ANZIAM J. 45(2004), 457–475

A PROBABILISTIC ALGORITHM FOR FINDING THE RATE MATRIX OF A BLOCK-$GI/M/1$ MARKOV CHAIN

EMMA HUNT[1] [2]

(Received 15 July, 2002)

Abstract

An efficient probabilistic algorithm is presented for the determination of the rate matrix of a block-$GI/M/1$ Markov chain. Recurrence of the chain is not assumed.

1. Introduction

Following the work of Neuts, consolidated in his book [16], there has been considerable interest in the structure and application of block-$GI/M/1$ Markov chains. Such a chain is customarily taken as one whose one-step transition matrix may be partitioned as

$$\widehat{Q} = \begin{bmatrix} D_1 & C_0 & 0 & 0 & \dots \\ D_2 & C_1 & C_0 & 0 & \dots \\ D_3 & C_2 & C_1 & C_0 & \dots \\ D_4 & C_3 & C_2 & C_1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{1.1}$$

where each matrix C_m is $k \times k$, the matrix $C := \sum_{m\geq 0} C_m$ is stochastic and irreducible and the block D_2 is nonzero. The sets of states corresponding to successive blocks are envisaged as constituting *levels*, and the states within each block *phases*.

Neuts remarked on a number of special cases that appear in the literature, particularly various elaborations of the basic $GI/M/1$ queueing model. With such

[1] Intelligence, Surveillance and Reconnaissance Division, DSTO Edinburgh, PO Box 1500, Edinburgh SA 5111, Australia; e-mail: emma.hunt@dsto.defence.gov.au.

[2] School of Applied Mathematics, The University of Adelaide, Adelaide SA 5005, Australia; e-mail: ehunt@maths.adelaide.edu.au.

applications, the primary question is again the determination of the invariant probability measure in the case when the chain is positive recurrent. This may be determined through the use of an auxiliary parameter, the rate matrix R.

Denote by $R_{j,\nu}$ the expected number of visits made by the process to state $(i+1, \nu)$ before the first revisit to level $i > 0$, given it begins in the state (i, j). Since the chain is skip-free from below, we may argue by homogeneity to the right that $R_{j,\nu}$ is well-defined and independent of i and is the minimal nonnegative solution to

$$R = C(R) := \sum_{i=0}^{\infty} R^i C_i. \tag{1.2}$$

This result was given by Neuts [16, Lemma 1.2.3] for the positive recurrent case, but careful examination reveals that the proof does not actually depend on positive recurrence or indeed even recurrence.

As before, denote the invariant probability measure in the positive recurrent case by $\pi = (\pi_0, \pi_1, \dots)$, with each π_i a k-vector. Neuts [16, Theorem 1.2.1] has shown that

- $\pi_{i+1} = \pi_i R$ for $i \geq 0$;
- the matrix $D(R) := \sum_{j=0}^{\infty} R^j D_{j+1}$ is stochastic and π_0 an invariant measure on it;
- the matrix $I - R$ is invertible and π_0 is normalised by $\pi_0 (I - R)^{-1} e = 1$.

Knowing R is thus central for the determination of the invariant probability measure of a positive recurrent block-$GI/M/1$ Markov chain. Neuts has provided an algorithm [16, p. 13] based on (1.2) for the evaluation of R, but this can converge very slowly. Our present aim is to provide a more efficient algorithm, which we shall call H*. The notation is chosen to provide consistency in a subsequent companion article where we demonstrate a natural duality, manifested by use of *, with Algorithm H* dual to Algorithm H, a procedure presented in [11]. The role of Algorithm H in evaluating the fundamental matrix G in a block-$M/G/1$ Markov chain (see, for example, [6, 17]) is comparable to that of Algorithm H* for determining R in the present article.

In fact we shall see via the duality how to construct some further efficient algorithms for calculating R and G in the case of a quasi-birth-and-death process. We shall also find relations between Algorithm H*, the logarithmic reduction algorithm of Latouche and Ramaswami [12] and the cyclic reduction algorithms of Bini and Meini (see, for example, [7, 8, 9] and [15]). Further, duality provides results on convergence rates. Accordingly we defer for the present comparison of Algorithm H* with those other algorithms relating to the QBD case.

In the following two sections we set up the probabilistic ideas involved in our construction. These ideas are drawn together for a succinct formulation of Algorithm H* in Section 4. The remainder of the article is concerned with five extended numerical experiments comparing Algorithm H* with algorithms other than those that will be considered in connection with duality. These are invariant subspace methods, introduced in Section 5 and applicable when $C(z)$ is rational and the chain positive recurrent, and the Neuts method.

2. Preliminaries

As with finding G in a structured $M/G/1$ Markov chain, it proves convenient to label the levels of the chain $\mathscr{C}$ as $-1, 0, 1, 2, \ldots,$ so that $\mathscr{C}$ is homogeneous in the one-step transition probabilities into all nonnegative levels. In evaluating R we are concerned with the numbers of visits of $\mathscr{C}$ to states of level 0 from initial level -1, with -1 as a taboo level. We may thus, without loss of generality, replace $\mathscr{C}$ with a chain $\widehat{\mathscr{C}}$ with levels $-1, 0, 1, 2, \ldots$ and structured one-step transition matrix

$$\widehat{P^*} = \begin{bmatrix} 0 & C_0 & 0 & 0 & \cdots \\ 0 & C_1 & C_0 & 0 & \cdots \\ 0 & C_2 & C_1 & C_0 & \cdots \\ 0 & C_3 & C_2 & C_1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

Our analysis will be mostly in terms of the (substochastic) subchain $\mathscr{C}_0$ with levels $0, 1, 2, \ldots$ and structured one-step transition matrix

$$P^{*(0)} = \begin{bmatrix} C_1 & C_0 & 0 & 0 & \cdots \\ C_2 & C_1 & C_0 & 0 & \cdots \\ C_3 & C_2 & C_1 & C_0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

The assumption that $\mathscr{C}$ is irreducible entails that every state in a nonnegative-labelled level has access to level -1. Hence all the states of $\mathscr{C}_0$ are transient or ephemeral.

For $t = 0, 1, 2, \ldots,$ denote by X_t, Y_t respectively the state and level of $\mathscr{C}_0$ at time t. For $r, s \in \mathscr{K} := \{1, 2, \ldots, k\}$ we define

$$U_{r,s} := P\left(\bigcup_{t>0} \{X_t = (0, s),\ Y_u > 0\ (0 < u < t)\} \,|\, X_0 = (0, r)\right).$$

Thus $U_{r,s}$ is the probability that, starting in $(0, r)$, the process $\mathscr{C}_0$ revisits level 0 at some subsequent time and does so with first entry into state $(0, s)$.

The matrix $U := (U_{r,s})$ may be regarded as the one-step transition matrix of a Markov chain $\mathscr{U}$ on the finite state space $\mathscr{K}$. The chain $\mathscr{U}$ is a censoring of $\mathscr{C}_0$. No state of $\mathscr{U}$ is recurrent, for if $r \in \mathscr{K}$ were recurrent, the state $(0, r)$ in $\mathscr{C}_0$ would be recurrent, a contradiction. Since no state of $\mathscr{U}$ is recurrent, $I - U$ is invertible and

$$\sum_{i=0}^{\infty} U^i = (I - U)^{-1}.$$

The matrix U is also strictly substochastic.

By elementary Markov chain theory, the (m, s) entry in $(I - U)^{-1}$ gives the expected number of visits made by $\mathscr{C}_0$ to $(0, s)$, given $\mathscr{C}_0$ begins in $(0, m)$. In $\widehat{\mathscr{C}}$, any path whose probability contributes to $R_{r,s}$ begins in $(-1, r)$, moves immediately to some state $(0, m)$ and then makes one or more visits to state $(0, s)$. Allowing for all possible choices of m, we derive that

$$R_{r,s} = \sum_{m \in \mathscr{K}} (C_0)_{r,m} \left(\sum_{i=0}^{\infty} U^i \right)_{m,s},$$

so that

$$R = C_0 \sum_{n=0}^{\infty} U^n = C_0 (I - U)^{-1}. \tag{2.1}$$

We proceed to determine R via U.

For $\ell \geq 0$, we write $U(\ell)$ for the matrix whose entries are given by

$$U(\ell)_{r,s} := P\left(\bigcup_{t>0} \{X_t = (0, s),\ 0 < Y_u < \ell\ (0 < u < t)\} \,|\, X_0 = (0, r) \right)$$

for $r, s \in \mathscr{K}$. Thus $U(\ell)$ corresponds to U when the trajectories in $\mathscr{C}_0$ are further restricted not to reach level ℓ or higher before a first return to level 0.

We may argue as above that $I - U(\ell)$ is invertible and

$$I - U(\ell) = \sum_{i=0}^{\infty} (U(\ell))^i.$$

Further, since U is finite, $U(\ell) \uparrow U$ and $[I - U(\ell)]^{-1} \uparrow [I - U]^{-1}$ as $\ell \to \infty$.

The probabilistic construction we are about to detail involves the exact algorithmic determination (to machine precision) of $U(\ell)$ for ℓ of the form 2^N with N a nonnegative integer. This leads to an approximation

$$T_N^* := C_0 \left[I - U(2^{N+1}) \right]^{-1}$$

for R. We have

$$T_N^* \uparrow R \quad \text{as} \quad N \to \infty.$$

The matrix T_N^* may be interpreted as the contribution to R from those trajectories from level -1 to level 0 in $\widehat{\mathscr{C}}$ that are restricted to pass through only levels below 2^{N+1}.

3. Probabilistic construction

We construct a sequence $(\mathscr{C}_j)_{j\geq 0}$ of censored processes, each of which has as its levels the nonnegative integers. For $j \geq 1$, the levels $0, 1, 2, \ldots$ of $\mathscr{C}_j$ are respectively the levels $0, 2, 4, \ldots$ of $\mathscr{C}_{j-1}$, that is, $\mathscr{C}_j$ is $\mathscr{C}_{j-1}$ censored to be observed in even-labelled levels only. Thus $\mathscr{C}_j$ is a process that has been censored j times. By the homogeneity of one-step transitions in $\mathscr{C}$, a straightforward induction gives that $\mathscr{C}_j$ has a structured one-step transition matrix of the form

$$P^{*(j)} = \begin{bmatrix} D_1^{(j)} & C_0^{(j)} & 0 & 0 & \cdots \\ D_2^{(j)} & C_1^{(j)} & C_0^{(j)} & 0 & \cdots \\ D_3^{(j)} & C_2^{(j)} & C_1^{(j)} & C_0^{(j)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

that is, each chain $\mathscr{C}_j$ is of structured $GI/M/1$ type. We have

$$D_n^{(0)} = C_n \quad (n \geq 1) \quad \text{and} \quad C_n^{(0)} = C_n \quad (n \geq 0).$$

We shall construct below the block entries of $P^{*(j+1)}$ in terms of those of $P^{*(j)}$.

In the previous section we saw that $\mathscr{C}_0$ contains no recurrent states, so the same must be true also for the censorings $\mathscr{C}_1, \mathscr{C}_2, \ldots$. Thus the substochastic matrices $D_1^{(j)}$, $C_1^{(j)}$ formed by restricting $\mathscr{C}_j$ to levels 0 and 1 respectively thus also contain no recurrent states. Hence $I - D_1^{(j)}$ and $I - C_1^{(j)}$ are both invertible.

We now consider how to derive the block entries in $P^{*(j+1)}$ from those in $P^{*(j)}$. First we extend our earlier notation and write $X_t^{(j)}$, $Y_t^{(j)}$ respectively for the state and level of $\mathscr{C}_j$ at time $t \in \{0, 1, \ldots\}$. For $n \geq 0$, define the $k \times k$ matrix $\mathbf{L}_n^{(j+1)}$ by

$$\left(\mathbf{L}_n^{(j+1)}\right)_{r,s} := P\left[\bigcup_{t>0}\left\{X_t^{(j)} = (2\ell+1, s),\ Y_t^{(j)} - Y_u^{(j)} \text{ even } (0 < u < t)\right\} \,\middle|\, X_0^{(j)} = (2\ell + 2n, r)\right]$$

for $r, s \in \mathscr{K}$. By the homogeneity of the one-step transition probabilities in $\mathscr{C}_j$ for transitions into positive-labelled levels, the right-hand side is independent of the value of $\ell \geq 0$, justifying its absence from the notation on the left-hand side.

We may express the transitions in $\mathscr{C}_{j+1}$ in terms of those in $\mathscr{C}_j$ and the matrices $\mathbf{L}_n^{(j+1)}$ by an enumeration of possibilities. Suppose $i > 0$. A single-step transition from state $(i - 1 + n, r)$ to (i, s) $(n \geq 0)$ in $\mathscr{C}_{j+1}$ corresponds to a transition from $(2(i - 1 + n), r)$ to $(2i, s)$ in $\mathscr{C}_j$ in one or more steps without passage through any intermediate state in an even-labelled level. For $n > 0$, this can occur in a single step, with probability $(C^{(j)}_{2n-1})_{r,s}$. For a transition involving more than one step, we may condition on the last step. This gives

$$C^{(j+1)}_n = C^{(j)}_{2n-1} + \sum_{m=0}^{n} \mathbf{L}^{(j+1)}_{n-m} C^{(j)}_{2m} \quad (n \geq 1). \tag{3.1}$$

For $n = 0$, the transition always requires more than one step and we have

$$C^{(j+1)}_0 = \mathbf{L}^{(j+1)}_0 C^{(j)}_0. \tag{3.2}$$

Similarly we derive

$$D^{(j+1)}_n = D^{(j)}_{2n-1} + \sum_{m=1}^{n} \mathbf{L}^{(j+1)}_{n-m} D^{(j)}_{2m} \quad (n \geq 1). \tag{3.3}$$

The determination of the matrices $\mathbf{L}^{(j+1)}_n$ proceeds in two stages. For $n \geq 0$, define the $k \times k$ matrix $\mathbf{K}^{(j+1)}_n$ on $\mathscr{C}_j$ by

$$\big(\mathbf{K}^{(j+1)}_n\big)_{r,s} := P\left[\bigcup_{t>0}\{X^{(j)}_t = (2\ell+1, s),\ Y^{(j)}_t - Y^{(j)}_u \text{ even } (0 < u < t)\} \,\middle|\, X^{(j)}_0 = (2\ell+2n+1, r)\right]$$

for $r, s \in \mathscr{K}$. Again the left-hand side is independent of ℓ.

Any path in $\mathscr{C}_j$ contributing to $\mathbf{L}^{(j+1)}_n$ involves an initial step to an odd-labelled level followed by a sequence of steps avoiding even-labelled levels. By conditioning on the first step, we derive

$$\mathbf{L}^{(j+1)}_n = \sum_{m=0}^{n} C^{(j)}_{2(n-m)} \mathbf{K}^{(j+1)}_m \quad (n \geq 0). \tag{3.4}$$

To complete the specification of $P^{*j+1)}$ in terms of $P^{*(j)}$, we need to determine the matrices $\mathbf{K}^{(j+1)}_n$. We have by definition that

$$\left(\mathbf{K}^{(j+1)}_0\right)_{r,s} = P\left[\bigcup_{t>0}\{X^{(j)}_t = (2\ell+1, s),\ Y^{(j)}_t - Y^{(j)}_u \text{ even } (0 < u < t)\} \,\middle|\, X^{(j)}_0 = (2\ell+1, r)\right].$$

Since $\mathscr{C}_j$ is skip-free to the right, trajectories contributing to $\mathbf{K}^{(j+1)}_0$ do not change level and so

$$\mathbf{K}^{(j+1)}_0 = \sum_{i=0}^{\infty} \left(C^{(j)}_1\right)^i = \left(I - C^{(j)}_1\right)^{-1}. \tag{3.5}$$

For $n > 0$, paths contributing to $\mathbf{K}_n^{(j+1)}$ involve at least one change in level and do not visit even-labelled levels. We may condition on the last step involving a change of levels to obtain the recursive relation

$$\mathbf{K}_n^{(j+1)} = \sum_{m=0}^{n-1} \mathbf{K}_m^{(j+1)} C_{2(n-m)+1}^{(j)} \mathbf{K}_0^{(j+1)} \quad (n \geq 1). \tag{3.6}$$

We may also develop a recursion by conditioning on the first jump between levels. This gives the alternative relation

$$\mathbf{K}_n^{(j+1)} := \sum_{m=0}^{n-1} \mathbf{K}_0^{(j+1)} C_{2(n-m)+1}^{(j)} \mathbf{K}_m^{(j+1)} \quad (n \geq 1). \tag{3.7}$$

Since level 1 in $\mathscr{C}_N$ corresponds to level 2^N in $\mathscr{C}_0$, paths in $\mathscr{C}_N$ from $(0, r)$ to $(0, s)$ that stay within level 0 correspond to paths from $(0, r)$ to $(0, s)$ in $\mathscr{C}_0$ that do not reach level 2^N or higher. Hence $(D_1^{(N)})_{r,s} = (U(2^N))_{r,s}$ for $r, s \in \mathscr{K}$, or $D_1^{(N)} = U(2^N)$. Thus the recursive relations connecting the block entries in $P^{*(j+1)}$ to those in $P^{*(j)}$ for $j = 0, 1, \ldots, N-1$ provide the means to determine $U(2^N)$ and so lead to an approximation for R.

4. Algorithm H*

In the last section we considered the sequence $\mathscr{C}_0, \mathscr{C}_1, \ldots, \mathscr{C}_N$ of censored processes. The determination of $D_1^{(N)}$ requires only a finite number of the matrix entries in each $P^{*(j)}$ to be determined. For the purpose of calculating T_N^*, the relevant parts of the construction may be summarised as follows.

The algorithm requires initial input of $C_0, C_1, \ldots, C_{2^N-1}$. First we specify

$$D_n^{(0)} = C_n \quad (n = 1, \ldots, 2^N),$$
$$C_n^{(0)} = C_n \quad (n = 0, 1, \ldots, 2^N - 1).$$

We determine $D_1^{(j)}, D_2^{(j)}, \ldots, D_{2^{N-j}}^{(j)}$ and $C_0^{(j)}, C_1^{(j)}, \ldots, C_{2^{N-j}-1}^{(j)}$ recursively for $j = 1, 2, \ldots, N$ as follows. For obtaining the block matrices in $\mathscr{C}_{j+1}$ from those in $\mathscr{C}_j$, first find the auxiliary quantities

$$\mathbf{K}_0^{(j+1)} = \left[I - C_1^{(j)}\right]^{-1},$$

with

$$\mathbf{K}_n^{(j+1)} = \sum_{m=0}^{n-1} \mathbf{K}_0^{(j+1)} C_{2(n-m)+1}^{(j)} \mathbf{K}_m^{(j+1)}$$

for $n = 1, 2, \ldots, 2^{N-j-1} - 1$ and

$$\mathbf{L}_n^{(j+1)} = \sum_{m=0}^{n} C_{2(n-m)}^{(j)} \mathbf{K}_m^{(j+1)}$$

for $n = 0, 1, \ldots, 2^{N-j-1} - 1$.

Calculate $C_0^{(j+1)} = L_0^{(j+1)} C_0^{(j)}$ and

$$D_n^{(j+1)} = D_{2n-1}^{(j)} + \sum_{m=1}^{n} L_{n-m}^{(j+1)} D_{2m}^{(j)},$$

$$C_n^{(j+1)} = C_{2n-1}^{(j)} + \sum_{m=0}^{n} L_{n-m}^{(j+1)} C_{2m}^{(j)},$$

for $n = 1, 2, \ldots, 2^{N-j-1} - 1$.

The above suffices for the evaluation of $D_1^{(N)}$. The algorithm may be specified as a short MATLAB program.

5. Invariant subspace approaches

A number of invariant subspace techniques have been developed in connection with the solution of the matrix Riccati equation. Akar and Sohraby have adapted two of these for the determination of the rate matrix. For brevity we refer to these simply as TELPACK and Schur factorisation. Both apply only in the positive recurrent case.

5.1. TELPACK TELPACK can be used to determine R when $C(z)$ is rational in a way parallel to its use for the determination of G when $A(z)$ is rational.

Three $GI/M/1$ examples were also provided in the TELPACK package, two of which we consider below in our numerical experiments. The example omitted relates to the case of 1×1 matrices.

5.2. Schur factorisation A second approach for determining the left-invariant subspace of a matrix is the so-called Schur approach. Akar, Oğuz and Sohraby have implemented this for evaluating R in the special case of a QBD. Details are given in [1, 2] and [4].

The approach uses the fact that a real matrix X is orthogonally similar to a quasi-upper triangular matrix [10]. The term “quasi-upper” signifies that the (block) diagonal consists of 2×2 matrices corresponding to complex-conjugate eigenvalues of X and the 1×1 blocks to its real eigenvalues.

Its numerical implementation [14] includes balancing X, casting it into upper-Hessenberg form, obtaining the real Schur form using the double Francis QR iteration

TABLE 1. Experiment 1

Algorithm	**Iterations I**	$\|R - R_I\|_\infty$	**CPU Time (sec.)**
Neuts	268	7.1054e-15	0.010
Schur	-	1.2648e-16	0.050
TELPACK	7	1.9429e-16	0.070
H*	8	0	0.006

and ordering the eigenvalues appropriately using orthogonal transformations. The approach is numerically stable.

In the following section, all outputs designated as being TELPACK (the invariant subspace approach) or Schur (the Schur factorisation method) have been obtained running C programs downloaded from Khosrow Sohraby's home page http://www.cstp.umkc.edu/org/tn/telpack/home.html.

6. Numerical experiments

We now consider some numerical experiments. All code for the Neuts and H* Algorithms has been implemented by us in MATLAB.

No iteration counts are given for the Schur factorisation technique as these are not provided by the TELPACK package.

6.1. Experiment 1. A TELPACK QBD example

Our first example comes from the infinite QBD section of TELPACK (ex-QBD-1). We have chosen it because its simplicity enables us to calculate R exactly, and thus to use as an error measure the supremum norm of the difference between the exact and estimated values of R.

The defining transition matrices for the system are given by

$$C_0 = \begin{bmatrix} 0 & 0 \\ 0 & 0.8 \end{bmatrix}, \quad C_1 = \begin{bmatrix} 0 & 0.1 \\ 0.2 & 0 \end{bmatrix} \quad \text{and} \quad C_2 = \begin{bmatrix} 0.9 & 0 \\ 0 & 0 \end{bmatrix}.$$

The results are displayed in Table 1. The stopping criterion used was the difference between two iterations being less than $\epsilon = 10^{-14}$. We note that this example is a QBD and as such can be expected to favour the Schur factorisation method. The accuracy of all four algorithms considered is comparable, but CPU times are much longer for the Neuts Algorithm, Schur factorisation and TELPACK.

6.2. Experiment 2. An $M/M/1$ queue in a random environment

Our second example is drawn from Latouche [13] and Bini, Latouche and Meini [9]. The process is that of an $M/M/1$ queue in a random environment featuring 8 environmental phases.

TABLE 2. The four sets of service rate values for Experiment 2

Case	μ							
1	2	2	2	2	2	2	2	2
2	1	1	1	1	5	5	1	1
3	0.4	0.4	0.4	0.4	10	2	2	0.4
4	0.2	0.2	0.2	0.2	13	1	1	0.2

The matrix infinitesimal generator of the process is given by

$$\begin{bmatrix} -1 & 1 & \cdot & \cdots & \cdot & \cdot \\ \cdot & -1 & 1 & \cdot & \cdots & \cdot \\ \cdot & \cdot & -1 & \cdots & \cdot & \cdot \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \cdot & \cdot & \cdot & \cdots & -1 & 1 \\ 1 & \cdot & \cdot & \cdots & \cdot & -1 \end{bmatrix}.$$

The process cycles through the 8 phases in order from 1 to 8 and then starts again at phase 1. The process remains in each phase for an interval with exponential distribution and unit mean. Arrival rates are given by the vector

$$\lambda = \rho(0.2, 0.2, 0.2, 0.2, 13, 1, 1, 0.2)^T.$$

We consider four sets of values for the service rates in each phase (see Table 2).

It is noted in [13] that this process may model a situation where the arrival process occasionally experiences a sharp increase during a short period (note that for Cases 2–4 the sharp increase in arrival rate is matched in varying degrees by a sharp increase in service rate). We note that for all four cases the overall arrival rate is $\alpha\lambda = 2\rho$ customers per unit of time, where α is the stationary probability vector for the phase process. The four queues are positive recurrent if and only if $\rho < 1$. This process is continuous-time, but discretisation is simple.

Results are shown in Table 3. The stopping rule used was that the difference between two iterations be satisfied to within 10^{-8}. We note that this experiment is again a QBD system, which can be expected to favour the Schur factorisation method. For this experiment all methods except for the Neuts Algorithm had comparable accuracy. CPU times for the Schur factorisation method were twice as long on average as those for Algorithm H* and CPU times for TELPACK were 8.5 times longer on average. The Neuts Algorithm was the worst performer with respect to both accuracy and CPU time.

6.3. Experiment 3 Our third experiment concerns a QBD problem with 16 × 16 transition matrices $C_0 = C_1 = S$ and $C_2 = S + \delta I$, where S is a matrix with zero

TABLE 3. Results for Experiment 2

Case	Method	**Iterations I**	$\|R_I - C(R_I)\|_\infty$	**CPU Time (sec.)**
1	Neuts	4110	9.9969e-09	5.020
	TELPACK	11	2.4328e-13	0.080
	Schur	-	1.4384e-14	0.030
	H*	12	9.5847e-13	0.010
2	Neuts	3386	9.9925e-09	3.890
	TELPACK	11	2.8538e-13	0.090
	Schur	-	8.4030e-15	0.020
	H*	12	8.8818e-16	0.010
3	Neuts	1871	9.9700e-09	2.080
	TELPACK	10	2.6035e-14	0.080
	Schur	-	2.0067e-14	0.020
	H*	10	4.3280e-12	0.010
4	Neuts	1644	9.9736e-09	1.8000
	TELPACK	10	1.7153e-13	0.090
	Schur	-	2.6645e-14	0.010
	H*	10	4.6629e-14	0.010

TABLE 4. Results for Experiment 3

δ	**Method**	**Iterations I**	$\|R_I - C(R_I)\|_\infty$	**CPU Time (sec.)**
10^{-1}	Neuts	136	6.9597e-13	0.393
	TELPACK	8	2.2413e-15	0.017
	Schur	-	7.0083e-16	0.070
	H*	6	3.4694e-17	0.008
10^{-2}	Neuts	1133	6.6986e-13	3.247
	TELPACK	11	1.1595e-14	0.023
	Schur	-	2.6298e-15	0.012
	H*	9	4.8572e-17	0.012
10^{-3}	Neuts	8353	6.6669e-13	24.370
	TELPACK	13	1.3251e-12	0.250
	Schur	-	2.0720e-14	0.070
	H*	12	6.9389e-17	0.020
10^{-4}	Neuts	52940	6.6662e-13	228.45
	TELPACK	16	1.0926e-10	0.270
	Schur	-	2.8047e-13	0.070
	H*	15	4.1633e-17	0.018
10^{-5}	Neuts	226944	6.6668e-13	2125.800
	TELPACK	18	9.8710e-09	0.290
	Schur	-	3.2004e-12	0.070
	H*	18	4.1633e-13	0.022

TABLE 5. Results for Experiment 3 continued

δ	**Method**	**Iterations I**	$\|R_I - C(R_I)\|_\infty$	**CPU Time (sec.)**
10^{-6}	Neuts	*	*	*
	TELPACK	23	8.1815e-07	0.290
	Schur	-	4.1806e-11	0.060
	H*	21	2.7756e-17	0.028
10^{-7}	Neuts	*	*	*
	TELPACK	34	6.9935e-05	0.420
	Schur	-	2.4769e-10	0.080
	H*	24	8.3267e-17	0.030
10^{-8}	Neuts	*	*	*
	TELPACK	25	0.0023	0.310
	Schur	-	3.9794e-09	0.070
	H*	27	4.1633e-17	0.030
10^{-9}	Neuts	*	*	*
	TELPACK	41	0.0037	0.510
	Schur	-	7.3487e-09	0.070
	H*	28	4.1633e-17	0.030
10^{-10}	Neuts	*	*	*
	TELPACK	20	2.3243	0.260
	Schur	-	5.0173e-06	0.080
	H*	28	6.9389e-17	0.030

TABLE 6. Results for Experiment 4

Method	**Iterations I**	$\|R_I - C(R_I)\|_\infty$	**CPU Time (sec.)**
TELPACK	7	3.3307e-16	0.060
H*	9	5.5511e-17	0.010

diagonal and constant off-diagonal entries. The traffic intensity for this problem is $\rho = 1 - \delta$, where δ ranges between 10^{-1} and 10^{-10}.

This QBD model is a block-$GI/M/1$ version of a block $M/G/1$ example of Meini [15]. We have $\pi\beta^* = 1 + \delta$ in the usual notation, so that the condition $\pi\beta^* > 1$ for positive recurrence (see Neuts [16, Theorem 1.3.2]) is satisfied. This process, like the example in [15], is close to the null recurrent limit.

Results are shown in Tables 4 and 5. The stopping criterion used was that the difference between two iterations was less than 10^{-12}.

TELPACK does not perform well in this experiment, with errors of several orders of magnitude greater than those for the other algorithms as well as much larger CPU times. The errors increase considerably in size as the value of δ decreases. This parallels TELPACK's behaviour in Meini's $M/G/1$ example.

Again, as expected, the Neuts Algorithm is the worst performer with respect to both time and accuracy. The asterisks appearing in the Neuts column signify that we did not run the algorithm for values of $\delta < 10^{-5}$.

6.4. Experiment 4 Our fourth experiment comes from the TELPACK suite of examples. Here

$$C_n = \begin{bmatrix} (10/101)\,(1/101)^n & (4/21)\,(1/21)^n & (7/50)\,(4/5)^n \\ (90/101)\,(1/101)^n & (1/21)\,(1/21)^n & (1/100)\,(4/5)^n \\ (30/101)\,(1/101)^n & (2/7)\,(1/21)^n & (2/25)\,(4/5)^n \end{bmatrix}$$

for $n \geq 0$. This gives

$$C(z) = \begin{bmatrix} (1+(1-z)/100)^{-1} & 0 & 0 \\ 0 & (1+(1-z)/20)^{-1} & 0 \\ 0 & 0 & (1+4(1-z))^{-1} \end{bmatrix} \times \begin{bmatrix} 1/10 & 2/10 & 7/10 \\ 9/10 & 1/20 & 1/20 \\ 3/10 & 3/10 & 4/10 \end{bmatrix}.$$

The reason for our choice is that this example provides a very simple form of rational $C(z)$ for which every C_n is nonzero. This example can therefore be expected to favour TELPACK. As can be seen from Table 6 however, Algorithm H* has superior CPU time with comparable accuracy.

6.5. Experiment 5 The numerical experiments above all involve matrix functions $C(z)$ of rational form. We could find no examples in the literature for which $C(z)$ is not rational. The following is an original example showing how Algorithm H* performs in the general case when $C(z)$ is not rational.

A two-stage queueing system consists of $k-1$ homogeneous servers as a first stage and an overflow pool of unlimited capacity with a separate service facility as a second. An arrival is taken up by one of the first-stage servers if a free server is available; otherwise it overflows to the pool. During the time between consecutive arrivals to the system, each customer in the first stage has its service completed with probability p (whereupon it departs) and not completed with probability $q = 1 - p$ (whereupon it remains).

The number of pool customers that (if available) can be served (and depart) in an inter-arrival interval of the system has a Poisson distribution with mean r.

We may model this system as a block-$GI/M/1$ chain in which the level represents the number of customers in the pool and the phase the number $(0, 1, \ldots, k-1)$ of busy first-stage servers. The time points are taken immediately before arrival epochs.

We then have

$$C_0 = \Omega_0 e^{-r}, \quad C_n = \Omega_0 e^{-r} \frac{r^n}{n!} + \Omega_1 e^{-r} \frac{r^{n-1}}{(n-1)!} \quad (n \geq 1),$$

where

$$\Omega_0 = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ p^{k-1} & \binom{k-1}{1} p^{k-2} q & \cdots & \binom{k-2}{k-1} pq^{k-2} & q^{k-1} \end{bmatrix}$$

and

$$\Omega_1 = \begin{bmatrix} p & q & 0 & \cdots & 0 & 0 \\ p^2 & 2pq & q^2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ p^{k-1} & \binom{k-1}{1} p^{k-2} q & \binom{k-1}{2} p^{k-3} q^2 & \cdots & \binom{k-2}{k-1} pq^{k-2} & q^{k-1} \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

We remark that

$$C(z) := \sum_{m=0}^{\infty} C_m z^m = (\Omega_0 + z\Omega_1) e^{-r(1-z)} \quad (|z| \leq 1),$$

so that $C(z)$ is irrational, irreducible for $0 < z \leq 1$ and stochastic for $z = 1$.

With the choice $D_n := \sum_{m=n}^{\infty} C_m$ $(n \geq 1)$, the chain $\mathscr{C}$ is irreducible and

$$D(R) := \sum_{n=0}^{\infty} D_{n+1} R^n > D_1 = C - C_0 = \Omega_0 \left(1 - e^{-r}\right) + \Omega_1.$$

Hence $D(R)$ is irreducible. By the form of D_n it is also finite and so has a strictly positive left-invariant probability measure. Also $C = \Omega_0 + \Omega_1$ is irreducible and stochastic.

By Neuts [16, Theorem 1.3.2], a necessary and sufficient condition for $\mathscr{C}$ to be positive recurrent is that $\pi^T \beta^* > 1$, where $\pi^T = (\pi_0, \ldots, \pi_{k-1})$ is the left-invariant probability measure of C and $\beta^* := \sum_{n=1}^{\infty} nC_n e$.

We have

$$\sum_{n=1}^{\infty} nC_n e = \Omega_1 e + (\Omega_0 + \Omega_1) re,$$

so $\beta^* = (1, 1, \ldots, 1, 0)^T + re$ and $\pi^T \beta^* = (1 - \pi_{k-1}) + r$. Hence with the above scenario, $r = 1$ is a sufficient (but not necessary) condition for positive recurrence for all p with $0 < p < 1$.

TABLE 7. Experiment 5: matrices of size 2 × 2

k	p	Method	Iterations I	$\|R_I - C(R_I)\|_\infty$	CPU Time (sec.)
2	0.05	Neuts	283	5.8487e-13	3.430
		H*	8	1.1102e-16	0.008
	0.1	Neuts	147	6.1617e-13	1.810
		H*	6	1.7553e-13	0.007
	0.2	Neuts	75	5.6666e-13	0.950
		H*	6	2.2204e-16	0.007
	0.3	Neuts	50	4.0212e-13	0.620
		H*	5	2.2204e-16	0.004
	0.4	Neuts	36	5.0676e-13	0.490
		H*	5	1.1102e-16	0.004
	0.5	Neuts	28	3.0442e-13	0.340
		H*	4	2.2204e-16	0.003
	0.6	Neuts	22	3.7170e-13	0.280
		H*	4	1.1102e-16	0.003
	0.7	Neuts	18	1.7308e-13	0.230
		H*	4	2.7756e-17	0.003
	0.8	Neuts	14	1.9862e-13	0.180
		H*	3	5.5511e-16	0.003
	0.9	Neuts	10	4.1589e-13	0.130
		H*	3	1.1102e-16	0.003
	0.95	Neuts	8	3.2707e-13	0.120
		H*	2	1.2623e-13	0.002

TABLE 8. Experiment 5: matrices of size 3 × 3

k	p	Method	Iterations I	$\|R_I - C(R_I)\|_\infty$	CPU Time (sec.)
3	0.05	Neuts	148	5.6544e-13	1.860
		H*	7	1.1102e-16	0.007
	0.1	Neuts	76	5.5345e-13	1.030
		H*	6	1.1102e-16	0.007
	0.2	Neuts	38	4.5275e-13	0.490
		H*	5	5.5511e-17	0.003
	0.3	Neuts	25	2.5380e-13	0.330
		H*	4	5.5511e-17	0.002
	0.4	Neuts	18	2.5985e-13	0.240
		H*	4	5.5511e-17	0.002
	0.5	Neuts	14	1.1086e-13	0.180
		H*	3	1.9429e-15	0.002
	0.6	Neuts	11	6.6003e-14	0.160
		H*	3	5.5511e-17	0.002
	0.7	Neuts	8	5.5556e-13	0.120
		H*	3	2.7756e-17	0.002
	0.8	Neuts	7	1.8929e-14	0.120
		H*	2	1.6098e-15	0.002
	0.9	Neuts	5	4.4464e-14	0.080
		H*	2	5.5511e-17	0.002
	0.95	Neuts	4	4.4631e-14	0.070
		H*	1	1.8974e-13	0.001

TABLE 9. Experiment 5: matrices of size 4 × 4

k	p	Method	Iterations I	$\|R_I - C(R_I)\|_\infty$	CPU Time (sec.)
4	0.05	Neuts	101	5.1015e-13	1.690
		H*	6	5.4210e-20	0.006
	0.1	Neuts	51	5.5744e-13	0.810
		H*	5	2.7756e-17	0.003
	0.2	Neuts	25	4.1073e-13	0.430
		H*	4	5.5511e-17	0.002
	0.3	Neuts	16	4.2949e-13	0.270
		H*	3	4.2866e-13	0.002
	0.4	Neuts	12	9.9615e-14	0.210
		H*	3	5.5511e-17	0.002
	0.5	Neuts	9	1.3001e-13	0.150
		H*	3	2.7756e-17	0.002
	0.6	Neuts	7	1.4144e-13	0.130
		H*	2	3.4667e-14	0.001
	0.7	Neuts	6	2.3120e-14	0.110
		H*	2	5.5511e-17	0.001
	0.8	Neuts	5	5.8842e-15	0.100
		H*	2	2.7756e-17	0.001
	0.9	Neuts	4	7.2164e-16	0.080
		H*	1	3.8858e-15	0.001
	0.95	Neuts	3	6.7724e-15	0.070
		H*	1	6.9389e-18	0.001

Table 10. Experiment 5: matrices of size 5×5

k	p	Method	Iterations I	$\|R_I - C(R_I)\|_\infty$	CPU Time (sec.)
5	0.05	Neuts	76	5.9908e-13	1.360
		H*	6	3.3307e-16	0.006
	0.1	Neuts	39	3.2402e-13	0.770
		H*	5	5.5511e-17	0.004
	0.2	Neuts	19	2.1871e-13	0.340
		H*	4	1.3878e-17	0.002
	0.3	Neuts	12	2.3789e-13	0.230
		H*	3	2.2204e-16	0.002
	0.4	Neuts	9	7.0111e-14	0.170
		H*	3	8.3267e-17	0.002
	0.5	Neuts	7	6.7696e-13	0.150
		H*	2	2.4286e-14	0.001
	0.6	Neuts	6	8.8124e-15	0.120
		H*	2	2.7756e-17	0.001
	0.7	Neuts	5	4.4686e-15	0.100
		H*	2	5.5511e-17	0.001
	0.8	Neuts	4	4.0523e-15	0.080
		H*	1	2.1982e-14	0.001
	0.9	Neuts	3	3.1641e-15	0.070
		H*	1	3.4694e-18	0.001
	0.95	Neuts	2	6.2450e-13	0.050
		H*	1	1.1102e-16	0.001

The calculations were performed with $r = 1$. The stopping criterion used was that the difference between two iterations was less than 10^{-12}. Results for the Neuts and H* Algorithms are given in Tables 7–10. We note that these are the only two algorithms which can be applied here. As expected, the Neuts Algorithm is less accurate with much greater CPU times.

References

[1] N. Akar, N. C. Oğuz and K. Sohraby, "TELPACK: An advanced TELetraffic analysis PACKage", *IEEE Infocom'97*, http://www.cstp.umkc.edu/personal/akar/home.html.

[2] N. Akar, N. C. Oğuz and K. Sohraby, "An overview of TELPACK", *IEEE Commun. Mag.* **36** (1998) 84–87.

[3] N. Akar, N. C. Oğuz and K. Sohraby, "A novel computational method for solving finite QBD processes", *Comm. Statist. Stochastic Models* **16** (2000) 273–311.

[4] N. Akar and K. Sohraby, "On computational aspects of the invariant subspace approach to teletraffic problems and comparisons", *preprint*.

[5] N. Akar and K. Sohraby, "An invariant subspace approach in $M/G/1$ and $G/M/1$ type Markov chains", *Comm. Statist. Stochastic Models* **13** (1997) 381–416.

[6] S. Asmussen and V. Ramaswami, "Probabilistic interpretations of some duality results for the matrix paradigms in queueing theory", *Comm. Statist. Stochastic Models* **6** (1990) 715–733.

[7] D. Bini and B. Meini, "On cyclic reduction applied to a class of Toeplitz-like matrices arising in queueing problems", in *Proc. 2nd Intern. Workshop on Numerical Solution of Markov Chains*, (Raleigh, North Carolina, 1995) 21–38.

[8] D. Bini and B. Meini, "Improved cyclic reduction for solving queueing problems", *Numer. Algorithms* **15** (1997) 57–74.

[9] D. A. Bini, G. Latouche and B. Meini, "Quadratically convergent algorithms for solving matrix polynomial equations", Technical Report 424, Université Libre Bruxelles, 2000.

[10] G. H. Golub and C. F. Van Loan, *Matrix computations* (Johns Hopkins University Press, Baltimore, 1989).

[11] E. Hunt, "A probabilistic algorithm for determining the fundamental matrix of a block $M/G/1$ Markov chain", *Math. Comput. Modelling* **38** (2003) 1203–1209.

[12] G. Latouche and V. Ramaswami, "A logarithmic reduction algorithm for Quasi-Birth-Death processes", *J. Appl. Probab.* **30** (1993) 650–674.

[13] G. Latouche and P. Taylor, *Matrix-analytic methods* (World Scientific, Singapore, 2002).

[14] A. J. Laub, "A Schur method for solving algebraic Riccati equations", *IEEE Trans. Auto. Control* **24** (1979) 913–925.

[15] B. Meini, "Solving QBD problems: the cyclic reduction algorithm versus the invariant subspace method", *Adv. Perf. Anal.* **1** (1998) 215–225.

[16] M. F. Neuts, *Matrix geometric solutions in stochastic models* (Johns Hopkins University Press, Baltimore, 1981).

[17] V. Ramaswami, "A duality theorem for the matrix paradigms in queueing theory", *Comm. Statist. Stochastic Models* **6** (1990) 151–161.