CANADIAN THESES

THÈSES CANADIENNES

NOTICE

The quality of this microfiche 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.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

AVIS

La qualité de cette microfiche 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.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

The University of Alberta

LINEAR SYSTEMS OF POLYNOMIAL EQUATIONS

by

Bart Courtney Domzy

A thesis
submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree
of Master of Science

Department of Computing Science

Edmonton, Alberta
Fall, 1983

ABSTRACT

Described are the modular and power series methods for solving the system of linear equations

$$A(z)\,x(z) = b(z),$$

where the components of $A(z)$ and $b(z)$ are polynomials with coefficients over the integers modulo p. Theoretical cost estimates for obtaining the solution in rational form are shown to be $4\partial^2 N^5 + \frac{2}{3}\,\partial N^4$ and $10\partial^2 N^3$ operations for each of these methods, respectively, where N is the order of the system and ∂ is the maximum degree of the component polynomials. These estimates, together with empirical timing results, are used to identify classes of problems for which each method is superior. Finally, the results of the modular and power series methods are used to describe an algorithm for solving systems of linear equations with component polynomials over the integers.

ACKNOWLEDGEMENTS

I would like to thank the members of my supervisory committee Cliff Addison and Gerald Cliff. The attention and time they gave to this thesis is very much appreciated. Thanks are especially due to my supervisor Stanley Cabay, who introduced me to this area of research and directed me toward a solution.

I would also like to thank the members of my family for their support and assistance without which I would have not succeeded.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Let Z denote the integers, $\frac{Z}{(p)}$ the integers modulo the prime p, $Z[z]$ the polynomials in z with integer coefficients, and $\frac{Z[z]}{(p)}$ the polynomials in z with coefficients in $\frac{Z}{(p)}$. The primary purpose of this thesis is the comparison of methods for symbolically solving the linear system of equations

$$A(z)\, x(z) = b(z), \tag{1.1}$$

where $A(z)$ is a matrix of order N and b(z) is a vector of length N with components being polynomials in z.

The established method of solving (1.1) is the modular method, described in careful detail by McClellan[13]. This method is essentially one of determination of the value of the solution at certain evaluation points, followed by interpolation over these points to yield, finally, the solution in rational form. Extensive theoretical and empirical investigations have been performed comparing the modular method for solving (1.1) with other viable methods, such as the Gaussian Elimination (exact-division) algorithm, the two-step, fraction-free method, and the multi-step methods (see McClellan[14], Bariess[2], Cabay and Lam[8] and Higginson[10]). The conclusions reached are that, as a general problem solving algorithm (that is, for random $A(z)$ and $b(z)$), the modular method is clearly superior. Consequently, in this thesis, from all these methods, only the modular method is given consideration.

In this thesis, the modular method is compared theoretically and empirically with the power series method, introduced recently by Cabay[5]. This method essentially determines the derivatives of the solution at a single evaluation point, from which the power series form, and eventually the rational function form of the solution is then constructed.

The thesis commences by describing in Chapter 2, an established method for solving systems over $\frac{Z}{(p)}$. These solutions are basic in the development of both the modular and power series methods for solving systems over $\frac{Z[z]}{(p)}$.

Chapter 3 describes the modular method for solving systems over $\frac{Z[z]}{(p)}$. Included in the development are detailed cost estimates and a new termination criterion.

Chapter 4 describes the power series method for solving systems over $\frac{Z[z]}{(p)}$. In addition to deriving cost estimates, some intrinsic problems for the implementation of the method are highlighted.

The major purpose of this thesis is a theoretical and empirical comparison of costs of the modular and power series methods in order to determine classes of problems for which each is superior. Chapter 5 describes the implementation of these methods and the testing procedures used. Empirical results confirm the theoretical cost estimates of Chapters 3 and 4 and identify practical trade-offs between the methods.

The final chapter uses the the methods of Chapters 3 and 4 to solve systems over $Z[z]$. Having obtained the solutions of (1.1) over $\frac{Z[z]}{(p_i)}$ for several primes p_i, $i = 0,1, \ldots, \gamma$, the Chinese Remainder Theorem permits the construction of the solution over $Z[z]$. Cost estimates are obtained, problems with bad primes are identified, and a new termination criterion is derived.

CHAPTER 2

The Adjoint Solution over $\frac{Z}{(p)}$

2.1. Introduction

Given the matrix A of order N with entries over $\frac{Z}{(p)}$, let A^{adj} denote the adjoint of A. Then,

$$A^{adj} A = d I, \tag{2.1}$$

where d is the determinant of A and I is the identity matrix of order N.

For the system of linear equations

$$A x = b, \tag{2.2}$$

the adjoint solution is defined to be the pair (y,d) where

$$y = A^{adj} b. \tag{2.3}$$

If $d \neq 0$, then from (2.1) and (2.2), it follows that

$$x = y/d. \tag{2.4}$$

On the other hand, if $d = 0$, then x may not exist, whereas the adjoint solution always does. In subsequent chapters, it is the adjoint solution that is usually required.

To obtain the adjoint solution, a variant of the Gaussian elimination method (see Forsythe and Moler [9] and Lipson [11]) is used. The method consists of three steps.

In the first step, forward elimination yields the triangular decomposition

$$PA = LU, \tag{2.5}$$

where P is a permutation matrix, L is a lower triangular matrix of multipliers and

$$U = \begin{bmatrix} u_{11} & & & & & u_{1N} \\ & \ddots & & & & \vdots \\ & & & & & \vdots \\ & u_{k-1,k-1} & u_{k-1,k} & & \cdots & u_{k-1,N} \\ & & 0 & u_{k,k+1} & \cdots & u_{k,N} \\ & & & & & \vdots \\ & & & & \ddots & \vdots \\ & & & & & \vdots \end{bmatrix} \tag{2.6}$$

is the upper echelon form of A.

In the second step, forward substitution is used to find a vector b' such that

$$Lb' = Pb. \tag{2.7}$$

A solution of (2.7) always exists since L is a lower triangular matrix with 1's along the diagonal.

It now follows from (2.3), (2.5) and (2.7) that

$$\begin{aligned} y &= A^{adj}\, b \\ &= \left(P^{-1}L\, U\right)^{adj} b \\ &= U^{adj}\left(L^{-1}(P^{-1})^{adj}\, b\right) = U^{adj}\left(det(P)\cdot L^{-1}\cdot P\; b\right) \\ &= det(P)\; U^{adj}\, b' \end{aligned} \tag{2.8}$$

and, in addition, that

$$d = det(P)\; det(U) = det(P)\cdot\prod_{i=1}^{N} u_{ii} \tag{2.9}$$

where $det(P) = \pm 1$. Thus, except for the sign of $det(P)$, the adjoint solution of (2.2) is also the adjoint solution of

$$U\, x = b'. \tag{2.10}$$

It remains, therefore, in the last step to solve (2.10) for (y,d). If $rank(U) = N$, then x is obtained from (2.10) by back substitution and then

$$y = d\, x \tag{2.11}$$

follows. If $rank(U) \leq N-2$, (i.e. the last two rows of U are zero) then replacing any column of U

by b ′ results in a matrix with rank at most N-1. It now follows by Cramer's rule that

$$y = 0. \tag{2.12}$$

If rank(U) = N-1 (i.e. the last row only of U is zero), then Cramer's rule for the last N-k+1 components of y yields

$$y_k = (-1)^{N-k}\left(\prod_{j=1}^{k-1} u_{jj}\right)\left(\prod_{j=k+1}^{N} u_{j-1,j}\right) b'_N \tag{2.13}$$

and

$$y_i = 0 \ , \ i = k+1,k+2, \ldots ,N \tag{2.14}$$

where k is as in (2.6). The remaining components of y are obtained by back substitution applied to

$$U\, y = d\, b' = 0. \tag{2.15}$$

That is,

$$y_i = -u_{ii}^{-1} \sum_{j=i+1}^{k} u_{ij}\, y_j \ , \ i = k-1,k-2, \ldots ,1. \tag{2.16}$$

2.2. Cost Analysis

The number of arithmetic operations required to find the adjoint solution (y,d) to (2.2) is counted as a measure to the cost of finding the solution. In the count, only the dominating terms are retained.

The first step is to find the triangular decomposition (2.5). This step requires $\frac{2N^3}{3}$ operations (counting additions and multiplications)(see Atkinson [1]).

The forward substitution of the second step to solve (2.7) requires N^2 operations.

The third step involves finding the adjoint solution to (2.10). If d ≠ 0, then the back substitution requires N^2 operations. The calculation of the determinant, d, and its subsequent use in (2.11) to find y requires 2N operations. Thus the total number of operations to find the solution (y,d) of (2.10) and (2.11) when $d \neq 0$ is N^2. If d = 0, then to find (2.12) or (2.13) and (2.14)

requires at most N operations and the subsequent back substitution to find (2.16) requires at most k^2 operations. Thus, since $k \leq N$, the back substitution requires at most N^2 operations to find the adjoint solution to (2.10).

Thus the total cost of finding the adjoint solution to (2.2) requires

$$\frac{2N^3}{3} \tag{2.17}$$

operations. To find the solution for another b when the triangular decomposition (2.5) is available requires only

$$2N^2 \tag{2.18}$$

additional operations for each b.

CHAPTER 3

Modular Methods over $\frac{Z[z]}{(p)}$

3.1. Description

Let A(z) be a matrix of order N and b(z) be a vector of length N, where the elements of A(z) and b(z) are univariate polynomials of degree at most ∂, whose coefficients are over $\frac{Z}{(p)}$, the integers modulo the prime p. This chapter describes the modular technique for finding the adjoint solution (y(z),d(z)) where

$$y(z) = A(z)^{adj}\, b(z) \tag{3.1a}$$

and

$$d(z) = \det\left(A(z)\right). \tag{3.1b}$$

Then

$$A(z)\, y(z) = d(z)\, b(z), \tag{3.2}$$

where y(z) is a vector of length N of polynomials over $\frac{Z}{(p)}$. All arithmetic considered in this chapter is done over $\frac{Z}{(p)}$.

The modular methods make use of the Chinese Remainder Theorem (CRT) for polynomials to find the solution over the polynomials over $\frac{Z}{(p)}$. To use the CRT, the system (3.2) is first solved modulo the prime polynomial $m_i(z)$ to obtain the adjoint solution $\left(\overline{y_i(z)},\overline{d_i(z)}\right)$ of

$$\overline{A_i(z)}\cdot\overline{y_i(z)} = \overline{d_i(z)}\cdot\overline{b_i(z)} \ \ mod\ m_i(z), \tag{3.3}$$

where

$$\overline{y_i(z)} = y(z)\ mod\ m_i(z) \tag{3.4}$$

$$\overline{d_i(z)} = d(z)\ mod\ m_i(z)$$

$\overline{A_i(z)} = A(z) \bmod m_i(z)$

$\overline{b_i(z)} = b(z) \bmod m_i(z)$.

These residuals, $\overline{y_i(z)}$ and $\overline{d_i(z)}$, for $i = 0,1, \ldots, \gamma$, are then used to reconstruct the solution (y(z),d(z)) via the CRT, where γ is such that

$$deg\left(\prod_{i=0}^{\gamma} m_i(z)\right) \leq \delta, \tag{3.5}$$

δ being the maximum degree of the solution's polynomials.

Since the linear polynomials, $\left(z - z_i\right)$, $z_i \in \frac{\mathbf{Z}}{(p)}$, are irreducible over $\frac{\mathbf{Z}[z]}{(p)}$, the prime polynomials $m_i(z)$ are set to $m_i(z) = \left(z - z_i\right)$, and γ is set to $\gamma = \delta$ in (3.5). Using these linear polynomials $\left(z - z_i\right)$, the next step is to solve (3.2) modulo $\left(z - z_i\right)$. But finding a polynomial's value modulo a linear polynomial, $\left(z - z_i\right)$, is equivalent to evaluating that polynomial at the point z_i. This implies that

$$A(z) \bmod \left(z - z_i\right) = A(z_i) \tag{3.6}$$

$$b(z) \bmod \left(z - z_i\right) = b(z_i)$$

$$d(z) \bmod \left(z - z_i\right) = d(z_i)$$

and

$$y(z) \bmod \left(z - z_i\right) = y(z_i)$$

for each polynomial $m_i(z) = \left(z - z_i\right)$, $i = 0,1, \ldots, \delta$.

Therefore, to find the solution $\left(\overline{y_i(z)},\overline{d_i(z)}\right)$, the system [A(z),b(z)] is evaluated at the point z_i and the resultant system $A(z_i)\,\overline{y_i(z)} = \overline{d_i(z)}\,b(z_i)$ is solved over $\frac{\mathbf{Z}}{(p)}$. The solution to the system over $\frac{\mathbf{Z}}{(p)}$ has results whose elements are in $\frac{\mathbf{Z}}{(p)}$ as well, and as such

$$\overline{y_i(z)} = \overline{y_i} \tag{3.7}$$

$\overline{d_i(z)} = \overline{d_i}$.

These $\overline{y_i}$ and $\overline{d_i}$ for the different prime polynomials $m_i(z) = \left(z - z_i\right)$ are then the residuals used by the CRT to reconstruct the polynomials of the solution.

A point to note is that these residuals $\overline{y_i}$ and $\overline{d_i}$ are exactly the values of their respective polynomials y(z) and d(z) evaluated at the point z_i and therefore are interpolation points of the polynomials. As well, the polynomials can be constructed uniquely up to a degree δ whenever their values are known at δ+1 distinct evaluation points. So the application of the CRT for the reconstruction of the polynomials is basically the interpolation of the polynomials from their values at these points.

The Newtonian form of the CRT is used for the reconstruction of the polynomials $\left(y(z), d(z)\right)$ from their residuals $\overline{y_i}$ and $\overline{d_i}$, $i = 0,1, \ldots, \delta$. This algorithm, explained below in terms of $\overline{d_i}$, to find the polynomial d(z), can be applied for each element in the vector y(z). The algorithm defines the resultant polynomial $d^{[k]}(z)$ of degree k, at the kth stage, iteratively as

$$d^{[k]}(z) = d^{[k-1]}(z) + a_k \prod_{i=0}^{k-1}\left(z - z_i\right), \tag{3.8}$$

where $a_k \epsilon \frac{\mathbf{Z}}{(p)}$, $k = 0,1,2, \ldots, \delta$ and

$$a_0 = d^{[0]}(z) = \overline{d_0} \tag{3.9}$$

$$a_k = \left(\overline{d_k} - d^{[k-1]}(z)\right) \times S_k \mod \left(z - z_i\right).$$

The S_k are defined as

$$S_k = \prod_{i=0}^{k-1}\left(z_k - z_i\right)^{-1} = \left(\prod_{i=0}^{k-1}\left(z_k - z_i\right)\right)^{-1} \mod p. \tag{3.10}$$

In the implementation of this algorithm with the maximum degree of the solution known, the S_k's can be calculated prior to the initiation of the algorithm and stored for use provided the z_i are known.

3.2. Implementation of the Modular Technique

The first step in the implementation is to find a bound on the degree of the adjoint solution (3.11). The bound, δ, put forth by McClellan[13], is defined as

$$\delta = \left(\delta_b + \sum_{i=1}^{N} \delta_i\right) - MIN\left(\delta_b, \delta_1, \ldots, \delta_N\right) \tag{3.11}$$

where δ_i, $i = 1,2, \ldots, N$, is the maximum degree of an entry in column i of A(z) and δ_b is the maximum degree of the entries in the right hand side, b(z).

The next step is to set the evaluation points z_i, $i = 0,1, \ldots, \delta$. For this implementation, the z_i were set to be $z_i = i$, $i = 0,1, \ldots, \delta$. With the z_i so set, the S_k values become

$$S_k = \prod_{i=0}^{k-1} (k-i)^{-1} = \prod_{i=1}^{k} i^{-1} = (k\,!)^{-1} \ mod\ p, \tag{3.12}$$

for $k = 1,2, \ldots, \delta$.

The third step is the evaluation of the matrix A(z) and the vector b(z) at z_k to get the integer system $\left[\overline{A_k}, \overline{b_k}\right]$ such that

$$\overline{A_k} = A(k)$$

$$\overline{b_k} = b(k)$$

for $k = 0,1, \ldots, \delta$. The evaluation of matrix A(z) and the vector b(z) at the point z_k is straightforward and is done using Horner's method.

The fourth step is solving $\overline{A_k}\, x_k = \overline{b_k}$ to obtain the adjoint solution

$$\overline{A_k}\, \overline{y_k} = \overline{d_k}\, \overline{b_k} \ mod\ p, \tag{3.13}$$

for $k = 0,1, \ldots, \delta$.

The adjoint solution of (3.13) over $\frac{Z}{(p)}$ is found using the LU decomposition procedures of Chapter 2. First, the decomposition into triangular matrices is applied to $\overline{A_k}$, then the solution is extracated from this triangular decomposition and the right hand side $\overline{b_k}$. The decomposition into triangular matrices returns the determinant, $\overline{d_k}$, of $\overline{A_k}$. If $\overline{A_k}$ is singular then back substitution

returns the adjoint solution vector, $\overline{y_k}$, so that $\left(\overline{y_k}, 0\right)$. On the other hand, when $\overline{A_k}$ is non-singular, the triangular decomposition process will return $\overline{d_k} \neq 0$. Thus, to get the adjoint solution vector, $\overline{y_k}$, the result , x_k, from the application of forward and back substitution, must be multiplied by $\overline{d_k}$.

The fifth step is the application of the CRT (3.9) to the residuals $\left(\overline{y_i}, \overline{d_i}\right)$ for $i = 0,1, \ldots, \delta$. This yields

$$y(z) = \sum_{k=0}^{\delta} y'_k \prod_{l=0}^{k-1} (z-i) \tag{3.14}$$

$$d(z) = \sum_{k=0}^{\delta} d'_k \prod_{l=0}^{k-1} (z-i),$$

where y'_k is a vector of length N with elements in $\frac{\mathbf{Z}}{(p)}$ and $d'_k \epsilon \frac{\mathbf{Z}}{(p)}$. Equation (3.14) is called t mixed radix representation of the adjoint solution.

In the sixth and final step, this mixed radix representation (3.14) is transformed into the more usual fixed radix form

$$y(z) = \sum_{k=0}^{\delta} y_k z^k \tag{3.15}$$

$$d(z) = \sum_{k=0}^{\delta} d_k z^k,$$

where y_k is vector of length N with elements in $\frac{\mathbf{Z}}{(p)}$ and $d_k \epsilon \frac{\mathbf{Z}}{(p)}$.

3.3. Cost Analysis

To analyse the cost of obtaining the adjoint solution $\left(y(z), d(z)\right)$ to satisfy (3.2), the measure used is the number of arithmetic operations required to find the solution with only the dominating terms retained.

The cost of precalculating the S_k for the CRT using (3.12) requires δ operations where δ is the bound on the degree of the solution's polynomials.

The cost of evaluating the matrix A(z) and the vector b(z) at the $\delta+1$ evaluation points, where Horner's rule is used, is $2\delta\partial N^2$ operations.

Calculating the $\delta+1$ adjoint solutions to the systems over $\frac{Z}{(p)}$, using the procedure of Chapter 2, requires $\frac{2}{3}\delta N^3$ operations.

To calculate the coefficient, a_k, of the term at the k^{th} stage of the CRT application, using (3.9) and the precalculated S_k, involves the evaluation of the polynomial $d^{[k-1]}(z)$ of degree k-1 at the point z_k, plus 2 operations to find a_k. Thus, to find the coefficient a_k requires 2k operations. Therefore, the total number of operations for the application of the CRT to the N+1 elements of the solution to get polynomials of degree δ requires $\delta^2 N$ operations.

The last count to consider is that required to transform the solution from the mixed radix form (3.14) to the fixed radix from (3.15). This transformation requires $\delta^2 N$ operations.

Therefore, the total operation count for finding the adjoint solution in the form (3.15), given the bound $\delta \leq \partial N$, has the dominating terms

$$4\partial^2 N^3 + \frac{2}{3}\partial N^4. \tag{3.16}$$

3.4. Terminating the Iteration

When calculating the terms of the polynomial solution up to the bound, δ, unnecessary calculations are performed when the bounds are too pessimistic. The following theorem can be used to determine if sufficient terms of the polynomial solution have been found to guarantee that the adjoint solution $\left(y(z),d(z)\right)$ has been found such that, when $d(z) \neq 0$,

$$x(z) = \frac{y(z)}{d(z)}. \tag{3.17}$$

The theorem is an extension to the polynomial case of the theorem presented by Cabay [4] for solving linear systems over the integers.

One point to notice about the theorem is that the coefficients of the polynomials need not be restricted to a field and may be over an arbitrary intergal domain.

Theorem 3.1

Suppose $A(z) = \left[a_{ij}(z)\right]$ is a matrix of order N and $b(z) = \left[b_i(z)\right]$ is a vector of length N, where

$$a_{ij}(z) = a_{ij}^{(0)} + a_{ij}^{(1)}\, z + a_{ij}^{(2)}\, z^2 + \cdots + a_{ij}^{(\partial)}\, z^{\partial} \tag{3.18}$$

$$b_{ij}(z) = b_{ij}^{(0)} + b_{ij}^{(1)}\, z + b_{ij}^{(2)}\, z^2 + \cdots + b_{ij}^{(\partial)}\, z^{\partial}.$$

are such that $deg\left(a_{ij}(z)\right) \le \partial$ and $deg\left(b_i(z)\right) \le \partial$ for all i,j , $0 \le i,j \le N$.

If the mixed radix representation of the adjoint solution $\left(y(z),d(z)\right)$ is

$$y(z) = y_I(z) + 0\cdot\left(z - z_0\right)\cdots\left(z - z_M\right) \tag{3.19}$$

$$+ \cdots + 0\cdot\left(z - z_0\right)\cdots\left(z - z_{M+\partial-1}\right) + y_{M+\partial+1}(z)\cdot\left(z - z_0\right)\cdots\left(z - z_{M+\partial}\right)$$

$$d(z) = d_I(z) + 0\cdot\left(z - z_0\right)\cdots\left(z - z_M\right)$$

$$+ \cdots + 0\cdot\left(z - z_0\right)\cdots\left(z - z_{M+\partial-1}\right) + d_{M+\partial+1}(z)\cdot\left(z - z_0\right)\cdots\left(z - z_{M+\partial}\right),$$

where

$$y_I(z) = y_0 + y_1\cdot\left(z - z_0\right) \tag{3.20}$$

$$+ y_2\cdot\left(z - z_0\right)\left(z - z_1\right) + \cdots + y_M\cdot\left(z - z_0\right)\cdots\left(z - z_{M-1}\right)$$

$$d_I(z) = d_0 + d_1\cdot\left(z - z_0\right)$$

$$+ d_2\cdot\left(z - z_0\right)\left(z - z_1\right) + \cdots + d_M\cdot\left(z - z_0\right)\cdots\left(z - z_{M-1}\right),$$

then

$$A(z)\, y_I(z) = d_I(z)\, b(z). \tag{3.21}$$

Proof

Suppose the contrary, i.e. $A(z)\, y_l(z) - d_l(z)\, b(z) \neq 0$. Then for some element of $A(z)\, y(z) - d(z)\, b(z)$, there exists a nonzero polynomial $p(z)$ such that

$$0 = \Big[A(z)\, y(z) - d(z)\, b(z)\Big]_l = \Big[A(z)\, y_l(z) - d_l(z)\, b(z)\Big]_l + p(z).$$

But

$$\Big[A(z)\, y(z) - d(z)\, b(z)\Big]_l = \Big[A(z)\, y_l(z) - d_l(z)\, b(z)\Big]_l + \Big[A(z)\, y_{M+\partial+1}(z) - d_{M+\partial+1}(z)\, b(z)\Big]_l \cdot \Big(\big(z - z_0\big) \cdots \big(z - z_{M+\partial}\big)\Big).$$

Therefore,

$$p(z) = \big(z - z_0\big) \cdots \big(z - z_{M+\partial}\big) \Big[A(z)\, y_{M+\partial+1}(z) - d_{M+\partial+1}(z)\, b(z)\Big]_l$$

$$= \big(z - z_0\big) \cdots \big(z - z_{M+\partial}\big) \left(\sum_{j=1}^{N} a_{lj}(z) \Big[\, y_{M+\partial+1}(z)\,\Big]_j - d_{M+\partial+1}(z)\, b_l(z)\right).$$

Since $p(z) \neq 0$,

$$\sum_{j=1}^{N} a_{lj}(z) \Big[\, y_{M+\partial+1}(z)\Big]_j - d_{M+\partial+1}(z)\, b_l(z) \neq 0$$

and its degree is ≥ 0. This implies

$$deg\Big(p(z)\Big) = deg\left(o\Big(z^{M+\partial+1}\Big) \cdot \left(\sum_{j=1}^{N} a_{lj}(z) \Big[\, y_{M+\partial+1}(z)\Big]_j - d_{M+\partial+1}(z)\, b_l(z)\right)\right)$$

$$> M + \partial.$$

Thus,

$$deg\Big(A(z)\, y_l(z) - d_l(z)\, b(z)\Big) > M + \partial.$$

But

$$\Big[A(z)\, y_l(z) - d_l(z)\, b(z)\Big]_l = \sum_{j=1}^{N} a_{lj}(z) \Big[\, y_l(z)\,\Big]_j - d_l(z)\, b_l(z),$$

where $deg\Big(\Big[\, y_l(z)\,\Big]_j\Big) \leq M$ and $deg\Big(\, d_l(z)\,\Big) \leq M$. Thus,

$$deg\left(\sum_{j=1}^{N} a_{lj}(z) \Big[\, y_l(z)\,\Big]_j - d_l(z)\, b_l(z)\right)$$

$$\leq \left[\sum_{j=1}^{N} \left(o\left(z^{\partial}\right) o\left(z^{M}\right)\right) - o\left(z^{M}\right) o\left(z^{\partial}\right)\right] = deg\left(o\left(z^{M+\partial}\right)\right)$$

$$\leq M + \partial.$$

Therefore,

$$deg\left[A(z)\, y_i(z) - d_i(z)\, b(z)\right] \leq M + \partial,$$

and thus the contradiction.

QED

Thus, bounds for the degree of the adjoint solution are now unnecessary. All that is required to terminate the iteration is that a sufficient number of consecutive zero coefficients for all of the polynomials of the solution be encountered. If the bounds on the degree of the adjoint solution are tight then as many as ∂ extra iterations are done to find a sufficient number of zeros to satisfy the theorem than would have been required using the bound. But when the bounds are not tight, the theorem can result in substantial savings by terminating the iteration long before the bounds would be reached.

If at stage $M+\partial+1$, where ∂ and M are defined as in Theorem 3.1, the theorem is applied to terminate the iteration of the algorithm, the result is the adjoint solution in the mixed radix form (3.19). Thus if $d_i(z)\neq 0$ then

$$x(z) = \frac{y_i(z)}{d_i(z)} = \frac{y(z)}{d(z)} = \frac{y_i(z) + y_{M+\partial+1}(z)\left(z - z_0\right)\cdots\left(z - z_{M+\partial}\right)}{d_i(z) + d_{M+\partial+1}(z)\left(z - z_0\right)\cdots\left(z - z_{M+\partial}\right)}, \tag{3.22}$$

where $\left(y(z),d(z)\right)$ is the complete solution to the system. Therefore,

$$A(z)\left(y_i(z) + y_{M+\partial+1}(z)\left(z - z_0\right)\cdots\left(z - z_{M+\partial}\right)\right) \tag{3.23}$$

$$= \left(d_i(z) + d_{M+\partial+1}(z)\left(z - z_0\right)\cdots\left(z - z_{M+\partial}\right)\right) b(z).$$

This implies

$$0 = \left[A(z)\, y_l(z) - d_l(z)\, b(z)\right] \tag{3.24}$$
$$+ \left(z - z_0\right) \cdots \left(z - z_{M+\delta}\right) \left[A(z)\, y_{M+\delta+1}(z) - d_{M+\delta+1}(z)\, b(z)\right].$$

Thus,

$$0 = A(z)\, y_{M+\delta+1}(z) - d_{M+\delta+1}(z)\, b(z).$$

As a result, $\left(y_{M+\delta+1}(z), d_{M+\delta+1}(z)\right)$ is a solution to the system as well. Assuming $d_{M+\delta+1}(z) \neq 0$, (for if it were then $y_{M+\delta+1}(z)=0$, as well) and since $d_l(z) \neq 0$, this results in

$$x(z) = \frac{y_l(z)}{d_l(z)} = \frac{y_{M+\delta+1}(z)}{d_{M+\delta+1}(z)}. \tag{3.25}$$

Assuming that the $GCD\left(y_l(z), d_l(z)\right) = 1$ (if it is not, then remove the common factor), then $\left(y_{M+\delta+1}(z), d_{M+\delta+1}(z)\right)$ look like

$$y_{M+\delta+1}(z) = M(z)\, y_l(z) \tag{3.26}$$
$$d_{M+\delta+1}(z) = M(z)\, d_l(z)$$

for some M(z). The result is that the complete solution, $\left(y(z), d(z)\right)$, is of the form

$$y(z) = y_l(z) \left(1 + M(z) \left(z - z_0\right) \cdots \left(z - z_{M+\delta}\right)\right) \tag{3.27}$$
$$d(z) = d_l(z) \left(1 + M(z) \left(z - z_0\right) \cdots \left(z - z_{M+\delta}\right)\right).$$

As will be seen later in Chapter 6, using the solution $\left(y_l(z), d_l(z)\right)$ can lead to problems when solving over the integers. Therefore, if it is possible to find the polynomial $M(z)$ without further iteration of the algorithm, the entire solution $\left(y(z), d(z)\right)$ could be constructed at this point. But (3.27) can be viewed as a system of N + 1 equations in the N + 2 unknowns y(z) (a vector of length N), d(z) and M(z). Thus M(z) cannot be solved for immediately. In addition, the following example illustrates that not enough information is available to determine M(z) directly.

Example

Let

$$A(z) = \begin{bmatrix} z\,(z-1)\cdots(z-7) & z \\ -1 & \Big(1 + (z-15)\Big)(z-8)(z-9)\cdots(z-14) \end{bmatrix} \tag{3.28}$$

$$b(z) = \begin{bmatrix} 3\,z\,(z-1)\cdots(z-9) + z\,(z-16)(z-17) \\ \Big(1 + (z-15)\Big)(z-8)(z-9)\cdots(z-14)(z-16)(z-17) - 3\,(z-8)(z-9) \end{bmatrix}.$$

The solution x(z) is then

$$x(z) = \begin{bmatrix} 3\,(z-8)(z-9) \\ (z-16)(z-17) \end{bmatrix}. \tag{3.29}$$

and the adjoint solution $\Big(y(z),d(z)\Big)$ is given by

$$y(z) = \Big(z + \Big(1 + (z-15)\Big)\,z\,(z-1)\cdots(z-14)\Big) \begin{bmatrix} 3\,(z-8)(z-9) \\ (z-16)(z-17) \end{bmatrix}, \tag{3.30}$$

$$d(z) = z + \Big(1 + (z-15)\Big)\,z\,(z-1)\cdots(z-14).$$

When solving the system using the modular techniques of this chapter, the complete adjoint solution $\Big(y(z),d(z)\Big)$, with coefficients in $\frac{\mathbf{Z}}{(p)}$, $p = 45233$, is

$$y(z) = y_I(z) + z\,(z-1)\cdots(z-13)\cdot y_{M+\delta+1}(z) \tag{3.31}$$

$$d(z) = d_I(z) + z\,(z-1)\cdots(z-13)\cdot d_{M+\delta+1}(z),$$

where the application of Theorem 3.1 at stage 14 results in the solution $\Big(y_I(z),\, d_I(z)\Big)$, in mixed radix form, of

$$y_I(z) = \begin{bmatrix} 168\,z - 42\,z\,(z-1) + 3\,z\,(z-1)(z-2) \\ 240\,z - 30\,z\,(z-1) + z\,(z-1)(z-2) \end{bmatrix}, \tag{3.32}$$

$$d_I(z) = z$$

and with $\Big(y_{M+\delta+1}(z),\, d_{M+\delta+1}(z)\Big)$ as

$$y_{M+\partial+1}(z) = (z-14) \times \begin{bmatrix} 126 + 210\,(z-15) + 51\,(z-15)(z-16) + 3\,(z-15)(z-16)(z-17) \\ 2 - 2\,(z-15) + (z-15)(z-16) + (z-15)(z-16)(z-17) \end{bmatrix}, \tag{3.33}$$

$$d_{M+\partial+1}(z) = (z-14)^2.$$

Since the $GCD\left(y_I(z), d_I(z)\right) = z$, the reduced form of the solution $\left(y_{I_r}(z), d_{I_r}(z)\right)$, when expanded, is

$$y_{I_r}(z) = \frac{y_I(z)}{z} = \begin{bmatrix} 3z^2 - 51z + 216 \\ z^2 - 33z + 272 \end{bmatrix} = \begin{bmatrix} 3\,(z-8)(z-9) \\ (z-16)(z-17) \end{bmatrix}, \tag{3.34}$$

$$d_{I_r}(z) = \frac{d_I(z)}{z} = 1.$$

Using (3.33) and (3.34) to find the $M(z)$ so that (3.26) holds results in

$$M(z) = (z-14)^2. \tag{3.35}$$

Therefore, there is no indication at stage $M + \partial + 1$ that the factor M(z) is $(z-14)^2$. One of the factors (z - 14) is due to the fact the non-zero portion of the remaining terms of the mixed radix solution does not start until 2 iterations after the current stage (i.e., the next iteration produces zero coefficients, as well). Thus the multiple $M(z)$ is not immediately available and, if the entire solution $\left(y(z),d(z)\right)$ is required, it will have to be calculated by the continued iteration of the algorithm beyond the stage at which the termination criteria can be applied.

CHAPTER 4

Power Series Methods over $\frac{Z[z]}{(p)}$

4.1. Generation of the Power Series Solution

Let A(z) be a matrix of order N and b(z) be a vector of length N, where the elements of A(z) and b(z) are polynomials of degree at most ∂ with coefficients in $\frac{Z}{(p)}$. This section describes the generation of the power series solution x(z), of

$$A(z)\,x(z) = b(z). \tag{4.1}$$

For the generation of the solution, $A(z_0)$, $z_0 \in \frac{Z}{(p)}$, is required to be non-singular, for some z_0. Given the system A(z) and b(z) in the form

$$A(z) = \sum_{k=0}^{\partial} A'_k\, z^k \tag{4.2}$$

$$b(z) = \sum_{k=0}^{\partial} b'_k\, z^k,$$

where A'_k is a matrix of order N and b'_k a vector of length N, with entries in $\frac{Z}{(p)}$, the first step is to find the element z_0 such that $A(z_0)$ has rank N. As pointed out by Cabay [5], if $A(z_0)$ is singular for $\delta+1$ distinct points z_0, where δ is a bound on the degree of the determinant of A(z), then A(z) mod p is singular. To facilitate finding the matrix $A(z_0)$, the system (4.2) is put into the form

$$A(z) = \sum_{k=0}^{\partial} A_k \left(z - z_0\right)^k, \tag{4.3}$$

$$b(z) = \sum_{k=0}^{\partial} b_k \left(z - z_0\right)^k$$

where the elements of A_k and b_k are in $\frac{Z}{(p)}$. Now, with $A(z_0) = A_0$, the LU decomposition of Chapter 2 is applied to determine the rank of A_0. If the rank is not N, the process of transforming

the system (4.2) to the form (4.3) for another z_0 and determining the rank of the resultant constant matrix A_0 is repeated. If the rank is N then a non-singular $A(z_0)$ has been found and the transformed system (4.3), for this particular z_0, is used to find the power series solution to (4.1). This is done using the following theorem discussed by Cabay[5] to find the coefficient x_k of

$$x(z) = \sum_{k=0}^{\infty} x_k \left(z - z_0\right)^k. \tag{4.4}$$

Theorem 4.1

$$A_0\, x_k = -\sum_{i=1}^{\min(k,\partial)} A_i\, x_{k-i} + \begin{cases} b_k & 0 \le k \le \partial \\ 0 & \partial < k \end{cases} \tag{4.5}$$

■

This method is iterative and uses, at most, the last ∂ terms A_i, b_i, and x_i to compute the next coefficient x_k. Thus, the calculations of these coefficients can continue as long as required. A bound on the number of terms necessary to determine the solution uniquely is arrived at by noting that the conversion of a power series to rational function form, with both numerator and denominator of degree at most δ, requires at most $2\delta + 1$ terms of the power series.

The final step in producing the power series solution consists of converting the solution $x(z)$ in (4.4) to an expansion about the point zero. This involves transforming the power series (4.4), truncated to $2\delta + 1$ terms, into the form

$$x(z) = \sum_{k=0}^{2\delta} x'_k\, z^k. \tag{4.6}$$

Obviously, this transformation is required only when the z_0 of the first step is non-zero.

4.2. Cost Analysis of Power Series Solution Generation

To find a measure of the cost of the algorithm to generate the power series solution, the number of arithmetic operations needed to find the solution are counted with only the dominating terms being retained.

The cost arises from the three parts of the algorithm. The first step is the transformation of the system from (4.2) into (4.3) and the subsequent LU decomposition of A_0 to determine its rank. If $A(0)$ is non-singular then the system is already in the required form for the remainder of the algorithm, and the transformation is therefore required only when $z_0 \neq 0$. The transformation requires $\partial^2 N^2$ operations for each trial z_0. The LU decomposition, using the methods of Chapter 2, requires $\frac{2}{3}N^3$ operations for each such z_0.

Once the matrix A_0 is found with rank N, the LU decomposition from the determination of the rank of A_0 is available to be used to solve for the coefficients of the power series solution using (4.5). To solve for these coefficients, x_k, requires the calculation of the right hand side of (4.5) before the forward and back substitution is applied. The calculation of this right hand side requires $2\partial N^2$ operations for each successive k. Forward and back substitution to find each x_k requires an additional $2N^2$ operations. Therefore, the operation count to find each coefficient x_k is $2\partial N^2$ operations; thereby, to find $2\delta + 1$ coefficients of (4.4) requires $4\delta\partial N^2$ operations.

Thus the number of operations to find the solution (4.4) truncated to $2\delta + 1$ terms is

$$4\delta\partial N^2 + \frac{2}{3}N^3, \tag{4.7}$$

whenever $A(0)$ is non-singular. If $A(0)$ is singular, then the additional cost for each trial z_0 in step 1 to obtain the form (4.3) is

$$\partial^2 N^2 + \frac{2}{3}N^3 \tag{4.8}$$

operations.

To convert the solution from the form (4.4) to the form (4.6) when $z_0 \neq 0$ requires $4\delta^2 N$ operations.

Thus, using the bounds $\delta \leq \partial N$, though better bounds are given by McClellan[13], the operation count for finding the solution (4.6) to (4.1), where A_0 is non-singular, is

$$4\partial^2 N^3. \tag{4.9}$$

When $A(0)$ is singular the cost is

$$4\partial^2 N^3 + k\left(\frac{2}{3}N^3 + \partial^2 N^2\right). \tag{4.10}$$

where k is such that $A(i)$ is singular for $i = 0,1, \ldots, k-1$.

4.3. Pade Conversion of the Power Series

Once the power series solution, x(z), to (4.1) is obtained, the next step is to convert these power series into rational form. The method used for this conversion is the diagonal algorithm described by Cabay and Kao[6].

The method takes a power series a(z) and makes successive approximates $p_{i_k}(z)$ and $q_{i_k}(z)$, each of degree i_k such that

$$q_{i_k}(z)\, a(z) - p_{i_k}(z) = o\left(z^{2 i_k + 1}\right), \tag{4.11}$$

where $q_{i_k}(z)$ and p_{i_k} are in reduced from. The method is iterative on i_k for $k = 0,1,\ldots,m$ and as such the maximum degree i_m of $p_{i_m}(z)$ and $q_{i_m}(z)$ need not be known apriori. The iterative technique fits in with the generation of the power series solution of the previous section as it generates the terms in increasing degrees. Consequently, whenever the conversion technique requires another term in the power series, it can be generated upon request. The entire process can be halted when the conversion process has reached a sufficient degree, i_m, for the rational function approximates.

The algorithm obtains the next approximates $p_{i_{k+1}}(z)$ and $q_{i_{k+1}}(z)$ from the previous two approximates, $p_{i_k}(z)$ and $q_{i_k}(z)$ and $p_{i_{k-1}}(z)$ and $q_{i_{k-1}}(z)$, by working with their remainder power

series

$$q_{i_k}(z)\, a(z) - p_{i_k}(z) = o\left(z^{2\, i_k+1}\right) = o_E\left(z^{i_{k+1}\, +\, i_k}\right) \tag{4.12}$$

and

$$q_{i_{k-1}}(z)\, a(z) - p_{i_{k-1}}(z) = o\left(z^{2\, i_{k-1}+1}\right) = o_E\left(z^{i_k+i_{k-1}}\right), \tag{4.13}$$

where $o_E\left(z^{i_{k+1}\, +\, i_k}\right)$ indicates that the power series begins exactly with the power $z^{i_{k+1}\, +\, i_k}$. The approximates of degree i_{k+1} are determined by first defining the remainder power series (4.12) as follows

$$q_{i_k}(z)\, a(z) - p_{i_k}(z) = \left(r_{i_k}^{(0)} + r_{i_k}^{(1)} z + \cdots + r_{i_k}^{(S_k)}\, z^{S_k}\right) z^{i_{k+1}+i_k} + o\left(z^{2\, i_{k+1}+1}\right) \tag{4.14}$$

where $S_k = i_{k+1} - i_k$ and $r_{i_k}^{(0)} \neq 0$.

This is basically finding the first non-zero terms, $r_{i_k}^{(0)}$, in the remainder power series and defining the value i_{k+1} using $r_{i_k}^{(0)}$ such that

$$q_{i_k}(z)\, a(z) - p_{i_k}(z) = r_{i_k}^{(0)}\, z^{i_{k+1}+i_k} + o\left(z^{i_{k+1}+i_k+1}\right). \tag{4.15}$$

Thus, the polynomial $r_{i_k}(z)$, of degree S_k, is defined such that

$$r_{i_k}(z) = r_{i_k}^{(0)} + r_{i_k}^{(1)}\, z + \cdots + r_{i_k}^{(S_k)}\, z^{S_k} \tag{4.16}$$

by calculating the remainder power series' terms $r_{i_k}^{(1)}, \ldots, r_{i_k}^{(S_k)}$.

The next step in the algorithm is to make the remainder power series (4.13) such that the polynomial $r_{i_{k-1}}(z)$ is of the same degree as $r_{i_k}(z)$. Therefore, (4.13) becomes

$$\begin{aligned} q_{i_{k-1}}(z)\, a(z) - p_{i_{k-1}}(z) &= \left(r_{i_{k-1}}^{(0)} + r_{i_{k-1}}^{(1)} z + \cdots + r_{i_{k-1}}^{(S_k)}\, z^{S_k}\right) z^{i_k\, +\, i_{k-1}} \\ &\quad + o\left(z^{i_{k+1}\, +\, i_{k-1}+1}\right) \\ &= r_{i_{k-1}}(z)\, z^{i_k\, +\, i_{k-1}} + o\left(z^{i_{k+1}\, +\, i_{k-1}\, +\, 1}\right). \end{aligned} \tag{4.17}$$

The third step finds a polynomial $b(z)$ of degree S_k, such that $b(0) = 1$, and a constant c such that

$$c\, r_{l_{k-1}}(z) = b(z)\, r_{l_k}(z) + o\left(z^{S_k+1}\right). \tag{4.18}$$

Using the method pointed out by Cabay and Kao[6], this step becomes the solution of the lower triangular system

$$\begin{bmatrix} r_{l_k}^{(0)} & & & \\ r_{l_k}^{(1)} & r_{l_k}^{(0)} & & \\ \vdots & & \ddots & \\ r_{l_k}^{(S_k)} & r_{l_k}^{(S_k-1)} & \cdots & r_{l_k}^{(0)} \end{bmatrix} \begin{bmatrix} y^{(0)} \\ y^{(2)} \\ \vdots \\ y^{(S_k)} \end{bmatrix} = \begin{bmatrix} r_{l_{k-1}}^{(0)} \\ r_{l_{k-1}}^{(1)} \\ \vdots \\ r_{l_{k-1}}^{(S_k)} \end{bmatrix} \tag{4.19}$$

for the $y^{(l)}$. Then by normalizing the vector $\left[y^{(l)}\right]$ such that $b(z) = \sum_{l=0}^{S_k} \left(\left(y^{(0)}\right)^{-1} y^{(l)}\right) z^l$ and letting $c = \left(y^{(0)}\right)^{-1}$, they are as required. These c and b(z) are used to compute the new approximates by noting that by multiplying (4.14) by b(z) and subtracting $c \cdot z^{l_{k+1} - l_{k-1}}$ times (4.17) from it, $2S_k$ more terms in the resultant remainder series are zeroed out. Therefore, this is a better set of approximates to the power series a(z).

This arithmetic then produces the following form.

$$\begin{aligned} & b(z)\left(q_{l_k}(z)\, a(z) - p_{l_k}(z)\right) - c\, z^{l_{k+1} - l_{k-1}}\left(q_{l_{k-1}}(z)\, a(z) - p_{l_{k-1}}(z)\right) \\ &= \left(b(z)\, q_{l_k}(z) - c\, z^{l_{k+1} - l_{k-1}}\, q_{l_{k-1}}(z)\right) a(z) - \left(b(z)\, p_{l_k}(z) - c\, z^{l_{k+1} - l_{k-1}}\, p_{l_{k-1}}(z)\right) \\ &= q_{l_{k+1}}(z)\, a(z) - p_{l_{k+1}}(z), \end{aligned} \tag{4.20}$$

by setting

$$q_{l_{k+1}}(z) = b(z)\, q_{l_k}(z) - c\, z^{l_{k+1} - l_{k-1}}\, q_{l_{k-1}}(z) \tag{4.21}$$

and

$$p_{l_{k+1}}(z) = b(z)\, p_{l_k}(z) - c\, z^{l_{k+1} - l_{k-1}}\, p_{l_{k-1}}(z).$$

Evaluation of (4.20) using (4.14), (4.17) and (4.18) yields

$$b(z)\left(r_{i_k}(z)\,z^{i_{k+1}+i_k}+o\left(z^{2i_{k+1}+1}\right)\right) \tag{4.22}$$

$$-c\,z^{i_{k+1}-i_{k-1}}\left(r'_{i_{k-1}}(z)\,z^{i_k+i_{k-1}}+o\left(z^{i_{k+1}+i_{k-1}+1}\right)\right)$$

$$=z^{i_{k+1}+i_k}\left(b(z)\,r_{i_k}(z)-c\,r'_{i_{k-1}}(z)\right)+o\left(z^{2i_{k+1}+1}\right)$$

$$=o\left(z^{2i_{k+1}+1}\right).$$

Therefore, these new estimates $p_{i_{k+1}}(z)$ and $q_{i_{k+1}}(z)$ satisfy the condition set out by (4.11) and are of degree i_{k+1}.

This process is iterated for $k = 0,1,\ldots,m+1$, where $i_{m+1} > \delta$, where completion of the next approximation step would create approximates of higher degree than required.

4.4. Cost Analysis of the Pade Conversion

The cost of the algorithm to find the rational function solution to (4.1) using the power series generation of section 4.1 is calculated by noting that the costs of both algorithms, the power series generation and the Pade conversion, are independent. The algorithm for the Pade conversion to rational form requires

$$6\delta^2 N \le 6\partial^2 N^3 \tag{4.23}$$

operations for the N elements of the power series solution (see Cabay and Kao [6]). Therefore, using the estimates (4.9) and (4.10), the total cost of the power series method for computing the solution in rational form is

$$10\,\partial^2 N^3 + k\left[\frac{2}{3}N^3+\partial^2 N^2\right] \tag{4.24}$$

operations where k is such that $A(i)$ is singular for $i = 0,1,\ldots,k-1$.

4.5. Problems with the Power Series Methods

In the calculation of the solution to the system $\left[A(z),b(z)\right]$ in rational form, the solution elements obtained by the Pade conversion algorithm are rational functions, $\frac{y_j(z)}{d_j(z)}$, $j = 1,2, \ldots, N$, in reduced form. As mentioned before, the $y_j(z)$, $j = 1,2,..,N$, when calculated to the degree δ, will divide the adjoint solution elements. Similarly, the $d_j(z)$ will divide the determinant $d(z)$ of $A(z)$. However, in practice it does become desirable to express the solution in the form

$$\frac{y_H(z)}{d_H(z)} \tag{4.25}$$

where $d_H(z)$ is the least common multiple (LCM) of the $d_j(z)$, $j = 1,2,...,N$. For example, if the partial solutions in the form (4.25) were available at every step k, $k = 0,1, \ldots, m$, then the termination test proposed by Cabay[5] could be applied.

One method of finding the LCM of two polynomials, $d_l(z)$ and $d_j(z)$, $l \neq j$, of degree i_k, is first to determine the power series, $a(z)$, of their quotient, $\frac{d_l(z)}{d_j(z)}$, then to apply the Pade algorithm to it. The resulting rational form, $\frac{t(z)}{s(z)}$, is in reduced form. Thus

$$d_l(z) = t(z)\, g(z) \tag{4.26a}$$

and

$$d_j(z) = s(z)\, g(z), \tag{4.26b}$$

where $g(z) = GCD\left(d_l(z),d_j(z)\right)$. Then, the least common multiple, $d(z) = LCM\left(d_l(z),d_j(z)\right)$ is given by

$$d(z) = t(z)\, s(z)\, g(z) = t(z)\, d_j(z) = s(z)\, d_l(z). \tag{4.27}$$

Finding the truncated power series of degree $2\, i_k$, where i_k is the maximum of the degrees of $d_l(z)$ and $d_j(z)$ at the k^{th} stage, $k = 0,1, \ldots, m$, can be done using the method of solving the lower triangular system (4.19) with the two polynomials $d_l(z)$ and $d_j(z)$ augmented by trailing zero terms to get the appropriate degree $2\, i_k$. The number of operations to complete this is $4\, i_k^2$. Then

the application of the Pade conversion to the resultant power series requires at most 6 i_k^2 operations. The multiplication of the polynomials to get common denominators and the appropriate numerators is bounded by 6 i_k^2 operations. Therefore, the cost of placing the two rational functions $\frac{y_i(z)}{d_i(z)}$ and $\frac{y_j(z)}{d_j(z)}$ over a common denominator is 16 i_k^2 operations.

To calculate the LCM of the N denominators of the vector of rational functions, a divide and conquer technique can be used to reduce the growth of the power series. The degree of the truncated power series representing a quotient must be 2 i_k where i_k is the largest degree of the two polynomials making up the quotient at the k^{th} stage. Using this technique, at the first stage 2 i_k + 1 terms are needed, 4 i_k + 1 terms at the second , . . . , and $\left(2^{\lceil \log_2 N \rceil}\right) i_k + 1$ terms at the final stage as the denominators at each stage, j, have a degree bounded by $2^{j-1} i_k$, $j = 1,2, \ldots, \lceil \log_2 N \rceil$.

The cost of calculating the common denominator is then

$$2 \cdot \sum_{j=1}^{\lceil \log_2 N \rceil} \left\lceil \frac{N}{2^j} \right\rceil \left(2^j \left(2^{j-1} i_k\right)^2 + 6 \left(2^{j-1} i_k\right)^2\right). \tag{4.28}$$

Setting $m = \lceil \log_2 N \rceil$, (4.28) becomes

$$\frac{2^m i_k^2}{2} \left[\sum_{j=1}^{m} 4^j + 6 \sum_{j=1}^{m} 2^j\right] \tag{4.29}$$

$$= \frac{2}{3} i_k^2 2^m \left((2^m)^2 - 1\right) + 6 i_k^2 2^m \left(2^m - 1\right)$$

Using the fact that $2N > 2^m$, the number of operations required is

$$\frac{2}{3} i_k^2 \left(2N\right)^3. \tag{4.30}$$

To calculate the common denominator at every step for the rational function approximations of degree i_k for $k = 0,1,2, \ldots, m$, where $i_k \geq k$, in the worst case, results in an algorithm requiring

$$\frac{16}{9}\delta^3 N^3 \tag{4.31}$$

operations. Using the bound $\delta \leq \partial N$, (4.31) becomes

$$\frac{16}{9}\partial^3 N^6. \tag{4.32}$$

The calculation of the common denominators is, therefore, more expensive than the calculation of the solution elements in rational form.

When $A(z)$ modulo p is singular, there exists another factor in the cost of the algorithm for finding the power series solution. In this case, the attempt to find a z_0 such that $A(z_0)$ is non-singular fails for $\delta + 1$ distinct points $z_0 \in \frac{\mathbf{Z}}{(p)}$. This, as pointed out before, is sufficient criteria to guarantee $A(z)$ is singular. As a result, the power series solution $x(z)$ of the system cannot be found as the existence of $x(z)$ relies on the fact $A(z)$ *mod p* is non-singular. Chapter 6 investigates the problems arising out of this inability to find a solution when $A(z)$ mod p is singular when finding a solution of a system of polynomials over the integers. First, the cost of finding that $A(z)$ modulo p is singular is investigated.

The operation count of finding $A(z)$ singular requires that for $\delta + 1$ points, z_0, the system be transformed to the form (4.3) and the LU decomposition procedure applied to determine that the rank of A_0 is $< N$. This requires $\partial^2 N^2 + \frac{2}{3}N^3$ operations for each point (by 4.8). Using the bounds $\delta \leq \partial N$, finding out that $A(z)$ is singular requires

$$\partial^3 N^3 + \frac{2}{3}\partial N^4 \tag{4.33}$$

operations. The terms of this operation count for finding $A(z)$ mod p singular, add a ∂ and N order of complexity to the algorithm, respectively.

Therefore, the cost of finding that the matrix $A(z)$ is singular over $\frac{\mathbf{Z}}{(p)}$ is greater than the cost of finding the solution if $A(z)$ is not singular as ∂ and N increase. The only saving grace to this, is that, with the random matrices, it is rare that more than one value z_0 is tried before finding

a non-singular $A(z_0)$ and even rarer that $A(z)$ is singular. Thus, the costs involved to find an appropriate z_0 are rarely encountered and of finding $A(z)$ singular even more so.

CHAPTER 5

Comparison of Methods over $\frac{Z[z]}{(p)}$

5.1. Introduction

In this chapter, the power series methods of solving systems of polynomial equations over $\frac{Z}{(p)}$ are compared with the modular method on the basis of empirical timing. The implementation of the algorithms is done in the language ALGOLW in the multi-programming environment provided by the operating system MTS running on an AMDAHL 580/5860. The times presented in this chapter are in milliseconds of CPU time on the 5860.

The programming of the algorithms is done with the philosophy that any steps that are similar between the algorithms use the same routines to perform these steps. For example, the LU decomposition of Chapter 2 is implemented in a routine called by all the methods' procedures. Also since the Pade algorithm for the conversion of the power series to rational function form is inserted into the procedure to generate the power series solution, the code is exactly the same for the portion of the algorithm to generate the terms of the power series in both. In addition, the polynomials output from the procedures are in the same form as the polynomials of the system that is inputted (that is, that they are polynomials in z).

Linked lists are used to represent the polynomials of the systems to give flexibility to the programs when the systems get large. To avoid excessive overhead due to the system's allocation of the records of the linked lists, as these records are dynamically allocated by the system, record collection routines were written and used to collect discarded records for reuse when possible. This garbage collection was used in all the programs.

The primes, p, for the implementation are chosen so that $p^2 < L$ where L is the bound of the magnitude of single precision integer arithmetic for the computer. On the AMDAHL 580/5860, a 32-bit machine, $L = 2^{31} - 1$, thus $p < +\sqrt{2^{31} - 1}$. As will be seen in Chapter 6, when solving systems over the integers, several p_l are required. All these primes satisfy the above condition.

By so limiting the value of p, all arithmetic is performed using single precision operations. That is, finding a value modulo p is done whenever an arithmetic operation threatens to have a result whose magnitude is larger than p and before the magnitude exceeds the bound L. In particular, this then allows for the operation $\left(a \cdot b + c\right)$ to be computed before finding the value modulo p is required, where a, b, and c have magnitudes less than p. The cost of finding the values modulo p is ignored when obtaining cost estimates for all methods. Inclusion would increase the cost estimates of both methods by the same factor of $\frac{3}{2}$.

For this implementation, the inverse, b, of an element $a \neq 0$ in $\frac{\mathbf{Z}}{(p)}$ is obtained by applying Euclid's extended algorithm to a and p. Since a and p are relatively prime, this yields the values b and c where $b < p$ such that

$$ab + cp = 1. \tag{5.1}$$

Clearly, then

$$b \equiv a^{-1} \bmod p. \tag{5.2}$$

5.2. Description of Testing Procedure

The programs were set up as procedure calls that solve the polynomials systems modulo a prime in a manner transparent to the calling routing except for the format of the results. These procedures are set up in a driver that allows for the input and output of data with very little overhead. The input data is assumed to have coefficients that are smaller in magnitude than the primes, p_l, where the system is solved modulo each of these primes. This driver then iterates several times solving the system modulo a different prime for each iteration. The amount of CPU time elapsed

during the procedure call was recorded and output along with the average for the CPU times elapsed for all the calls.

The number of iterations performed varied with the size of the problem solved due to cost constraints but a sufficient number is used in every case until a stabilization of the times required per iteration has occurred. This allows for startup effects to become visible when they exist.

With the power series techniques, a startup effect became noticeable in the first iteration for large systems. Most of the effect is likely due to the larger memory requirements of the methods as compared to the modular method. Therefore, a larger number of calls to the system to get the dynamically allocated records of the linked lists is executed at the beginning of each test for these methods as compared to the modular method. On subsequent iterations of the procedure, the garbage collection provided for the reuse of the records and thus the times for these iterations stabilize at a lower level than the first. In the calculations of the averages for the cases when a startup effect was noticed, the time of the first iteration is not incorporated.

For the input data, once the size of problem (i.e., N and ∂) to be done is determined, the coefficients of the polynomials are randomly generated such that their magnitude is smaller than the smallest prime used, 45233. Each of the methods is then run independently on the set of data generated for each variation of the values of ∂ and N investigated and the times collected and averaged as explained above. The data for these timing runs is presented in Table 5.1.

It should be noted that with the power series methods, a value of $z_0 \neq 0$ does increase the cost of the algorithms. But, as was previously stated, a value of $z_0 \neq 0$ is rare. In fact, in the timing runs of Table 5.1, there are no cases where $A(0)$ *mod* p is singular and thus, $z_0 = 0$ for all the runs.

In Table 5.1, the column labelled MODULAR refers to the time, in milliseconds, required to solve a particular problem by the modular method, the column labelled POWER SERIES refers to the time required to produce the power series solution of the same problem, and the column

Times of Test Runs (msec.)				
Degree	N	Method		
		MODULAR	RATIONAL FUNCTION	POWER SERIES
1	4	4.45	9.64	3.41
	8	36.62	47.81	18.15
	11	77.68	98.68	41.15
	12	115.68	122.99	51.98
	13	134.86	151.20	64.67
	14	173.28	181.88	79.38
	15	217.16	219.48	95.95
	16	269.23	264.66	115.52
	17	344.17	303.76	136.39
	18	408.81	355.60	160.69
	32	3365.76	1671.44	830.26
	40	7762.85	3270.10	1605.95
2	2	1.92	4.80	1.81
	3	4.59	11.53	4.10
	4	8.87	21.78	7.78
	8	59.83	114.38	45.20
	16	579.28	725.98	307.76
	22	1760.91	1751.39	765.07
	24	2410.11	2242.81	986.18
	26	3222.26	2809.22	1246.24
	32	6917.71	5138.47	2279.87
3	4	14.08	37.38	14.03
	8	95.66	211.72	84.32
	10	191.75	387.06	156.61
	16	936.60	1427.64	589.03
	20	2133.63	2663.59	1126.08
	30	8634.27	8679.44	3672.65
	31	10154.73	9502.14	4042.52
	32	11020.56	10396.16	4438.68
4	3	9.80	29.16	11.05
	4	20.41	58.61	22.40
	5	36.55	102.39	40.05
	6	61.29	163.63	63.70
	7	93.82	247.69	95.94
	8	144.93	345.22	136.16

Table 5.1 Times of Test Runs

Times of Test Runs (msec.)				
Degree	N	Method		
		MODULAR	RATIONAL FUNCTION	POWER SERIES
5	8	185.92	497.57	200.59
	9	265.20	685.30	277.96
	10	391.70	-	373.57
6	12	861.16	-	868.21
	13	1135.16	-	1092.81
	14	1438.39	-	1350.11
7	13	1396.64	-	1446.18
	14	1881.31	-	1792.65
	16	2811.05	-	2628.96
8	3	24.86	83.78	35.19
	8	369.88	1158.85	468.21
	14	2185.56	-	2299.46
	15	2839.40	-	2801.70
	16	3514.71	-	3374.68
16	3	72.67	274.33	124.06
	8	1134.52	4255.95	1731.98
32	3	229.73	982.22	467.94
	8	3878.86	-	6653.35

Table 5.2 Times of Test Runs (cont'd)

labelled RATIONAL FUNCTION refers to the time required by the power series method plus the additional time required to convert the power series solution to rational form. As a summary, the theoretical results of the costs of each of these methods is

$$\begin{aligned} &\textit{Modular} && 4\partial^2 N^3 + \frac{2}{3}\partial N^4 \\ &\textit{Power Series} && 4\partial^2 N^3 \\ &\textit{Rational Function} && 10\partial^2 N^3 \end{aligned} \tag{5.3}$$

Recognizing that overhead significantly effects these cost estimates for small ∂ and N, the following observations relating to the close correspondence between theory and implementation can be made.

(1) For small ∂ fixed and large N, the cost of the modular method increases by approximately a factor of 16 when N doubles.

(2) For ∂ fixed and large N, the cost of producing the power series solution and its rational form increases by approximately a factor of 8 when N doubles.

(3) For N fixed, the cost of producing the power series solution and its rational form increases by approximately a factor of 4 when ∂ doubles.

(4) The cost of producing the solution in rational form using the power series methods is approximately $\frac{5}{2}$ the cost of producing the power series solution.

The theoretical cost estimates given above suggest that for small ∂ and small N, the modular method is superior to the power series method for obtaining solutions in rational from. On the other hand, with ∂ fixed, this superiority is quickly lost as N grows in size. The crossover points for $\partial = 1$ and $\partial = 2$ are illustrated in Figure 5.1 and Figure 5.2, respectfully. Indeed, the primary objective for the implementations is to determine the class of problems for which each of the methods is superior. Using the theoretical estimates with ∂ fixed, the modular method should be superior to the power series method for obtaining solutions in rational form whenever

$$4\partial^2 N^3 + \frac{2}{3}\partial N^4 \leq 10\partial^2 N^3, \tag{5.4}$$

or equivalently, when

$$N \leq 9\partial. \tag{5.5}$$

Perhaps due partly to overhead and to implementation pecularities, but primarily due to the omission of lower order terms in the theoretical cost estimates, the crossovers occur somewhat later than expected. In Figure 5.3 the dotted curve illustrates that the experimental crossovers actually occur when

$$N = 7.5\,\partial + 8.0. \tag{5.6}$$

The solid curve in figure 5.3 representing the crossover points between the power series generation and the modular method of solution is exhibiting a quadratic trend in N. This is as expected since comparing the operation counts of the two methods gives

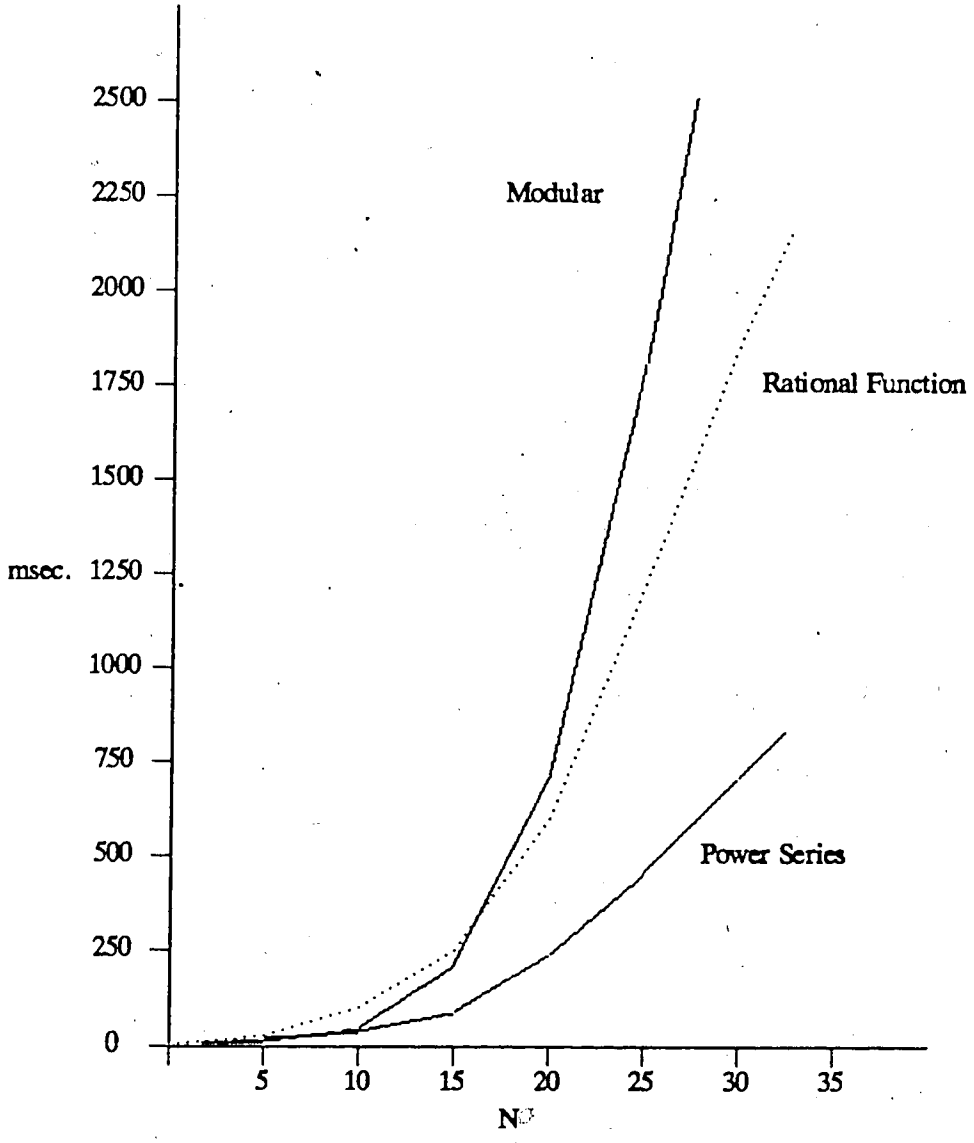

Figure 5.1 Timing of Methods($\partial = 1$)

$$\frac{2}{3}\,\partial N^4 + 4\,\partial^2 N^3 = 4\,\partial^2 N^3 + o\left(\partial^2 N^2\right). \tag{5.7}$$

Thus,

$$\frac{2}{3}\,N^2 = o\left(\partial\right).$$

Therefore,

$$\partial = c\,N^2 + o(N),$$

for some constant c. This cancelation of the terms of order $\partial^2 N^3$ in the difference of their operation counts results in the quadratic crossover curve for the two methods. Thus, as N increases, the power series solution quickly becomes the least costly of the two methods for a fixed degree ∂.

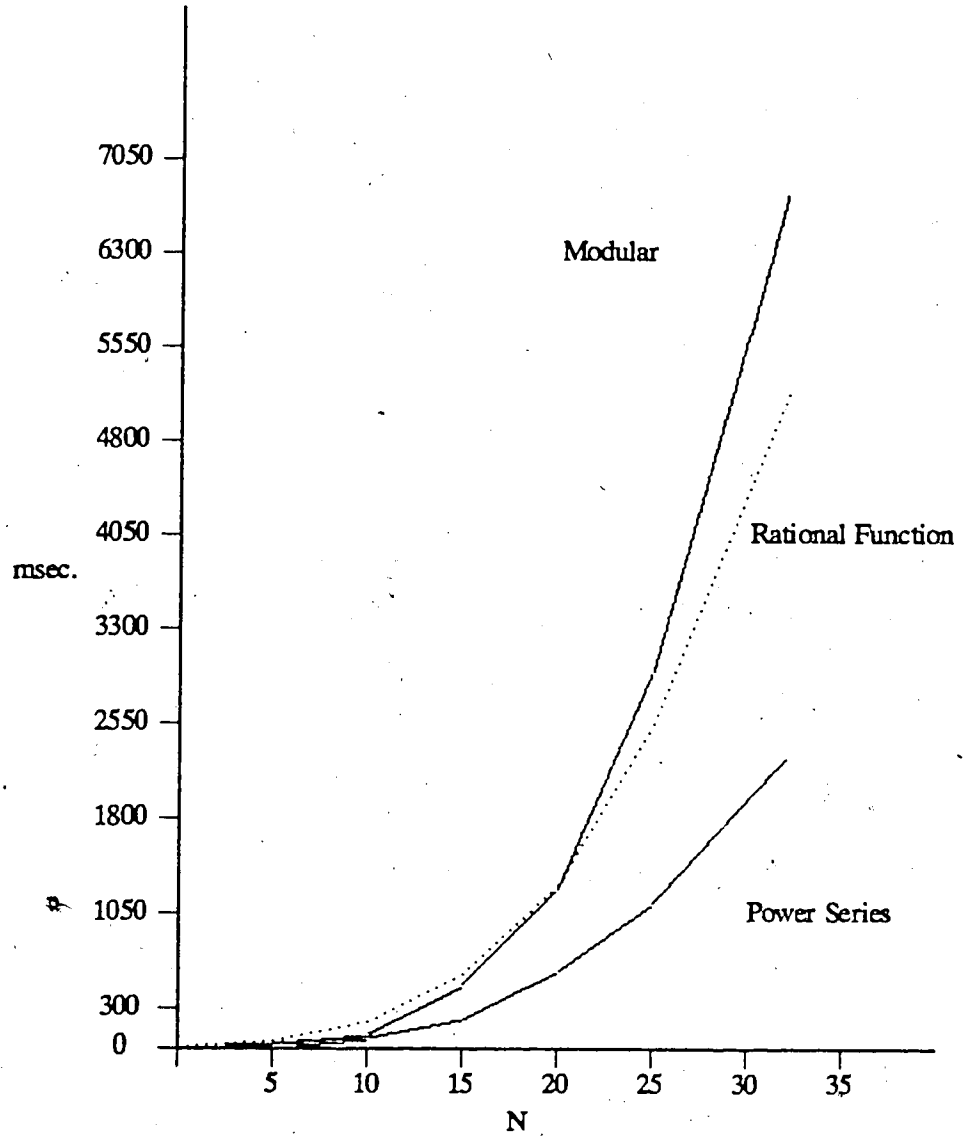

Figure 5.2 Timing of Methods ($\partial = 2$)

5.3. Conclusions

For solving systems over $\frac{Z[z]}{(p)}$, if solutions in rational form are required, the results of the previous section show that for small ∂ the power series method, together with the Pade conversion algorithm, is superior to the modular method. The converse is true for large ∂, that is, $\partial \geq \frac{2}{15} N$.

This anomaly between the choice of methods does not exist if solutions in truncated power series form only are required. The class of problems for which the power series method is superior to the modular method grows quadratically with N.

It can be observed that, in producing the solution in rational form by the power series method, 60 per cent of the cost is attributed to the Pade conversion algorithm. The conversion of

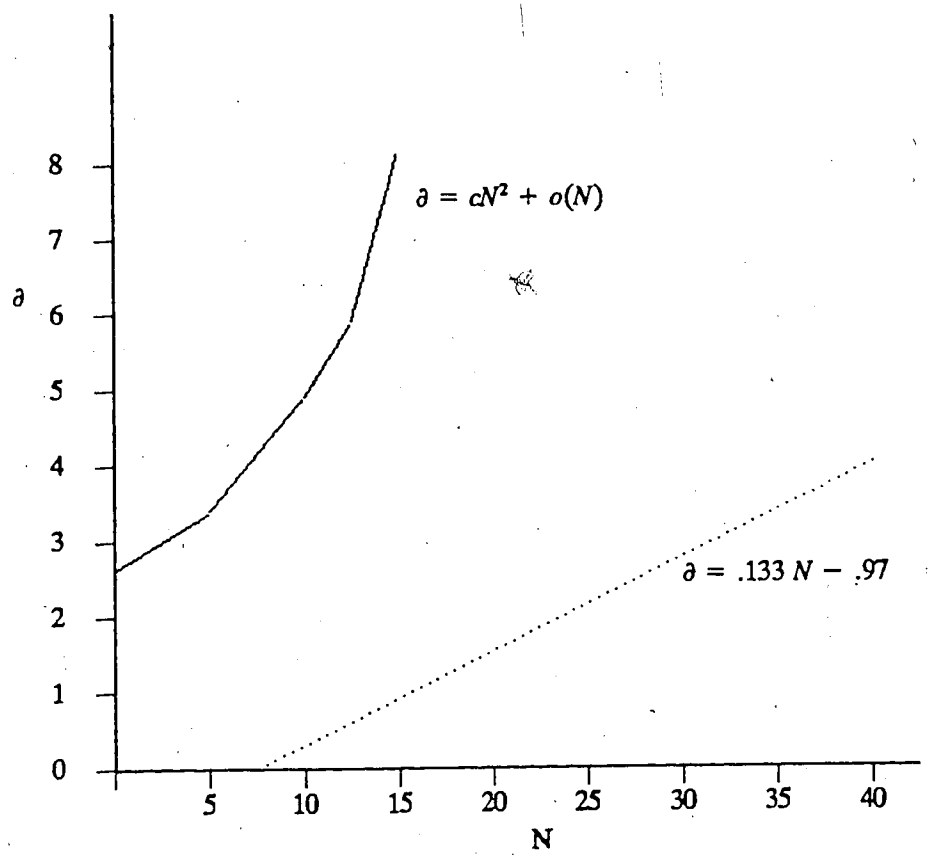

Figure 5.3 Crossover Curves for Modular vs Power Series Methods

one truncated power series requires $6\delta^2$ operations, where δ is the maximum degree of the polynomials in the resultant rational function. Better algorithms, which use fast polynomial arithmetic methods and which are of complexity $o\left(\delta \log^2\delta\right)$, are currently under theoretical and practical investigation (see Brent, Gustavson and Yun[3], Verheijen[15] and Choi[8]). Using these methods, the cost of producing the solution in rational form by the power series method is

$$4\partial^2 N^3 + c\cdot\partial N^2 \log^2(\partial N), \tag{5.8}$$

for some constant c. Thus, using fast methods, the Pade conversion algorithm does not add to the overall cost. Practically, however, the constant c is large, approximately 93 according to Verheijen[15], and the advantage of fast methods is again not realized until N is large. The determination of the crossover points where, using fast methods, the power series method becomes superior to the modular method for finding the solution in rational form is left as a subject for future research.

CHAPTER 6

Solving Systems over Z[z]

6.1. Description of the Method

Given the matrix $A(z)$, of order N, and the vector $b(z)$, of length N, where their entries are univariate polynomials with coefficients over the integers, finding the solution $x(z)$ such that

$$A(z)\,x(z) = b(z) \tag{6.1}$$

can be done using the methods of the previous chapters in conjunction with the CRT for integers.

The CRT for integers allows for the construction of the integer a from its residual modulo the distinct primes $p_0, p_1, p_2, \ldots, p_\gamma$

$$\bar{u}_i = u \bmod p_i,\ i = 0,1,2,\ldots,\gamma, \tag{6.2}$$

such that u is uniquely determined in the range 0 to $p_0p_1p_2\cdots p_\gamma - 1$ (or when a range symmetric about 0 is required, then in the range $-\frac{p_0p_1p_2\cdots p_\gamma - 1}{2}$ to $\frac{p_0p_1p_2\cdots p_\gamma - 1}{2}$). The Newtonian version of the CRT algorithm is used for this construction. Define the value $u^{[k]}$ at the k^{th} stage to be

$$u^{[k]} = u^{[k-1]} + a_k \prod_{i=0}^{k-1} p_i, \tag{6.3}$$

where

$$a_0 = u^{[0]} = u_0 \mod p_0 \tag{6.4}$$

$$a_k = \left(\left(u_k - u^{[k-1]}\right) \times S^{(k)}\right) \mod p_k$$

and

$$S^{(k)} = \prod_{i=0}^{k-1}\left(p_i^{-1}\right) \mod p_k = \left(\prod_{i=0}^{k-1} p_i\right)^{-1} \mod p_k. \tag{6.5}$$

To ensure $u^{[k]}$ is symmetric about 0, the a_i's must be such that

$$-\frac{p_i}{2} < a_i < \frac{p_i}{2} \quad , \; i = 0,1,2, \ldots ,k.$$

(If $u^{[k]}$ is required to be positive, the a_i must be positive.)

The result is the integer $u^{[\gamma]}$ in the mixed radix form

$$u^{[k]} = \sum_{k=0}^{\gamma} a_k \left(\prod_{i=0}^{k-1} p_i \right) = a_0 + a_1 p_0 + a_1 p_0 p_1 + \cdots + a_\gamma p_0 p_1 \cdots p_{\gamma-1}, \tag{6.6}$$

where

$$\left| a_k \right| \leq \frac{p_k - 1}{2} \; , \; i = 0,1, \ldots ,\gamma. \tag{6.7}$$

When γ is such that

$$\left| u \right| \leq \frac{p_0 p_1 \cdots p_\gamma - 1}{2}, \tag{6.8}$$

then $u = u^{[\gamma]}$.

To obtain the solution of (6.1), the first step is to determine for each prime p_i, $i = 0,1, \ldots ,\gamma$, the matrix $\overline{A_i(z)}$ and the vector $\overline{b_i(z)}$ such that

$$\overline{A_i(z)} = A(z) \; mod \; p_i \tag{6.9}$$

$$\overline{b_i(z)} = b(z) \; mod \; p_i.$$

The next step involves solving the system

$$\overline{A_i(z)} \; x_i(z) = \overline{b_i(z)} \quad mod \; p_i, \tag{6.10}$$

for each prime p_i. Using the modular method, this results in the adjoint solution $\left(\overline{y_i(z)}, \overline{d_i(z)}\right)$ of (6.10), where, for $i = 0,1, \ldots ,\gamma$,

$$\overline{y_i(z)} = \overline{A_i(z)}^{adj} \; \overline{b_i(z)} \quad mod \; p_i \tag{6.11}$$

is a vector of polynomials over $\frac{\mathbf{Z}}{(p_i)}$ and

$$\overline{d_i(z)} = \det\left(\overline{A_i(z)}\right) \quad mod \; p_i \; , \tag{6.12}$$

is a polynomial over $\frac{\mathbf{Z}}{(p_i)}$. Using the power series method, together with the Pade conversion algorithm, this results in vectors of rational functions

$$\overline{x_i(z)} = \overline{A_i(z)}^{-1}\,\overline{b_i(z)}\ ,\ i = 0,1,\ \ldots\ ,\gamma. \tag{6.13}$$

The numerators and denominators in the vector $\overline{x_i(z)}$ are, therefore, polynomials over $\frac{\mathbf{Z}}{(p_i)}$. Thus, with some regard for "bad primes", p_i, which are addressed in section 6.4, the modular method gives

$$y(z) = \overline{y_i(z)}\ mod\ p_i \tag{6.14}$$

$$d(z) = \overline{d_i(z)}\ mod\ p_i,$$

and the power series method gives

$$x(z) = \overline{x_i(z)}\ mod\ p_i. \tag{6.15}$$

In (6.15), it is intended that congruences are taken seperately for the numerators and denominators of the components.

Therefore, regardless of the method used, each coefficient in the polynomials in (6.11) and (6.12), or in (6.13), is computed modulo p_i, $i = 0,1,\ \ldots\ ,\gamma$. In the third and final step, the CRT is applied to each of these coefficients This yields the integer coefficients in the polynomials of $y(z)$ and $d(z)$, or in $x(z)$, as long as the integers are all bounded by

$$\frac{p_0 p_1 \cdots p_\gamma - 1}{2}.$$

Such a bound is determined in the next section.

6.2. Cost Analysis

To get a measure of the cost of finding the solution of (6.1), the first step is to find the cost for the reconstruction of a single integer coefficient from its residues. This reconstruction is done using modular arithmetic to avoid multi-precision arithmetic.

Let γ be the bound on the number of primes required to represent the solution's coefficients, such that the coefficient, u, satisfies (6.8). It is advantageous for the primes $p_0, p_1, \ldots, p_\gamma$ to be ordered such that $p_i < p_{i+1}$, $i = 0,1,\ \ldots\ ,\gamma-1$, as will be seen in section 6.3.

The cost of calculating the integer $u^{[k]}$ at the k^{th} stage (i.e., for the prime p_k), using modular arithmetic, involves the calculation of $u^{[k-1]} \bmod p_k$ and requires $2\left(k-1\right)$ operations. The additional operation of finding (6.4), given that the $S^{(k)}$ are precalculated, means the cost of finding (6.3) is $2k$ operations. Thus the calculation of a coefficient u in the form (6.6) requires γ^2 operations.

For each of the methods used to solve the system over the integers modulo p_i, the cost varies. To calculate the third step of algorithm, the CRT application, the number of polynomials in the solution and their degrees must be taken into account. Therefore, using ∂N as the bound on the degree of the polynomials of the adjoint solution, the number of operations to apply the CRT for each method are

Modular Solution	$\partial N^2\gamma^2$	
Power Series Solution	$2\partial N^2\gamma^2$	(6.16)
Rational Function Solution	$2\partial N^2\gamma^2$	

Using the cost analysis of Chapters 3 and 4, the total number of operations required to find the solutions modulo the $\gamma + 1$ primes (the second step), is

Modular Solution	$4\partial^2 N^3\gamma + \frac{2}{3}\partial N^4\gamma$	
Power Series Solution	$4\partial^2 N^3\gamma$	(6.17)
Rational Function Solution	$10\partial^2 N^3\gamma$	

Given that the magnitude of the coefficients of $A(z)$ and $b(z)$ is bounded by v, let t be such that

$$v < \frac{p_0 p_1 \cdots p_t}{2}.$$

Since finding the multiprecision coefficient v modulo the prime p_i require $t + 1$ operations, the calculation of (6.9) for the $\gamma + 1$ primes, (the first step), requires $t\gamma\partial N^2$ operations.

Using the bound on the size of the coefficient of the solution given by McClellan[13], γ is such that

$$N!\,\partial^N v^N < \frac{p_0 p_1 \cdots p_\gamma}{2}.$$

It follows then that $t \ll \gamma$. In fact γ can be approximated by $\gamma \approx tN$, by retaining the dominating terms only.

Thus the cost of finding the solution over the integers of (6.1) using each of the methods, in number of operations, is

Modular Solution	$\partial t^2 N^4 + 4\partial^2 t N^4 + \frac{2}{3}\partial t N^5$	(6.18)
Power Series Solution	$2\partial t^2 N^4 + 4\partial^2 t N^4$	
Rational Function Solution	$2\partial t^2 N^4 + 10\partial^2 t N^4$	

6.3. Terminating the Iteration

When calculating the solution to the system of equations of polynomials over the integers, unnecessary computation is done when the bound, γ, is larger than the actual number of primes required. Theorem 6.1 is designed to terminate the iteration when sufficient accuracy has been achieved in the integer coefficients of the adjoint solution $\left(y(z), d(z)\right)$ to (6.1) to guarantee a solution over the polynomials over the integers.

The theorem is an extension to polynomial systems with integer coefficients of the theorem presented by Cabay[4] for terminating the recursion for integer systems.

Theorem 6.1

Assume that the primes, p_i are ordered such that

$$p_0 < p_1 < p_2 < \cdots .$$

Suppose $A(z) = \left[a_{ij}(z)\right]$ is a matrix of order N and $b(z) = \left[b_j(z)\right]$ is a vector of length N with

$$a_{ij}(z) = a_{ij}^{(0)} + a_{ij}^{(1)}z + a_{ij}^{(2)}z^2 + \cdots + a_{ij}^{(\partial)}z^{\partial} \tag{6.19}$$

$$b_j(z) = b_j^{(0)} + b_j^{(1)}z + b_j^{(2)}z^2 + \cdots + b_j^{(\partial)}z^{\partial}$$

where ∂ is the maximum degree of the polynomials in A(z) and b(z) and the coefficients $a_{ij}^{(k)}$ and $b_j^{(k)}$ are integers. Suppose A(z) and b(z) are such that

$$\sum_{j=1}^{N}\sum_{k=0}^{\partial}\left|a_{ij}^{(k)}\right| \le p_0p_1p_2\cdots p_\gamma \tag{6.20}$$

$$\sum_{k=0}^{\partial}\left|b_i^{(k)}\right| \le p_0p_1p_2\cdots p_\gamma$$

for all i, $1 \le i \le N$.

If the mixed radix form of the adjoint solution (y(z),d(z)) is

$$y(z) = y_I(z) + \tag{6.21}$$

$$\sum_{k=0}^{\delta} z^k\left(0\cdot p_0p_1\cdots p_M + \cdots + 0\cdot p_0p_1\cdots p_{M+\gamma} + y_R^{(k)}\cdot p_0p_1\cdots p_{M+\gamma+1}\right)$$

$$d(z) = d_I(z) +$$

$$\sum_{k=0}^{\delta} z^k\left(0\cdot p_0p_1\cdots p_M + \cdots + 0\cdot p_0p_1\cdots p_{M+\gamma} + d_R^{(k)}\cdot p_0p_1\cdots p_{M+\gamma+1}\right)$$

where

$$y_I(z) = \sum_{k=0}^{\delta} z^k\cdot\left(y_0^{(k)} + y_1^{(k)}\cdot p_0 + \cdots + y_M^{(k)}\cdot p_0p_1\cdots p_{M-1}\right) = \sum_{k=0}^{\delta} z^k\cdot y_I^{(k)} \tag{6.22}$$

$$d_I(z) = \sum_{k=0}^{\delta} z^k\cdot\left(d_0^{(k)} + d_1^{(k)}\cdot p_0 + \cdots + d_M^{(k)}\cdot p_0p_1\cdots p_{M-1}\right) = \sum_{k=0}^{\delta} z^k\cdot d_I^{(k)}$$

with δ being the maximum degree of the solution and $y_I^{(k)} = \left[y_{I_i}^{(k)}\right]$, a vector of length N with entries from the integers mod p_I such that

$$\left|d_I^{(k)}\right| \le \left(p_I - 1\right)/2 \tag{6.23}$$

$$\left|y_{I_i}^{(k)}\right| \le \left(p_I - 1\right)/2$$

for all $l = 0,1, \ldots, M+\gamma+1$.

Then,

$$A(z)\, y_l(z) = d_l(z)\, b(z). \tag{6.24}$$

Proof

Suppose the contrary. This implies

$$A(z)\, y_l(z) - d_l(z)\, b(z) \neq 0 \tag{6.25}$$

but

$$A(z)\, y(z) - d(z)\, b(z) = 0. \tag{6.26}$$

Thus, there exists an i, $1 \leq i \leq N$, such that the i^{th} element of the vector (6.26), denoted

$$\left[A(z)\, y(z) - d(z)\, b(z)\right]_i,$$

there exists a polynomial c(z) with integer coefficients such that

$$\left[A(z)\, y_R(z) - d_R(z)\, b(z)\right]_i = c(z) = \sum_{k=0}^{\delta} c^{(k)} z^k \neq 0 \tag{6.27}$$

where

$$y_R(z) = \sum_{k=0}^{\delta} y_R^{(k)} z^k \tag{6.28}$$

$$d_R(z) = \sum_{k=0}^{\delta} d_R^{(k)} z^k,$$

such that

$$0 = \left[A(z)\, y(z) - d(z)\, b(z)\right]_i = \left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_i - c(z) \cdot p_0 p_1 \cdots p_{M+\gamma+1}. \tag{6.29}$$

Denote the coefficient of the k^{th} term in the polynomial

$$\left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_i \tag{6.30}$$

by

$$\left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_i^{(k)} \tag{6.31}$$

Thus,

$$\left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_l = \sum_{k=0}^{\delta} z^k \cdot \left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_l^{(k)} \tag{6.32}$$

and, using (6.27) and (6.29), this gives

$$0 = \sum_{k=0}^{\delta} z^k \cdot \left(\left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_l^{(k)} + c^{(k)} \cdot p_0 p_1 \cdots p_{M+\gamma+1}\right) \tag{6.33}$$

This implies that for all k such that $c^{(k)} \neq 0$, that

$$\left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_l^{(k)} = -c^{(k)} p_0 p_1 \cdots p_{M+\gamma+1} \tag{6.34}$$

Thus,

$$\left|\left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_l^{(k)}\right| = \left| c^{(k)} \right| p_0 p_1 \cdots p_{M+\gamma+1} \tag{6.35}$$

$$\geq p_0 p_1 \cdots p_{M+\gamma+1}.$$

But,

$$\left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_l = \left(\sum_{j=1}^{N} a_{lj}(z)\, y_{l_j}(z)\right) - \left(d_l(z)\, b_l(z)\right) \tag{6.36}$$

$$= \left(\sum_{j=1}^{N}\left(\sum_{k=0}^{\delta} z^k \sum_{l+h-k} a_{lj}^{(h)}\, y_{l_j}^{(l)}\right)\right) - \left(\sum_{k=0}^{\delta} z^k \sum_{l+h-k} d_l^{(h)}\, b_l^{(l)}\right)$$

$$= \sum_{k=0}^{\delta} z^k \left(\left(\sum_{j=1}^{N} \sum_{l+h-k} a_{lj}^{(h)}\, y_{l_j}^{(l)}\right) - \left(\sum_{l+h-k} d_l^{(h)}\, b_l^{(l)}\right)\right)$$

Combining (6.32) and (6.36) with (6.22) results in

$$\left[A(z)\, y_l(z) - d_l(z)\, b(z)\right]_l^{(k)} = \left(\sum_{j=1}^{N} \sum_{l+h-k} a_{lj}^{(h)}\, y_{l_j}^{(l)}\right) - \left(\sum_{l+h-k} d_l^{(h)}\, b_l^{(l)}\right) \tag{6.37}$$

$$= \left(\sum_{j=1}^{N}\left(\sum_{l+h-k} a_{lj}^{(k)} \left(y_{0_j}^{(l)} + y_{1_j}^{(l)}\, p_0 + \cdots + y_{M_j}^{(l)}\, p_0 p_1 \cdots p_{M-1}\right)\right)\right.$$

$$\left. - \left(\sum_{l+h-k} b_l^{(l)} \left(d_0^{(h)} + d_1^{(h)}\, p_0 + \cdots + d_M^{(h)}\, p_0 p_1 \cdots p_{M-1}\right)\right)\right.$$

$$= \left(\sum_{j=1}^{N} \sum_{s=0}^{M} \left(\prod_{v=0}^{s-1} p_v\right) \left(\sum_{l+h-k} a_{lj}^{(h)}\, y_{s_j}^{(l)}\right)\right) - \left(\sum_{s=0}^{M} \left(\prod_{v=0}^{s-1} p_v\right) \left(\sum_{l+h-k} b_l^{(l)}\, d_s^{(k)}\right)\right)$$

$$= \sum_{s=0}^{M} \left(\prod_{v=0}^{s-1} p_v \right) \left(\left(\sum_{l+h-k} a_{ij}^{(h)} y_{sj}^{(l)} \right) - \left(\sum_{l+h-k} b_i^{(l)} d_s^{(h)} \right) \right) \tag{6.37a}$$

Working with (6.19), (6.23) and (6.37a) results in

$$\left| \left(\sum_{j=1}^{N} \sum_{l+h-k} a_{ij}^{(h)} y_{sj}^{(l)} \right) - \left(\sum_{l+h-k} b_i^{(l)} d_s^{(h)} \right) \right| \tag{6.38}$$

$$\leq \left(\sum_{j=1}^{N} \sum_{l+h-k} \left| a_{ij}^{(h)} \right| \right) \left(\frac{p_s - 1}{2} \right) + \left(\sum_{l+h-k} \left| b_i^{(l)} \right| \right) \left(\frac{p_s - 1}{2} \right)$$

$$\leq \left(\frac{p_s - 1}{2} \right) \left(p_0 p_1 \cdots p_\gamma \right) + \left(\frac{p_s - 1}{2} \right) \left(p_0 p_1 \cdots p_\gamma \right) = \left(p_s - 1 \right) p_0 p_1 \cdots p_\gamma .$$

Combining (6.37) and (6.38) gives that

$$\left[A(z)\, y_j(z) - d_j(z)\, b(z) \right]_i^{(k)} \leq \sum_{s=0}^{M} \left(\left(\prod_{v=0}^{s-1} p_v \right) \left(p_s - 1 \right) p_0 p_1 \cdots p_\gamma \right) \tag{6.39}$$

But

$$\sum_{s=0}^{M} \left(\left(\prod_{v=0}^{s-1} p_v \right) \left(p_s - 1 \right) p_0 p_1 \cdots p_\gamma \right) = p_0 p_1 \cdots p_\gamma \left(\sum_{s=0}^{M} \left(\left(\prod_{v=0}^{s} p_v \right) - \left(\prod_{v=0}^{s-1} p_v \right) \right) \right) \tag{6.40}$$

$$= \left(p_0 p_1 \cdots p_\gamma \right) \left(p_0 p_1 \cdots p_M - 1 \right).$$

Therefore,

$$\left| \left[A(z)\, y_j(z) - d_j(z)\, b(z) \right]_i^{(k)} \right| \leq \left(p_0 p_1 \cdots p_\gamma \right) \left(p_0 p_1 \cdots p_M - 1 \right) \tag{6.41}$$

$$< \left(p_0 p_1 \cdots p_\gamma \right) \left(p_0 p_1 \cdots p_M \right)$$

$$< p_0 p_1 \cdots p_{M+\gamma+1},$$

since the primes are ordered.

Thus, (6.41) and (6.35) constitute the contradiction.

QED

The theorem is designed to be applied at stage $M + \gamma + 1$, where M and γ are defined as in Theorem 6.1, when the last $\gamma + 1$ applications of the CRT to each coefficient of the polynomials of the solution $\left(y(z), d(z) \right)$ has yielded only 0 terms in the mixed radix representation of the

coefficent.

6.4. Bad Prime Problems

When solving the polynomial systems with integer coefficients using the methods of Chapters 3 and 4 for solving the polynomial systems with coefficents in $\frac{\mathbf{Z}}{(p)}$, occasionally some problems are encountered with bad primes.

With the modular method the bad prime can only occur when Theorem 3.1 is utilized to terminate the iteration prior to finding the complete adjoint solution. Consider the system

$$A(z)\, y(z) = d(z)\, b(z) \quad mod\ p_i, \tag{6.42}$$

where $A(z)$ and $b(z)$ *mod* p_i have degree at most ∂. If Theorem 3.1 was applied to terminate the iteration while solving (6.42) then the system has its solution solved to a degree $M + \partial$, for some M as defined in Theorem 3.1. Assume, as well, that when solving the system (6.1) modulo another prime, p_j, $p_j \neq p_i$, the solution modulo p_j has degree δ where $\delta > M + \partial$. (Note, if the system modulo p_j used Theorem 3.1 to stop the iteration, the degree δ is defined since M is defined in the theorem.) Thus to apply the CRT for the integer coefficients using these 2 systems, the coefficients in the solution *mod* p_i consists of the terms of degree $M + \partial + 1$ and greater. But as pointed out in the example in Chapter 3, the nature of the terms of degree $M + \partial + 1$ and higher is unknown when the theorem is applied.

It is relatively simple to get around the problem when $j<i$ as the algorithm to solve (6.42) can be forced to continue until polynomials of degree δ are obtained. Thus, the coefficients to degree δ can be constructed using the CRT.

When $i<j$ though, the problem becomes more complicated. When this happens, for all previous primes p_k, $k<j$ the system *mod* p_k has been solved only up to a degree τ with $\tau < \delta$. Thus, the integer coefficients of the terms of the solution of the degrees $\tau + 1$ to δ cannot be constructed

using the information available. One method to find the solution when this happens involves discarding the solution so far constructed and starting the entire solution process over again with the prime p_j as the first prime in the sequence of primes. To do this shifting of the primes, the precalculated inverses, $S^{(k)}$, must be recalculated using (6.5). A second method to find the solution when a bad prime is encountered is to restart the solution process at the first prime, p_0, but ensuring that the solution modulo p_i, $i < j$, are guaranteed to be at least degree δ. This, again, involves discarding the solution thus far computed unless the solutions mod p_i,$i < j$, are retained in the state in which the algorithm terminated, for each prime p_i.

The power series form of the solution suffers when the degree of the solutions *mod* p_i vary for different p_i. Since the degree of the truncated power series outputted from the method over $\frac{\mathbf{Z}}{(p)}$ is dependent on the degree required to form the rational function approximates of sufficient degree, the degree of power series solution can vary. To get around this requires that the degree of the power series from each intermediate step (i.e., *mod* p_i) be the same. This can be done by determining the degree of the truncated power series with integer coefficients and ensuring the intermediate steps solve the power series *mod* p_i to this degree.

With the power series methods of Chapter 4, there exist other types of bad prime problems. One, that occurs when the matrix $A(z)$ *mod* p_i is singular, was discussed in Chapter 4. In this case, the power series solution to (6.1) (which is used in the Pade conversion method) does not exist. When this occurs, the prime, p_i, must be discarded and the next one in the sequence is used. In addition, the cost of finding $A(z)$ *mod* p_i singular which is higher than finding the solution of a non-singular system, is discarded with the prime. All that is required to determine if A(z) singular is that $A(z)$ *mod* p_i be singular for $\gamma + 1$ primes p_i.

With the rational function conversion of the power series, there exists another bad prime problem. It comes about because the conversion of the power series to rational function form returns the rational functions in reduced form. The problem occurs when factors are removed from

the rational function solution over $\frac{\mathbf{Z}}{(p)}$, for some prime p, that do not exist in the rational functions with integer coefficents.

To illustrate the problem, consider the adjoint solution $\left(y(z),d(z)\right)$, $d(z) \neq 0$, of (6.1) with integer coefficients. The reduced rational function form of the solution with integer coefficients is then

$$\frac{y'_i(z)}{d'_i(z)} \quad ,i = 1,2, \ldots ,N \tag{6.43}$$

where

$$y'_i(z) = \frac{y_i(z)}{GCD\left(y_i(z),d(z)\right)} \tag{6.44}$$

$$d'_i(z) = \frac{d_i(z)}{GCD\left(y_i(z),d(z)\right)}$$

and

$$GCD\left(y'_i(z),d'_i(z)\right) = u \ , \quad u \in \frac{\mathbf{Z}}{(p)}. \tag{6.45}$$

The problem occurs when

$$GCD\left(\left(y'_i(z) \bmod p\right), \left(d'_i(z) \bmod p\right)\right) = n(z) \tag{6.46}$$

where $deg\left((z)\right) > 0$ for some i. Thus, the conversion of the power series solution over $\frac{\mathbf{Z}}{(p)}$ will return the rational function with the factor $n(z)$ removed. But this factor did not exist over the integers and thus the degree of the rational function over $\frac{\mathbf{Z}}{(p)}$ is less than the degree of the corresponding rational function over the integers and its coefficients are not necessarily the residuals *mod p* of the coefficents of the corresponding terms of the rational function over the integers.

In addition, the factor $n(z)$ cannot be determined from the power series or its rational function. If the factor did not exist in all N of the rational functions of the solution, it can be determined by calculating a common denominator of the rational functions but if the factor was

removed from all N rational functions it cannot be determined. Thus, there is not any method of recovery when this type of problem occurs other than discarding the prime.

Thus, when a bad prime is encountered, it generally means that a sizeable amount of calculation has been done that cannot be used. In addition, as in the last case, the determination of whether a bad prime has been encountered or whether the solution is valid can be difficult.

BIBLIOGRAPHY

[1] Atkinson, K.E., *An Introduction to Numerical Analysis* , Wiley and Sons, N.Y., 1978.

[2] Bareiss, E.H., "Computational Solutions of Matrix Problems Over an Integral Domain", *J. Inst. Maths Applics.* 10 (1972), pp. 68-104.

[3] Brent, R., F.G. Gustavson and D.Y.Y. Yun, "Fast Solution of Toeplitz Systems of Equations and Computation of Pade Approximants", *J. of Algorithms* 1 (1980), pp. 259-295.

[4] Cabay, S., "Exact Solution of Linear Equations", Proc. Second Symp. on Symbolic and Algebraic Manipulation, ACM, N.Y., 1971, pp. 392-398.

[5] Cabay, S., "Solution of Linear Systems of Equations with Univariate Polynomial Coefficients", to appear.

[6] Cabay, S. and Kao, T.T.C., "The Diagonal Pade Table and the Triangular Decomposition of Hankel Matrices", to appear.

[7] Cabay, S. and Lam, T.P.L., "Congruence Techniques for the Exact Solution of Integer Systems of Linear Equations", *ACM Trans. Math. Software* 3, 4 (Dec. 1977), pp. 386-397.

[8] Choi, D.K., *Algebraic Computations of Scaled Pade Fractions*, Ph.D. Thesis, University of Alberta, in preparation.

[9] Forsythe, G. and Moler, C.B., *Computer Solution of Linear Algebraic Equations*, Prentice-Hall, 1967.

[10] Higginson, M., *Exact Methods for Systems of Polynomial Equations*, M.Sc. Thesis, University of Alberta, 1974.

[11] Lipson, J.D., *Elements of Algebra* and *Algebraic Computing*, Addison-Wesley, 1981.

[12] Knuth, D.E., *Seminumerical Algorithms*, Addison-Wesley, 1969.

[13] McClellan, M.,"The Exact Solution of Systems of Linear Equations with Polynomial Coefficients," *JACM* 20, 4 (October 1973), pp. 563-589.

[14] McClellan, M., "A Comparison of Algorithms for the Exact Solution of Linear Equations", *ACM Trans. Math. Software* 3, 2 (June 1977), pp. 147-158.

[15] Verheijen, A., *Evaluation of Diagonal Pade Computations*, M.Sc. Thesis, University of Alberta, in preparation.