Boundary Value Methods for Solving Transient Solutions of Markovian Queueing Networks

Raymond H. Chan[a] * Ka-Chun Ma [b] Wai-Ki Ching[c†]

[a]Department of Mathematics, The Chinese University of Hong Kong,
Shatin, Hong Kong, China.

[b]Department of Mathematics, The Chinese University of Hong Kong,
Shatin, Hong Kong, China.

[c]Department of Mathematics, The University of Hong Kong,
Pokfulam Road, Hong Kong, China.

In this paper, we consider Boundary Value Methods (BVMs) for finding transient solutions of Markovian queueing networks. Algebraic Multigrid (AMG) methods with modified restriction operator are applied to solve the resulting system of linear equations. Numerical examples are given to demonstrate the efficiency of our proposed method.

1. Introduction

Markovian queueing networks are common stochastic models for a number of physical systems such as telecommunication systems [9], manufacturing systems [10] and inventory systems [11]. For long-run system performance analysis, the steady-state probability distribution of the system is required. The steady-state probability distribution can be obtained by solving a large linear system. Direct methods [5,12,18] and iterative methods [3,7,9,10] have been developed for this purpose.

However, to analyze the system in a finite horizon, the transient solution of the queueing system is required, and it can be found by solving a system of ordinary differential equations (ODEs). Many classical numerical methods can be applied to solve the ODE systems. The Initial Value Methods (IVMs) such as the Runge-Kutta method are good explicit methods for its efficiency and easy implementation. But they may require small time step in order to converge. A survey on numerical methods for solving transient solutions of homogeneous irreducible Markov chains can be found in [20].

In this paper, we propose to use the Boundary Value Methods (BVMs) [2,15] to solve the ODE systems. BVMs are implicit stable methods and hence there is no restriction on the size of the time step for the method to converge. However, the disadvantage is that they require solutions of large linear systems, and hence may require longer computational time

*Research supported in part by RGC Grant No. CUHK 4243/01P and CUHK DAG 2060220.
†Research supported in part by RGC Grant No. HKU 7126/02P and HKU CRCG Grant No. 10205105 and 10204436.

when compared with the IVMs. Here we propose to use the algebraic multigrid (AMG) method to solve the resulting linear systems from BVMs.

AMG methods have been developed for more than two decades [22] and have been applied to many applications such as solving partial differential equations [1] and imaging problems [17]. They also have been used successfully for finding the steady-state probability distributions of queueing networks by using an appropriate coarse grid approximation, see [7]. In this paper, we use it for queues in transient states. The ODEs are first discretized by BVMs and the resulting linear systems are solved by AMG methods. For overflow queueing networks, we will see that the resulting method is much more efficient than IVMs, especially when the systems are ill-conditioned. We will illustrate the effectiveness of our method through 2-queue overflow networks. A comparison with other iterative methods will also be given.

The paper is organized as follows. In Section 2, we present the overflow queues. IVMs and BVMs are introduced in Section 3 while the AMG method is given in Section 4. In Section 5, numerical examples are given to demonstrate the efficiency of our method.

2. Queueing Networks

For continuous-time Markovian queueing networks, the transient probability distribution can be found by solving Kolmogorov's backward equations [8,23]. We introduce the equations here. We remark that our proposed method can be applied to many different queueing networks. For simplicity of discussion, here we only consider overflow queues, see [3,16] for instance.

We begin our discussion with a simple two-queue free queueing network. In this queueing network, there are no interactions between the two Markovian $M/M/s_i/(n_i - s_i - 1)$ queues. Here $i = 1, 2$, and s_i and $(n_i - s_i - 1)$ denote the number of parallel servers and the number of queueing spaces in Queue i. By state (i, j), we mean that there are i customers in Queue 1 and j customers in Queue 2 respectively. Let $p_{i,j}(t)$ be the probability that the network is in state (i, j) at time t. If we let

$$\mathbf{p}(t) = (p_{0,0}(t), \ldots, , p_{0,n_2-1}(t), p_{1,0}(t), \ldots, p_{1,n_2-1}(t), \ldots, p_{n_1-1,0}(t), \ldots, p_{n_1-1,n_2-1}(t))^T,$$

then the Kolmogorov backward equations can be written as

$$\frac{d\mathbf{p}(t)}{dt} = -(G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2)\mathbf{p}(t), \tag{1}$$

where

$$G_i = \begin{bmatrix} \lambda_i & -\mu_i & & & & & & 0 \\ -\lambda_i & \lambda_i + \mu_i & -2\mu_i & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & -\lambda_i & \lambda_i + s_i\mu_i & -s_i\mu_i & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & -\lambda_i & \lambda_i + s_i\mu_i & -s_i\mu_i & \\ 0 & & & & & -\lambda_i & s_i\mu_i & \end{bmatrix}, \tag{2}$$

and λ_i and μ_i are the mean arrival rate and the mean service rate of the servers in Queue i. Since there are no interactions between the queues, the transient solution can be obtained

in tensor product forms from the solutions of individual queues, i.e., if $\mathbf{q}_i(t)$ is the transient solution for Queue i, i =1, 2, then $(\mathbf{q}_1(t) \otimes \mathbf{q}_2(t))$ is the transient solution for (1). In fact,

$$\begin{aligned} \frac{d}{dt}(\mathbf{q}_1(t) \otimes \mathbf{q}_2(t)) &= \frac{d\mathbf{q}_1(t)}{dt} \otimes \mathbf{q}_2(t) + \mathbf{q}_1(t) \otimes \frac{d\mathbf{q}_2(t)}{dt} \\ &= -G_1\mathbf{q}_1(t) \otimes \mathbf{q}_2(t) - \mathbf{q}_1(t) \otimes G_2\mathbf{q}_2(t) \\ &= -(G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2)(\mathbf{q}_1(t) \otimes \mathbf{q}_2(t)). \end{aligned}$$

See [13] for the transient solutions of one-dimensional Markovian queues.

Next we present two-queue overflow networks. Unlike two-queue free networks, they allow overflow from one queue to another. Here we consider the following overflow discipline: (i) when Queue 1 is full, customers arriving at Queue 1 are allowed to overflow to Queue 2 if it is not yet full, and (ii) overflow from Queue 2 to Queue 1 is not allowed, see [16] for instance. Then the transient solution $\mathbf{p}(t)$ satisfies

$$\frac{d\mathbf{p}(t)}{dt} = -(G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2 + \mathbf{e}_{n_1}\mathbf{e}_{n_1}^T \otimes R_1)\mathbf{p}(t), \tag{3}$$

where $\mathbf{e}_{n_1}^T = (0, \ldots, 0, 1) \in \mathbf{R}^{n_1}$ and

$$R_1 = \lambda_1 \cdot \begin{bmatrix} 1 & & & & 0 \\ -1 & 1 & & & \\ & \ddots & \ddots & & \\ & & -1 & 1 & \\ 0 & & & -1 & 0 \end{bmatrix}.$$

For two-queue overflow networks, there are no product form solutions, and thus numerical methods must be used to find the solutions. One can easily extend these ideas to obtain the backward equations for more general q-queue overflow networks, see [4].

3. ODE Solvers

In this section, we present examples of IVMs and BVMs for solving Kolmogorov's backward equations. Suppose we want to find $\mathbf{p}(T)$ for some final time $T < \infty$. Then we divide the time horizon into N steps, with step size $h = T/N$. Denote $\mathbf{p}_k = \mathbf{p}(kh)$, $0 \le k \le N$, the probability distribution that we want to find.

One of the most powerful IVMs for solving a general ODE is the Runge-Kutta method of order 4 (RK4), see [21, p. 414]. For a general ODE

$$\frac{d\mathbf{p}(t)}{dt} = -H\mathbf{p}(t), \tag{4}$$

RK4 can be simplified as follows:

$$\mathbf{p}_{k+1} = (I - hH + \frac{1}{2}h^2H^2 - \frac{1}{6}h^3H^3 + \frac{1}{24}h^4H^4)\mathbf{p}_k, \quad k = 0, 1, \ldots, N-1,$$

with $\mathbf{p}_0$ being the given initial condition. The method is fast, requiring only matrix-vector multiplications. But it is stable only if $\|I - hH + \frac{1}{2}h^2H^2 - \frac{1}{6}h^3H^3 + \frac{1}{24}h^4H^4\| < 1$.

For the overflow network in (3), we have

$$H = G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2 + \mathbf{e}_{n_1}\mathbf{e}_{n_1}^T \otimes R_1. \tag{5}$$

We note that for some queueing parameters, $\|H\|$ may increase as the sizes of the queues increase. In these cases, h has to be small in order that RK4 converges.

To alleviate the requirement on h, one can solve (4) by BVMs which are stable implicit methods, see [2]. An example of BVMs is the third order generalized backward differentiation formulae (GBDF3):

$$\begin{array}{lcll} \frac{1}{6}(2\mathbf{p}_{k+1} + 3\mathbf{p}_k - 6\mathbf{p}_{k-1} + \mathbf{p}_{k-2}) & = & h\mathbf{f}_k, & k = 2, 3, \ldots, N-1, \\ \frac{1}{6}(-\mathbf{p}_3 + 6\mathbf{p}_2 - 3\mathbf{p}_1 - 2\mathbf{p}_0) & = & h\mathbf{f}_1, & \\ \frac{1}{6}(11\mathbf{p}_N - 18\mathbf{p}_{N-1} + 9\mathbf{p}_{N-2} - 2\mathbf{p}_{N-3}) & = & h\mathbf{f}_N, & \end{array}$$

see [2, p. 132]. In matrix form, it is

$$[A \otimes I_{n_1} \otimes I_{n_2} + hI_N \otimes H]\mathbf{x} = -\mathbf{a} \otimes \mathbf{p}_0, \tag{6}$$

where $\mathbf{a}^T = (-\frac{1}{3}, \frac{1}{6}, 0, \ldots, 0) \in \mathbf{R}^N$, and A is the N-by-N matrix

$$A = \begin{bmatrix} -\frac{1}{2} & 1 & -\frac{1}{6} & & & & 0 \\ -1 & \frac{1}{2} & \frac{1}{3} & & & & \\ \frac{1}{6} & -1 & \frac{1}{2} & \frac{1}{3} & & & \\ & \frac{1}{6} & -1 & \frac{1}{2} & \frac{1}{3} & & \\ & & \ddots & \ddots & \ddots & \ddots & \\ & & & \frac{1}{6} & -1 & \frac{1}{2} & \frac{1}{3} \\ 0 & & & -\frac{1}{3} & \frac{3}{2} & -3 & \frac{11}{6} \end{bmatrix}.$$

GBDF3 is stable in the sense that a very large time step can be used, while the drawback is that we need to solve the big linear system (6) which has size N times the size of H. There are many alternative ways to solve (6), such as the GMRES method [19, p. 164]. But we will see in the numerical examples that if H is ill-conditioned, the GMRES method does not work, even if a preconditioner is used. Hence, we propose to solve it by the AMG method.

4. The Algebraic Multigrid Method

For the 2-queue overflow network in (3), (6) becomes

$$[A \otimes I_{n_1} \otimes I_{n_2} + hI_N \otimes (G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2 + \mathbf{e}_{n_1}\mathbf{e}_{n_1}^T \otimes R_1)]\mathbf{x} = -\mathbf{a} \otimes \mathbf{p}_0. \tag{7}$$

Here we describe our method for solving (7). Let $A = PDP^{-1}$ be the spectral decomposition of A with D being a diagonal matrix with diagonal entries ρ_i, $1 \le i \le N$, where ρ_i are complex numbers with positive real part, see for instance [2, Figure 5.2].

Let $\mathbf{y} = (P \otimes I_{n_1} \otimes I_{n_2})\mathbf{x}$. Then (7) becomes

$$[D \otimes I_{n_1} \otimes I_{n_2} + hI_N \otimes (G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2 + \mathbf{e}_{n_1}\mathbf{e}_{n_1}^T \otimes R_1)]\mathbf{y} = -P\mathbf{a} \otimes \mathbf{p}_0.$$

We decompose this system of equations into N sub-systems of smaller size:

$$[\rho_i(I_{n_1} \otimes I_{n_2}) + h(G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2 + \mathbf{e}_{n_1}\mathbf{e}_{n_1}^T \otimes R_1)]\mathbf{y}_i = c_i\mathbf{p}_0, \quad 1 \le i \le N, \tag{8}$$

where c_i is the ith entry of $-P\mathbf{a}$ and $\mathbf{y}^T = (\mathbf{y}_1^T, \mathbf{y}_2^T, \ldots, \mathbf{y}_N^T)$.

In [7], an AMG method has been used successfully to solve a system of the form

$$(G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2 + \mathbf{e}_{n_1}\mathbf{e}_{n_1}^T \otimes R_1)\mathbf{x} = \mathbf{b},$$

which is the system corresponds to the steady-state queues, and is equal to the transient system in (8) but without the first term. Here we will use the same AMG method to solve (8).

Suppose in the finest grid we have $(2^{m_2} - 1)$ equations and at the coarsest grid we have $(2^{m_1} - 1)$ equations. We use the V-cycle algorithm here with one pre-smoothing and one post-smoothing at each grid. The smoother we used is the Gauss-Seidel iterative method [14, p. 49]. Traditionally, the one-dimensional restriction operator is defined as

$$I_{m+1}^m = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 & & & & & & 0 \\ & & 1 & 2 & 1 & & & & \\ & & & & \cdots & \cdots & \cdots & & \\ 0 & & & & & & 1 & 2 & 1 \end{bmatrix},$$

where its size is $(2^m - 1)$-by-$(2^{m+1} - 1)$. The prolongation operator I_m^{m+1} is equal to $2(I_{m+1}^m)^T$. For queueing networks, we use the modified restriction operator $\tilde{I}_{m+1}^m$ described in [7] which is defined as follows. Suppose $I_{m+1}^m = (c_{i,j})$, then $\tilde{I}_{m+1}^m \equiv (\tilde{c}_{i,j})$ where

$$\tilde{c}_{i,j} = \frac{c_{i,j}}{d_j}, \quad \text{and} \quad d_j = \sum_i c_{i,j}.$$

If A^m is the matrix at grid m, i.e., there are $(2^m - 1)$ equations, then at grid $m - 1$, the matrix A^{m-1} satisfies $A^{m-1} = \tilde{I}_m^{m-1} A^m I_{m-1}^m$. The reason for using the modified restriction operator is to keep the singularity as well as the structure of G_1 and G_2, see [7] for more details.

For two-queue systems, the prolongation operator and the modified restriction operator are $I_m^{m+1} \otimes I_m^{m+1}$ and $\tilde{I}_{m+1}^m \otimes \tilde{I}_{m+1}^m$ respectively, and the two dimensional coarse grid matrix can be defined accordingly. Note that the coarse grid matrix of an identity matrix is a matrix with 3 bands. We will use this fact in the cost analysis.

5. Numerical Experiments

In this section, we first compare the cost of using IVMs and BVMs for solving overflow queues. Then we illustrate the efficiency of our method on three overflow queues. At the end of this section, we compare the results with the GMRES method [19].

Consider a general two-queue network with Queue i having n_i states. For a general IVM, in each time step, it requires only a few matrix-vector multiplications which are of order $O(n_1 n_2)$ as the matrix H in (5) is a banded matrix of size $n_1 n_2$-by-$n_1 n_2$ with 5 bands. If h_c is the largest time step guaranteed for convergence, then the IVM will require T/h_c's time steps to get to the final time T. But in order for the method to converge, h_c should be of order $O(1/\|H\|)$. Thus the total cost of the IVM is of $O(n_1 n_2 \|H\| T)$.

For a general BVM, there are T/h's sub-systems to be solved. We solve each sub-system by the AMG method. The cost for each V-cycle is of $O(n_1 n_2)$ operations since the coefficient matrix of each sub-system is of size $n_1 n_2$-by-$n_1 n_2$ and has at most 9 bands.

Thus the total cost is about $O(n_1 n_2 kT/h)$, where k is the maximum of the numbers of V-cycles required for convergence for each sub-system.

We remark that if $\|H\|$ is bounded independent of n_i, then both methods will be of the same order. However, if $\|H\|$ is increasing with n_i, then the BVM will be an order less costly than the IVM. We illustrate this by three examples. The first example describes the situation that the arrival and service rate are independent of the size of the queues. The second and the third examples, on the other hand describe the situations with arrival and service rate dependent on the size of the queues.

EXAMPLE 1: For $i = 1, 2$, let G_i^M be the same $(2^M - 1)$-by-$(2^M - 1)$ matrix as in (2) with $s_i = 5$, $\mu_i = 1$, $\lambda_i = s_i\mu_i - \frac{1}{2}(n_i - 1)^{-1} = 5 - \frac{1}{2}(2^M - 2)^{-1}$.

EXAMPLE 2: For $i = 1, 2$, let G_i^M be the $(2^M - 1)$-by-$(2^M - 1)$ matrix,

$$G_i^M = \begin{bmatrix} 2^{M-1} & -2^{M-1} & & & 0 \\ -2^{M-1} & 2^M & -2^{M-1} & & \\ & \ddots & \ddots & \ddots & \\ & & -2^{M-1} & 2^M & -2^{M-1} \\ 0 & & & -2^{M-1} & 2^{M-1} \end{bmatrix}.$$

EXAMPLE 3: For $i = 1, 2$, let G_i^M be the same $(2^M - 1)$-by-$(2^M - 1)$ matrix as in (2) with $\mu_i = 2^{M-1}$, $s_i = 5$, $\lambda_i = s_i\mu_i - \frac{1}{2}(n_i - 1)^{-1} = 5 \cdot 2^{M-1} - \frac{1}{2}(2^M - 2)^{-1}$.

In all examples, we solve for the probability distribution vector at $T = 10$. We assume that the initial state is $(0, 0)$. In solving (8) by the AMG method or the GMRES method, we use a stopping tolerance of 10^{-6}. For the AMG method, we set m_1, the coarsest grid level, to be 2.

In Example 1, $\|G_i^M\|$ and hence $\|H\|$ is bounded. Thus, the total costs for both the IVM and the BVM are of order $O(2^{2M})$. In Examples 2 and 3, $\|H\|$ is of order $O(2^M)$. So for the IVM, the total cost is $O(2^{3M})$. But for the BVM, h can be kept constant regardless of $\|H\|$. Thus the total cost for each V-cycle is still of order $O(2^{2M})$. To be more specific, in the following we estimate the total costs for RK4 and GBDF3 in terms of number of scalar multiplications.

For RK4, 4 matrix-vector multiplications, 4 vector-vector operations, and 6 scalar-vector multiplications are required in each time step. The matrix involved is a 5-band matrix of size $(2^M - 1)^2$-by-$(2^M - 1)^2$. Thus

$$4 \cdot 10(2^M - 1)^2 + 4(2^M - 1)^2 + 6(2^M - 1)^2 = 50(2^M - 1)^2$$

operations are required in each time step. In Example 1, $h_c = 0.25$. Since $T/h_c = 40$ time steps are required, the total cost is $(2000 \cdot (2^M - 1)^2)$ operations. In Example 2, since $(10 \cdot 2^{M+1})$ time steps are required, the total cost is $(1000 \cdot 2^M(2^M - 1)^2)$ operations. In Example 3, since $(10 \cdot 2^{M+3})$ time steps are required, the total cost is $(4000 \cdot 2^M(2^M - 1)^2)$.

For GBDF3 with the AMG method, we first estimate the cost for each V-cycle. As the cost of a Gauss-Seidel iteration for a 9-band matrix is 18 operations for each update of a variable, the cost for each V-cycle is

$$2\sum_{i=2}^{M} 18(2^i - 1)^2 \le 36\sum_{i=2}^{M}(4^i) \le 12 \cdot 4^{M+1} = 48 \cdot 2^{2M}$$

operations. We can take $h = 1$ for all three examples as the method is stable. Thus there are 10 sub-systems to be solved by the AMG solvers. The total cost is therefore less than $(480k \cdot 2^{2M})$ operations.

Tables 1 to 3 give h_c, k, the total number of operations as estimated above, and the CPU times in solving the problems by Matlab. We can see that the BVM together with the AMG method is more efficient in finding the transient solutions when compared with the IVM. We see also that the AMG method is very efficient in solving these systems—the number of V-cycles required for convergence is independent of the queue sizes even when $\|H\|$ increases.

Regarding Examples 2 and 3, it may seem that the matrix G_i^M in these two examples can be made better conditioned by scaling down by the factor 2^{M-1} as follows: divide 2^{M-1} from both sides in the ODE (4) and let $\tilde{t} = 2^{M-1}t$, then we have

$$\frac{d\mathbf{p}(\tilde{t})}{d\tilde{t}} = -\tilde{H}\mathbf{p}(\tilde{t}),$$

where $\|\tilde{H}\|$ is independent of M. However, in order to find $\mathbf{p}(t)$ at time T, one has to find $\mathbf{p}(\tilde{t})$ at time $\tilde{T} = 2^{M-1}T$. By the previous results, the total cost for the IVMs on this new ODE is of order $O(n_1 n_2 \|\tilde{H}\|\tilde{T}) = O(n_1 n_2 \|H\|T)$. That means the order of the total cost cannot be reduced by scaling.

Finally, we compare the AMG method with the preconditioned GMRES method [19]. We will use T. Chan's preconditioner as proposed in [6]. If $c(G_i)$ are T. Chan's preconditioners for G_i, then the preconditioner for (8) is

$$\rho_i(I_{n_1} \otimes I_{n_2}) + h(c(G_1) \otimes I_{n_2} + I_{n_1} \otimes c(G_2)).$$

Table 4 gives the maximum number of iterations for the (preconditioned) GMRES method over each sub-system in (8). We note that the cost per iteration of the GMRES method is of the same order as the AMG method, as it requires mainly matrix-vector multiplications. However, as we can see from Table 4, its number of iterations required for convergence increases with the size of queues for Examples 2 and 3. Hence it is more costly than the AMG method.

We will present a theoretical analysis of the method and its applications to more general Markovian queueing networks in a forthcoming paper.

Acknowledgments: We are indebted to Profs. H. W. Sun and L. Brugnano for their valuable comments and suggestions. This paper will not be possible without their helps.

REFERENCES

1. D. Braess, *Towards Algebraic Multigrid for Elliptic Problems of Second Order*, Computing, 55 (1995), 379–393.
2. L. Brugnano, and D. Trigiante, *Solving Differential Problems by Multistep Initial and Boundary Value Methods*, Gordon and Breach Science Publishers, 1998.
3. R. Chan, *Iterative Methods for Overflow Queueing Models I*, Numer. Math., 51 (1987), 143–180.
4. R. Chan, *Iterative Methods for Overflow Queueing Models II*, Numer. Math., 54 (1988), 57–78.

Table 1
Total number of operations and CPU times in seconds for Example 1.

	IVM			BVM	
M	h_c	Operations	CPU time	k	Operations	CPU time
3	2^{-2}	98,000	0.49	7	215,040	1.00
4	2^{-2}	450,000	2.37	7	860,160	5.85
5	2^{-2}	1,922,000	10.81	7	3,440,640	28.26
6	2^{-2}	7,938,000	49.05	7	13,762,560	111.71
7	2^{-2}	32,258,000	222.06	7	55,050,240	546.06

Table 2
Total number of operations and CPU times in seconds for Example 2.

	IVM			BVM	
M	h_c	Operations	CPU time	k	Operations	CPU time
3	2^{-4}	392,000	1.91	8	245,760	1.13
4	2^{-5}	3,600,000	18.66	8	983,040	6.49
5	2^{-6}	30,750,000	170.70	8	3,932,160	32.24
6	2^{-7}	250,016,000	1,649.07	9	17,694,720	161.99
7	2^{-8}	2,064,512,000	19,131.92	9	70,778,880	792.72

5. R. Chan and W. Ching, *A Direct Method for Stochastic Automata Networks*, Proceedings of the IMS Workshop on Applied Probability, 1–18, Hong Kong, Eds.: R. Chan, Y. Kwok, D. Yao, and Q. Zhang, Studies in Advanced Mathematics, 26, International Press, June 1999.
6. R. Chan and W. Ching, *Circulant Preconditioners for Stochastic Automata Networks*, Numer. Math., 87 (2000), 35–57.
7. Q. Chang, S. Ma, and G. Lei, *Algebraic Multigrid Method for Queuing Networks*, Inter. J. Comput. Math., 70 (1999), 539–552.
8. W. Ching, *Iterative Methods for Queuing and Manufacturing Systems*, Springer Monographs in Mathematics, Springer, London, 2001.
9. W. Ching, R. Chan, and X. Zhou, *Circulant Preconditioners for Markov Modulated Poisson Processes and Their Applications to Manufacturing Systems*, SIAM J. Matrix Anal. Appl., 18 (1997), 464–481.
10. W. Ching and A. Loh, *Iterative Methods for Flexible Manufacturing Systems*, J. Appl. Math. Comput., 141 (2003), 553–564.
11. W. Ching, W. Yuen, and A. Loh, *An Inventory Model with Returns and Lateral Transshipments*, J. Operat. Res. Soc., 54 (2003), 636–641.
12. J. Climent, L. Tortosa and A. Zamora, *A Note on the Recursive Decoupling Method for Solving Tridiagonal Linear Systems*, Appl. Math. Comput., 140 (2003), 159–164.
13. D. Everitt and T. Downs, *The Output of the M/M/s Queue*, Opns. Res., 32 (1984),

Table 3
Total number of operations and CPU times in seconds for Example 3.

	IVM			BVM	
M	h_c	Operations	CPU time	k	Operations	CPU time
3	2^{-6}	1,568,000	7.62	8	245,760	1.13
4	2^{-7}	14,400,000	75.02	8	983,040	6.66
5	2^{-8}	123,000,000	693.18	8	3,932,160	32.26
6	2^{-9}	1,000,064,000	6,634.67	9	17,694,720	166.58
7	2^{-10}	8,258,048,000	>50,000	9	70,778,880	819.20

Table 4
Maximum number of iterations needed for the GMRES method.

	No preconditioner			T. Chan's preconditioner		
M	Ex. 1	Ex. 2	Ex. 3	Ex. 1	Ex. 2	Ex. 3
3	27	30	30	19	20	21
4	50	58	69	26	32	35
5	56	89	133	31	51	59
6	54	132	244	28	79	102
7	54	168	>300	26	123	179

796–808.
14. W. Hackbusch, *Multi-Grid Methods and Applications*, Springer-Verlag Berlin, 1985.
15. X. Jin, V. Sin and L. Song, *Circulant-block Preconditioners for Solving Ordinary Differential Equations*, J. Appl. Math. Comput. 140 (2003) 409–418
16. L. Kaufman, *Matrix Methods for Queueing Problems*, SIAM J. Statist. Comput., 4 (1982), 525–552.
17. R. Kimmel and I. Yavneh, *An Algebraic Multigrid Approach for Image Analysis*, SIAM J. Sci. Comput., 24 (2003), 1218–1231.
18. L. Lu, W. Ching, and M. Ng, *Exact Algorithms for Singular Tridiagonal Systems with Applications to Markov Chains*, to appear in J. Appl. Math. Comput. (2004).
19. Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, 2003.
20. R. Sideje and W. Stewart, *A Numerical Study of Large Sparse Matrix Exponentials Arising in Markov Chain*, Comput. Stat. Data Anal., 29 (1999), 345–368.
21. J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag New York Inc., 1980.
22. K. Stuben, *Algebraic Multigrid (AMG): Experiences and Comparisons*, Appl. Math. Comput., 13 (1983), 3–4, 419–451.
23. H. Taha, *Operations Research: An Introduction*, Prentice Hall, 1997.