DISTRIBUTIONS OF REWARD FUNCTIONS ON CONTINUOUS-TIME MARKOV CHAINS

MOGENS BLADT
Department of Statistics, IIMAS, Universidad Nacional Autonoma de Mexico
Apartado Postal 20-726, 01000 Mexico, D.F. Mexico
E-mail: bladt@sigma.iimas.unam.mx

BEATRICE MEINI
Dipartimento di Matematica, Università di Pisa, via Buonarroti 2, 56127 Pisa, Italy
E-mail: meini@dm.unipi.it

MARCEL F. NEUTS
Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ 85721, USA
E-mail: marcel@sie.arizona.edu

BRUNO SERICOLA
IRISA - INRIA, Campus universitaire de Beaulieu, 35042 Rennes Cedex, France
E-mail: sericola@irisa.fr

We develop algorithms for the computation of the distribution of the total reward accrued during $[0, t)$ in a finite continuous-parameter Markov chain. During sojourns, the reward grows linearly at a rate depending on the state visited. At transitions, there can be instantaneous rewards whose values depend on the states involved in the transition. For moderate values of t, the reward distribution is obtained by implementing a series representation, due to Sericola, that is based on the uniformization method. As an alternative, that distribution can also be computed by the numerical inversion of its Laplace-Stieltjes transform. For larger values of t, we implement a matrix convolution product to compute a related semi-Markov matrix efficiently and accurately.

1 Introduction

We consider an irreducible, continuous-time m-state Markov chain $\{J(t)\}$ with generator Q. Our objective is to develop the theory of and numerical procedures for various probability distributions associated with a reward function defined on the Markov chain.

There is a continuous reward in that, for every unit of time spent in state j, a reward a_j accrues. In addition, there are instantaneous rewards associated with the various transitions. At each transition $h \to r$, an instantaneous, finite reward c_{hr} is received. We do not impose restrictions on the signs of the quantities $\{c_{hr}\}$.

We start by defining several random variables of interest and by introducing notation. The random variables $N_{hk}(t)$ are the numbers of transitions $h \to k$ during $[0,t)$. We make the convention that $N_{hh}(t) = 0$, for $1 \leq h \leq m$. For use in transforms, we let Z be a matrix with elements z_{hk} where $z_{hh} = 1$, for $1 \leq h \leq m$. We recall that the *Schur product*, $A \bullet B$, of $m \times m$ matrices A and B is the matrix with elements $A_{hk}B_{hk}$.

The piecewise constant random function $a_{J(t)}$ takes the value a_j when $J(t) = j$. The total continuous reward $R_j(t)$ earned during sojourns in the state j over an interval $[0,t)$ is given by

$$R_j(t) = \int_0^t a_{J(u)} 1_{\{J(u)=j\}} du, \quad \text{for} \quad 1 \leq j \leq m,$$

where $1_{\{c\}}$ equals 1 if condition c holds and 0 otherwise, and the total continuous reward $R(t)$ over an interval $[0,t)$ is given by

$$R(t) = \int_0^t a_{J(u)} du.$$

In the context of dependability analysis of fault-tolerant computer systems, the random variable $R(t)$ is referred to as a performability measure, see e.g. [3] and [2] and the references therein.

We shall derive a concise expression for the joint Laplace-Stieltjes transform and generating function of the random variables $\{R_j(t)\}$ and $\{N_{hk}(t)\}$ taking the initial and final states $J(0)$ and $J(t)$ of the Markov chain into account. By $\mathbf{s}$ we denote the vector with components $s_1, \ldots, s_m$. By $\Delta(\mathbf{s})$, we denote an $m \times m$ diagonal matrix with the quantities $s_1, \ldots, s_m$ as its diagonal elements. We are interested in the transform

$$V^*_{ij}(\mathbf{s}, Z; t) = E\left\{ \exp[-\sum_{h=1}^m s_h R_h(t)] \prod_{h,k} z_{hk}^{N_{hk}(t)} 1_{\{J(t)=j\}} \middle| J(0) = i \right\},$$

for $1 \leq i, j \leq m$. By $V^*(\mathbf{s}, Z; t)$, we denote the $m \times m$ matrix with elements $V^*_{ij}(\mathbf{s}, Z; t)$. For all $t \geq 0$, the matrix $V^*(\mathbf{s}, Z; t)$ is well-defined and analytic for all complex values of z_{hk} and s_ν, for $1 \leq h, k \leq m$, $1 \leq \nu \leq m$.

The remainder of the paper is organized as follows. In the next section, we present the main theorem which gives the expression of the transform $V^*(\mathbf{s}, Z; t)$. That theorem is used in Section 3 to derive formulas for the first two moments of various measures combining linear and instantaneous rewards. Section 4 is devoted to the total continuous reward distribution over $[0,t)$.

We first develop an algorithm based on explicit formulas leading to a stable method whose precision can be specified in advance. Secondly, we compute that distribution by the numerical inversion of Laplace-Stieltjes transform and we compare these two methods through numerical examples. Finally, we develop a new method based on a matrix convolution product. This method uses the explicit solution for moderate values of t and implements on that basis the matrix convolution product for larger values of t.

2 The Main Theorem

Theorem 2.1 *For $t \geq 0$, the matrix $V^*(\mathbf{s}, Z; t)$ is given by*

$$V^*(\mathbf{s}, Z; t) = \exp\{[Q \bullet Z - \Delta(\mathbf{a})\Delta(\mathbf{s})]t\}. \tag{1}$$

Proof. The conditional probability

$$P\{R_\nu(t) \leq x_\nu, 1 \leq \nu \leq m; N_{hk}(t) = K_{hk}, 1 \leq h, k \leq m; J(t) = j | J(0) = i\}$$

depends on t, on the m nonnegative variables $\{x_\nu\}$, and on the $m(m-1)$ nonnegative integer-valued variables $\{K_{hk}\}$. We concisely denote that probability mass-function by $V_{ij}(\mathbf{x}, K; t)$. Moreover, let $\mathbf{e}_i$ be the unit vector with i-th component equal to one and denote by $J(i, r)$ an $m \times m$ matrix with a single non-zero element equal to one at the indices i, r. The notation $U(\mathbf{x} - \mathbf{b})$ signifies the m-variate degenerate distribution at $\mathbf{b}$.

Now distinguishing the cases where the state of the Markov chain does not change in $[0, t)$ and where there is a first state change at some time u, $0 \leq u \leq t$, and applying a standard first passage argument, we obtain the equation

$$\begin{aligned} V_{ij}(\mathbf{x}, K; t) &= \delta_{ij} \exp(Q_{ii} t) U(\mathbf{x} - a_i t \mathbf{e}_i) \\ &\quad + \sum_{r \neq i} \int_0^t \exp(Q_{ii} u) Q_{ir} V_{rj}(\mathbf{x} - a_i u \mathbf{e}_i, K - J(i, r); t - u) du. \end{aligned} \tag{2}$$

By a simple change of variable, the second term may be rewritten as

$$\sum_{r \neq i} \int_0^t \exp[Q_{ii}(t - u)] Q_{ir} V_{rj}(\mathbf{x} - a_i(t - u)\mathbf{e}_i, K - J(i, r); u) du.$$

To facilitate the derivation of equation (1) we introduce and evaluate the transforms

$$V_{ij}^0(\mathbf{s}, Z; t) = \int_0^\infty \cdots \int_0^\infty \sum_K \prod_{h,k} z_{hk}^{K_{hk}} \exp\left(-\sum_{\nu=1}^m s_\nu x_\nu\right) V_{ij}(\mathbf{x}, K; t) dx_1 \cdots dx_m.$$

With respect to the variables $x_1, \ldots, x_m$, these are Laplace, not Laplace-Stieltjes, transforms. Equation (2) leads to

$$V^0_{ij}(\mathbf{s}, Z; t) = \delta_{ij} \exp[(Q_{ii} - a_i s_i)t](s_1 \cdots s_m)^{-1}$$
$$+\sum_{r\neq i} \sum_{K} \prod_{h,k} z_{hk}^{K_{hk}} \int_0^t \exp(Q_{ii}u) Q_{ir} du \int_0^\infty \cdots \int_{a_i u}^\infty \cdots \int_0^\infty \exp\left(-\sum_{\nu=1}^m s_\nu x_\nu\right)$$
$$\times V_{rj}(\mathbf{x} - a_i u \mathbf{e}_i, K - J(i,r); t-u) dx_1 \cdots dx_m.$$

By routine changes of variables that reduces to

$$V^0_{ij}(\mathbf{s}, Z; t) = \delta_{ij} \exp[(Q_{ii} - a_i s_i)t](s_1 \cdots s_m)^{-1}$$
$$+ \sum_{r\neq i} \int_0^t \exp[(Q_{ii} - a_i s_i)u] Q_{ir} z_{ir} V^0_{ij}(\mathbf{s}, Z; t-u) du. \quad (3)$$

In equation (3) we multiply both sides by $\exp[(a_i s_i - Q_{ii})t]$ and we differentiate the resulting equation with respect to t. Routine simplifications lead to the differential equations

$$\frac{d}{dt} V^0_{ij}(\mathbf{s}, Z; t) = -a_i s_i V^0_{ij}(\mathbf{s}, Z; t) + [(Q \bullet Z) V^0(\mathbf{s}, Z; t)]_{ij}, \quad (4)$$

for $1 \leq i, j \leq m$ with initial conditions $V^0_{ij}(\mathbf{s}, Z; 0) = \delta_{ij}(s_1 \cdots s_m)^{-1}$.

The Laplace-Stieltjes transforms $V^*_{ij}(\mathbf{s}, Z; t)$ are related to the Laplace transforms $V^0_{ij}(\mathbf{s}, Z; t)$ by $V^*_{ij}(\mathbf{s}, Z; t) = s_1 \cdots s_m V^0_{ij}(\mathbf{s}, Z; t)$. They satisfy the same differential equations with constant coefficients as in (4) but with initial conditions $V^*_{ij}(\mathbf{s}, Z; 0) = \delta_{ij}$. Integrating these equations we obtain (1). ■

Corollary 2.2 *The joint Laplace-Stieltjes transform of the total continuous rewards $R_\nu(t)$ and the total instantaneous rewards $c_{hk}N_{hk}(t)$ is given by the matrix $V^*(\mathbf{s}, \Xi; t)$, where the matrix Ξ is obtained by setting $z_{hk} = \exp(-c_{hk}\xi_{hk})$, for $1 \leq h, k \leq m$. The ξ_{hk} are the transform variables corresponding to the total instantaneous rewards $c_{hk}N_{hk}(t)$.*

Proof. Obvious from the definition of the Laplace-Stieltjes transform and the fact that

$$\exp\{-c_{hk} N_{hk}(t) \xi_{hk}\} = [\exp(-c_{hk}\xi_{hk})]^{N_{hk}(t)},$$

for $1 \leq h, k \leq m$. ■

3 Moment Formulas

We derive formulas for the mean and variance of the total reward accrued during an interval $[0, t)$ in the stationary version of the process. Using special choices of the quantities a_i and c_{hk}, we can immediately obtain the first two moments of various interesting quantities associated with finite Markov chains.

The matrix $\Xi^o(s)$ has elements $\exp(-c_{hk}s)$. The vector $\boldsymbol{\theta}$ is the stationary probability vector of the matrix Q and $\mathbf{e}$ is the column vector with all components equal to 1. The Laplace-Stieltjes transform of the total reward in the interval $[0, t)$ is then given by $\psi(s) = \boldsymbol{\theta} V^*(s, \Xi^o(s); t)\mathbf{e}$, where

$$V^*(s, \Xi^o(s); t) = \exp\{[Q \bullet \Xi^o(s) - s\Delta(\mathbf{a})]t\}.$$

The computation of the mean and variance amounts to evaluating the first two derivatives of $\psi(s)$ at zero. However, because of the matrix functions involved, that computation requires manipulations that we need to present in some detail. These are similar to those in Narayana and Neuts [4]. First some preliminaries: the quantity ω^* is defined by

$$\omega^* = \boldsymbol{\theta}[Q \bullet C + \Delta(\mathbf{a})]\mathbf{e} = \boldsymbol{\theta}(Q \bullet C)\mathbf{e} + \boldsymbol{\theta}\mathbf{a},$$

in which $C = \{c_{hk}\}$, where by convention, $c_{hh} = 0$ for $1 \leq h \leq m$. ω^* is the steady-state instantaneous reward rate. The first term is the contribution of the instantaneous rewards; the second term corresponds to the piecewise linear rewards. It is well-known that the matrix $\mathbf{e}\boldsymbol{\theta} - Q$ is invertible and that

$$\int_0^t \exp(Qu)du = \mathbf{e}\boldsymbol{\theta}t + [I - \exp(Qt)](\mathbf{e}\boldsymbol{\theta} - Q)^{-1}. \tag{5}$$

Theorem 3.1 *The mean total reward in $[0, t)$ is given by $\mu_1'(t) = \omega^* t$, and the corresponding variance $\sigma^2(t)$ by*

$$\begin{aligned}\sigma^2(t) &= t\boldsymbol{\theta}(Q \bullet C \bullet C)\mathbf{e} \\ &\quad + 2t\left\{\boldsymbol{\theta}[Q \bullet C + \Delta(\mathbf{a})](\mathbf{e}\boldsymbol{\theta} - Q)^{-1}[Q \bullet C + \Delta(\mathbf{a})]\mathbf{e} - {\omega^*}^2\right\} \\ &\quad - 2\boldsymbol{\theta}[Q \bullet C + \Delta(\mathbf{a})](\mathbf{e}\boldsymbol{\theta} - Q)^{-1}[I - \exp(Qt)](\mathbf{e}\boldsymbol{\theta} - Q)^{-1}[Q \bullet C + \Delta(\mathbf{a})]\mathbf{e}.\end{aligned}$$

Proof. We introduce the matrices $M_1(t)$ and $M_2(t)$, defined by

$$M_1(t) = -[\frac{\partial}{\partial s}V^*(s, \Xi^o(s); t)]_{s=0}, \qquad M_2(t) = [\frac{\partial^2}{\partial s^2}V^*(s, \Xi^o(s); t)]_{s=0}.$$

We twice differentiate with respect to s in the differential equation

$$\frac{\partial}{\partial t}V^*(s, \Xi^o(s); t) = V^*(s, \Xi^o(s); t)[Q \bullet \Xi^o(s) - s\Delta(\mathbf{a})],$$

we set $s = 0$, and we notice that

$$[\frac{\partial}{\partial s}\Xi^o(s)]_{s=0} = -C, \quad [\frac{\partial^2}{\partial s^2}\Xi^o(s)]_{s=0} = C \bullet C,$$

to obtain the differential equations

$$\frac{d}{dt}M_1(t) = M_1(t)Q + \exp(Qt)[Q \bullet C + \Delta(\mathbf{a})],$$

and

$$\frac{d}{dt}M_2(t) = M_2(t)Q + 2M_1(t)[Q \bullet C + \Delta(\mathbf{a})] + \exp(Qt)(Q \bullet C \bullet C).$$

We postmultiply by $\exp(-Qt)$ in both equations and integrate. That leads to

$$M_1(t) = \int_0^t \exp(Qu)[Q \bullet C + \Delta(\mathbf{a})]\exp[Q(t-u)]du, \tag{6}$$

and

$$\begin{aligned} M_2(t) - M_1(t) = & \int_0^t [2M_1(u) - \exp(Qu)][Q \bullet C + \Delta(\mathbf{a})]\exp[Q(t-u)]du \\ & + \int_0^t \exp(Qu)(Q \bullet C \bullet C)\exp[Q(t-u)]du. \end{aligned} \tag{7}$$

We premultiply by $\boldsymbol{\theta}$ in (6) and invoke the integration formula (5) to obtain that

$$\boldsymbol{\theta}M_1(t) = \omega^*\boldsymbol{\theta}t + \boldsymbol{\theta}[Q \bullet C + \Delta(\mathbf{a})][I - \exp(Qt)](\mathbf{e}\boldsymbol{\theta} - Q)^{-1}. \tag{8}$$

The equality (8) readily implies that $\boldsymbol{\theta}M_1(t)\mathbf{e} = \omega^* t$. Premultiplying by $\boldsymbol{\theta}$ in (7) leads to

$$\boldsymbol{\theta}M_2(t)\mathbf{e} = \boldsymbol{\theta}(Q \bullet C \bullet C)\mathbf{e}t + 2\boldsymbol{\theta}\int_0^t M_1(u)du[Q \bullet C + \Delta(\mathbf{a})]\mathbf{e}. \tag{9}$$

The integral is evaluated by using formulas (8) and (5) and performing routine simplifications. We obtain that

$$\begin{aligned} \boldsymbol{\theta}\int_0^t M_1(u)du = & \frac{1}{2}\omega^* t^2\boldsymbol{\theta} + \boldsymbol{\theta}[Q \bullet C + \Delta(\mathbf{a})](\mathbf{e}\boldsymbol{\theta} - Q)^{-1}t - \omega^* t\boldsymbol{\theta} \\ & - \boldsymbol{\theta}[Q \bullet C + \Delta(\mathbf{a})][I - \exp(Qt)](\mathbf{e}\boldsymbol{\theta} - Q)^{-2}. \end{aligned}$$

Upon substitution into the formula (9) for the second moment, the stated formula for the variance is obtained after simplifications. ∎

For selected choices of the parameter a_i and c_{hk}, we obtain moment formulas of special interest. For example, setting all $c_{hk} = 0$, and $a_i = 1$, for i belonging to a set B of indices and 0 otherwise, we obtain the moments of the total sojourn time of the Markov chain in the set of states B. Setting all $a_i = 0$, and $c_{hk} = 1$ if k belongs to B, and 0 otherwise, we obtain moments of the total number of visits to the set B during $[0, t)$.

4 The Total Continuous Reward Distribution

We recall that henceforth all the instantaneous rewards c_{hk} are zero. In this section we consider the semi-Markov matrix $W(x,t) = \Big(W_{ij}(x,t) \Big)$ where

$$W_{ij}(x,t) = P\{R(t) \le x, J(t) = j | J(0) = i\}.$$

We partition the state space $S = \{1, \dots, m\}$ of the Markov chain $\{J(t)\}$ into disjoint subsets containing the states with the same reward rates. The number of distinct rewards is $\phi + 1$ and their different values are

$$r_0 < r_1 < \cdots < r_{\phi-1} < r_\phi.$$

States in the subsets B_l, $l = 0, \dots, \phi$, have the same reward rate r_l. That is, for $l = 0, \dots, \phi$,

$$B_l = \{i \in S | a_i = r_l\}.$$

We then have $R(t) \in [r_0 t, r_\phi t]$ with probability one. Without loss of generality, we may set $r_0 = 0$. That can be done by considering the random variable $R(t) - r_0 t$ instead of $R(t)$ and the reward rates $r_l - r_0$ instead of r_l.

We denote by P the transition probability matrix of the uniformized discrete time Markov chain associated to the Markov chain $\{J(t)\}$, with the same initial distribution. The matrix P is related to the generator Q by $P = I + Q/\lambda$, where I is the identity matrix and λ satisfies $\lambda \ge \max\{-Q_{ii},\ i \in S\}$.

Using the partition $B_0, \dots, B_\phi$, the matrices Q, P, and $W(x,t)$ can be written, for $u, v = 0, \dots, \phi$, as

$$Q = \{Q_{B_u B_v}\}, \ P = \{P_{B_u B_v}\} \text{ and } W(x,t) = \{W_{B_u B_v}(x,t)\}.$$

The distribution of $R(t)$ has at most $\phi+1$ jumps at the points $r_0 t = 0, r_1 t, \dots, r_\phi t$. For $t > 0$, the jump at $x = r_l t$ is the probability that the Markov chain $\{J(t)\}$, starting in subset B_l, stays in that set during all of $[0, t)$. Therefore, for $t > 0$, and $0 \le l \le \phi$,

$$P\{R(t) = r_l t, J(t) = j | J(0) = i\} = \begin{cases} (e^{Q_{B_l B_l} t})_{ij} & \text{if } i, j \in B_l, \\ 0 & \text{otherwise,} \end{cases} \tag{10}$$

which can also be written as

$$P\{R(t)=r_l t, J(t)=j|J(0)=i\} = \sum_{n=0}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}(P_{B_l B_l})^n_{ij} 1_{\{i,j\in B_l\}}.$$

4.1 Explicit Formulas

An explicit formula for the matrix $W(x,t)$, is given by the following theorem. It is derived in [5].

Theorem 4.1 *For every $t>0$, and $x\in[r_{h-1}t, r_h t)$, for $1\le h\le \phi$,*

$$W(x,t)=\sum_{n=0}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=0}^{n}\binom{n}{k} x_h^k (1-x_h)^{n-k} C^{(h)}(n,k), \tag{11}$$

where $x_h = \dfrac{x-r_{h-1}t}{(r_h-r_{h-1})t}$ and $C^{(h)}(n,k) = \left(C^{(h)}_{B_u B_v}(n,k)\right)_{0\le u,v\le\phi}$ are matrices given by the recurrence relations:
For $h\le u\le\phi$, and $0\le v\le\phi$:

for $n\ge 0 : C^{(1)}_{B_u B_v}(n,0) = 0_{B_u B_v},\ C^{(h)}_{B_u B_v}(n,0) = C^{(h-1)}_{B_u B_v}(n,n),$ *for* $h>1$;

for $1\le k\le n$:

$$C^{(h)}_{B_u B_v}(n,k)=\frac{r_u-r_h}{r_u-r_{h-1}}C^{(h)}_{B_u B_v}(n,k-1)+\frac{r_h-r_{h-1}}{r_u-r_{h-1}}\sum_{w=0}^{\phi}P_{B_u B_w}C^{(h)}_{B_w B_v}(n-1,k-1). \tag{12}$$

For $0\le u\le h-1$, and $0\le v\le\phi$:

for $n\ge 0 : C^{(\phi)}_{B_u B_v}(n,n) = (P^n)_{B_u B_v},\ C^{(h)}_{B_u B_v}(n,n) = C^{(h+1)}_{B_u B_v}(n,0),$ *for* $h<\phi$;

for $0\le k\le n-1$:

$$C^{(h)}_{B_u B_v}(n,k)=\frac{r_{h-1}-r_u}{r_h-r_u}C^{(h)}_{B_u B_v}(n,k+1)+\frac{r_h-r_{h-1}}{r_h-r_u}\sum_{w=0}^{\phi}P_{B_u B_w}C^{(h)}_{B_w B_v}(n-1,k). \tag{13}$$

Proof. See [5] ■

In what follows, we denote by $W'(x,t)$ the partial derivative of $W(x,t)$ with respect to x. That matrix, defined only for $t>0$ and $x\neq r_l t$, $l=0,\ldots,\phi$, is given in the following corollary.

Corollary 4.2 *For $t > 0$, and $x \in (r_{h-1}t, r_h t)$, for $1 \le h \le \phi$, we have*

$$W'(x,t) = \frac{\lambda}{r_h - r_{h-1}} \sum_{n=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} x_h^k (1-x_h)^{n-k} \times \left[C^{(h)}(n+1,k+1) - C^{(h)}(n+1,k) \right]. \tag{14}$$

Proof. Obvious from relation (11). ■

Note that in (12), that is for $h \le u$, we have

$$0 \le \frac{r_u - r_h}{r_u - r_{h-1}} = 1 - \frac{r_h - r_{h-1}}{r_u - r_{h-1}} \le 1,$$

and in (13), that is for $u \le h-1$, we have

$$0 \le \frac{r_{h-1} - r_u}{r_h - r_u} = 1 - \frac{r_h - r_{h-1}}{r_h - r_u} \le 1.$$

The following corollary gives some properties of the matrices $C^{(h)}(n,k)$. If M and K are square matrices of the same order, $M \le K$ signifies element-wise inequality.

Corollary 4.3 *For every $n \ge 0$, $0 \le k \le n$, and $1 \le h \le \phi$,*

$$0 \le C^{(h)}(n,k) \le P^n, \quad 0 \le k \le n,$$

$$C^{(h)}(n,k) \le C^{(h)}(n,k+1), \quad 0 \le k \le n-1.$$

Proof. For the first inequality, see [5]. The same recurrence mechanism is used to prove the second one. ■

These considerations yield a computational method that avoids numerical problems since, except for the ratio $\lambda/(r_h - r_{h-1})$ in (14), all the computed quantities are between 0 and 1 and require only additions and multiplications of nonnegative quantities. This leads to a stable algorithm whose precision can be specified in advance.

Let ε be the desired precision for the computation of $W(x,t)$. We define the integer N by

$$N = \min \left\{ n \ge 0 \;\middle|\; \sum_{i=0}^{n} e^{-\lambda t} \frac{(\lambda t)^i}{i!} \ge 1 - \varepsilon \right\}. \tag{15}$$

We thus have

$$W(x,t) = \sum_{n=0}^{N} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} x_h^k (1-x_h)^{n-k} C^{(h)}(n,k) + e(N).$$

From the first inequality of Corollary 4.3, we obtain that the remainder of the series $e(N)$ satisfies $e_{ij}(N) \le \varepsilon$, for every $i, j \in S$.

With regards to $W'(x,t)$, again from Corollary 4.3, we have that

$$0 \le C^{(h)}(n+1, k+1) - C^{(h)}(n+1, k) \le P^{n+1}, \;\; 0 \le k \le n,$$

and so we obtain that

$$W'(x,t) = \frac{\lambda}{r_h - r_{h-1}} \sum_{n=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} x_h^k (1-x_h)^{n-k} \times \left[C^{(h)}(n+1,k+1) - C^{(h)}(n+1,k) \right] + e^1(N),$$

where the remainder of the series $e^1(N)$ is such that, for every $i, j \in S$,

$$e_{ij}^1(N) \le \frac{\lambda \varepsilon}{r_h - r_{h-1}} \le \frac{\lambda \varepsilon}{r},$$

and $r = \min\{r_h - r_{h-1}, \; h = 1, \dots, \phi\}$.

4.1.1 Algorithmic aspects

In this section we consider the computation of matrix $W(x,t)$. The main effort goes into the computation of the matrices $C^{(h)}(n,k)$. With regards to storage requirements, since the values of the $C^{(h)}(n,k)$ at step n depend only on their values at step $n-1$, we need to store only two arrays of $(N+1)\phi$ matrices. At step n, we need to compute $n+1$ matrices for each $h = 1, \dots, \phi$. That can easily be seen from the algorithmic description in Table 2. The procedure **Accumulate**(n) is used to compute the approximate matrix $W^{\varepsilon}(x,t)$ defined, for $h = 1, \dots, \phi$ and $x \in [r_{h-1}t, r_h t)$, by

$$W^{\varepsilon}(x,t) = \sum_{n=0}^{N} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} x_h^k (1-x_h)^{n-k} C^{(h)}(n,k).$$

By the definition of N in (15) and from Corollary 4.3, we have, for a given value of the precision ε, that

$$\sup_{i \in S} \sum_{j \in S} (W_{ij}(x,t) - W_{ij}^{\varepsilon}(x,t)) \le \varepsilon.$$

The procedure **Accumulate**(n), described in Table 1, involves a fixed value $t > 0$ and M distinct values of x, denoted by $x(i)$, $1 \le i \le M$. We initialize $W^{\varepsilon}(x(i), t) = 0$, we denote by h_i the index such that $x(i) \in [r_{h_i - 1}t, r_{h_i}t)$, and we define

$$x_{h_i} = \frac{x(i) - r_{h_i - 1}t}{(r_{h_i} - r_{h_i - 1})t}.$$

Table 1. The procedure **Accumulate**(n)

for $i = 1$ **to** M **do**

$$W^{\varepsilon}(x(i),t,n) = e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=0}^{n}\binom{n}{k}x_{h_i}^k(1-x_{h_i})^{n-k}C^{(h_i)}(n,k)$$

$$W^{\varepsilon}(x(i),t) = W^{\varepsilon}(x(i),t) + W^{\varepsilon}(x(i),t,n)$$

endfor

Table 2. Computation of the matrices $C^{(h)}(n,k)$ and $W(x,t)$

for $h = 1$ **to** ϕ **do** $\forall\ u, v = 0,\ldots,\phi$, $C^{(h)}_{B_uB_v}(0,0) = 0_{B_uB_v}$ **endfor**
for $h = 1$ **to** ϕ **do** $\forall\ u = 0,\ldots,h-1$, $C^{(h)}_{B_uB_u}(0,0) = I_{B_uB_v}$ **endfor**
Accumulate(0)
for $n = 1$ **to** N **do**
  $\forall\ u = 1,\ldots,\phi$, $\forall\ v = 0,\ldots,\phi$, $C^{(1)}_{B_uB_v}(n,0) = 0_{B_uB_v}$
  for $h = 1$ **to** ϕ **do**
    for $k = 1$ **to** n **do**
      $\forall\ u = h,\ldots,\phi$, $\forall\ v = 0,\ldots,\phi$, compute relation (12)
    endfor
    $\forall\ u = h+1,\ldots,\phi$, $\forall\ v = 0,\ldots,\phi$, $C^{(h+1)}_{B_uB_v}(n,0) = C^{(h)}_{B_uB_v}(n,n)$
  endfor
  $\forall\ u = 0,\ldots,\phi-1$, $\forall\ v = 0,\ldots,\phi$, $C^{(\phi)}_{B_uB_v}(n,n) = (P^n)_{B_uB_v}$
  for $h = \phi$ **downto** 1 **do**
    for $k = n-1$ **downto** 0 **do**
      $\forall\ u = 0,\ldots,h-1$, $\forall\ v = 0,\ldots,\phi$, compute relation (13)
    endfor
    $\forall\ u = 0,\ldots,h-2$, $\forall\ v = 0,\ldots,\phi$, $C^{(h-1)}_{B_uB_v}(n,n) = C^{(h)}_{B_uB_v}(n,0)$
  endfor
  Accumulate(n)
endfor

Note that the integer N, defined in (15), is an increasing function of t, say $N(t)$. So, if the matrix $W(x,t)$ is to be computed at L different $t-$values, say $t_1 < \ldots < t_L$, we need only evaluate the matrices $C^{(h)}(n,k)$ for $n = 0,1,\ldots,N(t_L)$, as these matrices do not depend on the values of $t_1,\ldots,t_L$.

The main effort required for the computation of matrices $W(x,t)$ or $W'(x,t)$ is in the computation of matrices $C^{(h)}(n,k)$. We use for matrix P

a compact storage. If d denotes the connectivity degree of matrix P, that is the maximum number of nonzero entries in each row, then the computational cost of one matrix $C^{(h)}(n,k)$ is $O(dm^2)$. The number of such matrices that have to be computed (see Table 2) is equal to $\phi(N+1)(N+2)/2$, The total computational effort required is thus $O(\phi dm^2N^2/2)$. Concerning the storage requirements, it is easy to see, from Table 2, that we need to store two arrays of $\phi(N+1)$ matrices for the recursive computation of matrices $C^{(h)}(n,k)$. Thus the storage complexity is $O(\phi m^2 N)$.

Note also that if one only wants to compute the distribution $P\{R(t) \le x\}$, there is no need to evaluate the matrices $C^{(h)}(n,k)$. It then suffices to evaluate the vectors $b^{(h)}(n,k) = C^{(h)}(n,k)\mathbf{e}$. The algorithm thereby becomes more efficient, as the matrix-matrix products are replaced by matrix-vector products. In that case, the end product of the algorithm is the vector $G(x,t) = W(x,t)\mathbf{e}$ and the complexity is reduced by a factor m.

4.1.2 Numerical examples

Consider the Markov chain with $S = \{1,2,3\}$, the generator Q and the reward vector a, given by

$$Q = \begin{pmatrix} -1 & 1 & 0 \\ 0.5 & -1 & 0.5 \\ 0 & 1 & -1 \end{pmatrix} \quad \text{and} \quad a = \begin{pmatrix} 2 & 1 & 0 \end{pmatrix}.$$

We thus have $\lambda = 1$, $\phi = 2$ and $0 \le R(t) \le 2t$ with probability 1. For the error tolerance $\varepsilon = 10^{-10}$, we obtain the following results.
For $t = 1$

$$W(0,1) = \begin{pmatrix} 0.0000000000 & 0.0000000000 & 0.0000000000 \\ 0.0000000000 & 0.0000000000 & 0.0000000000 \\ 0.0000000000 & 0.0000000000 & 0.3678794412 \end{pmatrix}.$$

Note the high precision of the algorithm: the element $W_{33}(0,1)$ is the jump corresponding to the MC staying in the state 3 up to time $t=1$. That is also equal to $\exp(-1) \approx 0.36787944117$.

$$W(1-10^{-12},1) = \begin{pmatrix} 0.0010069669 & 0.0163117922 & 0.0499470501 \\ 0.0081558961 & 0.0998941002 & 0.2080102831 \\ 0.0499470501 & 0.4160205662 & 0.4667665745 \end{pmatrix},$$

$$W(1,1) = \begin{pmatrix} 0.0010069669 & 0.0163117922 & 0.0499470501 \\ 0.0081558961 & 0.4677735414 & 0.2080102831 \\ 0.0499470501 & 0.4160205662 & 0.4667665745 \end{pmatrix}.$$

Again, note the high precision of the algorithm: all the elements of $W(1-10^{-12},1)$ and $W(1,1)$ are equal except for the element of indices $(2,2)$. The difference between these two values is 0.3678794412; it corresponds to the jump at $x = 1$, the probability that the MC stays in state 2 beyond time $t = 1$, or $\exp(-1) \approx 0.36787944117$.

$$W(2^-,1) = \begin{pmatrix} 0.0998941002 & 0.4323323583 & 0.0998941002 \\ 0.2161661792 & 0.5676676416 & 0.2161661792 \\ 0.0998941002 & 0.4323323583 & 0.4677735414 \end{pmatrix}.$$

The elements of the matrix $W(2^-,1)$ are

$$W_{ij}(2^-,1) = P\{R(t) < 2, J(1) = j \mid J(0) = i\}.$$

Since $W(2,1) = e^Q$, we easily obtain the jump

$$\begin{aligned} P\{R(t) = 2, J(1) = 1 \mid J(0) = 1\} &= \sum_{j=1}^{3} W_{1j}(2,1) - \sum_{j=1}^{3} W_{1j}(2^-,1) \\ &= 1 - \sum_{j=1}^{3} W_{1j}(2^-,1) = 0.3678794412, \end{aligned}$$

whose value is $\exp(-1) \approx 0.36787944117$.

For $t = 100$, we obtain $W_{ij}(0,100) = 0$ for every $1 \le i,j \le 3$, and

$$W(100,100) = \begin{pmatrix} 0.1050278103 & 0.2299261526 & 0.1250000000 \\ 0.1149630763 & 0.2500000000 & 0.1350369237 \\ 0.1250000000 & 0.2700738474 & 0.1449721896 \end{pmatrix},$$

$$W(200^-,100) = \begin{pmatrix} 0.2500000000 & 0.5000000000 & 0.2500000000 \\ 0.2500000000 & 0.5000000000 & 0.2500000000 \\ 0.2500000000 & 0.5000000000 & 0.2500000000 \end{pmatrix}.$$

Note that in this case the jumps are invisible since $\exp(-100) \approx 0.372 \times 10^{-43}$.

4.2 Numerical transform inversion

The joint Laplace transform and generating function $V^*(\boldsymbol{s}, Z; t)$ can be numerically inverted, at least in some special cases. We only consider the case where we want to find the distribution (density) of the total continuous reward earned in some subset of the state–space, A say. In this case the joint transform of course reduces to a Laplace–transform for this reward. Since the density of interest is concentrated on the positive real axis we can use the

Bromwich inversion integral as follows. For simplicity let $V^*(t)$ denote the Laplace transform for the total continuous reward. The ij'th element is hence the Laplace transform corresponding to the case where the Markov jump process initiates in state i and is in state j at time t. Let the corresponding (defective) density of total reward earned in the set A be $\boldsymbol{f}(t)$, whose i,j's element corresponds to the conditional density given initiation of the Markov jump process in state i, and subject to being in state j at time t. Then the Bromwich inversion integral is

$$\boldsymbol{f}(t) = \frac{2}{\pi}\int_0^\infty \mathrm{Re}(V^*(iu))\cos(ut)du.$$

This integral is hence solved by numerical integration (trapezoidal rule) choosing discretization such that the cosine term becomes $(-1)^k$ and we approximate the integral by an alternating series, which in turn is calculated by Euler summation to approximate the infinite series. See Abate and Whitt (1992) [1] for details.

The transform $V^*(iu)$ in the integral above is essentially a matrix–exponential of a complex matrix. Such a matrix–exponential is obviously the solution to a system of linear differential equations, and we solve for the matrix–exponential by solving the system of differential equation using a fourth order Runge–Kutta method. In order to speed up the procedure there is also a scaling consideration involved where we use the property of the matrix–exponential $\exp(\boldsymbol{\Gamma}t) = \exp(\boldsymbol{\Gamma}t/n)^n$. If we choose n to be a power of 2, $n = 2^k$ say, then the power of the exponential is particularly fast to calculate by repeated squaring of the exponential k times with itself.

The numerical inversion of the transform requires the evaluation of the exponential of a matrix, here carried out by the fourth order Runge–Kutta method, the complexity of which is $O(m^2n_1)$, where n_1 is the number of discretization steps for solving the differential equations. The storage requirement for evaluating the transform is $O(m^2)$ as we need to store the intensity matrix $(m \times m)$ and reward vector (dimension m). The numerical integration depends linearly on the number of steps involved when we consider the total reward earned in some states. If we let n_2 denote the number of integration steps and n_3 the number of density points to be produced, the complexity of the total algorithm is $O(m^2n_1n_2n_3)$, while the storage requirements remain $O(m^2)$.

4.3 Comparison of the Two Methods

Next, we compare the performance of the explicit method and the numerical transform inversion by means of two examples. We refer to the former as the *exact* method and to the latter as the *inversion* method.

4.3.1 A 3-state model

We again consider the example of section 4.1.2. We compute the conditional density $W'_{13}(x, 10)$ for various values of x. For the *exact* method, the precision ε was set to 10^{-10}, while the error for the *inversion* method is estimated in the course of the computation. We obtained $W'_{13}(x, 10)$ for many x; only a few numerical results are shown in Table 3. The third column lists the estimated error for the *inversion* method. Among all the computed values, the maximum absolute difference between the results of both methods is 1.3×10^{-7}. In particular, this suggests that the error estimates in the *inversion* method are not accurate. All the values in the third column of Table 3 are much smaller than 1.3×10^{-7}. Given the high accuracy of the *exact* method, it appears that the error in the *inversion* method is larger than reported.

Table 3. Numerical results for the 3-state model

x	*exact*	*inversion*	Error Estim. on Inv.
1	6.90841D-05	6.9084099823223D-05	6.7762635780344D-21
5	8.0198771D-03	8.0198758928317D-03	1.0842021724855D-18
10	3.39710030D-02	3.3970937106725D-02	1.9185555921730D-10
15	8.0198771D-03	8.0198749959861D-03	1.6532381453410D-10
16	3.9259414D-03	3.9259417172244D-03	1.8769383399939D-10
17	1.5424709D-03	1.5424716317178D-03	1.6531584738286D-10
18	4.395360D-04	4.3953734438919D-04	1.7517860938427D-10
19	6.90841D-05	6.9086110168773D-05	9.4898452709599D-11
20	0.0000000000	1.3247682929818D-07	1.9573838184247D-10

4.3.2 A stiff model

Consider a system with N processors that, independently of each other, are subject to failure and repair. The times to failure and the repair times of each processor are exponential, respectively, with parameters β and μ. There is a single repairman. We denote by $J(t)$ the number of operational processors at time t. The transition rates of the Markov chain $J(t)$ are shown in Figure 1. We assign a reward equal to 1 to states N and $N-1$ and equal to 0 to all other states.

Such a model is called stiff when the ratio between the largest and the

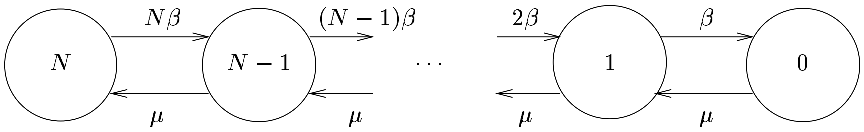

Figure 1. A stiff model

smallest transition rates is very large. By choosing $\beta = 10^{-9}$ and $\mu = 1$, we obtain an example of a *stiff* model.

We compute the conditional density of the total continuous reward earned up to time $t = 100$, that the state is then N, given that 1 is the initial state. For this example, $R(t)$ is called *the interval availability* over $[0, t)$. We computed $W'_{1N}(x, 100)$ for various values of the accumulated reward x and for $N = 10$. As before, for the *exact* method, the precision was specified as $\varepsilon = 10^{-10}$, while the error for the *inversion* method is estimated as we go.

We obtained $W'_{1N}(x, 100)$ for many values of x and, as for the preceding example, Table 4 lists only some representative values.

Among all computed values, the maximum absolute difference between the results of both methods is 5.5×10^{-6}. As for the previous example, this suggests that the error estimates in the *inversion* method are not accurate. All the values in the third column of Table 4 are much smaller than 5.5×10^{-6}. Given the high accuracy of the *exact* method, it appears again that the error in the *inversion* method is larger than reported.

Table 4. Numerical results for the stiff model

x	*exact*	*inversion*	Error Estim. on Inv.
70	4.061D-07	3.6675885125778D-07	1.0268233628745D-10
75	1.68185D-05	1.6742507306377D-05	1.4984542538130D-10
80	5.234676D-04	5.2333975210878D-04	1.7149709717655D-10
85	1.03702940D-02	1.0369952661451D-02	1.5675087616795D-10
90	9.00792255D-02	9.0078490790866D-02	1.0241073961081D-10
95	1.044448605D-01	1.0443933140437D-01	4.8018394815941D-11
96	0.0595403611D-02	5.9534948246358D-02	9.9827379002049D-11
97	0.0216040309D-02	2.1604262271277D-02	1.0371982179375D-10
98	0.0034370865D-03	3.4426169921810D-03	7.2716614413909D-11
99	0.0000729920D-05	7.7315641585370D-05	7.5063514621719D-11
100	0.0000000000	-2.2215910573491D-07	1.5172630167441D-10

These two examples show, as expected, that the *exact* method has a high precision that can be given in advance. So we can evaluate beforehand the time needed execute the corresponding algorithm. This execution time can be very important for large values of the mission time t and also for a large number of distinct rewards. Concerning the *inversion* method, it is not so accurate and the error estimated is not reliable, so we are not sure that it gives the correct result. The main advantage of that method is that the execution time is independent of the mission time t and of the number of distinct rewards ϕ.

4.4 Convolution Method

In this section we spell out the convolution properties of the matrix $W(x,t)$ and we use these properties to develop a new algorithm to deal with large values of t. The *exact* algorithm developed from the explicit formulas serves as a starting point for the convolution method. We thus initiate the convolutions with data having very high precision.

To simplify notation, we denote by Q_l the matrix $Q_{B_lB_l}$, for $0 \leq l \leq \phi$. For any real numbers a and b, we define $a \wedge b = \min(a,b)$. Recall that for $x \geq r_\phi(t+s)$, we have $W(x,t+s) = e^{Q(t+s)}$, and for $x = 0$, we have, $W(0,t+s) = e^{Q(t+s)}_{ij} 1_{\{i,j \in B_0\}}$.

Theorem 4.4 *For $0 < x < r_\phi(t+s)$, we have that*

$$W_{ij}(x,t+s) = \sum_{k \in S} \int_0^{x \wedge r_\phi t} W'_{ik}(v,t) W_{kj}(x-v,s)dv + \sum_{l=0}^{\phi} \sum_{k \in B_l} e^{Q_l t}_{ik} W_{kj}(x - r_l t, s) 1_{\{i \in B_l\}} 1_{\{x \geq r_l t\}}. \tag{16}$$

Proof. By $R(t,t+s)$, we denote the total continuous reward over the interval $(t,t+x]$. We thus have $R(t) = R(0,t)$ and $R(t+s) = R(t) + R(t,t+s)$. Using this relation, we have

$$\begin{aligned} W_{ij}(x,t+s) &= P\{R(t+s) \leq x, J(t+s) = j \mid J(0) = i\} \\ &= \sum_{k \in S} P\{R(t) + R(t,t+s) \leq x, J(t+s) = j, J(t) = k \mid J(0) = i\} \\ &= \sum_{k \in S} \int_{v \geq 0} P\{R(t,t+s) \leq x - v, J(t+s) = j | J(t) = k, R(t) = v, J(0) = i\} \\ &\qquad \times dP\{R(t) \leq v, J(t) = k \mid J(0) = i\} \end{aligned}$$

$$= \sum_{k\in S}\int_{v\geq 0} P\{R(t,t+s)\leq x-v, J(t+s)=j \mid J(t)=k\}$$
$$\times dP\{R(t)\leq v, J(t)=k \mid J(0)=i\}$$
$$= \sum_{k\in S}\int_{v\geq 0} P\{R(s)\leq x-v, J(s)=j \mid J(0)=k\}$$
$$\times dP\{R(t)\leq v, J(t)=k \mid J(0)=i\}$$
$$= \sum_{k\in S}\int_{v\geq 0} dW_{ik}(v,t)W_{kj}(x-v,s).$$

The fourth equality is due to the Markov property and the fifth comes from the homogeneity of J. The jumps arising in $W_{ik}(v,t)$ are described in relation (10). Using that relation, we have

$$\begin{aligned}
W_{ij}(x,t+s) &= \sum_{k\in S}\int_{v\geq 0} W'_{ik}(v,t)W_{kj}(x-v,s)dv \\
&\quad + \sum_{k\in S}\sum_{l=0}^{\phi} \Pr\{R(t)=r_l t, J(t)=k \mid J(0)=i\}W_{kj}(x-r_l t,s) \\
&= \sum_{k\in S}\int_{v\geq 0} W'_{ik}(v,t)W_{kj}(x-v,s)dv \\
&\quad + \sum_{k\in S}\sum_{l=0}^{\phi} e^{Q_l t}_{ik} 1_{\{i,k\in B_l\}} W_{kj}(x-r_l t,s) \\
&= \sum_{k\in S}\int_{v\geq 0} W'_{ik}(v,t)W_{kj}(x-v,s)dv \\
&\quad + \sum_{l=0}^{\phi}\sum_{k\in B_l} e^{Q_l t}_{ik} W_{kj}(x-r_l t,s)1_{\{i\in B_l\}},
\end{aligned}$$

and the result follows since $W_{kj}(x-v,s)=0$, for $v>x$, $W'_{ik}(v,t)=0$, for $v>r_\phi t$ and, $W_{kj}(x-r_l t,s)=0$, for $x<r_l t$. ■

The following corollary is a simplified version of relation (16). As usual, we define $x^+=\max(0,x)$, for any real number x.
Corollary 4.5 *For $0<x<r_\phi(t+s)$, we have*

$$W_{ij}(x,t+s) = \sum_{k\in S}\int_{(x-r_\phi s)^+}^{x\wedge r_\phi t} W'_{ik}(v,t)W_{kj}(x-v,s)dv$$

$$
\begin{aligned}
&+\sum_{k\in S} W_{ik}((x-r_\phi s)^+,t)e^{Qs}_{kj}\\
&+\sum_{l=0}^{\phi}\sum_{k\in B_l} e^{Q_l t}_{ik} W_{kj}(x-r_l t,s)1_{\{i\in B_l\}}1_{\{r_l t\le x<r_l t+r_\phi s\}}, \quad (17)
\end{aligned}
$$

Proof. Consider relation (16) and denote by β the integral part and by α the other part corresponding to the jumps. We thus have

$$
\alpha=\sum_{l=0}^{\phi}\sum_{k\in B_l} e^{Q_l t}_{ik} W_{kj}(x-r_l t,s)1_{\{i\in B_l\}}1_{\{x\ge r_l t\}},
$$

and

$$
\beta=\sum_{k\in S}\int_0^{x\wedge r_\phi t} W'_{ik}(v,t)W_{kj}(x-v,s)dv.
$$

Since $W_{kj}(x-r_l t,s)=e^{Qs}_{kj}$, if $x\ge r_l t+r_\phi s$ and as $x<r_\phi(t+s)$, we get

$$
\begin{aligned}
\alpha&=\sum_{l=0}^{\phi}\sum_{k\in B_l} e^{Q_l t}_{ik} W_{kj}(x-r_l t,s)1_{\{i\in B_l\}}1_{\{r_l t\le x<r_l t+r_\phi s\}}\\
&\quad+\sum_{l=0}^{\phi}\sum_{k\in B_l} e^{Q_l t}_{ik} e^{Qs}_{kj}1_{\{i\in B_l\}}1_{\{x\ge r_l t+r_\phi s\}}\\
&=\sum_{l=0}^{\phi}\sum_{k\in B_l} e^{Q_l t}_{ik} W_{kj}(x-r_l t,s)1_{\{i\in B_l\}}1_{\{r_l t\le x<r_l t+r_\phi s\}}\\
&\quad+\sum_{l=0}^{\phi-1}\sum_{k\in B_l} e^{Q_l t}_{ik} e^{Qs}_{kj}1_{\{i\in B_l\}}1_{\{x\ge r_l t+r_\phi s\}}. \qquad (18)
\end{aligned}
$$

In the same way, since $W_{kj}(x-v,s)=e^{Qs}_{kj}$, if $v\le x-r_\phi s$, we get

$$
\begin{aligned}
\beta&=\sum_{k\in S}\left(\int_0^{(x-r_\phi s)^+} W'_{ik}(v,t)dv\right)e^{Qs}_{kj}\\
&\quad+\sum_{k\in S}\int_{(x-r_\phi s)^+}^{x\wedge r_\phi t} W'_{ik}(v,t)W_{kj}(x-v,s)dv.
\end{aligned}
$$

Let us denote by θ the integral arising in the first sum. We have

$$
\theta=\int_0^{(x-r_\phi s)^+} W'_{ik}(v,t)dv
$$

$$= W_{ik}((x-r_\phi s)^+,t) - \sum_{h=1}^{\phi}\sum_{l=0}^{h-1} e_{ik}^{Q_l t} 1_{\{i,k\in B_l\}} 1_{\{(x-r_\phi s)^+\in[r_{h-1}t, r_h t)\}}$$

$$= W_{ik}((x-r_\phi s)^+,t) - \sum_{l=0}^{\phi-1}\sum_{h=l+1}^{\phi} e_{ik}^{Q_l t} 1_{\{i,k\in B_l\}} 1_{\{(x-r_\phi s)^+\in[r_{h-1}t, r_h t)\}}$$

$$= W_{ik}((x-r_\phi s)^+,t) - \sum_{l=0}^{\phi-1} e_{ik}^{Q_l t} 1_{\{i,k\in B_l\}} 1_{\{x\geq r_l t + r_\phi s\}}.$$

Finally, we obtain that

$$\beta = \sum_{k\in S} W_{ik}((x-r_\phi s)^+,t) e_{kj}^{Qs} - \sum_{l=0}^{\phi-1}\sum_{k\in B_l} e_{ik}^{Q_l t} e_{kj}^{Qs} 1_{\{i\in B_l\}} 1_{\{x\geq r_l t + r_\phi s\}}$$
$$+ \sum_{k\in S}\int_{(x-r_\phi s)^+}^{x\wedge r_\phi t} W'_{ik}(v,t) W_{kj}(x-v,s)dv. \qquad (19)$$

By adding the expressions (18) and (19), we obtain the desired result. ■

4.4.1 Algorithmic aspects

We wish to compute $W_{i,j}(x,2t)$ for some values of x, where $0 < x < 2r_\phi t$, assuming that we know $W'_{ij}(x,t)$, $W_{ij}(x,t)$ at the same points x. For this purpose, we use the relations (16) and (17) for $s=t$. Let us consider formula (16). The main difficulty consists in the evaluation of matrices $U(x,t)$ defined by

$$U_{i,j}(x,t) = \sum_{k\in S}\int_0^{x\wedge r_\phi t} W'_{ik}(v,t) W_{kj}(x-v,t)dv.$$

In order to simplify notations, let us write $W'_{ij}(x)$ instead of $W'_{ij}(x,t)$, $W_{ij}(x)$ instead of $W_{ij}(x,t)$ and $U_{ij}(x)$ instead of $U_{ij}(x,t)$. Moreover, we can write that

$$U(x) = \int_0^{x\wedge r_\phi t} W'(v) W(x-v)dv$$

where the integral of a matrix function is the matrix whose entries are the integrals of the entries of the matrix function.

Let us partition the interval $[0, 2r_\phi t]$ into subintervals $[x_i, x_{i+1}]$, $i = 0,\ldots,N-1$, where $x_i = 2ir_\phi t/N$, $i=0,\ldots,N$. In this way $x_0 = 0$, $x_{N/2} = r_\phi t$, $x_N = 2r_\phi t$.

We wish to approximate $U(x_i)$ by using the trapezoidal rule with knots x_k, $k = 0, \ldots, \min(i, N/2)$ (the trapezoidal rule can be naturally extended to the case of matrix functions).

Let us first consider the case $i \geq N/2$, i.e., $x_i \geq r_\phi t$. Then

$$U(x_i) = \int_0^{r_\phi t} W'(v) W(x_i - v) dv$$

can be approximated by

$$\begin{aligned}\widetilde{U}(x_i) &= \frac{2r_\phi t}{N} \left[W'(x_0) W(x_i) + W'(x_{N/2}) W(x_i - x_{N/2}) \right] \\ &+ \frac{4r_\phi t}{N} \sum_{k=1}^{N/2-1} W'(x_k) W(x_i - x_k). \end{aligned} \tag{20}$$

Now, $x_i - x_j = (2t(i-j)r_\phi)/N = x_{i-j}$. Thus, if we set $w_i = W(x_i)$, $w'_i = W'(x_i)$, and $\widetilde{u}_i = \widetilde{U}(x_i)$, $i = 0, \ldots, N$, we can write (20) as

$$\widetilde{u}_i = \frac{2r_\phi t}{N} \left(w'_0 w_i + w'_{N/2} w_{i-N/2} + 2 \sum_{k=1}^{N/2-1} w'_k w_{i-k} \right), \quad i = N/2, \ldots, N-1. \tag{21}$$

If $i = 1, \ldots, N/2 - 1$, then

$$U(x_i) = \int_0^{x_i} W'(v) W(x_i - v, s) dv$$

can be approximated by

$$\widetilde{U}(x_i) = \frac{2r_\phi t}{N} \left(W'(x_0) W(x_i) + W'(x_i) W(x_0) + 2 \sum_{k=1}^{i-1} W'(x_k) W(x_i - x_k) \right),$$

which yields

$$\widetilde{u}_i = \frac{2r_\phi t}{N} \left(w'_0 w_i + w'_i w_0 + 2 \sum_{k=1}^{i-1} w'_k w_{i-k} \right), \quad i = 1, \ldots, N/2 - 1. \tag{22}$$

If we write (21,22) in matrix form we obtain that

$$\begin{bmatrix} \widetilde{u}_1 \\ \widetilde{u}_2 \\ \vdots \\ \vdots \\ \widetilde{u}_{N-2} \\ \widetilde{u}_{N-1} \end{bmatrix} = \frac{2r_\phi t}{N} \left(\begin{bmatrix} w'_1 \\ \vdots \\ w'_{N/2} \\ 0 \\ \vdots \\ 0 \end{bmatrix} w_0 + T_N \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ \vdots \\ w_{N-2} \\ w_{N-1} \end{bmatrix} \right),$$

where

$$T_N = \begin{bmatrix} w'_0 & & & & & & \bigcirc \\ 2w'_1 & w'_0 & & & & & \\ \vdots & 2w'_1 & \ddots & & & & \\ 2w'_{N/2-1} & \vdots & \ddots & \ddots & & & \\ w'_{N/2} & 2w'_{N/2-1} & & \ddots & \ddots & & \\ 0 & \ddots & \ddots & & \ddots & \ddots & \\ \vdots & \ddots & \ddots & \ddots & & \ddots & w'_0 \\ 0 & \dots & 0 & w'_{N/2} & 2w'_{N/2-1} & \dots\ 2w'_1 & w'_0 \end{bmatrix}$$

is a block $(N-1) \times (N-1)$ banded lower triangular block Toeplitz matrix.

The product between T_N and the block vector $w_N = (w_i)_{i=1,N-1}$ can be computed by means of FFT's of length N with a computational cost of $O(m^2 N \log N + m^3 N)$ arithmetic operations, where the size of the blocks w_i is equal to m, the size of the state space of the Markov chain $\{J(t)\}$. More specifically, we may define the $N \times N$ block triangular Toeplitz matrix T_N^*, obtained by adding a block row and a block column to T_N, and consider the problem of computing the product $T_N^* w_N^*$, where $w_N^* = (w_i^*)_{i=1,N}$ is the block vector obtained by appending to w_N a null block component: in fact, the vector $T_N w_N$ is given by the first $N-1$ block components of $T_N^* w_N^*$. Assuming that N is a power of 2, we may partition the matrix T_N^* into a 2×2 block matrix,

$$T_N^* = \begin{bmatrix} T_{N,1} & 0 \\ T_{N,2} & T_{N,1} \end{bmatrix},$$

where $T_{N,1}$ and $T_{N,2}$ are respectively block lower and upper triangular block Toeplitz matrices with block size $N/2$, and write the vector $T_N^* w_N^*$ as

$$T_N^* w_N^* = \begin{bmatrix} T_{N,1} w_{N,1} \\ T_{N,2} w_{N,1} + T_{N,1} w_{N,2} \end{bmatrix},$$

where $w_{N,1} = (w_i^*)_{i=1,N/2}$, $w_{N,2} = (w_i^*)_{i=N/2+1,N}$. The block vectors $T_{N,1} w_{N,1}$, $T_{N,2} w_{N,1}$ and $T_{N,1} w_{N,2}$ can be obtained by means of the relation

$$C_N \left[\begin{array}{c|c} w_{N,1} & 0 \\ 0 & w_{N,2} \end{array}\right] = \left[\begin{array}{c|c} T_{N,1} w_{N,1} & T_{N,2} w_{N,2} \\ T_{N,2} w_{N,1} & T_{N,1} w_{N,2} \end{array}\right], \tag{23}$$

where

$$C_N = \begin{bmatrix} T_{N,1} & T_{N,2} \\ T_{N,2} & T_{N,1} \end{bmatrix}$$

is the block circulant matrix defined by the first block column of T_N^*. Since C_N is block circulant, the right hand side in (23) can be computed according to the following scheme:

1. *DFT associated with C_N:* Evaluate the matrix polynomial $w'_N(z) = w'_0 + 2\sum_{i=1}^{N/2-1} w'_i z^i + w'_{N/2} z^{N/2}$ at the N-th roots of 1, by means of m^2 FFT's of length N;
2. *DFT's associated with $w_{N,1}$ and $w_{N,2}$*: evaluate the matrix polynomials $w_{N,1}(z) = \sum_{i=0}^{N/2-1} w_{i+1} z^i$ and $w_{N,2}(z) = \sum_{i=N/2}^{N-2} w_{i+1} z^i$ at the N-th roots of 1, by means of $2m^2$ FFT's of length N;
3. *Convolution:* Compute the values of the matrix polynomials $p_1(z) = w'_N(z)w_{N,1}(z)$ and $p_2(z) = w'_N(z)w_{N,2}(z)$ at the N-th roots of 1 by means of $2N$ matrix products;
4. *IDFT's*: Interpolate the values obtained at the previous step by means of $2m^2$ FFT's of length N, thus obtaining the block coefficients of $p_1(z)$ and $p_2(z)$; the block coefficients of $p_1(z)$ and $p_2(z)$ coincide with the block entries of the vectors $\begin{bmatrix} T_{N,1}w_{N,1} \\ T_{N,2}w_{N,1} \end{bmatrix}$ and $\begin{bmatrix} T_{N,2}w_{N,2} \\ T_{N,1}w_{N,2} \end{bmatrix}$ in (23), respectively.

Thus the overall computational cost amounts to $O(m^2 N \log N + m^3 N)$ arithmetic operations.

Concerning the number N of knots which are sufficient to have a good approximation of the integral, we can use the estimates of the approximation error of the trapezoidal rule. In particular

$$\max_{i=0,\ldots,N-1} |u_i - \tilde{u}_i| \leq \frac{r_\phi t}{6N^2}\gamma,$$

where γ is an upper bound to the maximum norm of the second derivative of the argument of the integral.

References

1. J. Abate and W. Whitt, The Fourier-series method for inverting transforms of probability distributions, Queueing Systems, 10, 5-88, 1992.
2. E. de Souza e Silva and H. R. Gail, An algorithm to calculate transient distributions of cumulative rate and impulse based reward, Stochastic Models, 14(3), 1998.
3. H. Nabli and B. Sericola, Performability analysis: A new algorithm, IEEE Transactions on Computers, 45(4), April 1996.

4. S. Narayana and M. F. Neuts, The first two moments matrices of the counts for the Markovian arrival process, Stochastic Models, 8, 459-477, 1992.
5. B. Sericola, Occupation times in Markov processes, Stochastic Models, 16, 479-510, 2000.

Acknowledgments

This research of M. F. Neuts was supported in part by NSF Grant Nr. DMI-9988749.