On Computing the Exact Determinant of Matrices with Polynomial Entries

E. HOROWITZ

University of Southern California, Los Angeles, California

AND

S. SAHNI

University of Minnesota, Minneapolis, Minnesota

ABSTRACT. The problem of computing the determinant of a matrix of polynomials is considered. Four algorithms are compared: expansion by minors, Gaussian elimination over the integers, a method based on evaluation and interpolation, and a procedure which computes the characteristic polynomial of the matrix. Each method is analyzed with respect to its computing time and storage requirements using several models for polynomial growth. First, the asymptotic time and storage is developed for each method within each model. In addition to these asymptotic results, the analysis is done exactly for certain especially small, yet practical and important cases. Then the results of empirical studies are given which support conclusions about which of the methods will work best within an actual computing environment.

CR CATEGORIES: 5.14, 5.25

KEY WORDS AND PHRASES: determinants, matrix of polynomials, Gaussian elimination, expansion by minors, characteristic polynomial

1. *Introduction*

Though much work has been done on the problem of computing solutions of linear systems with rational number entries, far less has been done on matrices with polynomial entries. Yet such problems arise in many areas of scientific study. Perhaps the first such application is concerned with the computation of Jacobians, as for example in [11, pp. 113, 420]. Other applications are connected with the computation of generating functions for flow graphs; see [13]. In another area such matrices arose while solving a singular perturbation problem for a linear ordinary differential equation with an interior turning point; see [6]. Our own interest in this problem came from a desire to create a computer system which solves mathematical problems symbolically. In such computer systems, the solution of linear equations with symbolic polynomial entries arises repeatedly. As an aid to the mathematician, the object is to produce successive solutions for matrices of varying size. The hope is that the collection of these results will yield insights as to the makeup of the

This research was supported by the National Science Foundation under Grant GJ-33169 and GJ-44207.

Authors' addresses: E. Horowitz, Computer Science Program, University of Southern California, Los Angeles, CA 90007; S. Sahni, Department of Computer, Information, and Control Sciences, University of Minnesota, Minneapolis, MN 55455.

general formula. It may, in fact, be possible to produce the desired result solely by computer, but as we shall see later present speeds imply that only small problems can be solved. Thus the major advantage of such software is that it produces the exact solution. For notoriously ill-conditioned matrices such as the generalized Hilbert and Vandermonde, exact symbolic solution of linear systems provides a useful adjunct to conventional numerical techniques. In this paper we will specifically consider the problem of exact determinant calculation for matrices of polynomials.

Considerable work has already been done on the exact solution of linear systems with numerical entries. Bareiss [2] covers the background of early work by Borosh and Fraenkel and also the work by Takahasi and Ishibashi. More recent work by Howell and Gregory [10], Cabay [4], and McClellan [14] have tended to recommend the use of congruence (or modular) arithmetic for obtaining exact solutions instead of the more commonly used Gaussian elimination over an integral domain; see [1]. Bareiss [2] has compared a two-step elimination algorithm with a conventional multiprecision arithmetic routine versus a congruence technique. He concludes that the former is more efficient than the latter.

Only Bareiss [2] and McClellan [14] have attempted to extend their results to polynomial entries. Bareiss derives degree bounds for the resulting polynomials whereas McClellan actually derives cost functions. Both conclude that the congruence technique is preferable, though neither has presented an operation count for competing methods. In that sense this paper is a continuation of this work, since we give both asymptotic formulas and experimental data on several determinant calculating methods.

The conclusion that congruence methods, or what we shall refer to as evaluation-interpolation methods, are best for polynomial matrices is predicated on the assumption that all polynomials which arise in the course of the computations are completely dense (i.e. all possible terms occur). This assumption allows for the full exploitation of the evaluation-interpolation method, but it is misleading in the following sense: many problems in actual practice are not dense, e.g. the Vandermonde matrix, where each entry is a function of n variables but has only one term. Later on we will see why the evaluation-interpolation methods work poorly in such circumstances. A second difficulty is the question of the asymptotic analysis. Current computers cannot solve even a 6×6 system with dense polynomials in more than one variable. Thus asymptotic formulas may be misleading when we are concerned with only very small values of the parameters. A further factor is that different algorithms will behave radically different on different classes of polynomials. Therefore, in this paper we have attempted to model several classes of polynomials which do arise in practice. Then within each model we have analyzed the performance factors of several important algorithms, carefully preserving the constants so that we may estimate behavior on small problems. Finally, a summary of extensive empirical studies is given in order to assess the global performance of the methods within the confines of the practical world of computation.

2. *The Algorithms*

In this section we describe and analyze four algorithms for computing the determinant of a matrix. All these methods have been known before, but their use on matrices with polynomials of different types has been studied only partially. Although on the surface these methods look disparate, they are in fact all related. Their relationships have been brought out by Bareiss and we refer any interested reader to [2].

For all four algorithms, the input is an $n \times n$ matrix M whose elements are polynomials in the variables $x_1 \cdots , x_r$. The output is an r-variable polynomial $R(x_1, \cdots , x_r)$ such that $R = \det(M)$. The coefficients may be fixed or floating-point, single-precision numbers, or elements from the finite field, $\mathrm{GF}(p)$, where p is a single-precision odd prime. For the case of fixed or floating-point coefficients there has been no attempt to do an error analysis since we are primarily interested in exact calculations. However, the asymptotic formulas obtained in Section 3 will apply to all polynomials whose coefficients are such that the time to perform any arithmetic operation is bounded. No attempt has been made to judge the numerical accuracy of these methods.

Recent work studying exact algorithms for polynomial computations has shown that where multiple precision integer coefficients can occur, the best strategy is to first reduce the input modulo a set of primes p_i and perform the desired operations over GF(p_i); see, for example, the articles by Brown (pp. 478–504), Collins (pp. 515–532), and Heindel (pp. 533–548) in *J. ACM*, Vol. 8, No. 4, October 1971. Then a reconstruction procedure is applied which produces the correct integer results. We will assume that such a "modulo" process has occurred and that all of the polynomial's coefficients are and remain single-precision numbers.

While analyzing the computing times of these algorithms, only multiplications will be counted, and we shall assume that initially all the elements of M have the same number of terms. We shall also assume that polynomial multiplication is carried out using the classical algorithm (i.e. the time required to multiply a p term by a q term polynomial is $p \cdot q$). In practice there is also the time needed to merge partial sums as we compute the product. A clever multiplication procedure will only require an extra log min $\{p, q\}$ factor in our equations, which we will uniformly ignore as we do the analysis. The empirical results of Section 5 will help to substantiate or perhaps alter any claims initially made on the basis of the equations obtained simply by counting coefficient multiplications.

The following notation will be used:

1. $| P(x_1, \cdots, x_r) |$ is the number of terms of the polynomial P;
2. $T(P, Q)$ is the number of coefficient multiplications required to multiply polynomials P and Q or $| P | \cdot | Q |$;
3. $e(M)$ is an element of the matrix M.

While analyzing the storage requirements we shall assume that we are not permitted to alter the original matrix M. The storage required to store M will not be explicitly considered. Then:

4. $T_{(y)}^{(n)}$ is the computing time and $S_{(y)}^{(n)}$ is the storage required by algorithm y using model n;
5. $\deg(P)$ in x_i is the degree of P in its ith variable.

2.1. Interpolation Method. The concept of evaluating the polynomial entries of a matrix, computing the resulting determinants, and interpolating back has been known for a long time. A recent study of this approach has been made by McClellan [14]. Specifically, he considers the case where the polynomials initially have integer coefficients of arbitrary size. His method then selects a sequence of single-precision odd primes, p_i, and solves the related problem produced by taking the entries modulo p_i. For the purpose of this paper we consider his algorithm at the stage where the elements of the matrix are polynomials with coefficients in the field GF(p_i).

A more precise specification of this algorithm is now presented.

ALGORITHM EVAL-INTERP

```
Input:   $M$ an $n \times n$ matrix, $e(M)$ polynomials in $r$ variables with coefficients in GF$(p)$.
Output: $R(x_1, \cdots, x_r) = \det(M)$.
1. [Initial case]       if $r = 0$ compute $R \leftarrow \det(M)$ using Gaussian elimination;
                        return$(R)$; end.
2. [Initialize]         $m_r \leftarrow \max\{\deg(e(M)) \text{ in } x_r\} + 1$;
   [Interpolation]      $C(x_1, \cdots, x_r) \leftarrow 0,\ D(x_r) \leftarrow 1$;
                        $b \leftarrow -1$;
3. [Choose next]        $b \leftarrow b + 1$;
   [point]
4. [Evaluation]         $\bar{M}(x_1, \cdots, x_{r-1}) \leftarrow M(x_1, \cdots, x_{r-1}, b)$;
   [Homomorphism]       (evaluate every element of $M$ at $b$ for its $r$th variable)
5. [Recursion]          $R(x_1, \cdots, x_{r-1}) \leftarrow$ EVAL-INTERP $(\bar{M})$;
6. [Interpolate]        $C(x_1, \cdots, x_r) \leftarrow \{R(x_1, \cdots, x_{r-1}) - C(x_1, \cdots, x_{r-1}, b)\} * \{D(x_r)/D(b)\}$
   [the determinant]    $+ C(x_1, \cdots, x_r)$;
```

```
7. [Done?]      if deg($D$) $\le m_r n$ then
                    ($D(x_r) \leftarrow (x_r - b)D(x_r)$; go to (3);)
8. [End!]       $R(x_1, \cdots, x_r) \leftarrow C(x_1, \cdots, x_r)$; end.
```

Proof that this algorithm works can be found in [14]. Note that the main loop between steps 3 and 7 is governed by the maximum degree in the main variable of $e(M)$ and the size of the matrix.

Computing time analysis. We assume that each element of M has degree $m - 1$ in each of the variables $x_1, \cdots, x_r$ (and hence has at most m^r terms). Thus $\det(M)$ has degree at most $(m - 1)n$ in each variable and at most $((m - 1)n + 1)^r$ terms.

Let $T_{(\text{INTRP})}(r)$ be the number of multiplications required to compute the determinant of an $n \times n$ matrix with r-variable polynomials using the EVAL-INTERP algorithm and let $a = (m - 1)n + 1$. Then we have:

Step	*Number of multiplications*	*Comments*
4.	$a[(m - 2) + n^2 m^{r-1}(m - 1)]$	powers of x_r + multiplications
5.	$aT(r - 1)$	recursion
6.	$\sum_{i=3}^{a} (i - 2)a^{r-1} + (i - 3)$	evaluating $C(x_1, \cdots, x_{r-1}, b)$
	$\sum_{i=1}^{a} i + \sum_{i=3}^{a} (i - 2)$	$D(x_r)/D(b)$
	$\sum_{i=1}^{a} ia^{r-1}$	$(R - C)^*D(x_r)/D(b)$

Thus

$$\begin{aligned} T(r) &= aT(r - 1) + a(m - 2) + an^2m^{r-1}(m - 1) + \tfrac{1}{2}(a - 1)(a - 2)a^{r-1} \\ &\quad + \tfrac{1}{2}(a - 2)(a - 3) + a(a + 1) + \tfrac{1}{2}(a - 1)(a - 2) + \tfrac{1}{2}a(a + 1)a^{r-1} \\ &= aT(r - 1) + a(m - 2) + an^2m^{r-1}(m - 1) + (a^2 - a + 1)a^{r-1} \\ &\quad + (2a^2 - 3a + 4). \end{aligned}$$

Working out the recurrence relation, one gets

$$\begin{aligned} T(r) = a^rT(0) &+ r(a^2 - a + 1)a^{r-1} + am(a^{r-1})/(a - 1) \\ &+ (2a^2 - 5a + r)(a^r - 1)/(a - 1) \\ &+ am^{r-1}n^2(m - 1)[(a/m)^r - 1]/[(a/m) - 1] \\ &+ am^{r-1}n^2(m - 1)[(a/m)^r - 1]/[(a/m) - 1]. \end{aligned} \tag{1}$$

We have maintained this rather unwieldy expression in order to retain the constant. However, we note that

$$T_{(\text{INTRP})}(0) = n(n - 1)(n + 1)/3, \tag{2}$$

and hence the asymptotic time for the algorithm is

$$T_{(\text{INTRP})}(r) = O((mn)^r(n^3 + mn^2)). \tag{3}$$

Storage. Bounded by $r(\,|\,R\,| + n^2\,|\,e(M)\,|\,)$,

$$\therefore\ S_{(\text{INTRP})} = O(ra^r). \tag{4}$$

For later comparison we give here the formulas for the exact computing time of the EVAL-INTERP method for the special cases of one and two variables ($r = 1$ and $r = 2$). Note that these exact formulas are derived assuming elements over a field.

For one variable the time is

$$T(1) = a[\tfrac{1}{3}n(n - 1)(n + 1) + 3a - 6 + m + n^2(m - 1)] + 5, \tag{5}$$

and for two variables the time is

$$\begin{aligned} T(2) = \tfrac{1}{3}a^2n(n - 1)(n + 1) &+ an^2(m - 1)(a + m) + am(a + 1) \\ &+ 4a^3 - 5a^2 + a + 4. \end{aligned} \tag{6}$$

2.2. Gaussian Elimination with Exact Division. Below is an algorithm given by Bareiss in [1] that is used for computing the determinant while remaining over the integers. A proof that this works and that the division in step 2.1.1.1 below is exact may be found in [1].

ALGORITHM EXACT DIVISION

```
Input:   $M_{ij}^{(1)}$ an $n \times n$ matrix;
Output:  $R = \det(M)$;
1.       $M_{0.0}^{(0)} \leftarrow 1$;
2.       for $i \leftarrow 1$ to $n - 1$ do;
2.1.         for $k \leftarrow i + 1$ to $n$ do:
2.1.1.           for $j \leftarrow i + 1$ to $n$ do;
2.1.1.1.         $M_{kj}^{(i+1)} \leftarrow (M_{ii}^{(i)} M_{kj}^{(i)} - M_{ki}^{(i)} M_{ij}^{(i)}) / M_{i-1,i-1}^{(i-1)}$
2.1.2.           end;
2.2.         end;
3.       end
4.       $R \leftarrow M_{nn}^{(n)}$;
```

Computing time analysis. Bareiss [1] and Lipson [13] show that each $M_{ij}^{(i)}$ is a determinant (minor) of some $i \times i$ submatrix of the original matrix. Since we assumed all entries in the original matrix to be of the same size, we may expect that all elements $M_{ij}^{(i)}$, for a given i, are the same size. Though this is not true in general, it does provide a worst case analysis.

For each k we have $(n - i)^2$ executions of step 2.1.1.1. We assume that an exact division of A by B to yield C has the same cost as a multiplication of B by C. Then

$$T_{(GE)} = \sum_{i=1}^{n-1} (n - i)^2 (2T(M_{kj}^{(i)}, M_{kj}^{(i)}) + T(M_{kj}^{(i+1)}, M_{kj}^{(i-1)})). \tag{7}$$

Storage.

$$S_{(GE)} \leq \max_{0 \leq i \leq n-1} \{(n - i)^2 \mid M_{kj}^{(i+1)} \mid \}. \tag{8}$$

2.3. Characteristic Polynomial. Here we compute the characteristic polynomial $P(\lambda) = \det(M - \lambda I)$ of the matrix M. The coefficient of λ^0 is the determinant of M. This method is described in detail in [13]. As may be expected from the results of [8] and [12], the $O(n^4)$ method for matrices with integer elements described in [13] turns out to be better than the $O(n^{3.5})$ method when applied to an M whose entries are polynomials in at least one variable.

ALGORITHM CHARACTERISTIC POLYNOMIAL

```
Input:   $M$ an $n \times n$ matrix;
Output:  $R = \det(M)$;
1. [Compute trace of powers of $M$]                 $S(1) \leftarrow \mathrm{Trace}(M)$;
                                                     for $i \leftarrow 2$ to $r$
                                                       do;
                                                         $M^i \leftarrow M^* M^{i-1}$;
                                                         $S(i) \leftarrow \mathrm{Trace}(M^i)$;
                                                         end;
2. [Compute coefficients of characteristic polynomial of   $r(0) \leftarrow 1$;
   $M$: $P(\lambda) = \sum r(i)\lambda^{n-1}$             $r(1) \leftarrow -S(1)$;
                                                     for $i \leftarrow 2$ to $n$ do;
                                                     $r(i) \leftarrow -\sum_{j=0}^{i-1} r(j)S(i-j)/i$;
3. [Return determinant]                              $R \leftarrow (-1)^n r(n)$; end.
```

Computing time analysis.

$$T_{(\text{step 1})} = n^3 \sum_{i=1}^{n-1} T(e(m), e(M^i)),$$

$$T_{(\text{step 2})} = \sum_{i=2}^{n} \{\sum_{j=0}^{i-1} T(r(j), S(i-j)) + \mid r(i) \mid \},$$

$$T_{(CP)} = T_{(\text{step 1})} + T_{(\text{step 2})}. \tag{9}$$

Storage. We assume $|e(M^i)| \geq |e(M^{i-1})|$ for $i > 1$. Then storage for step 1 is less than or equal to $n^2 |e(M^n)| + n^2 |e(M^{n-1})| + \sum_{i=1}^{n} |s(i)|$. The $r(i)$ of step 2 may be stored in space freed by $M^{(n)}$ and $M^{(n-1)}$. Therefore

$$S_{(\text{CP})} \leq n^2\{ |e(M^n)| + |e(M^{n-1})| \} + \sum_{i=1}^{n} |s(i)|. \tag{10}$$

2.4. Evaluation by Minors. This is a well-known method and can be found, for example, in [12, p. 440, Ex. 10].

Here we have $n - 1$ steps, indexed by i from 2 to n. At each step, we compute all of the $i \times i$ minors from the first i columns (there are $\binom{n}{i}$ of them), using the $i - 1 \times i - 1$ minors from the first $i - 1$ columns, computed in the previous step. Gentleman and Johnson consider this method in [7].

Computing time analysis. The cost of computing an $i \times i$ minor, assuming that all required $i - 1 \times i - 1$ minors are available, is i times the time to multiply an element of M by a minor of size $i - 1$. Therefore the total time becomes

$$\begin{aligned} T_{(\text{MS})} &= \sum_{i=2}^{n} \binom{n}{i} iT(e(M), \det_{i-1,i-1}) \\ &= n\sum_{i=2}^{n} [(n-1)!/(i-1)!\,(n-i)!]T(e(M), \det_{i-1,i-1}) \\ &= n\sum_{i=1}^{n-1} \binom{n-1}{i} T(e(M), \det_{i,i}). \end{aligned} \tag{11}$$

Storage. At step i we compute the $\binom{n}{i}$ minors of size $i \times i$. Concurrent with this, the $\binom{n}{i-1}$ minors of size $i - 1 \times i - 1$ have to be retained.

$$\therefore \text{ Storage required at step } i \leq \binom{n}{i} |\det_{i,i}| + \binom{n}{i-1} |\det_{i-1,i-1}|$$

$$\therefore\ S_{(\text{MS})} = \max_{2 \leq i \leq n} \left\{ \binom{n}{i} |\det_{i,i}| + \binom{n}{i-1} |\det_{i-1,i-1}| \right\}. \tag{12}$$

3. *Models and Analysis*

In this section we analyze the times for applying the methods of Section 2 to the problem at hand. One should note that the time and storage formulas derived for the last three methods are independent of any assumptions about the elements of the matrix, other than that they must be from an integral domain. But our particular concern is for elements which are multivariate polynomials. Such polynomials can be quite diverse and may behave radically different as operations are performed on them. Thus it is necessary to make some assumptions about the way in which the polynomials grow as the reductions are carried out. In this case we hypothesize models which will hopefully reasonably reflect reality. In order to provide some motivation for the models we have used, let us look at the two extreme possibilities that can occur.

Suppose that every element of M is a polynomial in r variables, degree $m - 1$ in each variable, and every possible term occurs. Then each polynomial initially has m^r terms and is said to be completely dense. Suppose further that as we apply Gaussian elimination the resulting submatrices have polynomials which remain dense. In particular at the ith stage, each element has degree $i(m - 1)$ in every variable and $(i(m - 1) + 1)^r$ terms. A generalization of this growth rate for the completely dense case is given in Section 3.1.

Now for the other extreme case suppose that every element of M has t terms and as we apply Gaussian elimination the resulting polynomials have $i!\,t^i$ terms at the ith stage,

$1 \leq i \leq n$. This is equivalent to saying that after every iteration every term produced in the multiplication process is distinct from every other term. This gives the maximum rate of growth that could occur without terms overlapping and implies that the polynomials were initially quite sparse. Section 3.3 contains a generalized model for this completely sparse case.

Then in Section 3.2 a model which is intermediate to the generalized dense and generalized sparse models is presented. Within each of the three models we will consider the four algorithms—evaluation-interpolation, Gaussian elimination, minors, and characteristic polynomial—and compare their relative performances.

However, before we begin let us consider separately the evaluation-interpolation method, whose computing time remains the same for all of the models. This circumstance arises from the fact that there is no way for it to take advantage of any existing sparsity within polynomials. If for every element of M the degree of $e(M)$ in each variable is $m - 1$, then the evaluations and interpolation must be carried on $(m - 1)n + 1$ times at every level. This is so, even if a polynomial has only t terms where t is much smaller than m^r. Therefore the asymptotic computing time for this algorithm remains $O((mn)^r(n^3 + mn^2))$ independent of any of the models. Thus we can already surmise that though the evaluation-interpolation method may work quite well on completely dense polynomials of reasonably small degree and number of variables, other methods will become superior as the matrix's polynomials become more sparse. It is the primary purpose of the three models to allow for both sparse and dense polynomials which grow at different rates.

We will abbreviate the generalized dense, average growth, and generalized sparse models by GD, AG, and GS. The methods of Gaussian elimination, characteristic polynomial, and minors will continue to be referred to as GE, CP, and MS, respectively. Much of the actual derivations which produced the results of Sections 3.1, 3.2, and 3.3 have been deleted for readability, but they may be found in [9].

3.1. Generalized Dense Model. In this model we assume that a minor of size $i \times i$ has $i^k t$ terms, where $t = |e(M)|$ and k is a nonnegative integer. We note that for $t = m^r$ and $k = r$ this model corresponds to the case where the $e(M)$ are completely dense polynomials in r-variables and remain completely dense throughout (actually the growth is as $(i(m - 1) + 1)^r = O(i^r m^r)$). On the other hand, small values of k and t allow for very sparse polynomials which grow at the same rate as the dense polynomials.

3.1.1. *Gaussian Elimination.* Using eq. (7), we get

$$\begin{aligned} T_{(\mathrm{GE})}^{(\mathrm{GD})} &= \sum_{1 \leq i \leq n-1} (n - i)^2 (2T(M_{kj}^{(i)}, M_{kj}^{(i)}) + T(M_{kj}^{(i+1)}, M_{kj}^{(i-1)})) \\ &= \sum_{1 \leq i \leq n-1} (n - i)^2 (2(i^k t)^2 + (i + 1)^k t (i - 1)^k t) = O(t^2 n^{2k+3}). \end{aligned}$$

For the completely dense case $t = m^r$, $k = r$ we get

$$T_{(\mathrm{GE})} = O(m^{2r} n^{2r+3}) \geq O((mn)^r (n^3 + mn^2)) = T_{(\mathrm{INTRP})}.$$

Thus evaluation-interpolation is asymptotically superior to Gaussian elimination on completely dense polynomials for all $r \geq 1$.

From eq. (8) we get $S_{(\mathrm{GE})}^{(\mathrm{GD})} = \max_{1 \leq i \leq n-1} (n - i)^2 (i + 1)^k t$. The maximum occurs for $i = (kn - 2)/(k + 2)$, and thus we get $S_{(\mathrm{GE})}^{(\mathrm{GD})} \leq O((n + 1)^{k+2} t / k^2)$.

As special cases, the exact equations for one and two variables are derived in Section 3.4.

3.1.2. *Characteristic Polynomial.* Let $|r_i|$, $|S_i|$, and $|e(M^i)| = i^k t$. Using eq. (9) we get:

$$\begin{aligned} T_{(\mathrm{CP})}^{(\mathrm{GD})} &= n^3 \sum_{i=1}^{n-1} i^k t^2 + \sum_{i=2}^{n} \sum_{j=1}^{i-1} j^k (i - j)^k t^2 + \sum_{i=2}^{n} i^k t \\ &< t^2 n^{k+4}/(k + 1) + \tfrac{1}{2} t^2 (n + 1)^{2k+2}/4^k (k + 1) + n^{k+1} t \\ &= O(t^2 n^k (4^k n^4 + n^{k+2})/4^k (k + 1)). \end{aligned}$$

Using eq. (10) for the storage requirements we get

$$S^{(\mathrm{GD})}_{(\mathrm{CP})} \le n^2\{n^k t + (n-1)^k t\} + \sum_{1}^{n} i^k t \le O(n^{k+2}t).$$

3.1.3. *Minors.* Here $|\det_{i,i}| = i^k t$, $T(e(M), \det_{i,i}) = i^k t^2$ and substituting into eq. (11) we get

$$T^{(\mathrm{GD})}_{(\mathrm{MS})} = n\sum_{i=1}^{n-1}\binom{n-1}{i} i^k t^2 \le n(n-1)^k 2^{n-2} t^2.$$

For the storage requirements we get, from eq. (12),

$$S^{(\mathrm{GD})}_{(\mathrm{MS})} \le \max_{2\le i\le n}\left\{\binom{n}{i} i^k t + \binom{n}{i-1}(i-1)^k t\right\} < 2tn^k\binom{n}{n/2}.$$

The exact values for one and two variables appear in Section 3.4.

3.2. Average Growth Model. In this model it is assumed that the multiplication of a p-term polynomial and a q-term polynomial ($p \ge q$) results in a kp-term ($1 \le k \le p$) polynomial, while their addition yields a p-term polynomial. Such behavior arises, for instance, when $k = q$, the polynomials P and Q are sparse, and the exponents occurring in Q are a subset of those occurring in P. Examples for other values of k may be obtained by varying the degree of sparsity and exponent distribution in P.

The determinant of an $i \times i$ matrix may be viewed as the sum of $i!$ terms, where each term is a product of i polynomials. If each polynomial entry in M has t terms, then under our assumptions each of the $i!$ terms in the sum for det (M) has $k^{i-1}t$ terms and det (M) is a polynomial with $k^{i-1}t$ terms (because of the assumption for addition of polynomials). We note that the case when all the entries of M are integers is a special case of this model, i.e. $k = 1$ and $t = 1$.

3.2.1. *Gaussian Elimination.* By assumption $|M^{(i)}| = k^{i-1}t$. Using eq. (7) we get $T^{(\mathrm{AG})}_{(\mathrm{GE})} = \sum_{1\le i\le n-1}(n-i)^2(2(k^{i-1}t)^2 + k^{i-2}t\cdot k^i t)$, with $k^{-1} = 1$. Then $T^{(\mathrm{AG})}_{(\mathrm{GE})} = \sum_{1\le i\le n-1}(n-i)^2(3k^{2(i-1)}t^2) \le O(t^2k^{2(n-2)}n^3)$. For the storage, we get from eq. (8): $S^{(\mathrm{AG})}_{(\mathrm{GE})} = \max_{0\le i\le n-1}\{(n-i)^2k^i t\}$, which is $\le k^{n-1}t$ for $k^{i-1}t \ge (i+1)^2$.

3.2.2. *Characteristic Polynomial.* Since in the average growth model we assume that the addition of a p-term and a q-term polynomial ($p \ge q$) results in a p-term polynomial while their product has k p-terms, we obtain $|s_i| = |e(M^i)| = k^{i-1}t$, as s_i is just the sum of the diagonal entries of M^i. We make the further assumption that $|r_i| = |s_i|$.

From eq. (9) we get: $T^{(\mathrm{AG})}_{(\mathrm{CP})} = T_{(\text{step 1})} + T_{(\text{step 2})}$.

$$T_{(\text{step 1})} = \begin{cases} n^3t^2(k^{n-1}-1)/(k-1), & k > 1,\\ n^3(n-1)t^2, & k = 1,\end{cases}$$

$$T_{(\text{step 2})} = \begin{cases} t^2[(n-1)k^n - nk^{n-1} + 1]/(k-1)^2 + (k^n - k)t/(k-1), & k > 1;\\ \tfrac{1}{2}t^2n(n-1) + t(n-1), & k = 1.\end{cases}$$

Therefore

$$T^{(\mathrm{AG})}_{(\mathrm{CP})} = \begin{cases} O(n^3t^2k^{n-2} + t^2nk^{n-2}) = O(n^3t^2k^{n-2}), & k > 1\\ O(n^4t^2), & k = 1.\end{cases}$$

Substituting in eq. (10) for the storage, we obtain $S^{(\mathrm{AG})}_{(\mathrm{CP})} = O(n^2k^{n-1}t)$.

3.2.3. *Minors.* Here $|\det_{i,i}| = k^{i-1}t$ and thus $T(e(M), \det_{i,i}) = k^{i-1}t^2$. Substituting into eq. (11) we get

$$T^{(\mathrm{AG})}_{(\mathrm{MS})} = n\sum_{i=1}^{n-1}\binom{n-1}{i}k^{i-1}t^2 = nt^2/k\left(\sum_{i=0}^{n-1}\binom{n-1}{i}k^i - 1\right) = nt^2/k\,[(1+k)^{n-1} - 1].$$

Equation (12) yields:

$$S^{(\mathrm{AG})}_{(\mathrm{MS})} = \max_{2\le i\le n}\left\{\binom{n}{i}k^{i-1}t + \binom{n}{i-1}k^{i-2}t\right\} \le 2t(1+k)^n/k.$$

3.3. Generalized Sparse Model. The assumptions for this model are similar to those for the average growth model. As in that model, here too, it is assumed that the product of a p-term and a q-term ($p \geq q$) polynomial is a kp-term ($1 \leq k \leq q$) polynomial. However, we now assume that the exponents present in Q are not present in P so that their addition results in a $(p + q)$-term polynomial. For $k = q$, this means that P, Q are totally sparse and have no exponents in common.

The consequences of this change in assumption for addition is that the determinant of an $i \times i$ matrix now has a number of terms equal to $i! \cdot$(number of terms in the product of i polynomials) $= i!k^{i-1}t$ terms. For $k = t$, this represents the maximum possible growth rate that can occur. It is readily seen that the $n \times n$ matrix whose i, j element is r_{ij}, where r_{ij} represents a variable and $r_{ij} = r_{lm} \Leftrightarrow i = l$ and $j = m$ falls under this model with $k = t = 1$. Examples for other values of k, t can easily be constructed.

3.3.1. *Gaussian Elimination.* From eq. (7) we obtain $T_{(GE)}^{(GS)} = O(t^2 n!^2 k^{2(n-2)} n^3)$, and eq. (8) gives, for the storage, $S_{(GE)}^{(GS)} = \max_{1 \leq i \leq n-1} \{(n - i)^2 (i + 1)! k^i t\}$, which is $\leq n! k^{n-1} t$ for $n, k \geq 2$. For the completely sparse case we get $T_{(GE)}^{(\text{sparse})} = O(n!^2 n^3 t^{2n-2})$ and $S_{(GE)}^{(\text{sparse})} = O(n! t^n)$.

3.3.2. *Characteristic Polynomial.* Since in this model we assume that multiplication of a t-term polynomial by a t-term polynomial results in a kt-term polynomial and that their addition yields a $2t$-term polynomial, we get $|e(M^i)| = (nk)^{i-1}t$.

Since $s_i = \sum_{j=1}^{n} a_{jj}^i$ where a_{jj}^i is a diagonal entry of M^i, we obtain $|s_i| = n|e(M^i)| = n^i k^{i-1} t$. In step 2 it is assumed that no new terms are generated, so $|r_i| = |s_i| = n^i k^{i-1} t$. Substituting for $|e(M^i)|$, $|r_i|$, $|s_i|$ in eq. (9), we get $T_{(GE)}^{(GS)} = T_{(\text{step 1})} + T_{(\text{step 2})}$:

$$T_{(\text{step 1})} = n^3 \sum_{i=1}^{n-1} t \cdot (nk)^{i-1} t = n^3 t^2 [(nk)^{n-1} - 1]/(nk - 1),$$

$$T_{(\text{step 2})} = n^2 t^2 [(n - 1)(nk)^n - n(nk)^{n-1} + 1]/(nk - 1)^2 + nt[(nk)^n - nk]/(nk - 1), \quad nk > 1.$$

Therefore $T_{(CP)}^{(GS)} = O(n^{n+1} k^{n-2} t^2)$. The storage required is

$$S_{(CP)}^{(GS)} \leq n^2 \{(nk)^{n-1} t + (nk)^{n-2} t\} + \sum_{i=1}^{n} n^i k^{i-1} t = O(n^{n+1} k^{n-1} t),$$

and for the completely sparse case we have $T_{(CP)}^{(\text{sparse})} = O(n^{n+1} t^n)$ and $S_{(CP)}^{(\text{sparse})} = O(n^{n+1} t^n)$.

3.3.3. *Minors.* Here $|\det_{i,i}| = i! k^{i-1} t$, $T(e(M), \det_{i,i}) = i! k^{i-1} t^2$ and thus $T_{(MS)}^{(GS)} = n \sum_{i-1}^{n-1} \binom{n-1}{i} i! k^{i-1} t^2 \leq n! t^2 k^{n-2} e^{1/k}$. For the completely sparse case ($k = t$), we get $T_{(MS)}^{(GS)} \leq n! t^n e^{1/t}$. Equation (12) yields, for the storage required,

$$S_{(MS)}^{(GS)} = \max_{2 \leq i \leq n} \left\{ \binom{n}{i} i! k^{i-1} t + \binom{n}{i-1} (i - 1)! k^{i-2} t \right\} = n! k^{n-1} t (1 + 1/k).$$

For the completely sparse case, this gives $S_{(MS)}^{(\text{sparse})} = n! t^n (1 + 1/t) \leq 1 \cdot 5 n! t^n$, $t \geq 2$.

3.4. Special Cases. In this section we derive the exact equations for the methods Gaussian elimination, characteristic polynomial, and minors under the assumption of dense polynomials in one and two variables. These particular cases are ones which are often useful in practice.

3.4.1. *Gaussian Elimination.* With $|e(M)| = t$, $|M^{(i)}| = i(t - 1) + 1$, and substituting in eq. (7) we get

$$T_{(GE)}^{(\text{dense univariate})} = \tfrac{1}{30} n(n - 1) \cdot [(3n^3 + 3n^2 - 7n + 8)(t - 1)^2 + 15n(n - 1)(t - 1) + 15(2n - 1)]. \quad (13)$$

With $m^2 = t$, $|M^{(i)}| = (i(m - 1) + 1)^2$, and substituting again in eq. (7) we get

$$\begin{aligned} T_{(GE)}^{(\text{dense bivariate})} = {} & (\tfrac{1}{35} n^7 - \tfrac{1}{15} n^5 + \tfrac{7}{30} n^3 - \tfrac{1}{2} n^2 + \tfrac{32}{105} n)(m - 1)^4 \\ & + (\tfrac{1}{5} n^6 - \tfrac{1}{3} n^4 + \tfrac{2}{15} n^2)(m - 1)^3 \\ & + (\tfrac{3}{5} n^5 - \tfrac{2}{3} n^3 + n^2 - \tfrac{14}{15} n)(m - 1)^2 \\ & + (n^4 - n^2)(m - 1) + n^3 - \tfrac{3}{2} n^2 + \tfrac{1}{2} n. \end{aligned} \quad (14)$$

3.4.2. *Characteristic Polynomial.* Here $|r_i|$, $|S_i|$, and $|e(M^i)| = i(t-1)+1$. Thus the time becomes

$$T_{(\mathrm{CP})}^{(\text{dense univariate})} = n(n-1)[\tfrac{1}{2}n^3(t-1)t + n^2t + \tfrac{1}{24}(n+1)(n+2)(t-1)^2 + \tfrac{1}{3}(n+1)(t-1)+1] + \tfrac{1}{2}n(n+1)(t-1) - t + n. \qquad (15)$$

For two variables we have $|r_i|, |S_i|, |e(M^i)| = (i(m-1)+1)^2$ and $T_{(\mathrm{CP})}^{(\text{dense bivariate})} = T_{(\text{step 1})} + T_{(\text{step 2})}$:

$$T_{(\text{step 1})} = n^3(n-1)t[\tfrac{1}{6}n(2n-1)(m-1)^2 + n(m-1)+1]$$

$$T_{(\text{step 2})} = \sum_{i=2}^{n}\sum_{j=1}^{i-1}(j^2(m-1)^2 + 2j(m-1)+1)((i-j)^2(m-1)^2 + 2(i-j)(m-1)+1) + \sum_{i=2}^{n}|r_i|$$

$$\begin{aligned}T_{(\mathrm{CP})}^{(\text{dense bivariate})} &= (m-1)^4\{\tfrac{1}{360}n^2(n+1)(2n^3+4n^2+n-1) - \tfrac{1}{60}n(n+1)\}\\ &+ (m-1)^3\{\tfrac{1}{90}n(n+1)(2n+1)(3n^2+3n-1) - \tfrac{1}{8}n(n+1)(2n+1)\}\\ &+ (m-1)^2\{\tfrac{1}{3}n^2(n+1)^2 - \tfrac{1}{6}n(n+1) - 1\}\\ &+ (m-1)\{\tfrac{1}{3}n(n+1)(2n+1) - 2\} + \tfrac{1}{2}n(n-1) + (n-1). \qquad (16)\end{aligned}$$

3.4.3. *Minors.* For the case of one variable we get $|e(M)| = t$ and therefore $\deg(e(M)) = t-1$, $|\det_{i,i}| = i(t-1)+1$ and $T(e(M), \det_{i,i}) = it(t-1)+t$.

$$T_{(\mathrm{MS})}^{(\text{dense univariate})} = nt[(t-1)(n-1)2^{n-2} + 2^{n-1} - 1]. \qquad (17)$$

This result was first given by Gentleman and Johnson in [7].

The special case of dense bivariate polynomials yields $|e(M)| = m^2$, $t = m^2$, and $\deg(e(M)) = m-1$. Then

$$|\det_{i,i}| = (i(m-1)+1)^2 \quad\text{and}\quad T(e(M), \det_{i,i}) = t(i(m-1)+1)^2.$$

$$T_{(\mathrm{MS})}^{(\text{dense bivariate})} = nt(2^{n-1} + 2^{n-2}\{(n-1)[(m-1)^2 + 2(m-1)]\} + 2^{n-3}\{(n-1)(n-2)(m-1)^2\} - 1). \qquad (18)$$

4. *Asymptotic Conclusions*

The table below summarizes the asymptotic computing times and storage requirements for each of the three methods under each of the three models. The methods are listed in increasing order of time.

Method	*Time*	*Space*
Generalized dense:		
Characteristic polynomial	$t^2(n^{k+4} + n^{2k+2}/4^k)$	tn^{k+2}
Gaussian elimination	t^2n^{2k+3}	$t(n+1)^{k+2}/k^2$
Minors	$t^2n^{k+1}2^{n-2}$	$tn^k\binom{n}{n/2}$
Average growth:		
Minors	t^2nk^{n-2}	$2t(k+1)^n/k$
Characteristic polynomial	$t^2n^3k^{n-2}(k > 1)$	$tk^{n-1}n^2$
Gaussian elimination	$t^2n^3k^{2(n-2)}$	tk^{n-1}
Generalized sparse:		
Minors	$t^2n!\,k^{n-2}e^{1/k}$	$tn!\,k^{n-1}(1+1/k)$
Characteristic polynomial	$t^2n^{n+1}k^{n-2}$	$tn^{n+1}k^{n-1}$
Gaussian elimination	$t^2n!^2k^{2(n-2)}n^3$	$tn!\,k^{n-1}$

Thus we can conclude that asymptotically the following is true: for both the average growth and generalized sparse models, *minors* is the best method for all values of k. As for storage, minors is either better than or no worse than any of the other methods by a constant factor. In fact for the completely sparse polynomial case, as considered by Gentleman and Johnson in [7] where $k = t$, minors is optimal to within a constant factor.

For the generalized dense model the characteristic polynomial method is superior for all $k > 1$, and equivalent to Gaussian elimination for $k = 1$.

The fact that minors is asymptotically the best method on certain models is important, but one must note that because of the exponential growth of this method only relatively small problems could ever be successfully computed. Thus it still remains to show the practical usefulness of adopting minors over the other more commonly used methods. However, we note that this conclusion is supported by recent work of Bareiss and Mazukelli [3], who show that a modified Cramers rule is useful for solving a numerical system when m is much larger than n.

But before doing that there is more to be gained from the formulas just derived, realizing that practical computation will imply small values for all of the variables. For example, in [7] it is shown that for univariate dense polynomials minors is better than Gaussian elimination for $n \leq 7$. If M has completely dense univariate polynomial entries, then, by eqs. (5), (13), and (17) which give the exact computing times for evaluation-interpolation, exact division, characteristic polynomial, and minors, we find that for any number of terms $t > 1$, minors is better on matrices whose size is $n \leq 5$. So in the completely dense univariate case it is better to use minors over evaluation-interpolation or characteristic polynomial as long as the size of the matrix is ≤ 5 no matter how many terms are in the polynomials.

Similarly examining eqs. (6), (14), (16), and (18) for the completely dense bivariate case, minors requires fewer multiplications as long as one of the following conditions hold: (a) $n = 2$ & $2 \leq m \leq 28$; (b) $n = 3$ & $2 \leq m \leq 11$; (c) $n = 4$ & $2 \leq m \leq 4$.

For $n > 4$, evaluation-interpolation requires fewer operations than minors. We further note that the Gaussian elimination method will be worse than either minors or characteristic polynomial for all $m, n > 1$. Thus the formulas imply that Gaussian elimination is a poor method to use in every case.

Thus at this point we see that for the broad possibilities of polynomials as subsumed within the generalized sparse and average growth models, minors is the choice. Also, for small problems with completely dense polynomials, minors is better. For problems as described by the generalized dense model, characteristic polynomial has a smaller operation count than Gaussian elimination but requires more storage. The next section on empirical tests will give a clearer picture of which method is favorable. Though in the case of generalized dense, minors is exponential, whereas the other two methods are polynomial functions of n and t, the feasible domain of computation implies small n and so we will continue to keep all three methods under scrutiny.

5. *Empirical Studies*

Using the SAC-1 system [5], which allows for symbolic operations on polynomials, and an IBM 360/65, several tests were run to determine the relative merits of the three methods: minors, Gaussian elimination, and characteristic polynomial. The results of these tests are given in Table I. Entries marked ([a]) indicate that the workspace provided, 30,000 words, was not enough to compute the corresponding determinant.

Let a_{ij} denote an element of the matrix M, whose determinant is to be found. Then the data sets used were:

Data set I (symmetric Toeplitz matrix): $a_{ij} = x_{|i-j|}$.
Data set II (Vandermonde matrix): $a_{ij} = (x_j)^{i-1}$.
Data set III: $a_{1j} = \sum_{i=0}^{j-1} x^i$; $a_{ij} = 1 + xa_{i-1,j}$, $i > 1$.
Data set IV: Let $M_1(x)$ and $M_2(y)$ be the two matrices constructed using data set III with variables x and y. Then $M = M_1(x) + M_2(y)$.
Data set V: This is similar to data set IV but now $M = M_1(x) + M_2(y) + M_3(z)$.

Data sets I and II are instances of the generalized sparse model; data set III is an instance of the generalized dense, and data sets IV and V are examples of the average growth model.

TABLE I. Time Required to Compute the Determinant of an $n \times n$ Matrix Using the SAC-1 System on the IBM 360/65

Times in seconds

Data set	n	Minors	Gaussian elimination	Characteristic polyomial
I	3	.1	.1	.5
Symmetric Toeplitz	4	.7	1.3	4.5
	5	3.5	9.6	32.3
	6	18.1	[a]	[a]
	7	[a]		
II	3	.1	.2	.8
Vandermonde	4	0.6	1.5	9.0
	5	3.6	17.4	[a]
	6	[a]	[a]	
III	3	.1	.1	1.3
	4	.4	.4	11.9
	5	1.1	.9	66.0
	6	2.6	2.0	[a]
	7	5.2	3.5	
	8	9.7	5.7	
	9	16.6	9.0	
	10	[a]	13.2	
	11		21.5	
	12		34.8	
	13		38.7	
	14		53.6	
	15		68.8	
IV	3	.8	.8	8.7
	4	5.3	4.5	[a]
	5	22.3	13.8	
	6	[a]	34.8	
	7		70.9	
	8		128.8	
	9		[a]	
V	3	2.7	3.2	30.8
	4	31.2	49.4	[a]
	5	[a]	[a]	

[a] $\Rightarrow$ the provided workspace, 30,000 words, was insufficient.

From the results reported in Table I and corroborated by the results of [7 and 14], it is clear that only relatively small problems can be solved in a reasonable amount of space and time. Rarely can n exceed 10 except for small polynomials in one variable. In [7] the maximum n before storage was exceeded was 7 and in [14] for two variables or more the maximum n was 5. Typically, only problems with an $n \leq 6$ can be solved.

Examining Table I more closely, we see that minors worked best as predicted for data sets I and II. For $n = 5$ it was doing at least three times better than its nearest competitor. But note that in both cases it only allowed us to compute for one higher value of n before storage was exceeded.

The case for data sets IV and V is more interesting in that it shows up the inadequacy of doing only a computing time analysis without testing. Minors is predicted as being the best method, but for all practical cases Gaussian elimination does as well or much better. For the case of $n = 4$ and three variables, minors does finally compute the solution faster than Gaussian elimination. We would expect that if we could handle larger problems, the scales would tip toward minors. But within the conceivably useful range, Gaussian elimination is preferable. Similarly for data set III, which is an instance of generalized dense, Gaussian elimination does much better than the other two methods.

Thus the conclusions resulting from Sections 4 and 5 seem to us to be as follows: for relatively sparse matrices minors is preferable, whereas for relatively dense matrices Gaussian elimination should be chosen.

We note that in all cases characteristic polynomial turned out to be poor in practice. We have run some experiments using another method. This is a hybrid approach which first uses Gaussian elimination, and if the polynomials become large it switches to minors. We have run this combined algorithm on several of our data sets, but have been unable to improve upon the simpler, single algorithm. Finally, an important practical reason for preferring Gaussian elimination over all the other methods is the user's ability to watch the matrix as it is being reduced. Sometimes even if one can't produce the final answer, a lot can be gained from watching several steps. The other methods, with their strategies of breaking up the problem, make it harder to see what is going on with only partial output.

Acknowledgment. We would like to thank Morven Gentleman for initially suggesting the problem to us. We also would like to thank Erwin Bareiss for his helpful suggestions, as well as the referees.

REFERENCES

1. BAREISS, E. H. Sylvester's identity and multistep integer-preserving Gaussian elimination. *Math. Compt. 22*, 103 (July 1968), 565–578.
2. BAREISS, E. H. Computational solutions of matrix problems over an integral domain. *J. Inst. Math. and Appls. 10* (1972), 68–104.
3. BAREISS, E. H., AND MAZUKELLI, D. Multistep elimination over commutative rings. Argonne National Lab. Rep. ANL-7898, April 1972.
4. CABAY, S. Exact solution of linear equations. Proc. Second Symposium on Symbolic and Algebraic Manipulation, ACM, Los Angeles, Calif., March 1971, pp. 392–398.
5. COLLINS, G. The SAC-1 system: An introduction and survey. Proc. of the Second Symposium on Symbolic and Algebraic Manipulation, ACM, Los Angeles, Calif., March 1971, pp. 144–152.
6. DORR, F. W. An example of ill-conditioning in the numerical solution of singular perturbation problems. *Math. Comput. 25*, 114 (April 1971), 271–284.
7. GENTLEMAN, W. M., AND JOHNSON, S. C. Analysis of algorithms, a case study: Determinants of polynomials. Proc. of the Fifth Annual ACM Symposium on Theory of Computing, Austin, Texas, April 30, 1973, pp. 135–142.
8. HOROWITZ, E. The efficient calculation of powers of a polynomial. *J. Comput. Syst. Sci. 7*, 5 (Oct. 1973), 469–481.
9. HOROWITZ, E., AND SAHNI, S. On computing the determinant of matrices with polynomial entries. Comput. Sci. TR 73-180, Cornell U., Ithaca, N.Y., June 1973.
10. HOWELL, J., AND GREGORY, R. T. Solving systems of linear algebraic equations using residue arithmetic I, II, and III. *BIT 9* (1969), 200–224, 324–337, and *BIT 10* (1970), 23–27.
11. ISAACSON, E., AND KELLER, H. B. *Analysis of Numerical Methods*. Wiley, New York, 1966.
12. KNUTH, D. E. *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*. Addison-Wesley, Reading, Mass., 1969.
13. LIPSON, J. D. Symbolic methods for the computer solution of linear equations with applications to flow graphs. Proc. of the 1968 Summer Institute on Symbolic Mathematical Computation, IBM, Boston, June 1969, pp. 233–303.
14. MCCLELLAN, M. T. The exact solution of systems of linear equations with polynomial coefficients. *J. ACM 20*, 4 (Oct. 1973), 563–588.

RECEIVED JANUARY 1973; REVISED FEBRUARY 1974