Rounding Errors in Certain Algorithms Involving Markov Chains

WINFRIED K. GRASSMANN
University of Saskatchewan

A number of algorithms involving Markov chains contain no subtractions. This property makes the analysis of rounding errors particularly simple. To show this, some principles for analyzing the propagation and generation of rounding errors in algorithms containing no subtraction are discussed first. These principles are then applied in the context of a simple recursive algorithm involving the transient solution of discrete-time Markov chains, Jensen's algorithm, and state reduction. Jensen's algorithm, also known as randomization or uniformization, is an algorithm for finding transient solutions of continuous-time Markov chains. State reduction is a method for finding equilibrium probabilities for discrete-time or continuous-time Markov chains

1. INTRODUCTION

There are a number of algorithms arising in Markov chain analysis that contain no subtractions. These algorithms tend to be resistant against rounding errors because they avoid subtractive cancellation. They also simplify the analysis of the generation and propagation of rounding errors considerably, as is shown in this paper. Specifically, Section 1 presents a methodology for analyzing rounding errors in algorithms containing no subtractions. Section 2 then applies this methodology to analyze the rounding errors arising when calculating transient probabilities in discrete-time Markov chains. The theory of discrete-time Markov chains is essential for the implementation of Jensen's method, also known as randomization, which is discussed in Section 3. Section 4 then presents an error analysis of the state reduction method, a method that finds the equilibrium probabilities of discrete-time and continuous-time Markov chains.

We hope that our effort is also of interest to researchers outside the area of stochastic processes. We feel that by restricting ourselves to algorithms containing no subtractions we gain a new perspective that may help in the analysis of rounding errors, in general. In particular, we complement the traditional analysis of error generation and propagation by what we call *long-term* analysis. Specifically, we identify a number of cases in which the

Author's address: Department of Computational Science, University of Saskatchewan, Saskatoon, Canada S7N 0W0.

effect of earlier errors decreases with the number of iterations, reaching zero in the limit.

2. THE COMPUTATIONAL MODEL

This section lays out the basic techniques to be used later. Many of the techniques are, of course, known, but since we use results from many different sources with often widely different philosophies, it becomes unavoidable to state our basic assumptions explicitly. First, we assume that we have two different algebras, namely, the infinite precision algebra and the computer algebra. Both algebras are algebras over $\mathscr{P}$, the set of nonnegative reals. The infinite precision algebra, which we refer to as the *ideal* algebra, contains the operations $+$, $\times$, and $\div$, whereas the corresponding operations of the computer algebra are denoted by $\hat{+}$, $\hat{\times}$, and $\hat{\div}$. The set $\{+, \times, \div\}$ is denoted by Φ, and the set of the corresponding computer operations is similarly denoted by $\hat{\Phi}$.

If $\circ \in \Phi$ is an ideal operation and $\hat{\circ}$ is the corresponding computer operation, we require that the following relation holds: There is a value of u such that

$$\frac{x \circ y}{1+u} \leq x \,\hat{\circ}\, y \leq (x \circ y)(1+u). \tag{1}$$

The formulation that we use is thus somewhat like the rounding-error analysis introduced by Wilkinson [1963], complemented with some ideas of Kulisch and Miranker [1991]. Moreover, we make use of an improvement suggested in De Boor and Pinkus [1977] and in Higham [1990], and we replace the value $1-u$ used by Wilkinson [1963] by $1/(1+u)$. In this approach, u must be set slightly higher than in Wilkinson's approach. Specifically, if u_c is Wilkinson's unit error and if u is the error underlying (1), then

$$u = u_c + \frac{u_c^2}{1-u_c}.$$

The difference between u and u_c is so small that it is irrelevant for all practical purposes. For simplicity, instead of (1) we write

$$x \,\hat{\circ}\, y = (x \circ y)(1+u)^{\pm 1}. \tag{2}$$

As is well known, computer algebra is no longer associative. In order to avoid parentheses, we assume that the normal priority rules of algorithmic languages, such as Pascal, are followed. For instance, $x \hat{+} y \hat{+} z$ is to be understood as $(x \hat{+} y) \hat{+} z$. Neither $x \circ y$ nor $x \,\hat{\circ}\, y$ can be zero unless one or both operands are zero. In other words, the algebras in question are also closed if we restrict the domain of the algebra to the position reals or to the set $\mathscr{P} - \{0\}$.

When using $\hat{\circ}$ instead of $\circ$, one *generates* rounding errors. Once an error is generated, it may *propagate*; that is, it may affect further results, even if the ideal algebra is used. In the *short-term* analysis, we investigate the error

propagation only for a finite and usually small number of steps. Classical error analysis tends to be restricted to short-term analysis. Here, we also introduce a *long-term* analysis; that is, we investigate whether or not errors committed earlier decrease and eventually fade out, or increase and thus restrict the size of the problem for which reasonably accurate results are obtainable.

To do the short-term analysis, we assume that we are given intermediate results $\bar{x}$ and $\bar{y}$ that may deviate from their true values x and y. Specifically, we assume that there are two values e_x and e_y such that

$$\bar{x} = xe_x^{\pm 1}, \tag{3}$$

$$\bar{y} = ye_y^{\pm 1}. \tag{4}$$

As in (2), the expression $\bar{x} = xe_x^{\pm 1}$ indicates that $\bar{x}$ is between x/e_x and xe_x, and the same holds for all other equations. From (3) and (4),

$$\bar{x} \times \bar{y} = (x \times y)(e_x e_y)^{\pm 1}, \tag{5}$$

$$\bar{x} \div \bar{y} = (x \div y)(e_x e_y)^{\pm 1}, \tag{6}$$

$$\bar{x} + \bar{y} = xe_x^{\pm 1} + ye_y^{\pm 1}. \tag{7}$$

In the domain of $\mathscr{P}$, (7) implies that

$$\bar{x} + \bar{y} = (x + y)(e_x e_y)^{\pm 1}. \tag{8}$$

Hence, in $\mathscr{P}$, we always have

$$\bar{x} \circ \bar{y} = (x \circ y)(e_x e_y)^{\pm 1}. \tag{9}$$

In the general case, one can improve the bounds given by (9) only if "$\circ$" = "+".

We now generalize the error propagation to general expressions. This generalization is easily done by complete induction. For this purpose, we need the following recursive definition of an expression (see, e.g., Andrews 1986):

Definition 1. Let A be a string consisting of numerical constants, of variable names, say $v_1, v_2, \ldots$, of operators $\circ \in \Phi$, and of parentheses. Then, A is an expression iff either one of the following conditions applies:

(1) A is a numerical constant or a variable name.

(2) A is $(A_1 \circ A_2)$, $\circ \in \Phi$, where A_1 and A_2 are themselves expressions.

If no ambiguity arises, parentheses may be omitted.

We also define val(A) to be the value to which A evaluates if all v_i are assigned the given values x_i. Moreover, we assume that there are approximate values $\bar{x}_i = x_i e_i^{\pm 1}$, where e_i is given. If the v_i are assigned the values $\bar{x}_i$, then A evaluates to $\overline{\text{val}}(A)$.

THEOREM 1. *If v_i occurs r_i, $i = 1, 2, 3, \ldots$, times in A, then*

$$\overline{val}(A) = val(A) \prod_{i=1}^{\infty} e_i^{\pm r_i}.$$

Moreover, this bound is tight, unless the expression contains additions.

PROOF. The theorem is correct for expressions of length 1, which must be numerical constants or variables. If the theorem holds for the subexpressions A_1 and A_2, then (2) implies that it also holds for $A_1 \circ A_2$, which completes the inductive case. The inductive case is tight unless "$\circ$" is "+". This completes the proof. □

If $\bar{x}_i = x_i e^{\pm 1}$, $i = 1, 2, \ldots, n$, where e is given, the theorem implies that

$$\bar{x}_1 \circ \bar{x}_2 \circ \cdots \circ \bar{x}_n = (x_1 \circ x_2 \circ \cdots \circ x_n) e^{\pm n}. \tag{10}$$

Equation (10) holds for any operator, that is, for $+$, $\times$, and $\div$, and the order in which the operations are carried out is irrelevant. One can even group the x_i into subexpressions, such as $(x_1 \circ x_2) \circ (x_3 \circ x_4)$, and one still has the error e^n, or e^4 for this specific case. If $\circ = +$, (10) can be strengthened considerably, as can be proved from first principles.

$$\bar{x}_1 + \bar{x}_2 + \cdots + \bar{x}_n = (x_1 + x_2 + \cdots + x_n) e^{\pm 1}. \tag{11}$$

Since (11) is much stronger than (10), it pays to treat summations separately.

The next theorem combines error propagation and error generation:

THEOREM 2. *Let A be an expression in $\mathscr{P}$, and let $op(A)$ be the number of operations in the expression A. Furthermore, let $\bar{A}$ be the expression one obtains by replacing all operators by their computer approximations. Then,*

$$val(\bar{A}) = val(A)(1 + u)^{\pm op(A)}. \tag{12}$$

Moreover, this bound is tight, unless the expression contains additions.

PROOF. The theorem is obviously true for expressions containing no operators. We now do complete induction. To do this, we assume that $A = A_1 \circ A_2$, with $\bar{A}_i = A_i(1 + u)^{\pm \operatorname{op}(A_i)}$, $i = 1, 2$. Because of (9), $\bar{A}_1 \circ \bar{A}_2$ becomes $A_1 \circ A_2(1 + u)^{\pm (\operatorname{op}(A_1) + \operatorname{op}(A_2))}$, and because of (2), the error of $A_1 \hat{\circ} A_2$ is $(1 + u)^{\pm (\operatorname{op}(A_1) + \operatorname{op}(A_2) + 1)}$. Since the exponent of this expression is equal to $\operatorname{op}(A)$, the inductive case is established. It is easy to see that the bound is tight except for additions. This establishes the theorem. □

Theorems 1 and 2 can be combined to yield

$$\overline{\operatorname{val}}(\bar{A}) = \operatorname{val}(\bar{A}) \prod_{i=1}^{\infty} e_i^{\pm r_i} = \operatorname{val}(A)(1 + u)^{\pm \operatorname{op}(A)} \prod_{i=1}^{\infty} e_i^{\pm r_i}.$$

In the case where there is only one type of operator, say, $\circ$, Theorem 2 specializes to

$$\bar{x}_1 \circ \bar{x}_2 \circ \cdots \circ \bar{x}_n = (x_1 \circ x_2 \circ \cdots \circ x_n)(1 + u)^{\pm n - 1}. \tag{13}$$

In the case of additions, Theorem 2 allows for tighter bounds than the ones given by (13). One has [Wilkinson 1963]

$$\begin{aligned} x_1 \hat{+} x_2 \hat{+} \cdots \hat{+} x_n &= x_1(1+u)^{\pm(n-1)} + x_2(1+u)^{\pm(n-1)} + x_2(1+u)^{\pm(n-2)} + \cdots \\ &\quad + x_n(1-u)^{\pm 1} \\ &= \sum_{i=1}^{n} x_i(1+u)^{\pm(n-i+1)}. \end{aligned} \tag{14}$$

As before, the priority of operators decreases from left to right. Note that the last line is less tight than the previous one. According to Wilkinson, this equation suggests that one should add the smallest terms first. In the case of the nonnegative reals, this is always true. Otherwise, there are exceptions to this rule [Grassmann 1989; Wilkinson 1963].

In computer programs, one does not normally use explicit formulas. Specifically, if a formula contains the same subexpression several times, one would tend to evaluate it only once. For instance, the expression

$$(x + y) \times (x + y)$$

would be programmed as

$$z = x + y,$$
$$\text{result} = z * z.$$

Of course, $\text{op}((x + y) \times (x + y)) = 3$, even though the program has only two operations. Hence, in order to apply Theorem 2, one has to count the operations needed, given that the program has been expanded into a closed form formula. In some cases, the operations count of these expansions is huge, and the resulting error bounds are poor. For instance, state reduction is similar to Gaussian elimination. If a program doing state reduction is converted into explicit expressions, one essentially obtains Cramer's rule. Since the number of operations in Cramer's rule is huge, the bound given by Theorem 2 is poor, and more elaborate methods are required (see, e.g., Stewart and Zhang [1991] or Bunch [1987]). On the other hand, in the case of calculating the probabilities of the Poisson distribution, Theorem 2 gives quite satisfactory results.

In algorithms with many iterations, the short-run analysis should be complemented by a long-run analysis. Some notation is needed to define what we mean. Clearly, the state of the system at the end of an iteration is a function of the state at the start. Hence, if S is the state at the beginning of the iteration and if f is the function implemented by one iteration, then the state at the end of the iteration is $f(S)$. Here, S includes all variables that affect the iteration. The state after n iterations is, of course, $f^n(S)$, where f^n is the n-fold functional composition of f with itself. Normally, we are interested in a particular measure, such as an equilibrium probability, and this measure can be interpreted as a function g, which maps a state into the set of reals. After iteration n, this measure is obviously $g \cdot f^n(S)$, where $\cdot$

denotes functional composition. In many cases of interest, $g \cdot f^n(S)$ becomes independent of the initial state S; that is,

$$\lim_{n \to \infty} g \cdot f^n(S) = \lim_{n \to \infty} g \cdot f^n(\hat{S}),$$

where $\hat{S}$ is an arbitrary state. This means that any error made in regard to the initial state S becomes irrelevant after a sufficient number of iterations. We express this by saying that the error is *transient* in respect to g. Errors that are not transient are called *persistent*. Note that transience and persistance are based on *absolute*, as opposed to relative, errors. If the result is bounded to be greater than some positive value, then the fact that an error is transient means that its relative error also goes to zero. However, if the measure $g \cdot f^n(S)$ converges to zero as $n \to \infty$, nothing can be said about the behavior of the relative error.

3. DISCRETE-TIME MARKOV CHAINS

In this section we assume that we are given the transition probabilities p_{ij}, $i,j = 1, 2, \ldots, N$, of an N-state Markov chain, together with the initial probabilities π_i^0, $i = 1, 2, \ldots, N$, and the problem is to determine the transient probabilities π_j^n, $j = 1, 2, \ldots N$. Assume that the π_i^n are calculated recursively according to the following formula:

$$\pi_j^n = \sum_{i=1}^{N} \pi_i^{n-1} p_{ij}.$$

Let $\bar{\pi}_j^n$ be the value found after replacing the operators by their computer equivalents. We now want to determine values e_n such that

$$\bar{\pi}_j^n = \pi_j^n e_n^{\pm 1}.$$

By Eqs. (2), (11), and (13),

$$\bar{\pi}_1^{n-1} \hat{\times} p_{1j} \hat{+} \bar{\pi}_2^{n-1} \hat{\times} p_{2j} \hat{+} \cdots \hat{+} \bar{\pi}_N^{n-1} \hat{\times} p_{Nj} = \sum_{i=1}^{N} \pi_i^{n-1} p_{ij} e_{n-1}^{\pm 1} (1+u)^{\pm N}.$$

Thus, the error increases by the factor $(1+u)^N$ in each iteration. Hence,

$$e_n = e_0 (1+u)^{\pm nN}.$$

Here, e_0 is the error of π_i^0. If the initial values are accurate, $e_0 = 1$. It may be the case that the p_{ij} are also afflicted by errors. To account for this, let

$$\bar{p}_{ij} = p_{ij} \gamma^{\pm 1}.$$

In this case, one obtains

$$\bar{\pi}_j^n = \pi_j^n (1+u)^{\pm nN} \gamma^{\pm n}. \tag{15}$$

If the matrix in question is a sparse matrix, the above result can be improved. In particular, if there are never more than v entries per column, the error is $(1+u)^{nv}\gamma^n$, rather than $(1+u)^{nN}\gamma^n$.

In ergodic Markov chains, the equilibrium probabilities are independent of the initial probabilities. This implies that the errors in π_i^n are transient in respect to the measure $S_m = \pi_j^m / \Sigma_k \pi_k^m$. Note, however, that the relative errors in respect to the nonnormalized expression π_j^m are persistent because an error in one of the π_j^n will cause the sum of all the π_k^n to deviate from unity, and this deviation will not disappear.

Even though errors in π_i^n are transient only after normalization, normalization does not, in general, reduce the relative errors committed. Indeed, normalization only decreases the relative error if $\pi_j^n > 0.5$. To show this, we derive an upper bound for $\overline{\pi}_i^n / \Sigma_k \overline{\pi}_k^n$. Such an upper bound is obtained by assuming that $\overline{\pi}_i^n$ is at its upper bound, while all other $\overline{\pi}_j^n$ are at their lower bound. To do this, let $h_n = e_n - 1$. Since $e_n \geq 1$, $h_n \geq 0$. Hence,

$$\begin{aligned}\frac{\overline{\pi}_i^n}{\overline{\pi}_i^n + \Sigma_{j \neq i} \overline{\pi}_j^n} &\leq \frac{\pi_i^n (1 + h_n)}{\pi_i^n (1 + h_n) + \Sigma_{j \neq i} \pi_j^n / (1 + h_n)} \\ &= \frac{\pi_i^n (1 + 2h_n + h_n^2)}{\pi_i^n (1 + 2h_n + h_n^2) + 1 - \pi_i^n} \\ &= \pi_i^n (1 + 2h_n + h_n^2) / \left(1 + \pi_i^n (2h_n + h_n^2)\right).\end{aligned}$$

This bound has to be compared to $e_n = 1 + h_n$, the error bound for π_i^n. Since h_n is small, all terms h_n^2 can be dropped, and one has to compare

$$\frac{1 + 2h_n}{1 + 2\pi_i^n h_n}$$

with $1 + h_n$. For small h_n, the above expression is approximately equal to $1 + 2h_n - 2\pi_i^n h_n$, and it is easily verified that this expression is smaller than $1 + h_n$ iff $\pi_i^n > 0.5$. If relative errors are to be minimized, one should normalize as late as possible. In fact, one should normalize only the final results.

4. JENSEN'S METHOD

Consider a continuous-time Markov chain with n states and with transition rates a_{ij}, $i \neq j$, $i, j = 1, 2, \ldots, N$. The initial probabilities $\pi_i(0)$ are given, and the problem is to find the $\pi_j(t)$, $t > 0$, $j = 1, 2, \ldots, N$, which are the probabilities of being in state j at time t. The following algorithm, due to Jensen [1953], Grassmann [1977], and Gross and Miller [1984] yields the $\pi_j(t)$.

1. $a_i = \Sigma_{j=1, i \neq j}^{N} a_{ij}$
2. $f = \max_i a_i$
3. $a_{ii} = f - a_i$
4. $p_{ij} = a_{ij}/f$, $i, j = 1, 2, \ldots, N$
5. $\pi_i^0 = \pi_i(0)$, $i = 1, 2, \ldots, N$
6. $\pi_j^{n+1} = \Sigma_{i=1}^{N} \pi_i^n p_{ij}$, $n = 0, 1, 2, \ldots$
7. $q = f \times t$
8. $\pi_j(t) = \Sigma_{n=0}^{\infty} \pi_j^n p(n; q)$

Here, $p(n;q) = e^{-q}q^n/n!$ is the Poisson distribution, which is calculated as $p(n;q) = p(n-1;q) \times q/n$, $n = 1, 2, \ldots$, starting with $p(0;q) = e^{-q}$. We now assume that the a_{ij} are integers, which implies that a_i, f, and a_{ii} are accurate. Consequently, $\bar{p}_{ij} = p_{ij}(1+u)^{\pm 1}$. We can thus use (15) with $\gamma = 1 + u$. Hence,

$$\bar{\pi}_j^n = \pi_j^n(1+u)^{\pm(N+1)n}. \tag{16}$$

Next, we consider the error of $p(n;q)$. For simplicity, we assume that t is an integer, which means that q is an integer as well and, therefore, accurate. Since $p(0;q) = \exp(-q)$, the accuracy of $p(0;q)$ is equal to the accuracy of the exponential function provided by the computer. Here, we assume that the approximation $\bar{p}(0;q)$ equals $p(0,q)r_0^{\pm 1}$. It is now easily verified that (see also Fox and Glynn [1988])

$$\bar{p}(n;q) = p(n;q)\left(r_0(1+u)^{2n}\right)^{\pm 1}. \tag{17}$$

Define

$$s_n = \pi_j^n \times p(n;q).$$

By using (16), (17), and (2), we get

$$\bar{s}_n = s_n(1+u)^{\pm(nN+3n+1)}r_0^{\pm 1}.$$

Clearly,

$$\pi_j(t) = \sum_{n=0}^{\infty} s_n.$$

We now define

$$\pi_j^m(t) = \sum_{n=0}^{m} s_n.$$

If we do the summation in reverse order, starting with s_m, (14) yields

$$\bar{\pi}_j^m(t) = \sum_{n=0}^{m} \bar{s}_n(1+u)^{\pm(n+1)} = \sum_{n=0}^{m} \bar{s}_n(1+u)^{\pm(n+1)}.$$

We can now let m go to infinity to obtain

$$\bar{\pi}_j(t) = \left(r_0(1+u)^2\right)^{\pm 1} \sum_{n=0}^{\infty} \pi_j^n p(n;q)(1+u)^{\pm n(N+4)}. \tag{18}$$

This use of (14) allows one to deal with infinite sums, provided the summation is done in reverse order. Of course, a sum can never be formed with an infinite number of terms. However, no matter how many terms one uses, (18) provides a bound. The addition having to be done in descending order of the subscript is not as crucial as it seems. If the sum is symmetric, the summation generates the same rounding error, no matter whether one forms the sum starting with the first or with the last term. Since the Poisson distribution becomes symmetric as q increases, the s_n are also approximately symmetric, and (18) is a good approximation, even when starting with s_0.

Now determine the upper and lower bound of (18) separately. To determine the upper bound, let

$$\delta = (1 + u)^{N+4}.$$

With this definition,

$$\begin{aligned}
&\left(r_0(1+u)^2\right) \sum_{n=0}^{\infty} \pi_j^n p(n;q)(1+u)^{n(N+4)} \\
&\quad = \left(r_0(1+u)^2\right) \sum_{n=0}^{\infty} \frac{e^{-q}(q\delta)^n}{n!} \pi_j^n \\
&\quad = r_0(1+u)^2 e^{-q} e^{q\delta} \sum_{n=0}^{\infty} \frac{e^{-q\delta}(q\delta)^n}{n!} \pi_j^n \\
&\quad = r_0(1+u)^2 e^{q(\delta-1)} \pi_j(t\delta).
\end{aligned}$$

The lower bound can be obtained in a similar fashion, as

$$\frac{1}{r_0(1+u)^2} e^{q(1/\delta-1)} \pi(t/\delta).$$

Consequently,

$$\frac{1}{r_0(1+u)^2} e^{q(1/\delta-1)} \pi_j(t/\delta) \le \bar{\pi}_j(t) \le r_0(1+u)^2 e^{q(\delta-1)} \pi_j(t\delta). \tag{19}$$

The precision of the results thus depends on δ, or on $(1 + u)^{N+4}$. If the matrix in question is sparse, the exponent $N + 4$ can be reduced. In particular, if each column of the transition matrix contains at most v elements, one can replace N by v. Moreover, the effect of rounding seems to be stronger if $\pi_j(t)$ changes strongly with t.

5. STATE REDUCTION

A number of people have used the state reduction algorithm, or, as it is also known, the GTH algorithm [Grassman et al. 1985; Heyman 1987; Heyman and Reeves 1989; Kohlas, 1986; Stewart 1993; O'Cinneide 1993]. An algorithm closely related to the state reduction algorithm was recently analyzed by Stewart and Zhang [1991]. In the state reduction algorithm, we consider a discrete-time Markov chain with states $1, 2, \ldots, N$, and the problem is to find the equilibrium probabilities π_i, given the transition probabilities p_{ij}, $i, j = 1, 2, \ldots, N$. We also assume that the chain has one recurrent class and that there are no transient states. This involves solving the steady-state equation, given as

$$\pi_j = \sum_{i=1}^{N} \pi_i p_{ij}, \qquad j = 1, 2, \ldots, N. \tag{20}$$

As usual, one has to require that the sum of all π_j is unity. From (20), eliminate $\pi_N, \pi_{N-1}, \ldots, \pi_{n+1}$, and use the symbols p_{ij}^n to denote the coefficients obtained after the elimination of these variables. Because of this definition,

$$\pi_j = \sum_{i=1}^{n} \pi_i p_{ij}^n, \qquad j = 1, 2, \ldots, N. \tag{21}$$

It is easy to verify that the p_{ij}^n can be calculated recursively by the following formula:

$$p_{ij}^{n-1} = p_{ij}^n + \frac{p_{in}^n p_{nj}^n}{1 - p_{nn}^n}. \tag{22}$$

It is known that the p_{ij}^n define a Markov chain with the states 1 to n [Grassmann et al. 1985; Kohlas 1986]. In fact, the p_{ij}^n describe an embedded Markov chain. It is the Markov chain obtained by using visits to states $i \leq n$ as regeneration points. This implies that $1 - p_{nn}^n = \sum_{j=1}^{n-1} p_{nj}^n$. In state reduction, we now rewrite (22) as

$$p_{ij}^{n-1} = p_{ij}^n + \frac{p_{in}^n p_{nj}^n}{\sum_{j=1}^{n-1} p_{in}^n}, \tag{23}$$

After all the p_{ij}^i and p_{ij}^j are calculated, one can use (21) to perform the normal backsubstitution step. Since the analysis of the backsubstitution is similar to the one presented earlier for discrete-time Markov chains, we do not discuss it here. We merely note that in order to simplify the backsubstitution step, and also the evaluations of (23), one normally defines new variables $a_{nj} = p_{nj}^n / \sum_{j=1}^{n-1} p_{nj}^n$.

We now perform the error analysis of (23). A similar analysis is given in O'Cinneide [1993]. Let $\bar{p}_{ij}^n = p_{ij}^n e_n^{\pm 1}$, where the e_n are to be determined. By counting the operations in (23), one finds that

$$e_{n-1} = e_n^3 (1 + u)^{n+1}.$$

If the p_{ij} are accurate, $e_N = 1$, and one finds that

$$e_{N-1} = (1 + u)^{N+1},$$
$$e_{N-2} = (1 + u)^{3(N+1)} (1 + u)^N = (1 + u)^{4N+3},$$
$$e_{N-3} = (1 + u)^{12N+9} (1 + u)^{N-1} = (1 + u)^{13N+8},$$

and so on. Notice that the term $(1 + u)^{n+1}$, which represents the error generation of iteration n, is small compared to the term e_n^3, which accounts for the error propagation. If one ignores the error generation after the first iteration, one finds that

$$e_{N-m} \geq (1 + u)^{(N+1) \times 3^m}. \tag{24}$$

These error bounds increase rapidly. Still, they are adequate for small matrices. However, as N increases, these bounds become unacceptably large.

In fact, if e_n is calculated according to (24), one can construct matrices such that $p_{ij}^n e_n$ exceeds 1 for some values n, i, and j. One merely has to choose large enough values for N and m. Note that the approximate bounds given by (24) are entirely due to error propagation. To find better bounds, one therefore must concentrate on the propagation, rather than on the generation of errors. Traditionally, this has been accomplished by backward analysis and perturbation theory. For further details on this in the context of state reduction; see Stewart [1993] and Stewart and Zhang [1990].

We now show that if we change the probabilities p_{ij} in the original matrix such that the row sums change by at most ϵ then all p_{ij}^n must remain between 0 and $1 + \epsilon$. To see this, note that (23) leaves the sums of the p_{ij}^n unchanged.

$$\sum_{j=1}^{n-1} p_{ij}^{n-1} = \sum_{j=1}^{n-1} p_{ij}^n + \frac{p_{in}^n p_{nj}^n}{\sum_{j=1}^{n-1} p_{nj}^n}$$
$$= \sum_{j=1}^{n-1} p_{ij}^n + p_{in}^n = \sum_{j=1}^{n} p_{ij}^n.$$

Since the p_{ij}^n cannot become negative, this implies that

$$0 \le p_{ij}^n \le 1 + \epsilon.$$

This result also holds if the matrix we perturb is not $[p_{ij}]$, but is $[p_{ij}^n]$. In particular, after the first iteration, $[p_{ij}^{N-1}]$ is perturbed in such a way that the row sums are at most $(1 + u)^{N+1}$. Hence, no error propagation from that point onward can ever result in a value $\bar{p}_{ij}^n > (1 + u)^{N+1}$. Consequently, the error bounds given in (24), which increase exponentially, cannot be tight for large m.

In many cases, the errors seem to be transient. Some experimental results reported in the literature can be interpreted in this way [Grassmann et al. 1985; Heyman 1987; Heyman and Reeves 1989], and for special types of matrices, this can even be proved mathematically. In particular, Grassmann and Heyman considered transition matrices that are banded, irreducible, and positive recurrent. Moreover, for all i, j between a certain lower limit c and an upper limit $N - d$,

$$p_{ij} = p_{i-k\,j-k}, \qquad i, j = c, c + 1, \ldots, N - d. \tag{25}$$

In this case, N can be increased as much as needed to investigate what happens to the error bound of p_{ij}^{N-m} as m gets large. It was shown that in this case, the probabilities p_{ij}^{N-m} become independent of the probabilities p_{rs}, $r, s > N - d$. In other words, if m and N are large enough, then any error committed for $r, s > N - d$ is transient. Numerical experimentation showed that the errors propagated decay geometrically with m. Now consider the matrix $[p_{ij}^n]$. Since the original matrix is banded, there is some value g such that $p_{ij} = 0$ for $|j - i| > g$, which implies that $p_{ij}^n = p_{ij}$ for $i, j < n - g$. As a consequence, the matrix $[p_{ij}^n]$ satisfies the conditions given by (25), which implies that the errors commited in the calculation of p_{ij}^n are also

Table I. Comparison between Gauss and State Reduction

ϵ	Gauss	State reduction	Condition number
0.01	+0.00000095	0.00000000	211
0.001	+0.00001287	0.00000000	2,011
0.0001	+0.00014406	0.00000000	20,011
0.00001	+0.00135624	0.00000000	200,011

transient. In fact, if the decay is geometric, and numerical experiments indicate that this is true, then all errors together will remain within constants bounds, as is easily verified.

Finally, we investigate the consequences of the error bounds being relative bounds. Norms, and, with them, condition numbers, are based on absolute rather than relative changes. Hence, they may not be appropriate in the context of state reduction. To test this, we applied both state reduction and Gaussian elimination to solve the following problem:

$$\begin{pmatrix} 0.4 & 0.6 & & \\ 0.6 & 0.4-\epsilon & \epsilon & \\ & \epsilon & 0.5-\epsilon & 0.5 \\ & & 0.5 & 0.5 \end{pmatrix}.$$

Since the matrix is doubly stochastic, the solution of the problem is always $\pi_i = 1/4$, $i = 1, 2, 3, 4$, no matter what value ϵ happens to have. We then solved the problem by both state reduction and normal Gaussian elimination for $\epsilon = 0.01$, 0.001, 0.0001, and 0.00001, and we calculated the condition number. The norm chosen was the infinity norm. All calculations were done in single precision on a Sun 3/50 workstation. The probabilities were left unscaled, which means that the correct results were $\pi_i = 1$, $i = 1, 2, 3, 4$. Table I gives the differences between π_4 and 1, and also shows the condition number. It can be clearly seen that in Gaussian elimination the precision deteriorates as the condition number increases. No such effect is noticeable in state reduction, which always gives the correct result with a precision of eight digits.

REFERENCES

ANDREWS, P. B. 1986. *An Introduction to Mathematical Logic and Type Theory.* Academic Press, Orlando, Fla.

BUNCH, J. F. 1987. The weak and strong stability of algorithms in numerical linear algebra. *Linear Algebra Appl. 88–89,* 46–66.

DEBOOR, C., AND PINKUS, A. 1977. Backward error analysis for totally positive linear systems. *Numer. Math. 27,* 4, 485–490.

FOX, B. L., AND GLYNN, P. W. 1988. Computing Poisson probabilities. *Commun. ACM, 31,* 4 (Apr.) 440–445.

GRASSMANN, W. K. 1989. A probabilistic analysis of rounding errors of floating point numbers. *Congr. Numeratium 68,* 171–182.

GRASSMANN, W. K. 1985. The factorization of queueing equations and their interpretation. *J. Opl. Res. Soc. 36,* 11 (Nov.) 1041–1051.

GRASSMANN, W. K. 1977. Transient solutions in Markovian queueing systems. *Comput Oper. Res. 4*, 47–53.

GRASSMANN, W. K., AND HEYMAN, D. P. 1993. Computation of steady-state probabilities for infinite-state Markov chains with repeating rows *ORSA J. Comput. 5*, 3 (Summer).

GRASSMANN, W. K., TAKSAR, M. I., AND HEYMAN, D. P. 1985. Regenerative analysis and steady state distributions for Markov chains. *Oper. Res. 33*, 5 (Sept.–Oct.), 1107–1116.

GROSS, D., AND MILLER, D. R. 1984. The randomization technique as a modelling tool and solution procedure for transient Markov processes. *Oper Res. 32*, 2 (Mar.–Apr.), 343–361

HEYMAN, D. P. 1987. Further comparisons of direct methods for computing stationary distributions of Markov chains. *SIAM J. Algebraic Discrete Methods 8*, 2 (Apr.), 226–232.

HEYMAN, D. P., AND REEVES, A. 1989. Numerical solutions arising in Markov chain models *ORSA J. Comput 1*, 1 (Winter), 52–60.

HIGHAM, N. J. 1990. Bounding the error in Gaussian elimination for tridiagonal systems *SIAM J. Matrix Anal Appl.*

JENSEN, A. 1953. Markov chains as an aid in the study of Markoff processes. *Scand. Actuar. 36*, 87–91.

KOHLAS, J. 1986 Numerical computation for mean passage times and absorption probabilities in Markov and semi-Markov models. *Z. Oper. Res. 30*, A197–A207.

KULISCH, U. L., AND MIRANKER, W. L. 1981. *Computer Arithmetic in Theory and Practice.* Academic Press, New York.

KEMENY, J. G., SNELL, J. L., AND KNAPP, A. W 1966. *Denumerable Markov Chains.* Van Nostrand, Princeton, N.J.

O'CINNEIDE, 1993. Entrywise perturbation theory and error analysis for Markov chains. *Numer. Math. 5*, 1 (May).

STEWART, G. W. 1993. Gaussian elimination, perturbation theory and Markov chains. In *Linear Algebra, Markov Chains, and Queueing Models.* C. Meyer and R. J. Plemmons, Eds. Springer-Verlag, New York, 59–69

STEWART, G. W., AND ZHANG, G. 1990. On a direct method for the solution of nearly uncoupled Markov chains. *Numer. Math. 59*, 1 (Apr.), 1–11.

WILKINSON, J. H. 1963. *Rounding Errors in Algebraic Processes.* Prentice-Hall, Englewood Cliffs, NJ.

Received April 1991; revised and accepted February 1993