On modeling network congestion using continuous-time bivariate Markov chains

Brian L. Mark and Yariv Ephraim
Dept. of Electrical and Computer Engineering
George Mason University, MS 1G5
4400 University Drive, Fairfax, VA
email: bmark@gmu.edu, yephraim@gmu.edu

***Abstract*—We consider a model of congestion for computer networks based on a continuous-time finite-state homogeneous bivariate Markov chain. The model can be used to evaluate, via computer simulation, the performance of protocols and applications in a network with random path delays and packet losses due to traffic congestion. Only one of the processes of the bivariate Markov chain is observable. In our application, that process represents the dynamics of traffic congestion along a network path in terms of packet delay or packet loss. The other is an underlying process which affects statistical properties of the observable process. Thus, for example, the interarrival time of observed events is phase-type. The general form of the bivariate process studied here makes no assumptions on the structure of the generator of the chain, and hence, neither the underlying process nor the observable process is necessarily Markov. We present an expectation-maximization procedure for estimating the generator of a bivariate Markov chain given a sample path of the observable process. We compare the performance of the estimation algorithm to an earlier approximate estimation procedure based on time-sampling.** [1]

***Index Terms*—network models, parameter estimation, Markov models**

I. Introduction

Computer simulation is often used to evaluate the performance of network protocols and applications. The validity of network performance evaluation depends heavily on the accuracy to which the relevant network behaviors are represented by the simulation model. Ideally, the simulation model should abstract the salient features of the network that impact the performance of the protocol or application to be evaluated. Just as importantly, the simulation model should be sufficiently simple so that performance results can be obtained in a reasonable amount of time or even in real-time.

In [21], a continuous-time finite-state bivariate Markov chain was proposed as a model to simulate packet delays and losses in a computer network. One of the processes of the bivariate Markov chain, the underlying process, is not observable, while the other process, the observable process, represents the degree of congestion observed along a network path at any given time. More precisely, the observable state represents either a delay value or a "packet loss" condition due to severe congestion along the path. The observation samples can be obtained by employing a sequence of probes sent from a source to a destination host in a live network. The training data could also be obtained from a realistic network emulation environment such as Dummynet [6] or NIST Net [7] or a relatively detailed network simulator environment such as *ns-2* [11] or OPNET [9]. The probes measure round-trip delays between the source and destination, which are then quantized and mapped to the set of observable states of the bivariate Markov chain. Packet loss may be viewed as "infinite" delay and represented by a specially designated observable state.

The parameter of the continuous-time bivariate Markov chain is estimated from the observation samples and then incorporated into a simulator to drive the simulation of a network protocol or application by providing delays to packets transmitted into the network at arbitrary time instances. Compared to the more conventional hidden Markov model, the bivariate Markov chain can capture a higher degree of correlation in the observable delay values and provide a more general distribution of the time between jumps of the observable process. A discrete-time hidden Markov model was proposed by [19] to model packet losses in network communication channels. However, a discrete-time model cannot assign losses or delay values to packets transmitted at arbitrary time points in an event-driven simulation [21].

The estimation scheme proposed in [21] involves time-discretization of the bivariate Markov chain and inference of its generator from an estimate of the transition probability matrix of the resultant discrete-time bivariate Markov chain. As discussed in [16], this approach may lead to ambiguous estimates and in some cases will not lead to any valid estimates for the generator of the continuous-time bivariate Markov chain.

In this paper, we present an expectation-maximization (EM) algorithm for maximum-likelihood (ML) estimation of the generator of a continuous-time bivariate Markov chain from a given sample path of the observable process. Properties of the bivariate Markov chain and the EM algorithm are presented in greater detail in [14]. To our knowledge, no EM algorithm has previously been developed to estimate the parameter of a general continuous-time bivariate chain in the maximum likelihood sense. EM algorithms have been developed for the BMAP [5], [13], MMMP [10], and MMPP [17], [18], which

[1]This work was supported in part by the U.S. National Science Foundation under Grant CCF-0916568.

are particular bivariate Markov chains [14]. Similar to [5], [10], [13], [17], the EM algorithm described in this paper is based on the approach of Rydén [18], originally developed for parameter estimation of an MMPP. The EM algorithm for the BMAP developed in [13] employs a randomization technique for numerical computation. By contrast, the EM algorithm developed in the present paper consists of closed-form, stable recursions employing scaling and Van Loan's result for computation of matrix exponentials, along the lines of [10], [17]. In Section IV, we provide a numerical example to compare our EM algorithm with the parameter estimation approach proposed in [21].

The remainder of this paper is organized as follows. Section II presents the likelihood of the continuous-time bivariate Markov chain. Section III presents the EM algorithm. In Section IV, we discuss the implementation of the EM algorithm and provide a numerical example. Concluding remarks are given in Section V. Due to space limitations, proofs for the propositions stated in this paper are omitted but may be found in [14].

II. Continuous-time Bivariate Markov Chain

We consider a continuous-time finite-state homogeneous bivariate Markov chain $Z = (X, S) = \{(X(t), S(t)),\ t \geq 0\}$ and assume that it is separable and irreducible. The process $S = \{S(t), t \geq 0\}$ is the underlying process with state space $\{a_1, \ldots, a_r\}$ and $X = \{X(t), t \geq 0\}$ is the observable process with state space $\{b_1, \ldots, b_d\}$. The state space of Z is given by $\{b_1, \ldots, b_d\} \times \{a_1, \ldots, a_r\}$. To simplify notation, we will, without loss of generality, frequently refer to $X(t) = b_l$ or $S(t) = a_i$ as $X(t) = l$ or $S(t) = i$, respectively. Neither X nor S need be Markov. Conditions for either process to be Markov were given in [3]. With probability one, all sample paths of Z are right-continuous step functions with a finite number of jumps in each finite interval [1, Theorem 2.1].

The bivariate chain is parameterized by a generator matrix $H = \{h_{ln}(ij),\ l, n = 1, \ldots d; i, j = 1, \ldots r\}$, where the joint states (l, i) are ordered lexicographically and for $(l, i) \neq (n, j)$,

$$h_{ln}(ij) = \lim_{h \to 0} \frac{1}{h} P(Z(t+h) = (n, j) \mid Z(t) = (l, i)). \quad (1)$$

The generator matrix can be expressed as a block matrix $H = \{H_{ln},\ l, n = 1, \ldots, d\}$, where $H_{ln} = \{h_{ln}(ij),\ i, j = 1, \ldots, r\}$ are $r \times r$ matrices. The number of independent scalar values that constitute the generator H is at most $rd(rd-1)$.

A. Density of observed sample path

Assume that the observable process X of a bivariate Markov chain $Z = (X, S)$ starts from some state X_0 at time $T_0 = 0$ and jumps N times in $[0, T]$ at $0 < T_1 < T_2 < \cdots < T_N \leq T$. Let $X_k = X(T_k)$, such that X_k denotes the state of X in the interval $[T_k, T_{k+1})$ for $k = 0, 1, \ldots, N-1$, and X_N denotes the state of X in the interval $[T_N, T]$. We remark that this convention differs slightly from that used in [10], where $X_k \triangleq X(T_{k-1})$, $k = 1, \ldots, N+1$. Let $\Delta T_k = T_k - T_{k-1}$, $k = 1, 2, \ldots, N$. We denote realizations of X_k, T_k, and ΔT_k by x_k, t_k, and Δt_k, respectively. The observed sample path can be represented as

$$\mathcal{X} = \{(X_0, \Delta T_1), (X_1, \Delta T_2), \ldots, (X_N, T - T_N)\}. \quad (2)$$

Let $Z_k = Z(T_k) = (X(T_k), S(T_k)) = (X_k, S_k)$. Consider the sample path defined by

$$\mathcal{Z} = \{(Z_0, \Delta T_1), (Z_1, \Delta T_2), \ldots, (Z_N, T - T_N)\}. \quad (3)$$

The density of $\mathcal{X}$ may be obtained from the density of $\mathcal{Z}$ as shown below in Proposition 1. Furthemore, the process $\{(Z_k, T_k)\}$ obtained by sampling the process Z at the jump times of the observable process X is a Markov renewal process $\{(Z_k, T_k)\}$ [8].

Consider the conditional probability

$$P(T_1 \in [t, t+dt), Z(t) = (n, j) \mid Z(0) = (l, i)),\ l \neq n, \quad (4)$$

and denote the corresponding density by $f_{ij}^{ln}(t)$. Let $f^{ln}(t) = \{f_{ij}^{ln}(t),\ i, j = 1, \ldots, r\}$ denote the transition density matrix of $\{(Z_k, T_k)\}$. To obtain the density of the observable process, we also need to consider the transition probability of the underlying process S from state i at time 0 to state j at time t while the observable process X remains in state l for at least time t, denoted as follows:

$$\bar{f}_{ij}^{l}(t) = P(T_1 > t, S(t) = j \mid X(0) = l, S(0) = i). \quad (5)$$

Let $\bar{f}^l(t) = \{\bar{f}_{ij}^{l}(t),\ i, j = 1, \ldots, r\}$ denote the corresponding transition matrix.

Proposition 1. *For $t \geq 0$ we have*

$$f^{ln}(t) = e^{H_{ll}t} H_{ln}, \quad l \neq n, \quad (6)$$

and

$$\bar{f}^l(t) = e^{H_{ll}t}. \quad (7)$$

Proof: See [14]. ∎

Let $\boldsymbol{x}$ denote a realization of $\mathcal{X}$ (cf. (2)):

$$\boldsymbol{x} = \{(x_0, \Delta t_1), (x_1, \Delta t_2), \ldots, (x_N, T - t_N)\}.$$

Define $\mu_{x_0}(i) = P(X_0 = x_0, S_0 = i)$, $i = 1, \ldots, r$ and let

$$\mu_{x_0} = [\mu_{x_0}(1), \mu_{x_0}(2), \ldots, \mu_{x_0}(r)].$$

Let $\mathbf{1}$ denote a column vector of all ones. Using the Markov renewal property of $\{(Z_k, T_k)\}$, the density of the observable process X in $[0, T]$, given that it jumped N times, can then be expressed as

$$P_{\mathcal{X}}(\boldsymbol{x}) = \mu_{x_0} \left\{ \prod_{k=1}^{N} f^{x_{k-1}x_k}(\Delta t_k) \right\} \bar{f}^{x_N}(T - t_N)\mathbf{1}. \quad (8)$$

B. Forward-backward recursions

The density $P_{\mathcal{X}}(\boldsymbol{x})$ can be evaluated using forward and backward recursions. For convenience, we shall assume that $t_N = T$, i.e., the last jump of the observable process occurs at time T. In this case, (8) becomes

$$P_{\mathcal{X}}(\boldsymbol{x}) = \mu_{x_0} \left\{ \prod_{k=1}^{N} f^{x_{k-1}x_k}(\Delta t_k) \right\} \mathbf{1}. \tag{9}$$

Let $\mathcal{X}_\eta^\gamma$ denote the observed sample path in $[T_\eta, T_\gamma]$, where $0 \le \eta \le \gamma \le N$:

$$\begin{aligned} \mathcal{X}_\eta^\gamma &= \{X(t), t \in [T_\eta, T_\gamma]\} \\ &= \{(X_\eta, \Delta T_{\eta+1}), \ldots, (X_{\gamma-1}, \Delta T_\gamma), X_\gamma\}. \end{aligned}$$

We define the *forward* row vector

$$L(k) = \{P(\mathcal{X}_0^k = \boldsymbol{x}_0^k, S_k = i),\ i = 1, \ldots, r\}, \tag{10}$$

for $k = 0, 1, \ldots, N$. The forward recursion implements the product of μ_{x_0} and the bracketed term in (9), i.e.,

$$\begin{aligned} &L(0) = \mu_{x_0}, \\ &L(k) = L(k-1) f^{x_{k-1}x_k}(\Delta t_k), \quad k = 1, \ldots, N. \end{aligned} \tag{11}$$

We define the *backward* column vector

$$R(k) = \{P(\mathcal{X}_k^N = \boldsymbol{x}_k^N \mid X_{k-1} = x_{k-1}, S_{k-1} = i), i = 1, \ldots, r\}', \tag{12}$$

where $'$ denotes matrix transpose, for $k = N, N-1, \ldots, 1$. The backward recursion implements the product of the bracketed term and the column vector $\mathbf{1}$ in (9):

$$\begin{aligned} &R(N+1) = \mathbf{1}, \\ &R(k) = f^{x_{k-1}x_k}(\Delta t_k) R(k+1), \quad k = N, \ldots, 1. \end{aligned} \tag{13}$$

Using the above forward and backward recursions, the observable path density is given by $P_{\mathcal{X}}(\boldsymbol{x}) = L(N)\mathbf{1}$ and also by

$$P_{\mathcal{X}}(\boldsymbol{x}) = L(k)R(k+1), \quad k = 0, \ldots, N. \tag{14}$$

To ensure numerical stability, it is necessary to scale the above recursions. Using an approach similar to that developed in [17], the scaled forward recursion is given by

$$\begin{aligned} &\tilde{L}(0) = \mu_{x_0}, \\ &\tilde{L}(k) = \frac{\tilde{L}(k-1) f^{x_{k-1}x_k}(\Delta t_k)}{c_k}, \quad k = 1, \ldots, N, \end{aligned} \tag{15}$$

where

$$c_k \triangleq \tilde{L}(k-1) f^{x_{k-1}x_k}(\Delta t_k)\mathbf{1}, \quad k = 1, \ldots, N. \tag{16}$$

The scaled backward recursion is given by

$$\begin{aligned} &\tilde{R}(N+1) = \mathbf{1}, \\ &\tilde{R}(k) = \frac{f^{x_{k-1}x_k}(\Delta t_k) R(k+1)}{c_k},\ k = 1, \ldots, N. \end{aligned} \tag{17}$$

One can show straightforwardly that the scaled and unscaled iterates of the forward and backward recursions are related by

$$\tilde{L}(k) = \frac{L(k)}{\prod_{m=1}^{k} c_m} \text{ and } \tilde{R}(k) = \frac{R(k)}{\prod_{m=k}^{N} c_m}, \tag{18}$$

respectively. The density of $P_{\mathcal{X}}(\boldsymbol{x})$ can be expressed as the product of the scaling constants c_k as follows:

$$P_{\mathcal{X}}(\boldsymbol{x}) = \prod_{m=1}^{N} c_m. \tag{19}$$

From (18) and (10), we see that for $k = 1, \ldots, N$, the scaled forward vector $\tilde{L}(k)$ can be interpreted as the probability distribution of the underlying process S at time T_k conditioned on the observable sample path up to and including time T_k:

$$\tilde{L}(k) = \{P(S_k = i \mid \boldsymbol{x}_0^k), i = 1, \ldots, r\}. \tag{20}$$

III. EM Algorithm

In this section, we describe an EM algorithm for maximum likelihood estimation of the true parameter of a bivariate chain, denoted by ϕ^0, given the sample path of the observable process, $\mathcal{X} = \{X(t),\ 0 \le t \le T\}$. In the EM approach, a new parameter estimate, say $\phi_{\iota+1}$ is obtained from a given parameter estimate, say ϕ_ι as follows:

$$\phi_{\iota+1} = \arg\max_{\phi} E\{\log P(\mathcal{Z}; \phi) \mid \mathcal{X}; \phi_\iota)\}, \tag{21}$$

where $\mathcal{Z}$ denotes the sample path of the bivariate process given in (3). The maximization is over ϕ, which consists of the off-diagonal elements of the bivariate generator H.

The reestimation formula is obtained by considering the dwell time D_i^l in each state (l, i) and the number of jumps m_{ij}^{ln} from each state (l, i) to another state (n, j). Let $\varphi_{(l,i)}(t) = I(Z(t) = (l, i))$, where $I(\cdot)$ denotes the indicator function and let $\#$ denote set cardinality. Then the dwell time and number of jumps can be expressed, respectively, as follows:

$$D_i^l = \int_0^T \varphi_{(l,i)}(t) dt, \tag{22}$$

$$m_{ij}^{ln} = \#\{t : 0 < t \le T,\ Z(t-) = (l, i), Z(t) = (n, j)\}. \tag{23}$$

Let $\hat{D}_i^l$ and $\hat{m}_{ij}^{ln}$ denote the conditional mean estimates given $\mathcal{X}$ of D_i^l and m_{ij}^{ln}, respectively:

$$\hat{D}_i^l = E[D_i^l \mid \mathcal{X}], \quad \hat{m}_{ij}^{ln} = E[m_{ij}^{ln} \mid \mathcal{X}],$$

where we have suppressed the dependency on ϕ_ι to simplify the notation. We have

$$\hat{D}_i^l = \int_0^T P(Z(t) = (l, i) \mid \mathcal{X}) dt, \tag{24}$$

where $P(Z(t) = (l, i) \mid \mathcal{X})$ denotes the conditional probability of Z at time t, and (see [14]):

$$\hat{m}_{ij}^{ln} = \int_0^T P(Z(t-) = (l, i), Z(t) = (n, j) \mid \mathcal{X}) dt, \tag{25}$$

where $P(Z(t-) = (l,i), Z(t) = (n,j) \mid \mathcal{X})$ denotes the conditional density of a jump of Z at time t. A result similar to (25) was originally stated in [1], [2], [18]. The proof given in [2] in the context of estimating phase-type distributions was adapted in [10] for estimating MMMPs.

Maximization of $E\{\log P(\mathcal{Z};\phi) \mid \mathcal{X};\phi_\iota\}$ over ϕ, which consists of the off-diagonal elements of $H = \{h_{ln}(ij)\}$, yields the following intuitive estimate in the $\iota+1$st iteration of the EM algorithm [1]:

$$\hat{h}_{ln}(ij) = \frac{\hat{m}_{ij}^{ln}}{\hat{D}_i^l}, \quad (l,i) \neq (n,j). \tag{26}$$

A. Number of jumps estimate

We shall present closed-form expressions for the estimates given by (24) and (25). To estimate the number of jumps, two cases are considered:

1) $l = n$, $i \neq j$, for which

$$\hat{m}_{ij}^{ll} = \int_0^T P(Z(t-) = (l,i), Z(t) = (l,j) \mid \mathcal{X}) dt. \tag{27}$$

and

2) $l \neq n$, for which the number of jumps can be written as

$$\begin{aligned} m_{ij}^{ln} &= \#\{t : 0 \leq t \leq T, Z(t-) = (l,i),\ Z(t) = (n,j)\} \\ &= \sum_{\substack{k:x_k=l,\\ x_{k+1}=n}} I(Z(t_k-) = (l,i), Z(t_k) = (n,j)). \end{aligned} \tag{28}$$

We begin with the first case. For convenience, we shall assume that $T = t_N$, such that the density $P_{\mathcal{X}}(\boldsymbol{x})$ of the observable sample path is given by (19).

Proposition 2 (Number of jumps, $l = n$, $i \neq j$)**.** *Assume that $T = t_N$. Then,*

$$\hat{m}_{ij}^{ll} = \left[H_{ll} \odot \sum_{k:x_k=l} \frac{\mathcal{I}'_k}{c_{k+1}} \right]_{(i,j)}, \tag{29}$$

where $\odot$ denotes element-by-element matrix multiplication and

$$\mathcal{I}_k = \int_0^{\Delta t_{k+1}} e^{H_{x_k x_k}(\Delta t_{k+1} - y)} H_{x_k x_{k+1}} \tilde{R}(k+2) \tilde{L}(k) e^{H_{x_k x_k} y} dy, \tag{30}$$

for $k = 0, \ldots, N-1$.

Proof: See [14]. ∎

Following the approach of [10], [17], we apply Van Loan's result [20] to evaluate the integral in (30). Define the $2r \times 2r$ matrix

$$C_k = \begin{bmatrix} H_{x_k x_k} & H_{x_k x_{k+1}} \tilde{R}(k+2) \tilde{L}(k) \\ 0 & H_{x_k x_k} \end{bmatrix}. \tag{31}$$

Then $\mathcal{I}_k$ is given by the $r \times r$ upper right block of the matrix exponential $e^{C_k \Delta t_{k+1}}$, denoted as follows:

$$\mathcal{I}_k = \left[e^{C_k \Delta t_{k+1}} \right]_{(1,2)}. \tag{32}$$

Next, we consider the second case in (28). The following result holds in general for $T \geq t_N$.

Proposition 3 (Number of jumps, $l \neq n$)**.**

$$\hat{m}_{ij}^{ln} = \left[H_{ln} \odot \sum_{\substack{k:x_k=l,\\ x_{k+1}=n}} \frac{\mathcal{J}'_k}{c_{k+1}} \right]_{(i,j)}, \tag{33}$$

where

$$\mathcal{J}_k = \tilde{R}(k+2) \tilde{L}(k) e^{H_{x_k x_k} \Delta t_k}, \quad k = 0, \ldots, N-1. \tag{34}$$

Proof: See [14]. ∎

B. Dwell time estimate

Next, we provide an expression for the dwell time estimate in (25). For convenience, we shall again assume that $T = t_N$.

Proposition 4. *Assume that $T = t_N$. Then,*

$$\hat{D}_i^l = \left[\sum_{k:x_k=l} \frac{\mathcal{I}'_k}{c_{k+1}} \right]_{(i,i)}, \tag{35}$$

where $\mathcal{I}_k$ is given in (32)*.*

Proof: See [14]. ∎

IV. Implementation and Numerical Example

The EM algorithm for continuous-time bivariate chains developed in Section III was implemented in Python using the SciPy and NumPy libraries. The matrix exponential function from the SciPy library is based on a Padé approximation, which has a computational complexity of $O(r^3)$ for an $r \times r$ matrix (see [15]). For comparison purposes, the approximate parameter estimation algorithm based on time-sampling proposed in [21] was also implemented in Python. We refer to this algorithm as the *Baum-based algorithm* for estimating the parameter of continuous-time bivariate Markov chains.

A. Baum-based algorithm

In the Baum-based algorithm described in [21], the continuous-time bivariate chain Z is time-sampled to obtain a discrete-time bivariate Markov chain $\tilde{Z} = (\tilde{X}, \tilde{S}) = \{\tilde{Z}_k = (\tilde{X}_k, \tilde{S}_k)\}$, where

$$\tilde{Z}_k = Z(k\Delta), \quad k = 0, 1, 2, \ldots,$$

and Δ is the sampling interval. Let P denote the transition probability matrix of $\tilde{Z}$. A variant of the Baum algorithm [4] is then employed to obtain a maximum likelihood estimate, $\hat{P}$, of P. An estimate of the generator of the continuous-time bivariate chain is then obtained from

$$\hat{H} = \frac{1}{\Delta} \ln(\hat{P}), \tag{36}$$

where $\ln(\hat{P})$ denotes the matrix logarithm of $\hat{P}$, which has the Taylor series expansion

$$\ln(\hat{P}) = \sum_{n=1}^{\infty} (-1)^{n-1} \frac{(\hat{P} - I)^n}{n}. \tag{37}$$

However, existence and uniqueness of a generator $\hat{H}$ corresponding to a transition matrix $\hat{P}$ is not guaranteed.

As pointed out in [16], Israel, Rosenthal, and Wei [12] show that a sufficient condition for the matrix logarithm in (37) to provide a unique generator $\hat{H}$ is that the eigenvalues of $\hat{P}$ must be distinct positive. Furthermore, if any of the distinct eigenvalues of $\hat{P}$ is negative, then no generator exists. When $\hat{P}$ has multiple corresponding generators, each provides a different transition matrix. Moreover, if a generator matrix $\hat{H}$ of a certain structure is desired (e.g., an MMPP), it is generally not possible to estimate a transition matrix $\hat{P}$ that yields the required structure. In practice, the uniqueness and existence of $\hat{H}$ for a given $\hat{P}$ depends on the sampling interval Δ [16].

B. Computational and storage requirements

The computational requirement of the EM algorithm developed in Section III depends linearly on the number of jumps, N, of the observable process. On the order of N matrix exponentials for the transition density matrix $f^{x_{k-1}x_k}(\Delta t_k)$ in (11) and (13) and for the matrix $\mathcal{I}_k$ in (32) are computed. Computation of the matrix exponential of an $r \times r$ matrix requires $O(r^3)$ arithmetic operations [15]. Thus, the computational requirement due to the matrix exponentials is $O(Nr^3)$. The element-by-element matrix multiplications in (29) and (33) contribute a computational requirement of $O(N(r^2d^2))$. Therefore, the overall computational complexity of the EM algorithm can be stated as $O(N(r^3 + r^2d^2))$. The storage requirement of the EM algorithm is dominated by the (scaled) forward and backward variables $\tilde{L}(k)$ and $\tilde{R}(k)$. Hence, the overall storage required is $O(Nr)$.

For comparison, the computational requirement of the Baum-based algorithm is $O(\tilde{N}r^2d^2)$, where $\tilde{N} = T/\Delta$ is the number of discrete-time samples. The storage requirement of the Baum-based algorithm is $O(\tilde{N}rd)$. Clearly, both the computational and storage requirement of this algorithm are highly dependent on the choice of the sampling interval Δ.

C. Numerical example

A numerical example demonstrating parameter estimation of a bivariate Markov chain using the EM procedure presented in Section III is given in Table I. For this example, the number of underlying states is $r = 2$ and the number of observable states is $d = 2$. The generator matrix H is displayed in terms of its block matrix components H_{ln}, which are 2×2 matrices for $l, n \in \{1, 2\}$. The column labelled ϕ^0 shows the true parameter value for the bivariate Markov chain. Similarly, the columns labelled ϕ_0 and $\hat{\phi}_{\rm em}$ show, respectively, the initial parameter and the EM-based estimate rounded to two decimal places. The observed data, generated using the true parameter ϕ^0, consisted of 10,000 observable jumps. The EM algorithm was terminated when the relative difference of successive log-likelihood values fell below 10^{-7}.

The bivariate Markov chain parameterized by ϕ^0 in Table I is neither a BMAP nor an MMMP (see [14]). The estimate $\hat{\phi}_{\rm em}$ was obtained after 63 iterations of the EM procedure. An important property of the EM algorithm is that whenever an off-diagonal element of the generator H is zero in the initial parameter, the corresponding element in any EM iterate remains zero. This can be easily seen from Propositions 2 and 3. Thus, if structural information about H is known, that structure is maintained by the EM estimate provided that the initial parameter estimate possesses this structure.

For comparison purposes, we implemented the Baum-based approach proposed in [21] and applied it to the example of Table I for the sampling intervals $\Delta = 10^{-2}$, 0.5×10^{-2}, 0.25×10^{-2}. The Baum algorithm, applied after time-sampling to obtain an estimate of the transition matrix of a discrete-time bivariate Markov chain, was terminated when the relative difference of successive log-likelihood values fell below 10^{-7}. The number of iterations required for the three sampling intervals were 446, 105, and 123, respectively, while the number of discrete-time samples were 10^6, 2×10^6, and 4×10^6, respectively. For this example, a generator matrix could always be obtained from each transition matrix estimate using (36) for all three sampling intervals.

The results are shown in Table II. For all three sampling intervals, the estimate of H_{21} is not a diagonal matrix, but the accuracy of this estimate appears to improve as Δ is decreased. The Baum-based estimates of the other block matrices H_{ln} also appear to become closer in value to the EM-based estimate $\hat{\phi}_{\rm em}$ shown in Table I when Δ decreases. On the other hand, as Δ decreases, the computational requirement of the Baum-based approach increases proportionally, as discussed in Section IV-B.

	ϕ^0		ϕ_0		$\hat{\phi}_{\rm em}$	
H_{11}	-70	10	-120	30	-77.03	14.76
	20	-55	2	-8	8.46	-47.95
H_{12}	50	10	70	20	51.80	10.46
	25	10	5	1	32.66	6.83
H_{21}	50	0	70	0	49.50	0
	0	10	0	1	0	9.54
H_{22}	-60	10	-100	30	-59.51	10.01
	20	-30	2	-3	19.61	-29.15

TABLE I
BIVARIATE MARKOV CHAIN PARAMETER ESTIMATION: $\phi^0 = \text{true}$; $\phi_0 = \text{initial}$; $\hat{\phi}_{\rm em} = \text{EM-based estimate}$.

V. CONCLUSION

We considered a model of network congestion based on a finite-state continuous-time bivariate Markov chain and developed explicit forward-backward recursions for estimating its parameter in the maximum likelihood sense from observation samples obtained from a real network. The observation or training data consists of packet delay values or packet loss indications obtained by sending probe packets along a network

	$\hat{\phi}_{\text{baum}}$					
Δ	10^{-2}		0.5×10^{-2}		0.25×10^{-3}	
H_{11}	-77.54	10.40	-78.18	15.40	-80.43	15.98
	11.30	-49.21	6.16	-48.20	8.73	-48.53
H_{12}	57.34	9.81	49.29	13.48	52.13	12.32
	20.63	17.29	33.49	8.55	31.44	8.36
H_{21}	47.10	2.91	52.04	1.18	51.92	0.45
	2.19	15.36	2.11	9.74	0.97	10.63
H_{22}	-56.29	6.28	-64.22	11.01	-64.51	12.13
	4.21	-21.77	16.09	-27.93	17.72	-29.31

TABLE II
BAUM-BASED BIVARIATE MARKOV CHAIN PARAMETER ESTIMATION WITH DIFFERENT SAMPLING INTERVALS Δ.

path. The parameter of the bivariate Markov chain model can be used to drive an efficient event-driven simulation for evaluating the performance of network protocols and applications.

We developed an EM algorithm for estimating the parameter of a bivariate Markov chain directly from the continuous time sample path of the observations. The algorithm does not require any sampling scheme or numerical integration, and it preserves the structure of a generator attributed to the process. None of these desirable properties is guaranteed by an earlier approach proposed in [21]. In ongoing work, we are studying the performance of the proposed estimation algorithm using training data obtained from realistic network scenarios.

REFERENCES

[1] A. Albert. Estimating the infinitesimal generator of a continuous time, finite state Markov process. *Annals of Mathematical Statistics*, 23(2):727–753, Jun. 1962.

[2] S. Asmussen, O. Nerman, and M. Olsson. Fitting phase-type distributions via the EM algorithm. *Scand. J. Stat.*, 23(4):419–441, 1996.

[3] F. G. Ball, R. K. Milne, and G. F. Yeo. Continuous-time Markov chains in a random environment, with applications to ion channel modelling. *Adv. in Appl. Prob.*, 26(4):919–946, Dec. 1994.

[4] L. E. Baum, T. Petrie, G. Solues, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *Ann. Math. Statist.*, 41:164–171, 1970.

[5] L. Breuer. An EM algorithm for batch Markovian arrival processes and its comparison to a simpler estimation procedure. *Annals of Operations Research*, 112:123–138, 2002.

[6] M. Carbone and L. Rizzo. Dummynet revisited. *ACM Computer Communication Review*, 40(2), April 2010.

[7] M. Carson and D. Santay. NIST Net – A Linux-based Network Emulation Tool. *ACM Computer Communication Review*, June 2003.

[8] E. Çinlar. Markov Renewal Theory: A Survey. *Management Science*, 21(7):727–752, Dec. 1975.

[9] X. Chang. Network Simulations with OPNET. In P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, editors, *Proc. Winter Simulation Conference*, pages 307–314, 1999.

[10] Y. Ephraim and W. J. J. Roberts. An EM algorithm for Markov modulated Markov processes. *IEEE Trans. Sig. Proc.*, 57(2), Feb. 2009.

[11] K. Fall and K. Varadhan, editors. *The* ns *Manual*. The VINT Project, May 2010.

[12] R. B. Israel, J. S. Rosenthal, and J. Z. Wei. Finding generators for Markov chains via empirical transition matrices with applications to credit ratings. *Math. Finance*, 11(2):245–265, Apr. 2001.

[13] A. Klemm, C. Lindemann, and M. Lohmann. Modeling IP traffic using the batch Markovian arrival process. *Performance Evaluation*, 54:149–173, 2003.

[14] B. L. Mark and Y. Ephraim. Parameter estimation for continuous-time bivariate Markov chains. Feb. 2011 (to be submitted for publication).

[15] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[16] W. J. J. Roberts and Y. Ephraim. An EM algorithm for ion-channel current estimation. *IEEE Trans. Sig. Proc.*, 56:26–33, Jan. 2008.

[17] W. J. J. Roberts, Y. Ephraim, and E. Dieguez. On Rydén's EM algorithm for estimating MMPPs. *IEEE Sig. Proc. Let.*, 13(6):373–377, June 2006.

[18] T. Rydén. An EM algorithm for estimation in Markov-modulated Poisson processess. *Computational Statistics and Data Analysis*, 21:431–447, 1996.

[19] K. Salamatian and S. Vaton. Hidden Markov modelling for network communication channels. In *Proc. ACM Sigmetrics*, June 2001.

[20] C. F. Van Loan. Computing integrals involving the matrix exponential. *IEEE Trans. on Automatic Control*, 23(3), June 1978.

[21] W. Wei, B. Wang, and D. Towsley. Continuous-time hidden Markov models for network performance evaluation. *Performance Evaluation*, 49:129–146, Sept. 2002.