MATHEMATICS OF COMPUTATION, VOLUME 27, NUMBER 124, OCTOBER 1973

# An Algorithm for the Exact Reduction of a Matrix to Frobenius Form Using Modular Arithmetic. II

By Jo Ann Howell*

**Abstract.** Part I contained a description of the single-modulus algorithm for reducing a matrix to Frobenius form, obtaining exact integral factors of the characteristic polynomial. Part II contains a description of the multiple-modulus algorithm. Since different moduli may yield different factorizations, an algorithm is given for determining which factorizations are not correct factorizations over the integers of the characteristic polynomial. Part II also contains a discussion of the selection of the moduli and numerical examples.

## C. The Multiple-Modulus Algorithm

**7. Introduction.** The algorithm described in Chapter B uses single-modulus residue arithmetic to reduce a matrix $A$ to Frobenius form (1.1). We recall that the size of the modulus $p$ depends on the bound $\beta$ (4.4). If $2\beta$ is too large to be representable in a computer as a single-precision integer, then $p$ will have to be stored as a multiple-precision integer, making computations modulo $p$ too difficult to be practical.

In order to avoid this problem, we select a set of prime moduli, $p_1, p_2, \cdots, p_s$, with

$$p = p_1 p_2 \cdots p_s, \tag{7.1}$$

because this enables us to obtain results modulo $p$ by doing most of the arithmetic modulo $p_i$, for $i = 1, 2, \cdots, s$. Choosing the moduli as primes also guarantees that** $(p_i, p_j) = 1$, for $i \neq j$. Furthermore, we choose the moduli so that

$$p \geqq 2\beta \geqq 2\cdot\max_{i,j} |b_j^{(i)}|, \tag{7.2}$$

where the $b_j^{(i)}$ are defined in (4.2).

We perform similarity transformations modulo $p_i$ on $|A|_{p_i}$, for $i = 1, 2, \cdots, s$, by using the single-modulus procedure described in Chapter B in order to obtain the residue representations (see Szabo and Tanaka [1967, p. 12]) for the factors of the characteristic polynomial modulo $p$ of $A$,

$$f_i(\lambda) \sim \{|f_i(\lambda)|_{p_1}, |f_i(\lambda)|_{p_2}, \cdots, |f_i(\lambda)|_{p_s}\}. \tag{7.3}$$

Received August 22, 1972.

*AMS (MOS) subject classifications* (1970). Primary 15A21, 68A10; Secondary 10A10, 12C05, 15A36.

*Key words and phrases.* Modular arithmetic, residue arithmetic, modulus, Frobenius form, Danilewski method, characteristic polynomial, similarity transformation, prime number, Chinese Remainder Theorem.

* Work on this paper was supported in part by NSF Grant GP-23655 and by U. S. Army Research Office (Durham) Grant DA-ARO(D)-31-124-72-G34 at the University of Texas at Austin.

** The restriction given here is required by the Chinese Remainder Theorem, mentioned below.

From these $s$-tuples we can determine $|f_i(\lambda)|_p$ using the Chinese Remainder Theorem or some variation of it such as the mixed-radix conversion procedure. (See Szabo and Tanaka [1967, pp. 27, 43], Lipson [1971], and Howell and Gregory [1970].) This means that since $p$ is chosen according to (7.2), we can determine $f_i(\lambda)$. Examples illustrating the algorithm are given in Chapter E.

Since different moduli may give us different factorizations, we must monitor the reductions, keeping a record of rows which are interchanged and pivots which vanish in order to use only the factorizations modulo $p_i$ which give us the correct factorization over the integers for det $(A - \lambda I)$. This monitoring scheme and the multiple-modulus reduction are described in the next sections.

**8. Block Structures in the Frobenius Form.** In the ideal situation all moduli used would yield the same block structure (blocks of the same order and arranged in the same pattern along the diagonal). This is not always the case, as Example (6.2) illustrates.

In order to guarantee that we can reconstruct the factors by means of the Chinese Remainder Theorem, the factors obtained using the modulus $p_i$ must be of the same degrees as those factors obtained using $p_k$ (for all $i$ and $k$). Moreover, the factors must appear in the same order along the diagonal. This implies that even if two or more factors of the same degree, $r_j$, are obtained using the modulus $p_i$, they must appear in the same order as their corresponding factors of degree $r_j$ obtained using the modulus $p_k$.

We now show that if we have obtained blocks of corresponding orders for two or more moduli, then these blocks can be combined using the Chinese Remainder Theorem to obtain the blocks we would have obtained had we done our calculations modulo $p_1 p_2 \cdots p_s = p$. That is to say, if the blocks obtained using the multiple-modulus algorithm are of corresponding sizes for different moduli, then they are in the proper order for obtaining the Frobenius form modulo $p$ using the Chinese Remainder Theorem.

We prove this by considering the transforming matrices $J_k^{(p_i)}$, where

$$\begin{aligned} |J_j^{(p_i)-1}(p_i) A_j^{(p_i)} J_j^{(p_i)}|_{p_i} &= |J_j^{(p_i)-1}(p_i) \cdots J_0^{(p_i)-1}(p_i) A_0^{(p_i)} J_0^{(p_i)} \cdots J_j^{(p_i)}|_{p_i} \\ &= |J^{(p_i)-1}(p_i) A_0^{(p_i)} J^{(p_i)}|_{p_i} = \left[\begin{array}{c|c} D_1^{(p_i)} & \ast \\ \hline O & H_1^{(p_i)} \end{array}\right], \end{aligned} \tag{8.1}$$

and $D_1^{(p_i)}$ is a $(j + 2) \times (j + 2)$ submatrix which is in Frobenius form except for the subdiagonal elements which are not yet reduced to unity. We must show that if

$$\begin{aligned} |J_j^{(p)-1}(p) A_j^{(p)} J_j^{(p)}|_p &= |J_j^{(p)-1}(p) \cdots J_0^{(p)-1}(p) A_0^{(p)} J_0^{(p)} \cdots J_j^{(p)}|_p \\ &= |J^{(p)-1}(p) A_0^{(p)} J^{(p)}|_p = \left[\begin{array}{c|c} D_1^{(p)} & \ast \\ \hline O & H_1^{(p)} \end{array}\right], \end{aligned} \tag{8.2}$$

where $D_1^{(p)}$ is a $(j + 2) \times (j + 2)$ submatrix in Frobenius form except for the nonunity subdiagonal elements, then

$$|J^{(p)}|_{p_i} = |J^{(p_i)}|_{p_i} \tag{8.3}$$

for all $i$, and hence

$$|D_1^{(p)}|_{p_i} = D_1^{(p_i)} \tag{8.4}$$

and

$$|H_1^{(p)}|_{p_i} = H_1^{(p_i)}. \tag{8.5}$$

The same arguments can then be applied to the submatrices $H_1^{(p_i)}$ and $H_1^{(p)}$ to show that

$$|D_2^{(p)}|_{p_i} = D_2^{(p_i)} \tag{8.6}$$

and

$$|H_2^{(p)}|_{p_i} = H_2^{(p_i)}. \tag{8.7}$$

Continuing in this manner we can show that

$$|D_k^{(p)}|_{p_i} = D_k^{(p_i)} \tag{8.8}$$

for all $k$. We now prove (8.4) and (8.5).

First we examine $J_k^{(p)}$ $(k = 1, \cdots, j)$. We shall assume for the moment that no row interchanges have taken place. We have

$$J_k^{(p)} = \left[\begin{array}{c|c|c} I_{k+1} & \begin{matrix} -\mu^{(p)}_{1,k+1} \\ \vdots \\ -\mu^{(p)}_{k+1,k+1} \end{matrix} & O \\ \hline 0 \ \cdots \ 0 & 1 & 0 \ \cdots \ 0 \\ \hline O & \begin{matrix} -\mu^{(p)}_{k+3,k+1} \\ \vdots \\ -\mu^{(p)}_{n,k+1} \end{matrix} & I_{n-k-2} \end{array}\right] \qquad (k = 1, \cdots, j) \tag{8.9}$$

where

$$-\mu^{(p)}_{l,k+1} = |a^{(k)(p)}_{l,k+1} \cdot a^{(k)(p)-1}_{k+2,k+1}(p)|_p. \tag{8.10}$$

Since

$$\begin{aligned} -\mu^{(p)}_{l,k+1} &= ||a^{(k)(p)}_{l,k+1} \cdot a^{(k)(p)-1}_{k+2,k+1}(p)|_p|_{p_i} \\ &= |a^{(k)(p_i)}_{l,k+1} \cdot a^{(k)(p_i)-1}_{k+2,k+1}(p_i)|_{p_i} = |-\mu^{(p_i)}_{l,k+1}|_{p_i}, \end{aligned} \tag{8.11}$$

then

$$|J_k^{(p)}|_{p_i} = |J_k^{(p_i)}|_{p_i} = J_k^{(p_i)}. \tag{8.12}$$

Thus,

$$\begin{aligned} |J^{(p)}|_{p_i} &= |J_0^{(p)} J_1^{(p)} \cdots J_j^{(p)}|_{p_i} \\ &= |J_0^{(p_i)} J_1^{(p_i)} \cdots J_j^{(p_i)}|_{p_i} = |J^{(p_i)}|_{p_i}, \end{aligned} \tag{8.13}$$

and hence

$$|J^{(p)-1}(p)|_{p_i} = |J^{(p_i)-1}(p_i)|_{p_i}. \tag{8.14}$$

Therefore,

$$\begin{aligned} |J^{(p)-1}(p)A_0^{(p)}J^{(p)}|_{p_i} &= |J^{(p)-1}(p)\,|A|_p\,J^{(p)}|_{p_i} \\ &= |J^{(p_i)-1}(p_i)\,|A|_{p_i}\,J^{(p_i)}|_{p_i} \\ &= |J^{(p_i)-1}(p_i)A_0^{(p_i)}J^{(p_i)}|_{p_i}, \end{aligned} \tag{8.15}$$

and thus,

$$|D_1^{(p)}|_{p_i} = D^{(p_i)} \tag{8.16}$$

and

$$|H_1^{(p)}|_{p_i} = H_1^{(p_i)}. \tag{8.17}$$

Applying the same arguments to $H_k^{(p)}$ and $H_k^{(p_i)}$ $(k = 2, \cdots, j-1)$ we thus prove (8.8). It follows immediately from this that

$$|F_k^{(p)}|_{p_i} = F_k^{(p_i)}. \tag{8.18}$$

We see from the above that applying the Chinese Remainder Theorem to the $F_k^{(p_i)}$ gives the $F_k^{(p)}$, the blocks we would have obtained had we done our arithmetic modulo $p$ instead of modulo $p_i$, $i = 1, \cdots, s$. It is important to note that this analysis is based on the assumption that partitioning occurred at the same point for all moduli.

If a zero pivot occurs somewhere between columns one and $j+1$ which can be removed by pivoting rows, then the same rows must be pivoted for all moduli. This necessitates a monitor on the rows being pivoted during the course of reduction. If it is impossible to pivot the same rows for all moduli, then the odd modulus (or moduli) must be discarded and another tried. This assures us that even when pivoting occurs, (8.8), (8.13), and (8.18) still hold.

From the above arguments and Theorem (6.4), we see that if $a_{j+2,j+1}^{(i)(p_k)} \neq 0$, then the same pivot must be nonzero in the rational arithmetic algorithm, provided previous pivots for the two algorithms vanished at the same point. Thus, if $a_{j+2,j+1}^{(i)(p_i)}$ vanishes for $p_k$ $(k \neq i)$, it must be vanishing because

$$|a_{j+2,j+1}^{(i)}(a_{j+1,j}^{(i-1)})^{b_1} \cdots (a_{21}^{(0)})^{b_r}|_{p_k} = 0 \tag{8.19}$$

or because $|a_{j+2,j+1}^{(i)}|_{p_k} = 0$ and *not* because $a_{j+2,j+1}^{(i)} = 0$. Thus, $p_k$ should be discarded.

This implies that if we compare the size of the initial (leading) blocks $(F_1^{(p_i)})$ obtained by reductions modulo $p_1, p_2, \cdots, p_t$, then only the moduli which have produced those blocks of maximum size should be retained and all others should be discarded. Thus, if the $s$ of the moduli produce an initial block of order $j+1$, then either $a_{j+2,j+1}^{(i)} = 0$, or $a_{j+2,j+1}^{(i)}$ is an integer and $|a_{j+2,j+1}^{(i)}|_p = 0$, or

$$|a_{j+2,j+1}^{(i)}(a_{j+1,j}^{(i-1)})^{b_1} \cdots (a_{21}^{(0)})^{b_r}|_p = 0, \tag{8.20}$$

where $p$ is the product of the $s$ moduli. By demanding that we have at least $K$ moduli which produce like factorizations ($K > 1$ being some input parameter dependent upon the size of the computer word and the size of the moduli used) we can make $p$ as large as we like. By choosing $p$ large we lessen the chance of having

$$a_{j+2,j+1}^{(i)}(a_{j+1,j}^{(i-1)})^{b_1} \cdots (a_{21}^{(0)})^{b_r} = C\cdot p, \tag{8.21}$$

where $C$ is an integer not equal to zero. Thus, a large $p$ will increase the probability that $a_{j+2,j+1}^{(i)(p)} = 0$ if and only if $a_{j+2,j+1}^{(i)} = 0$.

By comparing the sizes of blocks $2, \cdots, l$, in a similar manner we can eliminate "bad" choices of $p_i$ until we are left with a set of moduli which have all produced the same block structures with blocks of maximal size. If their pivoting patterns are all the same, if the number of "successful" moduli is greater than $K$, and if the product of the moduli is greater than $2\beta$, then we can apply the results of the last section and use the Chinese Remainder Theorem to get the factors modulo $p$, where $p$ is the product of the moduli.

We must emphasize that even though we have *at least* $K$ matching reductions, and this *increases* the probability that $a_{j+2,j+1}^{(i)(p)} = 0$ if and only if $a_{j+2,j+1}^{(i)} = 0$, we *cannot guarantee* that this is the case. For example, if $K = 3$ and $p_i$ are approximately $10^7$, then $p \doteq 10^{21}$. Then, for the method to fail to produce the correct factors, we must have $a_{j+2,j+1}^{(i)(p)} = 0$ and either

$$a_{j+2,j+1}^{(i)}(a_{j+1,j}^{(i-1)})^{b_1} \cdots (a_{21}^{(0)})^{b_r} = C\cdot p_1p_2p_3 \doteq C\cdot 10^{21}, \tag{8.22}$$

or

$$a_{j+2,j+1}^{(i)} = C\cdot p_1p_2p_3, \tag{8.23}$$

where $C$ is an integer. Although it is extremely unlikely for the left side of (8.22) or (8.23) to be an integer multiple of $p_1p_2p_3$, a product of primes, the possibility nevertheless still exists.

An example of the multiple-modulus algorithm is in Section 11.

The following is an algorithm for the reduction of a matrix to the form $F$ in (1.1).

*Algorithm* III. *Reduction of a Matrix A to the Form* (1.1) (*Multiple-Modulus Algorithm*).

*Input*: An $n \times n$ matrix $\bar{A}$, a set of stored moduli $(p_1, p_2, \cdots, p_q)$, min (minimum number of moduli to be used), ndigit (number of digits stored in each word for a multiple-precision integer).

*Output*: An $n \times n$ matrix $F$ in the form (1.1).

(1) Compute a bound $\beta$ for the product of the moduli. (We must have $p_1p_2 \cdots p_s \geqq 2\cdot\beta$.)

(2) Set $i \leftarrow 0$, product $\leftarrow 1$, errorcode $\leftarrow 0$, $\mathfrak{L}_t \leftarrow 0$ $(t = 1, \cdots, q)$.

(3) Set $i \leftarrow i + 1$, $s \leftarrow i$, $p \leftarrow p_i$, product $\leftarrow$ product$\cdot p_i$, $A \leftarrow |\bar{A}|_p$.

(4) Apply Algorithm I and Algorithm II to $A$.

(5) Set $\mathfrak{L}_i \leftarrow l'$, $k \leftarrow 0$.

(6) For $t = 1, \cdots, l'$, set $m \leftarrow k + 1$, $k \leftarrow k + r_t'^{(p)}$, $\mathfrak{N}_{j,i} \leftarrow a_{j,k}$ $(j = m, \cdots, k)$.

(7) [Check to see if enough moduli have been used.] If product $< 2\cdot\beta$, go to (10).

(8) If $i <$ min, go to (10).

(9) If errorcode $= 0$, set match $\leftarrow i$. Go to (12).

(10) [Compare $i$ with total number of moduli stored.] If $i < q$, go to (3).

(11) [An insufficient number of moduli is stored.] *Exit* (failure).

(12) For $t = 1, \cdots, q$, set modulus$_t \leftarrow 0$ and temp$_t \leftarrow t$; set $j \leftarrow s$, errorcode $\leftarrow 0$, $K \leftarrow \max_t \mathfrak{L}_t$ $(t = 1, \cdots, s)$.

(13) Set $k \leftarrow 0$.

(14) [Compute maximum block size.] Set $k \leftarrow k + 1$, maxblock $\leftarrow \max_{1 \leqq t \leqq j} r_k'^{(p_{\mathrm{temp}\,t})}$, counter $\leftarrow 0$.

(15) Set $t \leftarrow 0$.

(16) Set $t \leftarrow t + 1$.

(17) [Compare block sizes.] If $r_k'^{(p_{\mathrm{temp}\,t})} \neq$ maxblock, go to (19).

(18) Set counter $\leftarrow$ counter $+ 1$, $p_{\mathrm{temp}_{\mathrm{counter}}} \leftarrow p_{\mathrm{temp}_t}$.

(19) If $t < j$, go to (16).

(20) Set $j \leftarrow$ counter.

(21) If $j <$ match, set errorcode $\leftarrow 1$, and go to (10).

(22) Set $K \leftarrow \max_t \mathfrak{L}_{\mathrm{temp}_t}$ $(t = 1, \cdots, j)$. If $k < K$, go to (14).

(23) Set $k \leftarrow 1$.

(24) Set $k \leftarrow k + 1$.

(25) Set $t \leftarrow 0$.

(26) Set $t \leftarrow t + 1$.

(27) [Compare pivoting patterns.] If, for all $v$ $(1 \leqq v \leqq n - 2)$,

$$\mathrm{pivot}_{v,1}^{(p_{\mathrm{temp}\,k})} = \mathrm{pivot}_{v,1}^{(p_{\mathrm{temp}\,t})} \quad \text{and} \quad \mathrm{pivot}_{v,2}^{(p_{\mathrm{temp}\,k})} = \mathrm{pivot}_{v,2}^{(p_{\mathrm{temp}\,t})},$$

go to (28), otherwise go to (35).

(28) Set modulus$_1 \leftarrow p_{\mathrm{temp}_t}$, modulus$_2 \leftarrow p_{\mathrm{temp}_k}$, $ip \leftarrow 2$.

(29) Set $ii \leftarrow k$.

(30) Set $ii \leftarrow ii + 1$.

(31) If, for all $v$ $(1 \leqq v \leqq n - 2)$,

$$\mathrm{pivot}_{v,1}^{(p_{\mathrm{temp}\,k})} = \mathrm{pivot}_{v,1}^{(p_{\mathrm{temp}\,ii})} \quad \text{and} \quad \mathrm{pivot}_{v,2}^{(p_{\mathrm{temp}\,k})} = \mathrm{pivot}_{v,2}^{(p_{\mathrm{temp}\,ii})},$$

go to (32), otherwise go to (33).

(32) Set $ip \leftarrow ip + 1$, modulus$_{ip} \leftarrow p_{\mathrm{temp}_{ii}}$.

(33) If $ii < j$, go to (30).

(34) If $ip \geqq$ match, go to (38).

(35) If $t < k - 1$, go to (26).

(36) If $k < j$, go to (24).

(37) Set errorcode $\leftarrow 1$, and go to (10).

(38) Set errorcode $\leftarrow 0$, $l \leftarrow \mathfrak{L}_K$, and $r_k \leftarrow r_k^{(\mathrm{modulus}_{ip})}$ $(k = 1, \cdots, l)$.

(39) For $t = 1, \cdots, ip$, set $\mathfrak{N}_{k,t} \leftarrow \mathfrak{N}_{k,\mathrm{modulus}_t}$ $(k = 1, \cdots, n)$.

(40) For $k = 1, \cdots, n$, combine the residue digits $\mathfrak{N}_{k,1}, \cdots, \mathfrak{N}_{k,ip}$ using the Chinese Remainder Theorem Algorithm.

(41) Store the multiple-precision combined results in $\mathfrak{N}_{k,1}$ through $\mathfrak{N}_{k,ip}$, ndigit digits per word, with the most significant digits in $\mathfrak{N}_{k,ip}$.

(42) Set $F \leftarrow 0$, counter $2 \leftarrow 0$.

(43) For $i = 1, \cdots, l$, set counter $1 \leftarrow$ counter $2 + 1$, counter $2 \leftarrow$ counter $1 + r_i - 1$, $f_{k,\mathrm{counter}\,2} \leftarrow$ multiple-precision integer $\{\mathfrak{N}_{k,ip}, \cdots, \mathfrak{N}_{k,1}\}$ $(k =$ counter $1, \cdots,$ counter $2)$.

## D. Selection of the Moduli

**9. Introducton.** In practice, the moduli are chosen as large prime numbers. The choice of the moduli as primes is necessary in order to guarantee the existence of inverses for integers and matrices. We recall that when $p_k$ is a prime, the integers modulo $p_k$ form a field. Furthermore, by choosing the $p_k$ as primes, we guarantee that

$$(9.1) \qquad (p_i, p_j) = 1,$$

for $i \neq j$, as required by the Chinese Remainder Theorem.

Ideally, the primes should be chosen as large as possible and so that $p_ip_j$ does not overflow a fixed-point computer word, for all $i$ and $j$. This guarantees that an intermediate result will not overflow before it can be reduced modulo $p_k$ for all $k$. In addition to this, time can be saved by using a small number of large primes rather than a large number of small primes. Furthermore, by choosing the moduli as large prime numbers we greatly increase the probability that the disappearance of a pivot during the reduction modulo $p_k$ has occurred because the same pivot would disappear during the rational arithmetic algorithm. We must further have $p_k > n$ for all $k$ in order to be able to reconstruct the characteristic polynomial from its residue representation.

**10. Calculation of a Bound for $p$.** Let

$$(10.1) \qquad \begin{aligned} \det(A - \lambda I) &= (-1)^n(\lambda^n - x_1\lambda^{n-1} - \cdots - x_{n-1}\lambda - x_n) \\ &= f_1(\lambda) \cdots f_l(\lambda), \end{aligned}$$

where

$$(10.2) \qquad f_i(\lambda) = (-1)^{r_i}(\lambda^{r_i} - b_1^{(i)}\lambda^{r_i-1} - \cdots - b_{r_i-1}^{(i)}\lambda - b_{r_i}^{(i)}).$$

We wish to compute a lower bound for $p$ so that if we have a prestored set of primes we select and use as many moduli as necessary to guarantee a solution (i.e., to guarantee that $|\det(A - \lambda I)|_p = \det(A - \lambda I)$).

If it is known that the matrix $A$ has a characteristic polynomial which is irreducible over the integers, then $l = 1$. In this case we obtain a bound for $\max_j |x_j|$ by utilizing the fact that $x_j$ is plus or minus the sum of the principal minors of order $j$. From Hadamard's inequality we have

$$(10.3) \qquad |x_n| \leqq \left(\sum_{j=1}^{n} |a_{ij}|^2 \cdots \sum_{j=1}^{n} |a_{nj}|^2\right)^{1/2} = k.$$

Thus, any principal minor of order less than $n$ is also bounded by $k$.

Since the number of principal minors of order $j$ is equal to $\binom{n}{j}$, we have

$$(10.4) \qquad |x_j| \leqq \binom{n}{j} \cdot k.$$

Hence

$$(10.5) \qquad \max_j |x_j| \leqq \max_j \binom{n}{j} \cdot k = \binom{n}{[n/2]} \cdot k$$

and we should choose $p$ so that

$$(10.6) \qquad p \geqq 2 \cdot \binom{n}{[n/2]} \cdot k.$$

Therefore, in (4.4) we have

$$\beta = \binom{n}{[n/2]} \cdot k. \tag{10.7}$$

If it is not known that $l = 1$, then the bound (10.7) *may* not be sufficient. It is possible for some of the coefficients of the $f_i(\lambda)$ to be greater in absolute value than $\max_j |x_j|$ .

A method for bounding the coefficients $|b_j^{(i)}|$ is based on a suggestion by Collins (see Knuth [1969, p. 392]). We note that

$$\begin{aligned} f_i(\lambda) &= (-1)^{r_i}(\lambda^{r_i} - b_1^{(i)}\lambda^{r_i-1} - \cdots - b_{r_i-1}^{(i)}\lambda - b_{r_i}^{(i)}) \\ &= (-1)^{r_i}(\lambda - \gamma_1^{(i)}) \cdots (\lambda - \gamma_{r_i}^{(i)}), \end{aligned} \tag{10.8}$$

where the $\gamma_j^{(i)}$ are eigenvalues of the matrix $A$. Thus, if we have a bound $\alpha$ for the eigenvalues of $A$, then

$$|b_j^{(i)}| \leqq \binom{r_i}{j}\alpha^j. \tag{10.9}$$

Therefore

$$\max_{i,j}|b_j^{(i)}| \leqq \max_{i,j}\binom{r_i}{j}\alpha^j = \binom{n}{[n/2]}\alpha^n, \tag{10.10}$$

and we should choose $p$ so that

$$p \geqq 2\binom{n}{[n/2]}\alpha^n = 2\beta. \tag{10.11}$$

Bounds for $\alpha$ such as $||A||_\infty$, $||A||_1$, or the bound given by Ostrowski [1952] are suitable. In practice, the bounds computed using either (10.6) or (10.11) are larger than necessary to guarantee that $|b_j^{(i)}|_p = b_j^{(i)}$.

In the next section we give examples illustrating the computation of a bound $\beta$ and examples of the multiple-modulus algorithm.

## E. Examples and Numerical Results

**11. Two Examples.** Let

$$A = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 4 & 5 \end{bmatrix}.$$

We let the stored set of moduli be $\{7, 11, 13, 17, 19\}$. We shall assume that regardless of the computed bound $\beta$ we require *at least* two moduli to give the same block structures. If we compute a bound $\beta$ by (10.7) we have

$$\beta = \binom{3}{1}\cdot(4\cdot 1\cdot 41)^{1/2} \doteq 3(12.8) \doteq 38.4.$$

The bound from (10.11) with $\alpha = ||A||_1$ is

$$\beta = \binom{3}{1} \cdot 5^3 = 375,$$

and with $\alpha = ||A||_\infty$ it is

$$\beta = \binom{3}{1} \cdot 9^3 = 2187.$$

If $\alpha$ is computed from Ostrowski [1952], we have

$$\alpha = R - (1 - \sigma)K,$$

where

$$R = ||A||_\infty = 9, \qquad r = \min_i \sum_{j=1}^{n} |a_{ij}| = 1,$$

$$K = \min_{i,j} |a_{ij}| = 1 \quad \text{and} \quad \sigma = ((r - K)/(R - K))^{1/2} = 0.$$

Then

$$\alpha = 9 - 1 = 8 \quad \text{and} \quad \beta = \binom{3}{1} \cdot 8^3 = 1{,}536.$$

Clearly, all of these bounds are larger than necessary to guarantee that $|f_i(\lambda)|_p = f_i(\lambda)$. In practice, the bound given by (10.7) is usually adequate, even if it is not known that $l = 1$.

To illustrate the multiple-modulus algorithm for the matrix $A$, we choose $p_1 = 7$ and $p_2 = 11$. (Note that $77 \geqq 2\beta$, where $\beta$ is computed by (10.7).) Transforming the matrix modulo 7, we obtain

$$F^{(7)} = \left[\begin{array}{c|cc} 2 & 0 & 0 \\ \hline 0 & 0 & 2 \\ 0 & 1 & -1 \end{array}\right].$$

Thus, the residue modulo 7 of the factors of the characteristic polynomial are

$$|f_1(\lambda)|_7 = \lambda - 2 \quad \text{and} \quad |f_2(\lambda)|_7 = \lambda^2 + \lambda - 2.$$

Now rows were interchanged to produce a nonzero pivot. Transforming the matrix modulo 11, we have

$$F^{(11)} = \left[\begin{array}{c|cc} 2 & 0 & 0 \\ \hline 0 & 0 & -5 \\ 0 & 1 & -5 \end{array}\right].$$

Thus,

$$|f_1(\lambda)|_{11} = \lambda - 2 \quad \text{and} \quad |f_2(\lambda)|_{11} = \lambda^2 + 5\lambda + 5.$$

Again, no rows were interchanged to produce a nonzero pivot.

The residue representations for $f_1(\lambda)$ and $f_2(\lambda)$ for moduli $p_1 = 7$ and $p_2 = 11$ are thus

$$f_1(\lambda) \sim \{\lambda - 2, \lambda - 2\} \quad \text{and} \quad f_2(\lambda) \sim \{\lambda^2 + \lambda - 2, \lambda^2 + 5\lambda + 5\}.$$

Since the two moduli used yield the same block structures, and since the same pivoting strategies were used in both cases, we can apply the Chinese Remainder Theorem to the coefficients of the polynomials $|f_i(\lambda)|_{p_i}$, and obtain results modulo $p_1p_2 = 77$:

$$|f_1(\lambda)|_{77} = \lambda - 2 \quad\text{and}\quad |f_2(\lambda)|_{77} = \lambda^2 - 6\lambda + 5.$$

Hence,

$$f_1(\lambda) = \lambda - 2 \quad\text{and}\quad f_2(\lambda) = \lambda^2 - 6\lambda + 5.$$

In the next example we let $A$ be the matrix, used in (6.1) and (6.2),

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 7 & 1 & 0 & 1 \\ 5 & 0 & 0 & 0 \end{bmatrix},$$

and the stored set of moduli be $\{5, 7, 13, 17, 19\}$. Again we require that *at least* two moduli give the same block structures. Computing $\beta$ by (10.7) we obtain

$$\beta = \binom{4}{2}(1\cdot 51\cdot 25)^{1/2} \doteq 6(35.7) = 214.2.$$

Thus, we should have

$$p \geqq 428.4.$$

We saw in (6.3) that transforming $A$ modulo 5 leads to an interchange of rows 2 and 3, and we obtain

$$F^{(5)} = \left[\begin{array}{cc|c|c} 0 & 2 & 0 & 0 \\ 1 & 0 & 3 & 3 \\ \hline 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array}\right].$$

Thus, we have three factors

$$|\det(A - \lambda I)|_5 = |(\lambda^2 - 2)\cdot\lambda\cdot\lambda|_5.$$

For $p_2 = 7$, we interchange rows 2 and 4 and obtain

$$F^{(7)} = \left[\begin{array}{ccc|c} 0 & 0 & 3 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

and the factorization

$$|\det(A - \lambda I)|_7 = |(\lambda^3 - 3)\cdot\lambda|_7.$$

For $p_3 = 13$, we interchange rows 2 and 3 in step one, and later rows 3 and 4. We obtained, in Example (6.1),

$$F^{(13)} = \left[\begin{array}{ccc|c} 0 & 0 & 5 & -4 \\ 1 & 0 & -6 & 0 \\ 0 & 1 & 0 & -5 \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

and hence the factorization

$$|\det(A - \lambda I)|_{13} = |(\lambda^3 + 6\lambda - 5)\cdot\lambda|_{13}.$$

At this point we note that the product of moduli used so far exceeds the bound $2\beta\,(p_1p_2p_3 = 455)$. If all block structures are the same at this point, and if all pivoting strategies are the same, we can apply the Chinese Remainder Theorem to the factors obtained. However, this is not the case in this example. We can immediately discard $p_1$ since it produced an initial block which is smaller than the one obtained using $p_2$ and $p_3$. The remaining two moduli yield identical block structures, but different pivoting strategies. It is not apparent at this point which is the correct one. Hence we must try other moduli. Since the bound $2\beta$ indicates that at least three moduli will have to yield identical reductions (blocks of corresponding orders and the same pivoting strategies) we will have to try at least two more primes.

For $p_4 = 17$, we obtain

$$F^{(17)} = \left[\begin{array}{ccc|c} 0 & 0 & 5 & 2 \\ 1 & 0 & 7 & 0 \\ 0 & 1 & 0 & 7 \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

and the factorization

$$|\det(A - \lambda I)|_{17} = |(\lambda^3 - 7\lambda - 5)\cdot\lambda|_{17}.$$

The rows interchanged are 2 and 3 and rows 3 and 4.

For $p_5 = 19$, we obtain

$$F^{(19)} = \left[\begin{array}{ccc|c} 0 & 0 & 5 & -9 \\ 1 & 0 & 7 & 0 \\ 0 & 1 & 0 & 4 \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

and the factorization

$$|\det(A - \lambda I)|_{19} = |(\lambda^3 - 7\lambda - 5)\cdot\lambda|_{19}.$$

The rows interchanged are rows 2 and 3 and rows 3 and 4.

We compare the results obtained using $p_2$, $p_3$, $p_4$, and $p_5$. The block structures are all the same. For $p_3$, $p_4$, and $p_5$ the pivoting strategies are the same. Since $p_3p_4p_5 \geqq 2\beta$ we can use the Chinese Remainder Theorem to obtain the coefficients of the factors modulo $p_3p_4p_5 = 4199$. We thus obtain

$$|\det(A - \lambda I)|_{4199} = |(\lambda^3 - 7\lambda - 5)\cdot\lambda|_{4199},$$

and hence

$$\det(A - \lambda I) = (\lambda^3 - 7\lambda - 5)\cdot\lambda.$$

In a computer program it is more efficient to use the largest stored primes first, since this may decrease the number of primes which must be used to guarantee that $p \geqq 2\beta$.

**13. Results from a Computer Program.** A program for reducing a matrix to Frobenius form and obtaining a factorization of its characteristic polynomial by the method described in this paper was written in FORTRAN for the CDC 6600 at the University of Texas at Austin. The set of stored primes used are as follows (Lehmer [1914]):

10,000,019

10,000,079

10,000,103

10,000,121

10,000,139

10,000,141

10,000,169

10,000,189

10,000,223

10,000,229.

The bound $\beta$ was computed using (10.7). In each of the following examples we exhibit a matrix and the factorization of its characteristic polynomial obtained using the program. We required at least three like reductions, regardless of the size of $\beta$.

*Example* 1. (Slotnick [1963, p. 4-43])

$$A = \begin{bmatrix} 3 & -1 & -4 & 2 \\ 2 & 3 & -2 & -4 \\ 2 & -1 & -3 & 2 \\ 1 & 2 & -1 & -3 \end{bmatrix}.$$

Eigenvalues:

$$\lambda_1 = 1, \qquad \lambda_3 = 1,$$
$$\lambda_2 = -1, \qquad \lambda_4 = -1.$$

$\beta$ computed by program: $3.10 \times 10^3$.
Number of moduli used: 3.
Factorization of det $(A - \lambda I)$ from program:

$$\det(A - \lambda I) = (\lambda^3 - \lambda^2 - \lambda + 1)(\lambda + 1).$$

*Example* 2. (Eberlein [1962], Gregory and Karney [1969, p. 90])

$$A = \begin{bmatrix} 15 & 11 & 6 & -9 & -15 \\ 1 & 3 & 9 & -3 & -8 \\ 7 & 6 & 6 & -3 & -11 \\ 7 & 7 & 5 & -3 & -11 \\ 17 & 12 & 5 & -10 & -16 \end{bmatrix}.$$

Eigenvalues:

$$\begin{aligned} \lambda_1 &= 1.5 + (12.75)^{1/2} i, \\ \lambda_2 &= 1.5 + (12.75)^{1/2} i, \\ \lambda_3 &= 1.5 - (12.75)^{1/2} i, \\ \lambda_4 &= 1.5 - (12.75)^{1/2} i, \\ \lambda_5 &= -1. \end{aligned}$$

$\beta$ computed by program: $2.41 \times 10^7$.
Number of moduli used: 3.
Factorization*** of det $(A - \lambda I)$ from program:

$$\det(A - \lambda I) = (-1)^5(\lambda^5 - 5\lambda^4 + 33\lambda^3 - 51\lambda^2 + 135\lambda + 225).$$

*Example* 3. (Gregory and Karney [1969, p. 7])

$$A = \left[\begin{array}{c|c} B & C \\ \hline C & B \end{array}\right],$$

where

$$B = \left[\begin{array}{ccc|cc} -364{,}270 & 0 & 0 & & \\ 1 & -364{,}270 & 0 & & O \\ 0 & 1 & -364{,}270 & & \\ \hline & & & -918{,}326 & 0 \\ & O & & 1 & -918{,}326 \end{array}\right]$$

and

*** We note that this is a case in which the characteristic polynomial is factorable over the integers, $(-1)^5(\lambda + 1)(\lambda^4 - 6\lambda^3 + 39\lambda^2 - 90\lambda + 225)$, but the program finds only one factor, that factor being the characteristic polynomial itself.

$$C = \left[\begin{array}{ccc|cc} -694{,}488 & 0 & 0 & & \\ 0 & -694{,}488 & 0 & \multicolumn{2}{c}{O} \\ 0 & 0 & -694{,}488 & & \\ \hline & & & 965{,}197 & 0 \\ & O & & 0 & 965{,}197 \end{array}\right].$$

Eigenvalues:

$$\begin{array}{ll} \lambda_1 = 330{,}218, & \lambda_6 = -1{,}058{,}758, \\ \lambda_2 = 330{,}218, & \lambda_7 = -1{,}058{,}758, \\ \lambda_3 = 330{,}218, & \lambda_8 = -1{,}058{,}758, \\ \lambda_4 = 46{,}871, & \lambda_9 = -1{,}883{,}523, \\ \lambda_5 = 46{,}871, & \lambda_{10} = -1{,}883{,}523. \end{array}$$

$\beta$ computed by program: $1.79 \times 10^{62}$.
Number of moduli used: 9.
Factorization of det $(A - \lambda I)$ from program:

$$\begin{aligned} &(\lambda^2 + 1{,}836{,}652\lambda - 88{,}282{,}606{,}533)\cdot(\lambda^2 + 1{,}836{,}652\lambda - 88{,}282{,}606{,}533) \\ &\quad\cdot(\lambda^6 + 2{,}185{,}620\lambda^5 + 543{,}448{,}747{,}068\lambda^4 - 1{,}141{,}589{,}515{,}081{,}478{,}560\lambda^3 \\ &\qquad - 1{,}901{,}066{,}815{,}376{,}621{,}816{,}592\lambda^2 \\ &\qquad + 267{,}158{,}841{,}389{,}405{,}409{,}701{,}792{,}512{,}320\lambda \\ &\qquad\qquad - 42{,}735{,}849{,}656{,}157{,}591{,}523{,}087{,}007{,}405{,}518{,}784). \end{aligned}$$

## F. Concluding Remarks

Since different moduli may yield different Frobenius forms and, hence, different factorizations for the characteristic polynomial it is recommended that the multiple-modulus algorithm and *not* the single-modulus algorithm be used in designing a computer program. The multiple-modulus algorithm is also recommended when $2\beta$ is larger than $\alpha^{1/2}$, where $\alpha$ is the largest computer-representable integer.

Care must be taken in using the multiple-modulus algorithm, however. A check must be made on reduction modulo $p_i$ for all $i$ to insure that their pivoting patterns are identical. The moduli yielding reductions with identical block structures and pivoting patterns are then checked to see if their product is greater than some preset constant, $K$. If so, then the Chinese Remainder Theorem is applied to the residue representations for the coefficients of the factors.

The number $K$ should be some number greater than 1 such that a product of $K$ moduli yields some "large" number. As $K$ becomes larger, the probability becomes greater that the vanishing of a pivot for all moduli means that the same pivot would have vanished had we used rational arithmetic. (See Theorem (6.4).) It cannot be overemphasized that the method *can fail* if either $K$ is too small or if not enough moduli are stored to give $p > 2\beta$.

The bound $\beta$ can be computed by several methods, most of which yield bounds which are larger than necessary. By choosing the moduli as large as possible we can reduce the amount of work to be performed even though the bound is too large.

We emphasize that the block structure obtained for a given matrix is not unique. The form obtained depends upon the order in which the elements below the first subdiagonal are annihilated. Changing the order in which the elements are annihilated may change either the order of the blocks on the diagonal or the size of the blocks. Clearly, the form obtained is *not* a canonical form, as the following example illustrates.

*Example.* Let $p_1 = 13$ (we are assuming it is known that $13 \geqq 2 \cdot \max_{i,j} |b_j^{(i)}|$) and

$$A = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 4 & 5 \end{bmatrix} \cdot$$

The modified Danilewski algorithm transforms $A$ into the form

$$F = \left[\begin{array}{c|cc} 2 & 0 & 0 \\ \hline 0 & 0 & -5 \\ 0 & 1 & 6 \end{array}\right] \cdot$$

Hence

$$\det(A - \lambda I) = (\lambda - 2)(\lambda^2 - 6\lambda + 5).$$

If we select

$$A' = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 5 \end{bmatrix}$$

(a matrix similar to $A$), we obtain

$$F' = \left[\begin{array}{c|c|c} 2 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 5 \end{array}\right] \cdot$$

Hence

$$\det(A - \lambda I) = (\lambda - 2)(\lambda - 1)(\lambda - 5).$$

Thus, $A$ and $A'$ yield different factorizations for the same characteristic polynomial.

University of California
Los Alamos Scientific Laboratory
Los Alamos, New Mexico 87544

P. J. Eberlein [1962], "A Jacobi-like method for the automatic computation of eigenvalues and eigenvectors of an arbitrary matrix," *J. Soc. Indust. Appl. Math.*, v. 10, 1962, pp. 74–88. MR **25** #2699.

R. T. Gregory & D. Karney [1969], *A Collection of Matrices for Testing Computational Algorithms,* Wiley, New York. MR **40** #6752.

J. A. Howell & R. T. Gregory [1970], "Solving linear equations using residue arithmetic-algorithm II," *Nordisk Tidskr. Informationsbehandling,* v. 10, pp. 23–37. MR **41** #6389.

D. E. Knuth [1969], *The Art of Computer Programming.* Vol. II: *Seminumerical Algorithms,* Addison-Wesley, Reading, Mass. MR **44** #3531.

D. N. Lehmer [1914], *List of Prime Numbers from* 1 *to* 10,006,721, Carnegie Institute of Washington (165), Washington, D. C.

J. D. Lipson [1971], "Chinese remainder and interpolation algorithms," *Proceedings of the Second Symposium on Symbolic and Algebraic Manipulation*, SIGSAM-ACM.

A. Ostrowski [1952], "Bounds for the greatest latent root of a positive matrix," *J. London Math. Soc.*, v. 27, pp. 253–256. MR **14,** 126.

D. L. Slotnick [1963], *Modular Arithmetic Computing Techniques*, Westinghouse Electric Corporation, Technical Report ASD-TDR-63-280, Baltimore; Clearinghouse for Federal Scientific and Technical Information, Report No. AD410534, Springfield, Virginia 22151.

N. S. Szabó & R. I. Tanaka [1967], *Residue Arithmetic and Its Applications to Computer Technology,* McGraw-Hill, New York.