] Mahler gives the construction of K_w, the completion of the field K with respect to the valuation w, as a set of residue classes modulo a prime ideal. It follows from this construction that every element of K_w can be approximated arbitrarily closely relative to w by elements of K.

From the definition of p-adic absolute value of a rational integer r and its properties, we see that $r \longrightarrow |r|_p$ is a valuation. Therefore, by the above statement, we know that any p-adic integer can be approximated "arbitrarily closely" by a rational integer. We say that x and y are "close" if $p^k \mid (x - y)$ for some large positive integer k. (The larger k, the closer x and y.)

This result is very important for implementation of algorithms which use modular arithmetic.

Two more facts should be stated here :

1) In $\mathbb{Z}_p$, $\alpha \mid \beta$ or $\beta \mid \alpha$ or $\alpha = \beta = 0$.

2) The p-adic Greatest Common Divisor and Least Common Multiple of two p-adic integers are powers of p.

Example :

$$x = p^r q, \ p \not| q$$

$$y = p^s t, \ p \not| t$$

$$gcd(x, y) = p^{\min(r,s)}$$

$$lcm(x, y) = p^{\max(r,s)} .$$

Since q and t are units, we ignore them.

More pertinent results from linear algebra may be found in [Sch, ch.4] and [HK].

At this point, one question that we may ask is "How big is $\mathbb{Z}_p$?". A standard diagonalization argument can be used to prove that $\mathbb{Z}_p$ is uncountable. It can similarly be shown that $\mathbb{Q}_p$ is an uncountable extension of $\mathbb{Q}$.

Modular arithmetic has many well known applications. One easy example is "two's complement arithmetic" which is actually 2-adic arithmetic with values computed modulo 2^k. (Typically, $k = 16$ or 32 .) By convention, the "sign bit" is propagated to ∞.

Another useful and well known application is Hensel's lemma which leads to factorization of polynomials in $\mathbb{Z}[x]$ (see [Zas]). We shall apply p-adic properties in a somewhat similar fashion in Chapter 3.

CHAPTER 3

BACKGROUND

From this point on, unless otherwise specified, matrices and vectors have p-adic integer entries.

The elementary row (column) operations on matrices from a ring R are defined as

(i) interchange two rows (columns)

(ii) multiplication of a row (column) by a unit of R

(iii) addition of μ times one row (column) to another row (column) (where $\mu \in$ R).

Matrices which effect those operations are called elementary matrices.

Similarly, we define elementary p-adic row (column) operations on p-adic matrices as

(i) interchange two rows (columns)

(ii) multiplication of a row (column) by a unit of $\mathbb{Z}_p$

(iii) addition of μ times one row (column) to another row (column) (where $\mu \in \mathbb{Z}_p$).

We also define elementary p-adic matrices as the matrices which effect elementary

p-adic operations.

The first two sections deal with the concept of p-adic "lifting" by which we mean iterative (linear) substitutions producing a p-adically convergent sequence. Lifting originates with Hensel (see the standard proof of Hensel's lemma from number theory [**VdW2, p. 204**]), and was applied by Zassenhaus in [**Zas**] to give the well known algorithm for factorization of polynomials with rational integer coefficients.

3.1 Lifting a system of linear congruences

The Smith normal form (SNF) of a rational integer matrix is defined as

$$\begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix}$$

where $D = \mathrm{diag}(\delta_1, \cdots, \delta_r)$, with $\delta_1, \cdots, \delta_r$ positive integers such that $\delta_1 | \delta_2 | \cdots | \delta_r$, and r is the rank of the matrix.

Every rational matrix A can be transformed into Smith normal form using elementary row and column operations. [**Newm, ch.II §15**] proves this and gives a simple method to compute the Smith normal form. Consequently, for any rational integer matrix A there are unimodular matrices L and R such that the product LAR is in Smith normal form ([**Sch, ch. 4**]).

Similarly, for any $k \times k$ matrix $C = (c_{i,j})$ ($\in \mathbb{Z}_p^{k \times k}$) there exist $k \times k$ elementary p-adic matrices L and R each with determinant a p-adic unit (thus invertible in $\mathbb{Z}_p^{k \times k}$ by Cramer's rule (see [**ND, p. 207-208**])) such that the product LCR is in Smith normal form, with its non-zero entries all being powers of p.

To see that, assume $C \neq 0$ and suppose we have applied row and column exchanges so that

$$ORD_p(c_{1,1}) = \min\{ORD_p(c_{i,j}) \mid 1 \leq i,j \leq k\} .$$

Let $c_{1,1} = p^{\beta_1}q$ (q a unit) and write

$$C = \begin{pmatrix} p^{\beta_1}q & d \\ e & D' \end{pmatrix}$$

where d is a $1 \times (k-1)$ row vector, e is a $1 \times (k-1)$ column vector, and D' is a $(k-1) \times (k-1)$ submatrix.

Multiplying the first row by q^{-1} we get

$$C = \begin{pmatrix} p^{\beta_1} & q^{-1}d \\ e & D' \end{pmatrix}$$

and, by minimality of $c_{1,1}$, we can get

$$C = \begin{pmatrix} p^{\beta_1} & 0 \\ 0 & D' \end{pmatrix}$$

using elementary p-adic operation (iii). Again by minimality of $c_{1,1}$, p^{β_1} divides every entry of D'.

By induction, D' can be transformed by elementary p-adic operations into Smith normal form

$$\begin{pmatrix} D'' & 0 \\ 0 & 0 \end{pmatrix}$$

with $D'' = \text{diag}(p^{\beta_2}, \cdots, p^{\beta_r})$, so that we finally have

$$LCR = \begin{pmatrix} p^{\beta_1} & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & p^{\beta_2} & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & p^{\beta_r} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \end{pmatrix} \tag{3.1.1}$$

with $\beta_1 \le \beta_2 \le \cdots \le \beta_r = \beta$.

Let

$$D = \begin{pmatrix} p^{\beta-\beta_1} & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & p^{\beta-\beta_2} & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & p^{\beta-\beta_r} & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 1 \end{pmatrix}$$

then

$$DLCR = \begin{pmatrix} p^{\beta} & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & p^{\beta} & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & p^{\beta} & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \end{pmatrix} = \begin{pmatrix} p^{\beta} I_r & 0 \\ 0 & 0 \end{pmatrix}$$

where I_r is the $r \times r$ identity matrix and 0 is the zero matrix, so

$$(DLCR)^2 = p^{\beta}(DLCR) .$$

Therefore

$$0 = p^{\beta} DLCR - DLCRDLCR$$

$$= DL(p^{\beta} C - CRDLC)R$$

with D,L,R invertible, so that

$$0 = p^{\beta} C - CRDLC$$

$$= C(p^{\beta} I - RDLC) .$$

Defining $\hat{C} = RDLC$, we have

$$C(p^{\beta} I - \hat{C}) = 0 .$$

Suppose we have a solution vector X for the $k \times k$ system of linear congruences

$$CX \equiv 0 \pmod{p^{\alpha}} . \qquad (3.1.2)$$

Then

$$CX = p^{\alpha}Y \ ,$$

for some vector Y, and

$$\begin{aligned} \hat{C}X &= RDLCX \\ &= p^{\alpha}RDLY \\ &= p^{\alpha}W \ . \end{aligned}$$

If $\beta < \alpha$, we may set

$$\begin{aligned} \hat{X} &= X - p^{\alpha-\beta}W \\ &= p^{-\beta}(p^{\beta}I - \hat{C})X \end{aligned} \tag{3.1.3}$$

so that

$$\begin{aligned} C\hat{X} &= p^{-\beta}C(p^{\beta}I - \hat{C})X \\ &= 0 \end{aligned}$$

and

$$\hat{X} \equiv X \pmod{p^{\alpha-\beta}} \ .$$

In other words, for any square matrix C, if α is sufficiently large, any solution modulo p^{α} can be lifted to an exact solution in $\mathbb{Z}_p$. Loosely speaking, β measures the loss of precision in lifting.

3.2 Lifting a matrix similarity congruence

Now suppose we have the congruence

$$AT \equiv TB \pmod{p^{\alpha}} \tag{3.2.1}$$

$$TU \equiv p^{\gamma}I \pmod{p^{\alpha}} \tag{3.2.2}$$

where A, T, B, and U are $n \times n$ matrices, B is in cyclic form and $\alpha > \gamma$. Let the $n^2 \times 1$ matrix X be defined by

$$X = \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_n \end{pmatrix}$$

where t_i is the i^{th} column of T, and

$$C = \begin{pmatrix} A - b_{1,1}I & -b_{2,1}I & \dots & -b_{n,1}I \\ -b_{1,2}I & A - b_{2,2}I & \dots & -b_{n,2}I \\ \vdots & \vdots & \ddots & \vdots \\ -b_{1,n}I & -b_{2,n}I & \dots & A - b_{n,n}I \end{pmatrix} \tag{3.2.3}$$

where $B = (b_{i,j})$. Then the system (3.1.2) is equivalent to the congruence (3.2.1) with $k = n^2$.

We determine β as in (3.1.1) and $\hat{X}$ (and the corresponding $n \times n$ matrix $\hat{T}$) as in (3.1.3) so that

$$A\hat{T} = \hat{T}B \ ,$$

$$\hat{T} \equiv T \pmod{p^{\alpha-\beta}} \ ,$$

$$\hat{T}U \equiv p^{\gamma}I \pmod{p^{\alpha-\beta}} \ .$$

If $\gamma < \alpha - \beta$, it follows that $\hat{T}$ is non-singular, and therefore that A and B are similar.

(In practice we do not actually compute $\hat{T}$ — in general the entries of $\hat{T}$ are non-rational p-adic integers — but as long as we know that β is small enough we know that $\hat{T}$ exists and lifting is possible.)

Now we know that for any matrices A and B (in cyclic form), if α is sufficiently large, any solution T modulo p^{α} can be lifted to an exact solution in $\mathbb{Z}_p$. Here again, we can say that β measures the loss of precision in lifting.

We now show how to improve the precision of congruences (3.2.1) and (3.2.2).

Theorem 3.2.1 : *If* $TU \equiv p^c I \pmod{p^k}$, $k > c$ *then*

$$\hat{U} = U(2I - p^{-c}TU) \tag{3.2.4}$$

implies

1) $T\hat{U} \equiv p^c I \pmod{p^{2k-c}}$

2) $\hat{U} \equiv U \pmod{p^{k-c}}$

3) $p^c T^{-1} \in \mathbb{Z}_p^{n\times n}$

4) $\hat{U} \equiv p^c T^{-1} \pmod{p^{k-c}}$

i.e. $\hat{U}$ *can be made as precise an approximation to* p^cT^{-1} *as we want.*

PROOF :

1)

$$TU \equiv p^c I \pmod{p^k}$$

$$p^c I - TU \equiv 0 \pmod{p^k}$$

$$(p^c I - TU)^2 \equiv 0 \pmod{p^{2k}}$$

$$p^{2c} I - 2p^c TU + TUTU \equiv 0 \pmod{p^{2k}}$$

$$2p^c TU - TUTU \equiv p^{2c} I \pmod{p^{2k}}$$

$$2TU - p^{-c}TUTU \equiv p^c I \pmod{p^{2k-c}}$$

$$TU(2I - p^{-c}TU) \equiv p^c I \pmod{p^{2k-c}}$$

$$T\hat{U} \equiv p^c I \pmod{p^{2k-c}} .$$

2)

$$\hat{U} = U(2I - p^{-c}TU)$$

$$= U + U - p^{-c}UTU$$

$$= U + U(I - p^{-c}TU)$$

$$= U + U\frac{(p^c I - TU)}{p^c} .$$

But $\frac{p^c I - TU}{p^c} \equiv 0 \pmod{p^{k-c}}$, therefore

$$\hat{U} \equiv U \pmod{p^{k-c}} .$$

3) Repeatedly substitute

$$U \leftarrow U(2I - p^{-c}TU)$$

$$k \leftarrow 2k - c$$

until $p^k \not| \det(T)$ so $p^k T^{-1} \in \mathbb{Z}_p^{n\times n}$, then

$$TU = p^c I + p^k W$$

$$= p^c T^{-1} T + p^k W$$

$$p^c T^{-1} = U - p^k T^{-1} W \in \mathbb{Z}_p^{n\times n} .$$

4)

$$TU = p^c I + p^k W$$

$$U = p^c T^{-1} + p^k T^{-1} W$$

$$= p^c T^{-1} + p^{k-c}(p^c T^{-1} W)$$

$$\equiv p^c T^{-1} \pmod{p^{k-c}} \qquad \square$$

Theorem 3.2.2 : *Given the congruences (3.2.1) and (3.2.2), if* $\gamma < \alpha \leq \beta + \gamma$ *then we may*

1) lift U, using (3.2.4), until

$$T\hat{U} \equiv p^\gamma I \pmod{p^{2\alpha+3\gamma}}$$

2) substitute

$$\hat{B} \leftarrow p^{-\gamma}\hat{U}AT \tag{3.2.5}$$

so that

(i) $AT \equiv T\hat{B} \pmod{p^{2\alpha+\gamma}}$

(ii) $\hat{B} \equiv B \pmod{p^{\alpha-\gamma}}$

PROOF :

(i)

$$p^\gamma \hat{B} = \hat{U}AT$$

$$p^\gamma T\hat{B} = T\hat{U}AT$$

$$\equiv T\hat{U}AT \pmod{p^{2\alpha+3\gamma}}$$

$$\equiv p^\gamma TT^{-1}AT \pmod{p^{2\alpha+2\gamma}}$$

$$\equiv p^\gamma AT \pmod{p^{2\alpha+2\gamma}}$$

so $T\hat{B} \equiv AT \pmod{p^{2\alpha+\gamma}}$.

(ii)

$$p^\gamma \hat{B} = \hat{U}AT$$

$$\equiv \hat{U}TB \pmod{p^\alpha}$$

$$\equiv p^\gamma B \pmod{p^\alpha}$$

so $\hat{B} \equiv B \pmod{p^{\alpha-\gamma}}$ □

It is important to note that even though B is in cyclic form, $\hat{B}$ (3.2.5) may only be *congruent* to a cyclic form, i.e. some work may need to be done on $\hat{B}$ to bring it into cyclic form.

3.3 Similarity with a matrix in cyclic form

Recall that β is important since it gives us a criterion for determining if an approximate similarity can be lifted to an exact similarity (namely $\beta < \alpha - \gamma$). The computation of β for the matrix $\mathcal{J}$ (3.2.3) involves bringing an $n^2 \times n^2$ matrix into Smith normal form which in general requires $\mathcal{O}(n^6)$ operations.

Making the assumption that $B = (b_{i,j})$ is a companion matrix, the matrix C has the form

$$\begin{pmatrix} A & -I & 0 & \dots & 0 & 0 & 0 \\ 0 & A & -I & \dots & 0 & 0 & 0 \\ 0 & 0 & A & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & A & -I & 0 \\ 0 & 0 & 0 & \dots & 0 & A & -I \\ -b_{1,n}I & -b_{2,n}I & -b_{3,n}I & \dots & -b_{n-2,n}I & -b_{n-1,n}I & A - b_{n,n}I \end{pmatrix}. \quad (3.3.1)$$

We can use elementary row and column operations to bring C into the form

$$\begin{pmatrix} I & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & I & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & I & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & I & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & I & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & f(A) \end{pmatrix}$$

where $f(x)$ is the characteristic polynomial of B. It is evident that the Smith normal form of $f(A)$ is equivalent to the Smith normal form of C. Therefore, β can be determined from the Smith normal form of $f(A)$.

We should note here that the matrix $f(A)$ is of size n so only $\mathcal{O}(n^3)$ operations are required to bring it into Smith normal form. Taking into account that computing $f(A)$ requires $\mathcal{O}(n^4)$ operations, we can determine β in $\mathcal{O}(n^4)$ operations (compared to $\mathcal{O}(n^6)$).

In the case of B in cyclic form, the matrix C will be the direct sum of matrices of the type (3.3.1). The value of β is the maximum value of β_i determined from each summand of C individually.

3.4 Estimating the entries of the Smith normal form

In this section, we assume that matrices and vectors have rational integer entries.

For $A = (a_{i,j})$ we take

$$\|A\| = (\sum_i \sum_j a_{i,j}^2)^{\frac{1}{2}} ,$$

and for $U = (u_i)$ we take

$$\|U\| = (\sum_i u_i^2)^{\frac{1}{2}} .$$

We denote the j^{th} column of the matrix M by M_j.

The Schwarz inequality ([**Her, p. 153**]) states that for any two vectors $u,\ v \in V$

$$| < u, v > | \leq \|u\| \ \|v\|$$

where $< \cdot, \cdot >$ denotes the dot product of two vectors. It follows that, for any column vector W,

$$\begin{aligned} \|AW\| &= (\sum_i (\sum_j a_{i,j} w_j)^2)^{\frac{1}{2}} \\ &\leq (\sum_i ((\sum_j a_{i,j}^2)(\sum_j w_j^2)))^{\frac{1}{2}} \\ &= ((\sum_i (\sum_j a_{i,j}^2)) \ (\sum_j w_j^2))^{\frac{1}{2}} \\ &= (\sum_i \sum_j a_{i,j}^2)^{\frac{1}{2}} \ (\sum_j w_j^2)^{\frac{1}{2}} \\ \|AW\| &\leq \|A\| \ \|W\| . \end{aligned} \tag{3.4.1}$$

We know that

$$\|A_j\| \leq \|A\| ,$$

by definition of $\|\cdot\|$. From (3.4.1), we get

$$\|(A^2)_j\| = \|AA_j\| \leq \|A\| \, \|A_j\|$$

so

$$\|(A^2)_j\| \leq \|A\|^2 \, .$$

By induction, we get

$$\begin{aligned} \|(A^k)_j\| &= \|A(A^{k-1})_j\| \\ &\leq \|A\| \, \|(A^{k-1})_j\| \\ &\leq \|A\| \, \|A\|^{k-1} \\ \|(A^k)_j\| &\leq \|A\|^k \end{aligned}$$

for $k \geq 0$.

Let $F = f(A)$ where

$$f(x) = d_0 + d_1 x + \cdots + d_k x^k \qquad \in \mathbb{Z}[x] \, .$$

Then

$$\begin{aligned} \|F_j\| &= \|d_0 I_j + d_1 A_j + \cdots + d_k (A^k)_j\| \\ &\leq \|d_0 I_j\| + \|d_1 A_j\| + \cdots + \|d_k (A^k)_j\| \\ \|F_j\| &\leq |d_0| + |d_1| \, \|A\| + \cdots + |d_k| \, \|A\|^k \, . \end{aligned} \tag{3.4.2}$$

The diagonal entries of the Smith normal form of F can be expressed in terms of the subdeterminants of F (see [**Newm, p. 28**] and [**Sch, ch. 4**]). Hadamard's bound on determinants ([**Buch, p. 259**]) states that, for any matrix A

$$|\det(A)| \leq \prod_j \|A_j\| \, .$$

Let

$$\mathrm{SNF}(F) = \begin{pmatrix} \delta_1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \delta_r & 0 & \dots & 0 \\ 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 \end{pmatrix}$$

then

$$\delta_i \leq \prod_{\substack{j=1 \\ \|F_j\| \neq 0}}^{n} \|F_j\| \quad , \; i = 1, \cdots, r$$

where the $\|F_j\|$'s are bounded according to inequality (3.4.2).

We could use this last inequality to estimate β in §3.3 but, since this is usually a big over-estimate, we might end up lifting an exact similarity (useless work !).

3.5 Estimating the entries of a cyclic form

We now assume that the entries of the matrices A and B are rational integers, and that B is in cyclic form.

Let the characteristic polynomial of A be given by

$$c(x) = \det(xI - A) = c_0 + c_1 x + \cdots + c_n x^n \tag{3.5.1}$$

and let

$$b(x) = \prod_j (x + \|A_j\|) = b_0 + b_1 x + \cdots + b_n x^n \; . \tag{3.5.2}$$

To get c_k, the coefficient of x^k in (3.5.1), we need exactly k of the x's that appear on the diagonal of $(xI - A)$. For each choice of k x's , the corresponding k rows and columns of A are excluded from the factors contributing to the corresponding term of c_k.

There are $\binom{n}{k}$ ways to choose the x's, leaving a $(n-k)\times(n-k)$ subdeterminant in each case, so that $\pm c_k$ is given by the sum of these subdeterminants. Suppose for example that we are left with rows 1 through $(n-k)$ (and columns 1 through $(n-k)$), then Hadamard's bound $(= \|A_1\| \cdots \|A_{n-k}\|)$ can be applied and, since we only need $(n-k)$ entries from each column 1 through $(n-k)$, we get an upper bound for the absolute value of c_k.

So, we now have

$$|c_0| \leq \prod_{j=1}^{n} \|A_j\| ,$$

$$|c_1| \leq \sum_{i=1}^{n} \Big(\prod_{\substack{j=1 \\ j\neq i}}^{n} \|A_j\| \Big) ,$$

$$\cdots$$

$$|c_{n-1}| \leq \sum_{i=1}^{n} \|A_j\| ,$$

$$|c_n| = 1 .$$

It is therefore evident, by the definition of $b(x)$, that $|c_j| \leq b_j$ for $0 \leq j \leq n$.

Given

$$d(x) = d_0 + d_1 x + \cdots + d_k x^k ,$$

a divisor of $c(x)$ (with $k \leq n$), Mignotte's bound ([**Mig**]) directly gives

$$\begin{aligned} \sum_j |d_j| &\leq 2^k \, (\sum_j c_j^2)^{\frac{1}{2}} \\ &\leq 2^k \, (\sum_j b_j^2))^{\frac{1}{2}} \\ &\leq 2^k \sum_j b_j \\ \sum_j |d_j| &\leq 2^k \prod_j (1 + \|A_j\|) . \end{aligned} \tag{3.5.3}$$

Therefore, when $k = n$, (3.5.3) gives us an upper bound for the entries of a cyclic form of A.

Furthermore, if $d(x) \mid c(x)$ then $d(\|A\|x)$ must divide $c(\|A\|x)$ so, from Mignotte's bound ([**Mig**]), we have

$$\begin{aligned}
\sum_j |d_j| \, \|A\|^j &\leq 2^k \, (\sum_j (c_j \|A\|^j)^2)^{\frac{1}{2}} \\
&\leq 2^k \, (\sum_j (b_j \|A\|^j)^2)^{\frac{1}{2}} \\
&\leq 2^k \, \sum_j b_j \|A\|^j \\
&= 2^k \, \prod_j (\|A\| + \|A_j\|) \\
&\leq 2^k \, \prod_j 2\|A\| \\
&\leq 2^n \, \prod_j 2\|A\| \\
\sum_j |d_j| \, \|A\|^j &\leq 4^n \|A\|^n \, . \qquad (3.5.4)
\end{aligned}$$

The next chapter gives an algorithm to compute the cyclic form of a matrix using rational integer computations. Chapter 5 will combine the cyclic algorithm from chapter 4 and the material covered in this chapter to give the p-adic algorithm to compute the cyclic form of a matrix. The last two sections of this chapter will be used in the analysis of the complexity of the p-adic algorithm.

CHAPTER 4

EXACT CYCLIC FORM ALGORITHM

A well known algorithm for computing the characteristic polynomial of a matrix A is due to Danilevskii [**Fad**]. Algorithms derived from Danilevskii's algorithm can be used to find matrices B and T satisfying $AT = TB$, with B in rational normal form. Ozello developed such an algorithm in [**Ozel**]. In general, these algorithms require $\mathcal{O}(n^4)$ operations (additions and multiplications) to produce the rational normal form. (Here we neglect the effect of the size of the numbers.)

The algorithm that follows is similar to the original algorithm of Danilevskii. Given a matrix A with rational integer entries, the algorithm finds matrices B and T, with B in cyclic form. All operations are performed in $\mathcal{Q}$, the field of rational numbers (thus the algorithm is exact). Although the performance of this algorithm is much worse than that of Ozello's algorithm, it is of interest because it yields a superior modular algorithm.

We present the details of the exact algorithm now and in the next chapter we will draw a parallel between the modular algorithm and the exact one.

4.1 The algorithm

We use three elementary similarity transformations :

$$E_1(i,j) : \text{row } i \leftrightarrow \text{row } j \text{ , and}$$

$$\text{col } i \leftrightarrow \text{col } j \text{ ;}$$

$$E_2(i,j,q) : \text{row } i \leftarrow \text{row } i - q \times \text{row } j \text{ , and}$$

$$\text{col } j \leftarrow \text{col } j + q \times \text{col } i \text{ ;}$$

$$E_3(i,q) : \text{row } i \leftarrow q^{-1} \times \text{row } i \text{ , and}$$

$$\text{col } i \leftarrow q \times \text{col } i \text{ ;}$$

where row i (col i) is the i^{th} row (column) of a matrix and $q \in \mathbb{Q}$.

Given h, we consider the $k \times k$ matrix A to be composed of four sub-matrices :

$$A = \begin{pmatrix} Y & P \\ Q & X \end{pmatrix}$$

where Y is $h \times h$, X is $(k-h) \times (k-h)$, P is $h \times (k-h)$, and Q is $(k-h) \times h$.

We define the configurations "Q clear" and "P clear" as follows. (* indicates an arbitrary value; – indicates zero.)

Q clear :

$$\begin{pmatrix} * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ & & & & & & & & \\ * & * & * & * & * & * & * & * & * \\ - & - & - & - & - & 1 & * & * & * \\ - & - & - & - & - & - & 1 & * & * \\ - & - & - & - & - & - & - & 1 & * \end{pmatrix}$$

Q is zero if is it clear and its first row is zero.

P clear :

$$\begin{pmatrix} * & * & * & * & * & - & - & - & * \\ * & * & * & * & * & - & - & - & * \\ * & * & * & * & * & - & - & - & * \\ * & * & * & * & * & - & - & - & * \\ * & * & * & * & * & - & - & - & * \\ & & & & & & & & \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & 1 & * & * & * \\ * & * & * & * & * & - & 1 & * & * \\ * & * & * & * & * & - & - & 1 & * \end{pmatrix}$$

P is zero if it is clear and its last column is zero.

At any time during the computation, we will be in one of three situations :

1) Q is not zero, and either Q is clear or P is zero

2) P is not zero, and either P is clear or Q is zero (4.1.1)

3) P and Q are both zero.

Initially, $h = k - 1$ so both P and Q are clear (and possibly either or both of P and Q are zero).

We repeatedly apply the following steps until $h = 0$.

1) If Q is not zero:

If Q is not clear, we apply E_2 repeatedly between the columns of X and the columns of Q so as to leave Q clear. These operations do not affect P, but could leave $Q = 0$ (which would leave us in situation 3).

If $Q \neq 0$:

Determine j such that $a_{h+1,j} \neq 0$ has minimal absolute value among the non-zero elements of Q. If $j \neq h$, apply $E_1(j, h)$ to bring $a_{h+1,j}$ in position $(h + 1, h)$.

Apply $E_3(h, a_{h+1,h})$ to put a 1 in position $(h + 1, h)$.

Apply E_2 repeatedly among the columns of Q, to achieve the form

$$\begin{pmatrix} * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ & & & & & & & & \\ - & - & - & - & 1 & * & * & * & * \\ - & - & - & - & - & 1 & * & * & * \\ - & - & - & - & - & - & 1 & * & * \\ - & - & - & - & - & - & - & 1 & * \end{pmatrix} \tag{4.1.2}$$

Replace h by $h-1$ to re-define X, Y, P, Q. At this point, either Q is clear and not zero (in which case, we remain in situation 1) or Q is zero (in which case, we are in situation 2 or 3 depending on the effect those operations had on P).

2) If P is not zero:

If P is not clear, we apply E_2 repeatedly between the rows of X and the rows of P so as to leave P clear. These operations do not affect Q, but could leave $P = 0$ (which would leave us in situation 3).

If $P \neq 0$:

Determine i such that $a_{i,k} \neq 0$ has minimal absolute value among the non-zero elements of P. If $i \neq h$, apply $E_1(i, h)$ to bring $a_{i,k}$ in position (h, k).

Apply $E_3(h, a_{h,k})$ to put a 1 in position (h, k).

Apply E_2 repeatedly among the rows of P, to achieve the form

$$\begin{pmatrix} * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ & & & & & & & & \\ * & * & * & * & * & - & - & - & 1 \\ & & & & & & & & \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & 1 & * & * & * \\ * & * & * & * & * & - & 1 & * & * \\ * & * & * & * & * & - & - & 1 & * \end{pmatrix}$$

Finally, apply E_1 repeatedly to perform adjacent row and column interchanges until row h has been moved below row k (column h has moved to the right of column k).

$$\begin{pmatrix} * & * & * & * & - & - & - & - & * \\ * & * & * & * & - & - & - & - & * \\ * & * & * & * & - & - & - & - & * \\ * & * & * & * & - & - & - & - & * \\ & & & & & & & & \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & 1 & * & * & * & * \\ * & * & * & * & - & 1 & * & * & * \\ * & * & * & * & - & - & 1 & * & * \\ & & & & & & & & \\ * & * & * & * & - & - & - & 1 & * \end{pmatrix} \tag{4.1.3}$$

Replace h by $h-1$ to re-define X, Y, P, Q. At this point, either P is clear and not zero (in which case, we remain in situation 2) or P is zero (in which case, we are in situation 1 or 3 depending on the effect those operations had on Q).

3) If P and Q are both zero, we apply E_2 repeatedly to the rows and columns of X to bring X into companion form. (These operations do not affect P or Q.) X is now a direct summand of A and plays no further part in the computation. If

$h = 0$, A is in cyclic form and we are done. Otherwise, replace k by h, h by $h - 1$, and proceed to bring the sub-matrix Y into cyclic form.

The algorithm is short and simple (thus easy to implement). We will now show that it is also correct.

4.2 Correctness

Algorithm correctness means that the algorithm terminates and gives the mandated result given any proper input.

The cyclic algorithm has as input a square matrix of size k. Initially, we set $h = k - 1$. At any time during the computation we are in one of three situations (4.1.1). In each situation the value of h is decreased by 1 after performing the described operations. Eventually, h will be equal to 0 and the algorithm will terminate.

We know that a matrix is in cyclic form if it is a direct summand of companion blocks. (Recall that a companion block has 1's below the main diagonal and 0's elsewhere except maybe for the last column.)

In situation 1, if Q is not clear we have

$$\begin{pmatrix} * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ & & & & & & & & \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & 1 & * & * & * \\ * & * & * & * & * & - & 1 & * & * \\ * & * & * & * & * & - & - & 1 & * \end{pmatrix}$$

and repeated applications of E_2 between the columns of X and Q will leave Q clear (or zero) but will not affect P or the lower triangular part of X. At this point, if Q is not zero we can easily see that the subsequent operations leave 1's below the main diagonal and 0's below the first row of Q as shown in (4.1.2) (which implies 0's in the lower triangular part of X).

In situation 2, if P is not clear we have

$$\begin{pmatrix} * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * \\ & & & & & & & & \\ - & - & - & - & - & * & * & * & * \\ - & - & - & - & - & 1 & * & * & * \\ - & - & - & - & - & - & 1 & * & * \\ - & - & - & - & - & - & - & 1 & * \end{pmatrix}$$

and repeated applications of E_2 between the rows of X and the rows of P will leave P clear (or zero) but will not affect Q or the lower triangular part of X. At this point, if P is not zero we can easily see that the subsequent operations leave 1's below the main diagonal and 0's in the lower triangular part of X as shown in (4.1.3).

Therefore, in the first two stages we gradually bring X into "near companion" form; namely

$$\begin{pmatrix} * & * & * & * \\ 1 & * & * & * \\ - & 1 & * & * \\ - & - & 1 & * \end{pmatrix}.$$

In situation 3, we have

$$\begin{pmatrix} * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ * & * & * & * & * & - & - & - & - \\ & & & & & & & & \\ - & - & - & - & - & * & * & * & * \\ - & - & - & - & - & 1 & * & * & * \\ - & - & - & - & - & - & 1 & * & * \\ - & - & - & - & - & - & - & 1 & * \end{pmatrix}$$

with X in near companion form. Repeated applications of E_2 to the rows and columns of X will bring X into the desired companion form without affecting P or Q.

Therefore the last stage gives us one more companion block. Then we proceed if necessary to build the next companion block. Thus eventually, the input matrix is transformed into a direct summand of companion blocks — a cyclic form.

Now, since elementary similarity transformations preserve similarity, the resulting cyclic form is similar to the input matrix, and is therefore a cyclic form for the input matrix.

We have thus proved that the exact cyclic form algorithm is correct. We will see, in the next chapter, how we can use this algorithm in association with the results from chapter 3 to obtain an efficient modular algorithm to compute the cyclic form of a matrix.

CHAPTER 5

MODULAR ALGORITHM

As mentioned in chapter 4, algorithms derived from the Danilevskii's algorithm for the charateristic polynomial can be used to find B and T satisfying (3.2.1), with B in cyclic form (see [**How**], [**Lün**], [**Ozel**]).

We will use a modified version of the algorithm presented in chapter 4 (see §5.1) to find matrices B (explicitly), T and U (implicitly), and an exponent γ, with B in cyclic form, such that (3.2.1) and (3.2.2) are satisfied with $\alpha = \mu - \gamma$.

In §5.2, we describe how to use the results from chapter 3 in conjunction with the algorithm from §5.1 to compute the rational form. Sections 5.3 and 5.4 deal with the correctness and performance of the modular algorithm. The last section presents an alternate algorithm less efficient than the modular algorithm but still worth investigating.

5.1 The modular cyclic form algorithm

The algorithm that follows is a modified version of the algorithm presented in chapter 4. Given a matrix A with p-adic integer entries, the algorithm puts A into cyclic form performing all computations modulo p^{μ}. (If A has rational integer entries then all operations can be performed in $\mathbb{Q}$, the field of rational numbers.)

We take the p-adic (exponential) valuation of a p-adic integer to be

$$val(p^k q) = k \qquad \text{if } gcd(p,q) = 1$$

$$val(0) = \infty$$

We use the three elementary similarity transformations described in §4.1 with the difference that q is now a unit in $\mathbb{Z}_p$. The matrix A is still considered to be composed of four sub-matrices and the configurations "Q clear" and "P clear" are defined as in §4.1.

Again, at any time during the computation we will be in one of the three situations described by (4.1.1).

Initially, $h = k - 1$ and we repeatedly apply the steps described in §4.1 until $h = 0$ except for the following modifications.

1) If $Q \neq 0$:

Determine j such that $a_{h+1,j}$ has minimal p-adic value among the elements of Q. Let this p-adic value be v.

If $v > 0$, apply E_3 to rows and columns $h+1$ through k, with $q = p^v$. This increases γ by v and leaves $a_{h+1,j}$ a p-adic unit.

If $j \neq h$, apply $E_1(j,h)$ to bring $a_{h+1,j}$ in position $(h+1,h)$.

Then proceed as decribed to achieve the form (4.1.2) and re-define X, Y, P, Q.

2) If $P \neq 0$:

Determine i such that $a_{i,k}$ has minimal p-adic value among the elements of P. Let this p-adic value be v.

If $v > 0$, apply E_3 to rows and columns 1 through h, with $q = p^v$. This increases γ by v and leaves $a_{i,k}$ a p-adic unit.

If $i \neq h$, apply $E_1(i,h)$ to bring $a_{i,k}$ in position (h,k).

Then proceed as descibed to achieve the form (4.1.3) and re-define X, Y, P, Q.

There are no changes for situation 3.

We will now show how to use this algorithm to get the rational form.

5.2 Computing the rational form

For a given matrix A we define $\mu_0 = \mu_0(A)$ to be minimal such that $p^{\mu_0} > \max_j |b_j|$, with b_j defined as in (3.5.2). (This being an upper bound on the entries of the cyclic form of A, it has the effect of making the recomputation of the cyclic form modulo a higher power of p unlikely.)

Initially, we set $\mu = \mu_0$.

We apply the algorithm from §5.1, performing all arithmetic operations modulo p^μ. The result is a matrix B in cyclic form and an exponent γ such that (3.2.1) is satisfied with $\alpha = \mu - \gamma$. Moreover, there exist p-adic integer matrices T and U satisfying (3.2.2) with $\alpha = \mu - \gamma$.

If $\mu > 2\gamma$ we compute β as descibed in §3.3. First, for each component of B, we determine its characteristic polynomial, say f, and we compute $f(A)$ exactly then bring $f(A)$ into p-adic Smith normal form. If $\mu \leq 2\gamma$ we can avoid computing β (we set $\beta = 0$) since we already know that μ is too small.

If $\mu > \beta + 2\gamma$ then B is a correct cyclic form for A, and we are done.

If $\mu < \mu_0 + 2\gamma$, we replace $\mu \leftarrow \mu_0 + 2\gamma$; otherwise we replace $\mu \leftarrow 2\mu$. We then recompute B and γ using the algorithm from §5.1.

5.3 Correctness

We have already proved in §4.2 that the cyclic form algorithm terminates and gives a cyclic form for the input matrix. It is easily seen that the modifications described in §5.1 will not affect termination and the output is still a cyclic form.

All that we need to show now is that the modular cyclic form algorithm gives us a matrix B in cyclic form, matrices T and U (implicit) and an exponent γ, satisfying (3.2.1) and (3.2.2). From that and results from chapter 3, we will demonstrate that the entire algorithm (§5.2) is correct.

We know that U reflects elementary transformations performed on the rows of A, and similarly, T reflects elementary transformations performed on the columns of A.

Applying E_3 to rows and columns $h+1$ through k (or 1 through h) with $q = p^v$ (for $v > 0$) only diminishes the precision in computing the cyclic form. I.e. suppose we are at the i^{th} step in the computation and we find $v_i > 0$. At this point we have

$$\gamma = \sum_{j=0}^{i-1} v_j \ ,$$

$$AT^{(i)} \equiv T^{(i)}B^{(i)} \pmod{p^{\mu-\gamma}} \ ,$$

$$T^{(i)}U^{(i)} \equiv p^{\gamma}I \pmod{p^{\mu-\gamma}}$$

where $X^{(i)}$ represents the state of matrix X at the i^{th} step. The effect of applying E_3 with $q = p^{v_i}$ is

$$AT^{(i)} \equiv T^{(i)}B^{(i)} \pmod{p^{\mu-(\gamma+v_i)}} \ ,$$

$$T^{(i)}U^{(i)} \equiv p^{\gamma}I \pmod{p^{\mu-(\gamma+v_i)}} \ .$$

Therefore, when the algorithm terminates, we have γ, the summation of all the v_i's > 0 found, and (3.2.1) and (3.2.2) are satisfied.

Now, we know that the procedure descibed in §5.2 terminates since the algorithm to compute B is exact over the p-adic integers. Given that congruences (3.2.1) and (3.2.2) are satisfied, correctness follows from the result of §3.2.

5.4 Complexity

We will assume (justified experimentally) that the number of recomputations of β is $\mathcal{O}(1)$, and that the value of γ is dominated by μ_0. (In fact, chapter 7 will show that the choice of a large prime p will make values of $\gamma > 0$ unlikely.)

We define

$$a = \max_{i,j} |a_{i,j}| \ .$$

The time required to perform an addition or subtraction modulo p^μ is $\mathcal{O}(\mu)$; the time required to perform a multiplication or division modulo p^μ is $\mathcal{O}(\mu \log \mu)$ (assuming Fast Fourier Transform multiplication and division). Given b_j defined as in (3.5.2), we have

$$b_j = \sum_{j \text{ at a time}} \|A_i\| \leq \binom{n}{j} (na)^j$$

so that

$$\sum_j b_j \leq (1 + na)^n$$

and

$$\max_j |b_j| \leq (1 + na)^n \ .$$

Choosing $p^{\mu_0} > 2\max_j |b_j|$, we get

$$\begin{aligned}
\mu_0 &\geq \log_p 2\max_j |b_j| \\
&= \lceil \log_p 2\max_j |b_j| \rceil \\
&\leq \lceil \log_p 2(1+na)^n \rceil \\
&\leq \lceil \log_p 2^{n+1}(na)^n \rceil \\
&= \lceil (n+1)\log_p 2 + n\log_p(na) \rceil \\
&= \lceil (n+1)\log_p 2 + n(\log_p n + \log_p a) \rceil
\end{aligned}$$

so $\mu_0 = \mathcal{O}(n(\log n + \log a))$.

Therefore the time required to perform a single arithmetic operation modulo p^μ is

$$\begin{aligned}
\mathcal{O}(\mu \log \mu) &= \mathcal{O}(n(\log n + \log a)\log(n(\log n + \log a))) \\
&= \mathcal{O}(n(\log n + \log a)(\log n + \log(\log n + \log a))) \\
&= \mathcal{O}(n(\log n + \log a)(\log n + \log(\log na))) \ . \qquad (5.4.1)
\end{aligned}$$

Assuming that a is bounded, we certainly have

$$\log(\log(na)) = \mathcal{O}(\log n)$$

so that

$$\log n + \log(\log(na)) = \mathcal{O}(\log n)$$

and (5.4.1) reduces to

$$\mathcal{O}(n \log n(\log n + \log a)) \ .$$

Since the modular algorithm for the cyclic form requires $\mathcal{O}(n^4)$ operations, we find that the time required to construct the cyclic form of a $n \times n$ matrix is

$$\mathcal{O}(n^5 \log n(\log n + \log a)) \; . \tag{5.4.2}$$

From inequality (3.4.2), we have

$$p^\beta \leq (\sum_j |d_j| \|A\|^j)^n$$

and from inequality (3.5.4), we have

$$\sum_j |d_j| \|A\|^j \leq 4^n \|A\|^n \; .$$

Since $\|A\| \leq na$, we find

$$\begin{aligned} p^\beta &\leq (4^n \|A\|^n)^n \\ &\leq (4^n n^n a^n)^n \\ &= (4na)^{n^2} \; . \end{aligned}$$

Therefore, the p-adic Smith normal form of $f(A)$ can be computed modulo p^δ if $p^\delta > (4na)^{n^2}$. Since p^δ is

$$\mathcal{O}(n^2(\log n + \log a)) \; ,$$

the time required for a single arithmetic operation modulo p^δ is

$$\mathcal{O}(n^2 \log n(\log n + \log a))$$

(again assuming a bounded).

Computing $f(A)$ requires n matrix multiplications, each requiring $\mathcal{O}(n^3)$ operations for a total of $\mathcal{O}(n^4)$ operations; bringing $f(A)$ into Smith normal form

requires $\mathcal{O}(n^3)$ operations. Therefore, the time required to construct the p-adic Smith normal form of $f(A)$ is

$$\mathcal{O}(n^6 \log n(\log n + \log a)) \ . \tag{5.4.3}$$

It follows, from (5.4.2) and (5.4.3), that (5.4.3) is the complexity for the entire algorithm. Experiments suggest that the actual performance of the algorithm is closer to $\mathcal{O}(n^5)$ (see ch. 7, table 4).

5.5 Alternate algorithm

We use the algorithm descibed in §5.1 with the added modification that we actually compute T and U. Computations are done modulo p^μ and we find matrices B,T,U and an exponent γ, with B in cyclic form, such that (3.2.1) and (3.2.2) are satisfied with $\alpha = \mu - \gamma$. We then compute β according to §3.3 (or set $\beta = 0$ if $\mu \leq 2\gamma$).

If $\mu > \beta + 2\gamma$, then B is a correct cyclic form for A, and we are done.

If $\beta + 2\gamma \geq \mu > 2\gamma$, then from Theorem 3.2.1 we can apply p-adic lifting to refine U to $\hat{U}$ satisfying

$$T\hat{U} \equiv p^\gamma I \pmod{p^{2\mu}} \ .$$

Then from Theorem 3.2.2, taking

$$\hat{B} = p^{-\gamma}\hat{U}AT$$

we have

$$p^\gamma T\hat{B} = T\hat{U}AT \equiv p^\gamma AT \pmod{p^{2\mu}}$$

so that

$$AT \equiv T\hat{B} \pmod{p^{2\mu-\gamma}} \ .$$

After the substitutions

$$\mu \leftarrow 2\mu \ ,$$

$$B \leftarrow \hat{B} \ ,$$

$$U \leftarrow \hat{U} \ ,$$

$$\alpha \leftarrow \mu - \gamma$$

congruences (3.2.1) and (3.2.2) are satisfied, and we continue with the modular computation to bring B into cyclic form.

If $2\gamma \geq \mu > \gamma$ we replace $\mu \leftarrow 2\mu$; otherwise we replace $\mu \leftarrow 2\gamma + 1$. In either case, we perform the modular computation to determine B,T,U and γ satisfying (3.2.1) and (3.2.2) with B in cyclic form.

Correctness for this algorithm follows from an analysis similar to that made in §5.3.

This algorithm corresponds directly with the general lifting algorithm. There are two major reasons why we do not use it though. First, if the initial μ_0 is chosen as described in §5.2, then recomputation of B is unlikely, so that we do not benefit from the lifting procedure and the overhead induced by the computation of T and U makes the algorithm significantly slower than the algorithm in §5.2.

Secondly, for a "bad" initial choice of μ_0, in the case of $\beta + 2\gamma \geq \mu > 2\gamma$ we find that, during recomputation of B, γ grows quite large, inducing more recomputations of B and modulo a higher power of p than necessary.

If we were to find a way to control the growth of γ, we suspect that this algorithm would be more efficient than the algorithm from §5.2 for "bad" initial choice of μ_0.

The next chapter describes how we got the data to test and compare the algorithms from chapters 4 and 5 with the algorithm from [**Ozel**].

CHAPTER 6

TEST DATA

In computing the rational form an uninteresting case arises when the rational form of a given matrix consists of a single block (i.e, the matrix is "non-derogatory" [**Wilk**] having one $n \times n$ cyclic block). Filling up the entries of a matrix randomly will in practice almost always give a non-derogatory matrix.

We devised a way to generate random *derogatory* matrices that we used in testing the algorithms. First we generate polynomial degrees, and then use those to generate a matrix B in cyclic (in fact, rational) form. We then generate a non-singular transform matrix T and compute its adjoint, T^* ($T^* = \det(T)T^{-1}$). The matrix used for testing is given by $A = \frac{1}{g}A_0$, where

$$A_0 = TBT^* \ ,$$

$$g = \text{gcd of entries of } A_0 \ .$$

In each step, we use carefully chosen bounds to ensure some control over the size of the entries of A.

6.1 Generating polynomial degrees

The polynomial degrees are generated randomly. They will be used in §6.2 to generate $n \times n$ rational form matrices. The procedure is in three steps.

First we generate the number of cyclic blocks of the rational form matrix. Using a (uniform) random number generator, we get a number $0 < r < 1$, then compute $s = 2r - 1$ (so $-1 < s < 1$). The number of cyclic blocks, k, is given by

$$k = \lfloor \frac{(n-2)s^3 + (n+2)}{2} \rfloor$$

so that $2 \leq k \leq n-1$. (We thus exclude non-derogatory matrices and matrices with rational form having n cyclic blocks.)

The next step consists of partitioning n into k parts. Using an array $d = (d_0, d_1, \cdots, d_{n-1}, d_n)$, initialized to $(0, 0, \cdots, 0, 1)$, we repeatedly generate random indices, $1 \leq i \leq n-1$. If $d_i = 0$ then we set $d_i = 1$, otherwise we get another index. We repeat this until the array contains k 1's.

The partition p is given by the successive differences between indices for which $d_i = 1$. We then sort the partition in decreasing order.

Here is an example to illustrate the first two steps. Suppose

$$n = 6 \ , \ k = 4$$

$$d_0 = 0, 1, 0, 1, 1, 0, 1 = d_6$$

then

$$p_1 = 2, 2, 1, 1 = p_4 \ .$$

In the last step we derive the polynomial degrees, $[f_i]$ $1 \leq i \leq n$, from the partition p. Initially we set

$$[f_i] = 0 \quad i = 1, \cdots, n \ , \ i \neq k$$

$$[f_k] = p_k \ .$$

Then

$$[f_i] = p_i - (\sum_{j=i+1}^{k} [f_j]) \ , \quad i = k-1, k-2, \cdots, 1 \ .$$

This last step is explained by the fact that we subsequently want to generate a matrix in rational form. Let f_i be a polynomial of degree $[f_i]$ and F_h be the characteristic polynomial associated with the h^{th} cyclic block of the matrix. Then we will have

$$F_h = \prod_{j=h}^{k} f_j \ , \quad h = 1, \cdots, k$$

and

$$\deg(F_h) = \sum_{j=h}^{k} [f_j] \ .$$

Therefore $F_{h+1} | F_h$ as required, and $\sum_{h=1}^{k} \deg(F_h) = n$.

Using the example above as an illustration of the last step, we find

$$[f_1] = 0 \quad [f_4] = 1$$

$$[f_2] = 1 \quad [f_5] = 0$$

$$[f_3] = 0 \quad [f_6] = 0 \ .$$

Finally, we apply a criterion to ensure that there are not too many polynomials of degree 1; namely if

$$\sum_{\substack{j=1 \\ [f_j]=1}}^{n} j \geq \frac{n}{2}$$

we repeat the procedure for this instance. (Our example fails to meet this criterion so it would be rejected.)

We generated 100 polynomial degrees for each matrix size $11 \leq n \leq 20$ and used them to generate rational form matrices (as described in the next section).

6.2 Generating the rational form matrix

Using the polynomial degrees generated in §6.1 and a pseudo-normalized random number generator (described below) we will generate an $n \times n$ rational form matrix.

Standard random number generators give (approximately) uniformly distributed random variables. We want a random number generator that gives roughly normally distributed variables.

Given a distribution function $\phi(x) : (-\infty, +\infty) \rightarrow (0, \infty)$, its density function,

$$\Phi(x) = \int_{-\infty}^{x} \phi(t)dt \ ,$$

has an inverse $\Phi^{-1}(y)$, so that $\Phi^{-1}(\Phi(x)) = x, \Phi(\Phi^{-1}(y)) = y$.

Let u be uniformly distributed over the interval $(0, 1)$, and let $v = \Phi^{-1}(u)$. Then

$$\begin{aligned} \mathrm{P}(a < v < b) &= \mathrm{P}(a < \Phi^{-1}(u) < b) \\ &= \mathrm{P}(\Phi(a) < u < \Phi(b)) \\ &= \Phi(b) - \Phi(a) \\ &= \int_{a}^{b} \phi(t)dt \ . \end{aligned}$$

So given a uniformly distributed random variable u, taking $v = \Phi^{-1}(u)$ will produce a variable v having distribution $\phi(x)$.

Using

$$\Phi^{-1}(u) = \lambda \frac{2u - 1}{u(1 - u)} \tag{6.2.1}$$

as a random number generator, with u being (approximately) uniformly distributed [1] over the interval $(0, 1)$, gives a distribution function $\phi(x)$ roughly similar to a normal distribution about $x = 0$. The value λ $(0 < \lambda < 1)$ is a

[1] To generate pseudo-uniformly distributed random numbers we used an implementation in the ALGEB language [**Ford**] of the VAX VMS *mth$random* random number generator [**VMS,RTL-433**], which is of the linear congruential type [**Knuth I, ch.3**].

(rational) parameter chosen experimentally to make $\Phi^{-1}(u)$ close to 0 sufficiently often.

Using this random number generator, we generate the rational form matrix as follows.

Initially we set an array $e = ([f_1], \cdots, [f_n])$; at the end e will contain the initial partition from which $[f_1], \cdots, [f_n]$ were derived and will be used to construct the rational form matrix.

For each degree $[f_i]$, $1 \leq i \leq k$ (k = number of cyclic blocks), we generate $[f_i]$ coefficients, c_j $(0 \leq j \leq [f_j] - 1)$, for a monic polynomial f_i using (6.2.1) with

$$\lambda = \frac{\lfloor 100\sqrt{n} \rfloor}{100n}$$
$$\simeq \frac{1}{\sqrt{n}} .$$

If $|c_j| \geq 100$ we recompute it. If $c_j = 0$ for all j, we get another polynomial.

As we generate the polynomials, we compute the characteristic polynomials

$$F_j = F_j * f_i , \qquad j = 1, \cdots, k \ , \ i = j, \cdots, k$$

and the partition

$$e_j = e_j + [f_i] , \qquad j = 1, \cdots, k \ , \ i = j, \cdots, k ;$$

i.e.

$$F_j = \prod_{i=j}^{k} f_i$$
$$e_j = \sum_{i=j}^{k} [f_i] .$$

We are now ready to build the rational form matrix. The size of the h^{th} cyclic block is given by e_h and it has characteristic polynomial F_h.

The first row (column) number of the h^{th} cyclic block is given by

$$1 + \sum_{i=1}^{h-1} e_i$$

and the last by

$$\sum_{i=1}^{h} e_i \ .$$

The characteristic polynomial, F_h, goes in the last column of the block with its last coefficient, c_0, in the first row of the block; i.e. with $d = [f_h]$ we have

$$\begin{pmatrix} 0 & 0 & 0 & \dots & -c_0 \\ 1 & 0 & 0 & \dots & -c_1 \\ 0 & 1 & 0 & \dots & -c_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & -c_{d-1} \end{pmatrix} .$$

Initially we take the rational form matrix to be a single $n \times n$ cyclic block. Then we proceed to build the matrix according to the array e. The end of the h^{th} cyclic block is defined by changing the 1 at position $(1 + \sum_{i=1}^{h} e_i, \sum_{i=1}^{h} e_i)$ to 0 (except of course for the last cyclic block).

The rational normal form matrix just constructed has entries bounded by 100. It will be used in §6.4 to generate test data.

6.3 Generating the transform matrix

The entries of the $n \times n$ matrix T are generated randomly. Each entry $t_{i,j}$ is given by the normalized random number generator given in (6.2.1) with $\lambda = \frac{1}{n}$. If $|t_{i,j}| > \frac{100}{n}$ we recompute the entry.

Therefore the absolute value of each entry is bounded by $\lfloor \frac{100}{n} \rfloor$, so that the test matrix is likely to meet the criterion for acceptance (see §6.4).

6.4 The test matrix

We are now ready to generate the $n \times n$ matrix A. Our criterion for accepting a matrix as test data is that it should not be sparse and should have entries bounded by 10^5.

First we generate a matrix B in rational form as described in §6.2. We then repeatedly generate a matrix T as in §6.3 until its determinant is not zero (to ensure the existence of its adjoint).

We compute

$$A = TBT^*$$

then divide each entry of A by the greatest common divisor of its entries.

This matrix will be accepted if it passes the three following tests :

1) $|a_{i,j}| < 5000n \qquad i,j = 1,\cdots,n$

to ensure entries bounded by 10^5;

2) $\| \text{row}_j(A)\|^2 - a_{j,j}^2 > 10n \qquad j = 1,\cdots,n$, and

3) $\| \text{col}_j(A)\|^2 - a_{j,j}^2 > 10n \qquad j = 1,\cdots,n$,

to ensure that the matrix is not sparse.

If the matrix fails any of these tests it is rejected and we repeat the procedure.

We thus can generate derogatory matrices of degree n with entries bounded by 10^5 and a rational form having between 2 and $n-1$ cyclic blocks.

We tested all three algorithms using a set of matrices constructed using this procedure. The results are given in the next chapter.

CHAPTER 7

RESULTS

We generated 1000 example matrices, 100 for each degree from 11 to 20, with entries not exceeding five decimal digits in size (as descibed in chapter 6).

The first section presents the results of our computations using the exact algorithm described in chapter 4. As will be seen, the computations are quite slow; in some cases computations had to be aborted because of the excessive computing time required.

The second section presents the results using our modular algorithm (described in chapter 5). We used the second algorithm of Ozello ([**Ozel**]) as our standard for comparison.

7.1 Exact algorithm results

The computations were performed on a MicroVAX III in the Computer Science Department of Concordia University.

Table 1 shows the average time (in CPU-seconds) to compute the cyclic form of a matrix of degree n. In computing the average time for each degree, we exclude examples for which the time required to compute the cyclic form exceeds 5 times the average time of the previous examples of the same degree. We also exclude

examples for which computation was aborted for the reason stated above. So for each degree, table 1 also gives the number of examples, m (out of 100), used in getting the average time.

We see that the algorithm is quite slow (especially if we compare the results from table 1 to the results from table 4). This is due to the size of the numbers involved during the computation. (Chapter 8 will show an example in which numbers exceed 10000 digits.) To speed things up we decided to do computations using modular arithmetic (our modular algorithm). The results follow.

7.2 Modular algorithm results

The computations were performed on a MicroVAX II computer in the Computer Science Department of Concordia University. (Note that the ratio of speed between the MicroVAX III and the MicroVAX II is 5 : 2.)

The data in table 2 and table 3 was obtained using 10 matrices of each degree; in the other tables (4, 5, and 6) the complete set of 100 matrices of each degree was used. All times are expressed in CPU-seconds.

Our algorithm was run with six different choices for p, $2 \leq P_k \leq 2^{32}$:

$$P_1 = 2$$

$$P_2 = 17$$

$$P_3 = 199$$

$$P_4 = 8821$$

$$P_5 = 2090177$$

$$P_6 = 4294967291 \ .$$

Results from Ozello's algorithm are labelled P_0.

Table 2 shows the total execution times for initial values of μ given by $C\mu_0$ (μ_0 defined in §5.2), with $0.5 \leq C \leq 1.0$. The performance of the algorithm is highly sensitive to the initial value chosen for μ. The high execution times for low values of C are due to repeated computation of B. As can be seen in the table 3, B is frequently recomputed with $C = 0.5$ and $C = 0.6$, but for $C > 0.7$, B is not recomputed at all with this set of examples. These results convinced us that μ_0 is a good choice for the initial value of μ.

In table 4 we compare the performance of the modular algorithm, with various choices of p, with that of Ozello's algorithm. For each choice P_k, $k = 0, \cdots, 6$, a straight line was fit to the 1000 data points $(\log n, \log t)$ by the least squares method.

We can see that Ozello's algorithm is generally faster in the range $11 \leq n \leq 20$ and has a smaller standard deviation but the coefficients from the linear regression on the data points suggest that the performance our algorithm will surpass that of Ozello's somewhere between $n = 65$ and $n = 90$.

The performance of our algorithm is not sensitive to the choice of p, except that it is somewhat slower when p is very small. From table 5 we can see that the degradation in performance with small p is due to the substantial values of γ that arise. Since the algorithm computes a relatively small number of inverses, with a large prime the likelihood is very great that all of them will be p-adic units. (The value of γ changes only when the inverse of a non-unit is computed.) So one might expect the effect of γ on the performance to vanish with large p. (In fact none of

the examples gave $\gamma > 0$ with $p > 199$.)

In the case $p = 2$, machine-level implementation might offset the disadvantage associated with small primes because of the extreme ease with which arithmetic modulo a power of 2 may be performed.

Table 6 shows the percentage of the total time that is spent computing β. It is clear that this computation dominates the total time, and that, as n increases, it takes an increasing proportion of the total.

It would be highly desirable to avoid (or at least speed up) the computation of β.

A remark of Ozello ([**Ozel, p. 48**]) implies that, if (3.2.1) and (3.2.2) hold, and $p^{\mu-2\gamma} > (2na)^n$, then A and B are similar. Unfortunately, Ozello does not prove his remark, and it is not obvious why it should be true. The basic theoretical obstacle is that, while the coefficients of the characteristic polynomial, being sums of subdeterminants of the matrix, are continuous functions of the entries, the coefficients of the invariant factors, and even their degrees, are not.

On the other hand, none of our test cases provide a counterexample. In fact, in every case it was sufficient to have $\mu = \mu_0 + 2\gamma$ to obtain a correct cyclic form; the replacement $\mu \leftarrow 2\mu$ was never required. This bound is even weaker than Ozello's (his bound is chosen to exceed twice the magnitude of any coefficient of any polynomial dividing the characteristic polynomial of A), in that it is chosen merely to exceed twice the magnitude of any coefficient of the characteristic polynomial of A. We determined its effectiveness experimentally, and are not in a position to prove that it is sufficient.

Should Ozello's hypothesis be shown correct, computation of β could be bypassed entirely, improving the complexity of our algorithm by $\mathcal{O}(n)$.

The next chapter analyzes the behavior of the three algorithms for what we consider an interesting example.

TABLE 1 : EXACT ALGORITHM PERFORMANCE

n	m	average time
11	97	29
12	87	41
13	95	56
14	90	86
15	92	110
16	84	151
17	82	214
18	86	280
19	85	330
20	79	461

TABLE 2: CONSEQUENCES OF COMPUTING β PREMATURELY

Time for 10 examples of each degree, with initial $\mu = C\mu_0$

	C	11	12	13	14	15	16	17	18	19	20
P_1	0.5	352	1202	1565	1554	1144	3016	2157	2849	13197	3500
	0.6	266	908	1257	1231	1352	3021	1299	1602	10595	2498
	0.7	160	520	915	1401	718	979	1042	1752	4255	2524
	0.8	163	368	539	867	722	910	1054	3662	2124	2551
	0.9	163	290	346	637	635	991	1063	1539	2154	2585
	1.0	165	295	351	505	643	904	1078	1558	2180	2618
P_2	0.5	255	621	877	1552	1443	1297	982	2753	12613	2522
	0.6	153	303	754	883	1381	849	996	1500	3106	2548
	0.7	155	277	320	467	604	1165	1004	1517	2059	2573
	0.8	157	281	323	472	611	858	1014	1546	2095	2606
	0.9	159	285	327	481	620	874	1027	1563	2119	2650
	1.0	160	289	333	488	625	886	1043	1595	2153	2677
P_3	0.5	186	597	895	1576	1016	1370	999	1510	12551	2554
	0.6	153	306	535	759	1013	852	1005	1523	3077	2580
	0.7	156	281	324	476	607	866	1017	1542	2043	2604
	0.8	157	284	328	483	615	874	1027	1575	2084	2645
	0.9	160	289	332	490	621	891	1040	1591	2116	2683
	1.0	163	293	339	501	636	902	1063	1621	2143	2715
P_4	0.5	185	622	882	1509	1019	835	995	1531	8386	2549
	0.6	155	302	598	603	977	840	1003	1538	3091	2575
	0.7	157	280	535	476	614	853	1014	1556	2056	2605
	0.8	159	282	326	481	617	862	1025	1580	2078	2638
	0.9	161	286	332	489	627	877	1038	1599	2109	2666
	1.0	162	289	332	497	633	886	1051	1626	2124	2707
P_5	0.5	154	275	770	1504	975	835	987	1503	7529	2533
	0.6	155	275	316	806	599	840	996	1521	2017	2557
	0.7	158	279	321	467	608	852	1009	1539	2037	2580
	0.8	158	283	325	472	615	865	1021	1564	2073	2616
	0.9	159	283	329	479	618	875	1034	1591	2102	2653
	1.0	161	289	333	487	629	887	1052	1616	2131	2676
P_6	0.5	152	510	512	1311	791	816	968	1476	7160	2461
	0.6	152	273	314	455	796	824	975	1492	3019	2480
	0.7	156	276	314	458	593	828	986	1509	1992	2505
	0.8	155	278	317	467	601	846	997	1529	2024	2545
	0.9	160	287	325	473	612	852	1013	1556	2052	2578
	1.0	160	288	328	483	617	869	1030	1589	2088	2614

TABLE 3: TOTAL B COMPUTATIONS

For 10 examples of each degree, with initial $\mu = C\mu_0$

	C	11	12	13	14	15	16	17	18	19	20
P_1	0.5	14	17	16	15	15	14	13	17	17	13
	0.6	13	16	15	15	14	14	11	13	16	10
	0.7	10	13	14	14	11	12	10	13	12	10
	0.8	10	11	12	12	11	11	10	11	10	10
	0.9	10	10	10	12	10	11	10	10	10	10
	1.0	10	10	10	11	10	10	10	10	10	10
P_2	0.5	12	12	12	13	12	12	10	11	14	10
	0.6	10	11	12	13	12	11	10	10	11	10
	0.7	10	10	10	10	10	11	10	10	10	10
	0.8	10	10	10	10	10	10	10	10	10	10
	0.9	10	10	10	10	10	10	10	10	10	10
	1.0	10	10	10	10	10	10	10	10	10	10
P_3	0.5	11	12	12	13	11	11	10	10	14	10
	0.6	10	11	11	12	11	10	10	10	11	10
	0.7	10	10	10	10	10	10	10	10	10	10
	0.8	10	10	10	10	10	10	10	10	10	10
	0.9	10	10	10	10	10	10	10	10	10	10
	1.0	10	10	10	10	10	10	10	10	10	10
P_4	0.5	11	12	12	13	11	10	10	10	13	10
	0.6	10	11	11	11	11	10	10	10	11	10
	0.7	10	10	11	10	10	10	10	10	10	10
	0.8	10	10	10	10	10	10	10	10	10	10
	0.9	10	10	10	10	10	10	10	10	10	10
	1.0	10	10	10	10	10	10	10	10	10	10
P_5	0.5	10	10	12	13	11	10	10	10	13	10
	0.6	10	10	10	11	10	10	10	10	10	10
	0.7	10	10	10	10	10	10	10	10	10	10
	0.8	10	10	10	10	10	10	10	10	10	10
	0.9	10	10	10	10	10	10	10	10	10	10
	1.0	10	10	10	10	10	10	10	10	10	10
P_6	0.5	10	11	11	13	11	10	10	10	13	10
	0.6	10	10	11	10	11	10	10	10	11	10
	0.7	10	10	10	10	10	10	10	10	10	10
	0.8	10	10	10	10	10	10	10	10	10	10
	0.9	10	10	10	10	10	10	10	10	10	10
	1.0	10	10	10	10	10	10	10	10	10	10

TABLE 4 : COMPARATIVE PERFORMANCE

	11	12	13	14	15	16	17	18	19	20
P_0	990	1417	1992	2790	3965	5767	7202	10421	14841	18118
P_1	1767	2627	3935	5043	6640	10295	12785	16479	23769	27589
P_2	81	2564	3551	4737	6340	9720	12217	16124	21145	26702
P_3	1713	2563	3599	4779	6407	9913	12302	16224	21319	26967
P_4	1707	2562	3602	4808	6321	9944	12588	16373	21203	27055
P_5	1700	2547	3559	4745	6309	9961	12424	16413	21148	26710
P_6	1675	2512	3502	4700	6233	9697	12173	15870	20565	26041

P_0: $\mathrm{LOG}\, t = 4.849\mathrm{LOG}\, n - 9.433 \quad \sigma = 0.291$
P_1: $\mathrm{LOG}\, t = 4.527\mathrm{LOG}\, n - 8.039 \quad \sigma = 0.362$
P_2: $\mathrm{LOG}\, t = 4.562\mathrm{LOG}\, n - 8.179 \quad \sigma = 0.353$
P_3: $\mathrm{LOG}\, t = 4.559\mathrm{LOG}\, n - 8.160 \quad \sigma = 0.353$
P_4: $\mathrm{LOG}\, t = 4.574\mathrm{LOG}\, n - 8.198 \quad \sigma = 0.350$
P_5: $\mathrm{LOG}\, t = 4.573\mathrm{LOG}\, n - 8.200 \quad \sigma = 0.350$
P_6: $\mathrm{LOG}\, t = 4.542\mathrm{LOG}\, n - 8.135 \quad \sigma = 0.348$

TABLE 5: TOTAL γ VALUES

	11	12	13	14	15	16	17	18	19	20
P_1	1946	2420	2228	2716	2568	3227	3487	4089	4127	4707
P_2	54	88	36	59	72	82	97	94	24	149
P_3	0	10	0	1	0	1	1	1	2	1
P_4	0	0	0	0	0	0	0	0	0	0
P_5	0	0	0	0	0	0	0	0	0	0
P_6	0	0	0	0	0	0	0	0	0	0

TABLE 6: PERCENTAGE OF TIME COMPUTING β

	11	12	13	14	15	16	17	18	19	20
P_1	71.26	72.49	76.35	75.98	77.50	80.66	81.35	81.50	84.82	84.16
P_2	71.66	73.79	75.35	76.07	78.24	80.61	81.48	82.63	83.71	84.46
P_3	71.13	73.19	74.66	75.99	77.62	80.81	81.39	82.53	83.38	84.24
P_4	71.45	73.35	74.63	75.80	77.60	80.53	81.47	82.27	83.40	84.15
P_5	70.80	73.15	75.00	76.19	77.54	80.38	81.26	82.51	83.39	84.28
P_6	71.30	73.55	75.48	76.32	78.12	80.74	81.47	82.63	83.62	84.56

CHAPTER 8

AN INTERESTING EXAMPLE

An 18×18 matrix (p. 78) was used to trace the algorithms (exact, modular and Ozello's) in order to try to analyze their behavior.

We chose this matrix because it illustrates an important point, namely relative control over the size of the numbers involved during the computation of the rational form is intimately related to algorithm's efficiency (in terms of computing time).

The matrix has integer entries ranging from -44196 to 37324. Its cyclic form (p. 78) contains two blocks and integer entries between -3.53×10^{23} and 5.95×10^{19}. Here is how each algorithm behaved during the computation of the cyclic form of the matrix.

8.1 Using the exact algorithm

This is one of the few examples for which excessive computing time forced us to abort the computations (done on a MicroVAX III).

In monitoring the size of the numbers involved during the computations, we found that we get numbers with more than 13500 digits (after only a few minutes of computing time). We find that the numbers get much bigger than that as time goes on using more and more memory space and slowing down the algorithm to

the point of becoming totally impractical.

8.2 Using Ozello's algorithm

Ozello's algorithm gets the rational form of an $n \times n$ matrix A by finding vectors $f_1, \cdots, f_k$ such that $\pi_{f_i} = \pi_{A_{i-1}}$, for $i = 1, \cdots, k$, where k is the number of blocks, π_{f_i} is the minimal polynomial of f_i modulo ζ_{i-1}, the subspace induced by $f_1, \cdots, f_{i-1}$, and $\pi_{A_{i-1}}$ is the minimal polynomial of A_{i-1}, the endomorphism of the quotient space ε/ζ_{i-1} induced by A (ε is a vector space of size n over a field). (For detailed explanations of the algorithm see [**Ozel**].)

Even though finding these vectors is generally fast, this method can in certain cases fail to keep the size of the numbers involved small during the computations.

For this particular example, the required vector is found on the first try for each block but the algorithm has to deal with numbers as big as 700 digits in the process. These numbers are much smaller than those encountered in the exact algorithm but still, the total time to compute the rational form was 99.48 CPU-seconds. (Computations were done on a MicroVAX II.)

We will now see how this compares to our modular algorithm.

8.3 Using the modular algorithm

Recall that computations are done modulo p^{μ_0} (for some prime number p and μ_0 as defined in §5.2). Therefore throughout the computations, the numbers will always be between 0 and $|p^{\mu_0} - 1|$.

For this particular example, we have

p	μ_0	d
P_1	248	75
P_2	61	76
P_3	33	76
P_4	19	75
P_5	12	76
P_6	8	78

where d is the number of digits of the modulus. We see that the number of digits involved here are hundreds of times smaller than those of the exact algorithm, and ten times smaller than those of Ozello's. Accordingly, our algorithm is faster than the other two.

The times to compute the cyclic form range from 72.43 (for $p = P_6$) to 78.81 (for $p = P_5$) CPU-seconds (on a MicroVAX II). This is 1.37 times faster than Ozello's algorithm.

I believe that this example proves our point. Even though for most of the matrices tested Ozello's algorithm was faster and the exact algorithm manages to complete its computations, the advantage of the modular algorithm comes from the fact that it tackles one of the problems related to computing the rational form of a matrix, control of the size of intermediate values. (This is one of Ozello's goals that he failed to achieve ([**Ozel,p.58**]).) The next step will be to solve the problems related to the computation of β (see §7.2).

Original Matrix

-1764	882	76	-1110	-912	905	-228	1659	364	274	-2251	182	0	1536	114	-228	568	1954
37324	2086	-4180	14406	6688	3395	2660	-2663	-676	-42	5999	-3302	-3952	-10952	646	684	-664	-17146
21044	1334	-1202	8470	5624	2191	1900	1333	676	1790	1867	-3614	-1976	-5312	38	-76	-400	-10442
29184	-7182	1216	9462	9120	-4123	-2660	-16473	-2964	-2090	16777	-2470	0	-12084	1330	-3648	-2508	-6878
0	0	0	0	0	1976	0	1976	0	0	-1976	0	0	0	0	0	0	0
12456	-300	1520	1668	1520	1070	-2584	-1270	-312	-604	2950	-156	0	-1208	1292	-2584	1272	6164
1876	-2420	2052	-3736	1064	398	-5168	-6622	-936	-2040	414	520	1976	2836	1596	-4180	1764	11652
-4364	-782	76	-1422	-912	-1591	-228	411	-260	586	-379	-130	0	2160	114	-228	-992	2578
-13800	-4462	4560	-7242	608	-4603	-2812	-9569	-1716	-434	6497	130	1976	3084	418	-1824	-604	10874
-6604	832	988	-1820	0	260	0	1612	312	-156	-1924	156	0	676	0	988	780	1664
1084	-3506	2812	-978	1824	-2941	-2508	-7959	-1196	-314	6647	-598	0	360	1254	-2508	-824	8910
10200	-2136	2128	1680	2128	900	-2432	-4300	624	-1376	3220	312	0	-2752	1216	-2432	-112	3024
-412	3170	-1900	554	-912	2361	1748	8315	-260	586	-6307	-130	0	2160	114	1748	984	-1374
8468	-2752	2812	2220	3800	-328	-1520	-6776	-728	-548	5984	-364	0	-4060	760	-2508	-148	3008
-11000	3524	-1520	-2916	-3496	802	2584	8134	1768	1852	-7318	884	0	3704	-1292	4560	392	-3668
-22852	2534	532	-6106	-6384	875	2356	11353	1924	2230	-8945	962	0	7424	-1178	2356	440	4422
-44196	-10012	7980	15904	-912	-10002	-5168	-21702	-1560	-1728	12166	2184	5928	5436	-380	-2204	-1772	18204
2312	-1156	456	-212	456	-290	-1368	-1798	-728	-852	1158	-364	0	272	684	-1368	80	2780

Cyclic Form Matrix

$$\begin{pmatrix} C & 0 \\ 0 & C \end{pmatrix}$$

where C is a cyclic block having characteristic polynomial

$$c = x^9 - 3952x^8 + 3904576x^7 + 7715442176x^6 - x^5 + 30125530349797376x^4$$

$$- 5952804797119961497 6x^3 + 3528822683732713175777 28x^2 - x$$

REFERENCES

[**Buch**] B. Buchberger and al, "Computer Algebra Symbolic and Algebraic Computation," Springer-Verlag, New York, 1982.

[**Fad**] D. K. Fadeev and V. N. Fadeeva, "Computational Methods of Linear Algebra," W. H. Freeman & Co., San Francisco and London, 1963, pp. 251-260.

[**Ford**] D. Ford, "On the Computation of the Maximal Order in a Dedekind Domain," Ph.D. Dissertation, Ohio State University, 1978.

[**Her**] I. N. Herstein, "Topics in Algebra," Blaisdell Publishing Company, Waltham, Massachusetts, 1964.

[**HK**] K. Hoffman and R. Kunze, "Linear Algebra," Prentice Hall, Englewood Cliffs, 1961.

[**How**] J. Howell, *An algorithm for the Exact Reduction of a Matrix to Frobenius Form Using Modular Arithmetic, I + II*, Math. Comp. 124 **27** (1973), 887-920.

[**Knuth I**] Donald E. Knuth, "The Art of Computer Programming, vol. I," Addison-Wesley Pub. Co., Don Mills, Ontario, 1973.

[**Lün**] H. Lüneberg, "On the Rational Normal Form of Endomorphisms," BI-Wissenschafts-verlag, Mannheim, 1987.

[**Mah**] K. Mahler, "Introduction to p-adic Numbers and their Functions," Cambridge University Press, 1973.

[**MF**] M.H. Mathieu and D.J. Ford, *On p-adic Computation of the Rational Form of a Matrix*, Journal of Symbolic Computation, to appear.

[**Mig**] M. Mignotte, *An Inequality about Factors of Polynomials*, Math. Comp. 128 **28** (1974), 1153-1157.

[**ND**] B. Noble and J. W. Daniel, "Applied Linear Algebra," Prentice-Hall, Englewood Cliffs, 1977.

[**Newm**] M. Newman, "Integral Matrices," Academic Press, New York, 1972.

[**Ozel**] P. Ozello, "Calcul Exact des formes de Jordan et de Frobenius d'une Matrice," Doctoral Thesis, University of Grenoble, 1987.

[**Sch**] A. Schrijver, "Theory of Linear and Integer Programming," Wiley, New York, 1986.

[**VdW2**] B. L. Van der Waerden, "Algebra volume 2," Frederick Ungar Publishing Co., New York, 1970.

[**VMS**] Vax Record Management, "Services Reference Manual," Digital Equipment Corporation, Kanata, Ontario, 1986.

[**Wilk**] J. H. Wilkinson, "The Algebraic Eigenvalue Problem," Clarendon Press, Oxford, 1965.

[**Zas**] H. Zassenhaus, *Hensel Factorization I*, J. Number Theory 1 (1969), 291-311.