Fast Algorithms for Measurement-Based Traffic Modeling *

Hao Che *San-qi Li*
Department of Electrical and Computer Engineering
University of Texas at Austin
Austin, Texas 78712
{hche,sanqi}@globe.ece.utexas.edu

Abstract

This paper develops fast algorithms for construction of circulant modulated rate process to match with two primary traffic statistical functions: distribution $f(x)$ and autocorrelation $R(\tau)$ of the rate process. Using existing modeling techniques, $f(x)$ has to be limited to certain forms such as Gaussian or binomial; $R(\tau)$ can only consist of one or two exponential terms which are often real exponentials rather than complex. In reality, these two functions are collective from real traffic traces and generally expressed in much complicated form. Our emphasis here is placed on the algorithmic design for matching complicated $R(\tau)$ in traffic modeling. The typical CPU time for the traffic modeling with $R(\tau)$ consisting of five or six complex exponential terms is found in the range of a few minutes by the proposed algorithms. Our study further shows an excellent agreement between original traffic traces and sequences generated by the matched analytical model.

1 Introduction

Many studies indicate the importance of high burstiness and strong correlation nature of multimedia traffic to network control and resource management design (e.g., [1, 2, 3]). Such traffic dynamics in statistical measurement are mainly captured by the collective rate distribution function $f(x)$ and autocorrelation function $R(\tau)$. Real traffic measurement shows that both $f(x)$ and $R(\tau)$ are generally expressed in complicated form. The central task of *measurement-based* traffic modeling is to develop algorithms for construction of random process models, whose statistics match with the collective $f(x)$ and $R(\tau)$. This is difficult especially if the models should fit into feasible analytical queueing solutions for network performance evaluation.

Most queueing analyses have used two-state Markov chain modulated process as a basic element for construction of multimedia traffic. Using this modeling technique, $R(\tau)$ has to be in the form $\sum_k \psi_k e^{\lambda_k \tau}$ with real ψ_k and real λ_k; $f(x)$ is limited to a convolved binomial function [4]. They are certainly insufficient to capture the diversed traffic correlation and burstiness behavior, especially the strong pseudo-periodic nature as found in MPEG video traces and feedback-controlled ABR traffic. In [5, 6], *Li* and *Hwang* proposed a structure of circulant modulated Poisson process (CMPP), whose $R(\tau)$ has the form $\sum_k \psi_k e^{\lambda_k \tau}$ with real ψ_k but complex λ_k and whose cumulative function $F(x) = \int_0^x f(u)du$ is expressed as a piece-wise step function. It is obvious that the sum of complex exponentials represent a much wider class of correlation functions than the sum of real exponentials.

One may question the validity of using circulant for traffic modeling. A comprehensive, blind numerical study in [6] compared queueing/loss-rate solutions between vastly different circulant and Markov chain, both of which are designed to match with identical $R(\tau)$ and $f(x)$. The two solutions were found generally in excellent agreement. In other words, the underlying structural difference between circulant and Markov chain has no significant impact on queueing/loss-rate performance, as long as the steady state statistics $f(x)$ and second-order statistics $R(\tau)$ of a given range are properly matched. Such study was recently extended in [7] to the comparison between circulant and nonMarkovian process. The nonMarkovian process includes auto regressive moving average model (ARMA), nonlinearized ARMA model for non-Gaussian distribution, and correlated batch arrival process with nonexponential interarrival distribution. Similarly, all the comparison results between circulant and nonMarkovian process in [7] show excellent agreement, provided matching with identical $f(x)$ and $R(\tau)$. That is, once the important statistics of traffic are captured by its modeling, the actual underlying structure of selected models, whether circulant, Markovian or nonMarkovian, does not have significant impact on queueing/loss-rate performance.

The measurement-based construction of CMPP consists of three steps [6]. The first step is the identification of ψ_k's and λ_k's to match a collective correlation sequence, which is formulated as a standard nonlinear programming problem. Due to the constraint of real ψ_k's, however, most existing algorithms such as Prony, MUSIC and ESPRIT [8, 9, 10]

*The research reported here was supported by NSF under grant NCR-9314387, Southwestern Bell, and by Texas Advanced Research Program under grant TARP-33.

0-8186-7780-5/97 $10.00 © 1997 IEEE

fail to apply. In this paper we develop a heuristic algorithm by taking the combination of the Prony algorithm with the non-negative least squares (NNLS) method. The algorithm is found fast and robust by comprehansive numerical studies. The second step is the formation of a circulant transition rate matrix whose eigenvalues must contain all the λ_k's of $R(\tau)$, which is originally an inverse eigenvalue problem but transformed to an index search problem in [6]. The ad hoc approach used in [6] is based on random search of entire index space, which is rather slow and practically unusable when $R(\tau)$ contains more than three exponential terms (i.e., three λ_k's). The major contribution of this paper is the development of a fast index search algorithm for increased number of λ_k's. For instance, the average CPU time on a Sun workstation for construction of circulant by this algorithm takes about 30 seconds for five λ_k's and 3 minutes for eight λ_k's in $R(\tau)$. In contrast, the random search algorithm could take days or monthes for the construction. The third step of the construction is the design of input rate vector for CMPP to optimally match a collective rate distribution $F(x)$, which is formulated as a minimization problem and effectively solved by the Nelder-Meade simplex search method in [6].

Finding fast and practically useful algorithms for measurement-based construction of traffic model is of paramount importance to network performance evaluation. In most of our numerical examples, the total CPU time of the three steps is typically in the range of a few minutes, which makes this traffic modeling technique readily attractive to field traffic engineers. In practice, one can develop a source library from collected representative traces of multimedia sources, each of which is described by its own $f(x)$ and $R(\tau)$. One can then build a *matched* CMPP for each individual source. For the modeling of aggregate traffic at a multiplexer, one can simply take the convolution of individual $f(x)$'s and $R(\tau)$'s to form the aggregate $f(x)$ and $R(\tau)$ and then build a single CMPP. Such CMPP's can be used either for network performance analysis, or as traffic generator to load the real network for testing. Note that traffic generator is a critical component for network testing. A few traffic generator equipments available to date are practically not useful without measurement-based traffic modeling.

Other than using the superposition of two-state Markov chains, a few works are available for measurement-based traffic modeling to match with collective $R(\tau)$ and $f(x)$. In [13], a multi-state Markov chain is constructed to model a MPEG video, but its correlation function still contains a single real exponential as a two-state Markov chain. In [14], *Jagerman* and *Melamed* proposed a measurement-based modeling technique called transform expand sample (TES), which is mainly suitable for computer simulation. In most TES examples [15], $R(\tau)$ only consists of one or two exponentials. Recent traffic measurement [1, 2] also identifies the significance of long range dependencies, which are described by the correlation behavior in large time scales (or, equivalently by the dominant power spectrum in low-frequency band). The study in [4, 17] indicates that a finite-buffer queueing system is non-effective for transport of low-frequency traffic subject to negligible loss rate. That is, the link bandwidth should be at least equal to the peak rate of low-frequency traffic whose flow stays intact through the queueing system. In this paper we focus on the traffic whose correlation function can be approximated by the sum of complex exponentials.

The paper is organized as follows. Section 2 provides the background knowledge of the algorithmic design for CMPP construction. Section 3 describes a heuristic fast algorithm for the identification of λ_k's and ψ_k's to match with a given correlation sequence. Section 4, which is the major part of this paper, develops a fast index search algorithm for the construction of a circulant transition rate matrix whose eigenvalues contain all the λ_k's. Section 5 provides modeling examples based on real traffic traces to show the potential of the algorithm.

2 Background

Our objective here is the construction of a CMPP to statistically match with a stationary random arrival point process $a(t)$. Denote the rate autocorrelation function and steady-state cumulative rate distribution of $a(t)$ by $R(\tau)$ and $F(x)$ and that of CMPP by $R_c(\tau)$ and $F_c(x)$, where the subscript "c" indicates the function of CMPP model. The algorithmic design is to achieve

$$R_c(\tau) \approx R(\tau), \qquad F_c(x) \approx F(x).$$

In reality, $R(\tau)$ is either expressed analytically or described by a correlation sequence collective from real traffic trace. For CMPP, $R_c(\tau)$ must be represented by

$$R_c(\tau) = \sum_{l=0}^{M} \psi_l e^{\lambda_l |\tau|} \tag{1}$$

with real nonnegative ψ_l and complex λ_l [6]. Define $\vec{\lambda} = [0, \lambda_1, \lambda_2, ..., \lambda_M]$ and $\vec{\psi} = [\psi_0, \psi_1, \psi_2, ..., \psi_M]$ where M is the order of $R_c(\tau)$. The first step of our design is the identification of $\vec{\lambda}$ and $\vec{\psi}$ to match the original $R(\tau)$, i.e.,

$$\min_{\lambda_l, \psi_l} \int_0^\infty [R(\tau) - R_c(\tau)]^2 d\tau \qquad \text{s.t.} \qquad \psi_l \geq 0 \quad \forall l \tag{2}$$

which will be addressed in Section 3.

The next step is the construction of a circulant transition rate matrix $\mathbf{Q}$. Denote the first row of $\mathbf{Q}$ by vector $\vec{a} = [a_0, a_1, ..., a_{N-1}]$ for N-state circulant. Each row of $\mathbf{Q} = circ(\vec{a})$ is then formed by a forward shift permutation of the previous row. Define the eigenvalues of $\mathbf{Q}$ in vector form by $\vec{\lambda}_c = [\lambda_{c0}, \lambda_{c1}, ..., \lambda_{c(N-1)}]$. One can write

$$\vec{\lambda}_c = \sqrt{N} \vec{a} \mathbf{F}^\star. \tag{3}$$

$\mathbf{F}$ is a Fourier matrix with its (i,j)-th element given by $\frac{1}{\sqrt{N}} e^{-\frac{2\pi ij}{N}\sqrt{-1}}$. We have $\mathbf{F}^{-1} = \mathbf{F}^\star$ where $\mathbf{F}^\star$ is the conjugate transpose of $\mathbf{F}$. For CMPP, each eigenvalue contributes an exponential term to its correlation function,

$$R_c(\tau) = \sum_{k=0}^{N-1} \psi_{ck} e^{\lambda_{ck}|\tau|}. \tag{4}$$

Since the power vector $\vec{\psi}_c = [\psi_{c0}, \psi_{c1}, ..., \psi_{c(N-1)}]$ is assigned independent of $\vec{a}$ [6], the basic construction of $\vec{a}$ is to include

$\vec{\lambda}$ in $\vec{\lambda}_c$. For the *wanted* eigenvalues, namely $\lambda_{cl} \in \vec{\lambda}, \forall l$, one can assign ψ_{cl} by their corresponding value in $\vec{\psi}$. For the remaining *unwanted* eigenvalues, namely $\lambda_{cl} \notin \vec{\lambda}, \forall l$, one can simply remove them by assigning $\psi_{cl} = 0$.

As a transition rate vector, $\vec{a}$ must be constructed under the nonnegative constraint $\vec{a} \geq 0$, $a_k \geq 0$ for $k = 1, 2, ..., N-1$ with $a_0 = -\sum_{k=1}^{N-1} a_k$, which substantially limits its eigenvalue space. Nevertheless, only a subset of $\vec{\lambda}_c$ needs to match with $\vec{\lambda}$. One can significantly expand the eigenvalue space of $\vec{\lambda}_c$ by increasing its dimension N, i.e., making $N >> M$, such that a solution $\vec{a}$ for $\vec{\lambda} \subset \vec{\lambda}_c$ is more likely to exist.

The construction of $\vec{a}$ from $\vec{\lambda}$ is therefore transformed into an index search problem, which is to find an m-dimension index of $\vec{\lambda}$ within the N-dimension index space of $\vec{\lambda}_c$ for solution $\vec{a}$ in (3) to satisfy the nonnegative constraint $\vec{a} \geq 0$. Note that the unwanted eigenvalues in $\vec{\lambda}_c$ are arbitrarily assigned under the nonnegative constraint. Such index search has the space complexity of N^M. Brute force search method will fail to apply, for large N and M. Developing a fast index search algorithm therefore becomes crucial to the success of the proposed modeling technique. This is achieved in Section 4.

Define the input rate at each circulant state by $\vec{\gamma} = [\gamma_0, \gamma_1, ..., \gamma_{N-1}]$. Once $\vec{a}$ is fixed by $\vec{\lambda}$, the last step of our design is to find an input rate vector $\vec{\gamma}$ for $F_c(x) \approx F(x)$. For CMPP, one can write [6],

$$\gamma_i = \overline{\gamma} + \sum_{l=1}^{N-1} \sqrt{\psi_{cl}} \cos(2\pi i l/N - \theta_{cl}), \qquad \text{for } i = 0, 1, ..., N-1. \tag{5}$$

which is equivalent to $\vec{\psi}_c = \frac{1}{N} |\vec{\gamma}\mathbf{F}^*|^2$ and $\vec{\theta}_c = \arg\{\vec{\gamma}\mathbf{F}^*\}$ with phase vector $\vec{\theta}_c = [\theta_{c0}, \theta_{c1}, ..., \theta_{c(N-1)}]$. Since $\vec{\psi}_c$ is already fixed by $R_c(\tau)$, one can only use $\vec{\theta}_c$ to change $\vec{\gamma}$. Further, because $\vec{\psi}_c$ contains M nonzero elements, only their corresponding elements in $\vec{\theta}_c$ are effective in the $\vec{\gamma}$ design.

The steady-state probability of each state in circulant is equally likely, i.e., $\pi_l = \frac{1}{N}, \forall l$. Thus, $F_c(x)$ for CMPP is only dependent on $\vec{\gamma}$, which is a piecewise multi-step function and jumps by $\frac{1}{N}$ at each individual value of $x \in \vec{\gamma}$ in ascending order, expressed by

$$F_c(x) = \lim_{t \to \infty} Pr(\gamma(t) \leq x) = \frac{n_x}{N}, \tag{6}$$

where n_x represents the number of input rates in $\vec{\gamma}$ less than or equal to x. For the distribution matching, we first need to discretize the original $F(x)$ by partitioning its range of x into a set of N equal-probability rates. The rate at the partitioning point is denoted by $\vec{\gamma}' = [\gamma'_0, \gamma'_1, ..., \gamma'_{N-1}]$ in ascending order with $Pr(x = \gamma'_k) = \frac{1}{N}, \forall k$. Similarly for $F_c(x)$, we sort out $\vec{\gamma}$ in ascending order as denoted by $\vec{\gamma}_p = [\gamma_{p0}, \gamma_{p1}, ..., \gamma_{pN-1}]$, subject to $\gamma_{p_i} \geq \gamma_{p_k}$ for $i < k$. The distribution matching $F_c(x) \approx F(x)$ can then be formulated into a minimization problem:

$$\min_{\vec{\theta}_c} \frac{1}{N} \sum_{k=0}^{N-1} |\gamma'_k - \gamma_{p_k}|, \qquad \text{s. t. } \gamma_{p_k} \geq 0, \forall k, \tag{7}$$

which was effectively solved in [6] using the Nelder-Meade simplex search method. The total computation time in this step is typically in the range of a few seconds on a workstation. This paper focuses on the algorithmic design of the first two steps.

3 Matching Correlation Function

This section developes a heuristic fast algorithm for identification of $\vec{\lambda}$ and $\vec{\psi}$ in correlation function matching. The traffic statistics in real measurement are often collective in the discrete time domain using computer and digital signal processing technologies. Without loss of generality, the original correlation function $R(\tau)$ can be expressed in the discrete time domain by a correlation sequence

$$\vec{r} = [R(0), R(1), ..., R(L-1)].$$

Similarly, one can replace $R_c(\tau)$ in (1) by $R_c(n) = \sum_{l=0}^{M} \psi_l e^{\lambda_l n}$ for $0 \leq n < L$. Define

$$\vec{\phi}_l = [1, e^{\lambda_l}, e^{2\lambda_l}, ..., e^{(L-1)\lambda_l}]^T \qquad \text{for } 0 \leq l \leq M. \tag{8}$$

For the matching in the discrete domain, one can rewrite (2) as

$$\min_{\{\vec{\lambda}, \vec{\psi}\}} E = ||\vec{r} - \sum_{l=0}^{M} \psi_l \vec{\phi}_l||^2, \qquad \text{s. t. } \vec{\psi} \geq 0 \tag{9}$$

where $||.||$ is the Euclidian norm.

This is a nonlinear least square (NLS) problem with constraints, i.e., a nonlinear programming problem, which is generally very difficult to solve since no efficient algorithms are available to date. The generic nonlinear programming techniques cannot be applied here due to its high time complexity and inability to locate the globe optimal solution. In the signal processing and detection field, a variety of techniques were developed to solve the NLS problem of the sinusoidal type without constraint. They are classified into *direct* and *indirect* approaches (refer to [20] for details). The direct one involves a multidimensional search of $\vec{\lambda}$, $\vec{\psi}$ and M for the minimization of E. Although the constraint can be easily incorporated, the direct approach suffers from its extremely high time complexity. The indirect one extracts $\vec{\lambda}$, $\vec{\psi}$ and M as accurately as possible without explicit attempt to minimizing E. The standard methods such as Prony, MUSIC and ESPRIT are examples of the indirect approach. In contrast to the direct approach, the indirect approach is computationally much more efficient but at the cost of suboptimal. Yet, none of these indirect methods can be applied here because of the constraint $\vec{\psi} \geq 0$.

Here we propose a heuristic algorithm which is composed of three steps. In the first step, we simply use the standard Prony method [20] to identify an intermediate solution $(\vec{\lambda}_{int}, \vec{\psi}_{int})$ without the real nonnegative constraint on $\vec{\psi}_{int}$. In the second step, the intermediate solution vector $\vec{\lambda}_{int}$ is much expanded to $\vec{\lambda}_{exp}$ with $\vec{\lambda}_{int} \subset \vec{\lambda}_{exp}$. The extra eigenvalues in $\vec{\lambda}_{exp}$ are generated through interpolation of the eigenvalues in $\vec{\lambda}_{int}$. It is based on the intuition that the eigenvalues of the constraint solution $\vec{\lambda}$ are likely in the neighborhood of the eigenvalues of the nonconstraint solution $\vec{\lambda}_{int}$. In other words, the correlation function captured by $(\vec{\lambda}_{int}, \vec{\psi}_{int})$, is

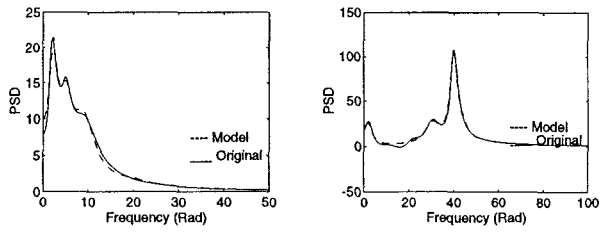

Figure 1: Two cases of power spectrum match

likely to resemble that by $(\vec{\lambda}, \vec{\psi})$, since the former is formulated with less constraint than the latter. The purpose of the expansion is to approximately achieve $\vec{\lambda} \subset \vec{\lambda}_{exp}$. Various interpolation techniques can be used to expand $\vec{\lambda}_{int}$ to $\vec{\lambda}_{exp}$. The one used here is a simple linear interpolation in the complex domain among all the eigenvalues in $\vec{\lambda}_{int}$, which is found sufficient in most our applications. The last step is to find $\vec{\psi}_{exp}$ of the given $\vec{\lambda}_{exp}$ with the real nonnegative constraint of $\vec{\psi}_{exp} > 0$, which will minimize the difference between the original correlation sequence $\vec{r}$ and the correlation function of $(\vec{\lambda}_{exp}, \vec{\psi}_{exp})$. This is achieved by the well known non-negative least squares (NNLS) method [20]. Note that most power elements in $\vec{\psi}_{exp}$ will be zero since the order of the correlation function $(\vec{\lambda}_{exp}, \vec{\psi}_{exp})$ should not be significantly greater that the order of the correlation function $(\vec{\lambda}_{int}, \vec{\psi}_{int})$. Finally, the power vector $\vec{\psi}$ consists of the nonzero power elements of $\vec{\psi}_{exp}$; the eigenvalue vector $\vec{\lambda}$ consists of those eigenvalues in $\vec{\lambda}_{exp}$, whose corresponding power element in $\vec{\psi}_{exp}$ are nonzero.

Note that if the original solution $(\vec{\lambda}_{int}, \vec{\psi}_{int})$ by the Prony method already satisifies the real nonnegative constraint, both expansion and minimization processes in steps 2 and 3 will not change the solution.

Two examples are taken to demonstrate the effectiveness of the heuristic algorithm. Note that the correlation function in the time domain is equivalently characterized by the power spectrum in the frequency domain. Taking the Fourier transform of $R_c(\tau)$ in (1), the power spectrum is expressed by

$$P_c(\omega) = \sum_{l=0}^{M} \frac{-2\psi_l \lambda_l}{\lambda_l^2 + \omega^2} \tag{10}$$

where each eigenvalue contributes a bell component. Each bell is described by its central frequency $Im\{\lambda_l\}$ and half power bandwidth $-2Re\{\lambda_l\}$ with average power ψ_l. Every pair of complex conjugate eigenvalues then contributes two bells which are symmetric at the central frequency $\pm Im\{\lambda_l\}$ Since lower frequency power has much more impact on the queueing performance than higher frequency power [5], it is more convenient to use the power spectrum than the correlation function for the second-order input statistical measurement. This is true especially since we are interested in the power spectral matching in the low-frequency region.

For simplicity, in the following two examples we assume that the original correlation sequence is exactly described by $(\vec{\lambda}_{int}, \vec{\psi}_{int})$ as the Prony method would get. In the first example, we describe the original power spectrum at $M = 12$ by

$$\vec{\lambda}_{int} = \{-1 \pm 5i,\ -1 \pm 2i,\ -3 \pm 9i,\ -1,\ -3 \pm 4i,\ -4 \pm 6i\},$$

$$\vec{\psi}_{int} = \{2 \pm 1i,\ 8 \pm 2i,\ 8 \pm 4i, 1 \pm 0.5i,\ 12 \pm 3i,\ 3 \pm 2i\}.$$

as shown in Fig. 1a. Here we have purposely chosen complex $\vec{\psi}$ for the original function with six pairs of complex conjugate eigenvalues. After the linear interpolation, the given $\vec{\lambda}_{int}$ at $M = 12$ is expanded to $\vec{\lambda}_{exp}$ at $M = 40$. Using the NNLS method for minimization, we then identify the following matched two vectors,

$$\begin{aligned}\vec{\lambda} \;=\; &\{-1.57 \pm 5.13i,\ -1.16 \pm 2.18i,\ -2.99 \pm 9.0i,\\ &-1.21 \pm 2.51i, -16.9 \mp 8.0i,\ -3.45 \pm 20.4i\},\end{aligned}$$

$$\vec{\psi} = \{12.78,\ 9.0,\ 23.7,\ 7.63,\ 12.39,\ 1.49\}.$$

Notice that with respect to each pair of complex conjugate eigenvalues, the two real positive power elements are identical and therefore only one of them is included in the definition of $\vec{\psi}$. It is obvious that $\vec{\lambda}$ cannot exactly match with $\vec{\lambda}_{int}$ because of the constraints. Yet, as one can see, the first three pairs of eigenvalues in $\vec{\lambda}$ are very close to the first three pairs of eigenvalues in $\vec{\lambda}_{int}$. This explains why the conctructed power spectral curve closely matches with the original one as shown in Fig. 1a.

In the second example shown in Fig. 1b, we choose the following original power spectrum at $M = 10$,

$$\vec{\lambda}_{int} = \{-2 \pm 2i,\ -2 \pm 40i,\ -4 \pm 30i,\ -4 \pm 20i,\ -6 \pm 10i\},$$

$$\vec{\psi}_{int} = \{30 \pm 5i,\ 100 \pm 20i,\ 50 \pm 20i,\ 10 \pm 20i,\ 20 \pm 10i\}.$$

An expansion set $\vec{\lambda}_{exp}$ at $M = 60$ is first generated by linear interpolation from the given set $\vec{\lambda}$ at $M = 10$. The minimization process then identifies 19 nonzero ψ pairs in $\vec{\psi}_{exp}$, many of which have very small values (representing negligible energy of the corresponding bell components in $P(\omega)$). Based on $\vec{\psi}_{exp}$, 5 pairs of complex conjugate eignevalues are then selected from $\vec{\lambda}_{exp}$, each of which contributes more than 5% of the total energy, given by

$$\begin{aligned}\vec{\lambda} \;=\; &\{-2.0 \pm 40.0i,\ 4.0 \pm 30.0i, -2.0 \pm 2.0i,\\ &-3.8 \pm 31.1i,\ -2.32 \pm 44.1i\},\end{aligned}$$

$$\vec{\psi} = \{199.52,\ 65.76,,\ 29.5,\ 26.05,\ 29.78\}.$$

Again, the first three pairs of the eigenvalues in $\vec{\lambda}$ is basically identical to the first three pairs in $\vec{\lambda}_{int}$. The two correlation functions, $(\vec{\lambda}_{int}, \vec{\psi}_{int})$ and $(\vec{\lambda}, \vec{\psi})$ are compared in Fig. 1b.

The CPU time for the matching in both cases is about 3 seconds on a workstation. In Section 5, we shall use this heuristic algorithm to match the power spectrum of various real traffic traces.

4 Index Search Algorithm

This section develops a fast index search algorithm (ISA) for construction of $\vec{a}$ from $\vec{\lambda}$, based on $\vec{\lambda}_c = \sqrt{N}\vec{a}\mathbf{F}^*$ in (3) and $\vec{\lambda} \subset \vec{\lambda}_c$ subject to $\vec{a} \geq 0$. Subsection 4.1 formulates the problem. Subsection 4.2 presents the ISA algorithm, which is further optimized in Subsection 4.3 and 4.4. Subsection 4.5 discusses the complexity issues for the ISA.

4.1 *Index Search Problem*

Since $\vec{a}$ is real, a complex λ_{cj} must be the conjugate of $\lambda_{c(N-j)}$ in $\vec{\lambda}_c$. In other words, the equation set by λ_{cj} in $\vec{\lambda}_c = \sqrt{N}\vec{a}\mathbf{F}^\star$ is equivalent to the equation set by $\lambda_{c(N-j)}$; only one complex eigenvalue in each conjugate pair needs to be taken into account in $\vec{\lambda}_c = \sqrt{N}\vec{a}\mathbf{F}^\star$. Further, since all the unwanted eigenvalues, $\lambda_{cj} \in \vec{\lambda}_c$ but $\notin \vec{\lambda}$, are arbitrarily assigned, they shall be completely removed from $\vec{\lambda}_c = \sqrt{N}\vec{a}\mathbf{F}^\star$ for the derivation of $\vec{a}$. In consequence, one can rewrite $\vec{\lambda}_c = \sqrt{N}\vec{a}\mathbf{F}^\star$ as

$$\mathbf{A}(\vec{i}_m)\vec{a} = \vec{b}(\vec{i}_m) \qquad \text{and} \qquad \vec{a} \geq 0 \tag{11}$$

with $\vec{a} = [-a_0, a_1, ..., a_{N-1}]^T$, $\vec{i}_m = [i_1, i_2, ..., i_m]$ and

$$\vec{b}(\vec{i}_m) = [-\mathrm{Re}(\lambda_{i_1}), \mathrm{Im}(\lambda_{i_1}), ..., -\mathrm{Re}(\lambda_{i_m}), \mathrm{Im}(\lambda_{i_m})]^T,$$

$$\mathbf{A}(\vec{i}_m) = \begin{bmatrix} -1 & 1 & 1 & \cdots & 1 \\ 1 & -c_{i_1,1} & -c_{i_1,2} & \cdots & -c_{i_1,N-1} \\ 0 & s_{i_1,1} & s_{i_1,2} & \cdots & s_{i_1,N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & -c_{i_m,1} & -c_{i_m,2} & \cdots & -c_{i_m,N-1} \\ 0 & s_{i_m,1} & s_{i_m,2} & \cdots & s_{i_m,N-1} \end{bmatrix}$$

where $c_{i,j} \stackrel{\text{def}}{=} \cos(2\pi ij/N)$ and $s_{i,j} \stackrel{\text{def}}{=} \sin(2\pi ij/N)$. m represents the number of distinctive, nonzero eigenvalues in $\vec{\lambda}$, where each pair of complex conjugate eigenvalues is counted as one distinctive eigenvalue. For $\vec{\lambda} \subset \vec{\lambda}_c$, the index vector $\vec{i}_m$ provides a specific index assignment of $\vec{\lambda}$ in $\vec{\lambda}_c$, i.e., $\vec{\lambda} = [\lambda_{i_1}, \lambda_{i_2}, ..., \lambda_{i_m}]$. A feasible solution $\vec{a}$ is obtained by identifying a specific $\vec{i}_m$ to satisfy (11).

Denote the set of entire index assignment space by $\mathbf{I}_m = \{\vec{i}_m : i_l \in \{1, 2, ..., N-1\}, \forall l \text{ for } i_l \neq i_j \text{ if } l \neq j\}$. The index space of $\mathbf{I}_m$ is then on the order of N^m. Our objective is to find an index vector $\vec{i}_m \in \mathbf{I}_m$ for a feasible solution $\vec{a}$ in (11). For convenience, we also denote the set of all feasible index assignments by $\mathbf{I}'_m$ with $\mathbf{I}'_m \subset \mathbf{I}_m$. Typically, $\mathbf{I}'_m$ is a small fraction of $\mathbf{I}_m$.

Here we briefly discuss the ad-hoc scheme developed in [6] for solving (11) and explain the crucial importance of the proposed fast algorithm to the measurement-based modeling. In [6], (11) was formulated as a linear programming problem with an arbitrarily assigned objective function; the simplex method was employed to search for a feasible solution $\vec{a}$ at each selected $\vec{i}_m \in \mathbf{I}_m$. Since $\vec{i}_m$ is randomly selected at each time with incremental change of every index $i_l \in \vec{i}_m$, we call the scheme as random selection algorithm (RSA). The random search of this kind is found extremely ineffective for large N and m. Note that N has to be large for the expansion of solution space, which is typically in the range of several hundreds in our application. For instance, taking $N = 100$ at $m = 3$, a complete search of the index space $\mathbf{I}_3 = 10^6$ by RSA requires several hours of CPU time on SUN workstation. The exponential expansion of the index space with m renders any brute force search algorithm to be virtually impractical for $m > 3$. In the RSA implementation, the number of allowable index searches at each given N was limited to a few hundreds, which only covers about 0.01% of the index space for $N = 100$ at $m = 3$. Since $\mathbf{I}'_m$ is only a small fraction of $\mathbf{I}_m$ such as 10^{-5} or less in most our experiments, the probability of finding a feasible $\vec{i}_m$ in $\mathbf{I}_m$ by RSA diminishes rapidly as m increases. Yet, confining $m \leq 3$ is certainly insufficient to capture the complex correlation behavior of multimedia traffic.

4.2 *Tree Search Algorithm*

Consider the index search problem of order n for $0 < n \leq m$. At each given n we have the notation of $\vec{i}_n$, $\mathbf{I}_n$ and $\mathbf{I}'_n$. Further define $\vec{i}_n = [\vec{i}_{n-1}, i_n]$, where $\vec{i}_{n-1}$ represents the first $(n-1)$ indices of $\vec{i}_n$. We introduce a new set $\mathbf{J}_{n-1} = \{\vec{i}_{n-1} : \vec{i}_n \in \mathbf{I}'_n\}$, i.e., $\mathbf{J}_{n-1}$ is the set of the index vectors which are composed of the first $(n-1)$ indices of $\vec{i}_n \in \mathbf{I}'_n$. The ISA algorithm is developed on the basis of the following relation:

$$\mathbf{J}_n \subset \mathbf{I}'_{n-1} \qquad \forall n, \tag{12}$$

which is proved by contradiction. Consider an $\vec{i}_n \in \mathbf{I}'_n$, whose first $(n-1)$ elements are denoted by the corresponding $\vec{i}_{n-1} \in \mathbf{J}_{n-1}$. Suppose $\vec{i}_{n-1} \notin \mathbf{I}'_{n-1}$. By definition, $\vec{i}_n$ must be a feasible solution of (11). Since $\vec{i}_{n-1}$ represents the first $(n-1)$ elements of $\vec{i}_n$, it must also be a feasible solution of (11) with its last two equations removed. Yet , removing the two last equations in (11) is equivalent to formulating the index search problem of order $n-1$, whose solutions are denoted by $\vec{i}_{n-1} \in \mathbf{I}'_{n-1}$, which contradicts to the assumption $\vec{i}_{n-1} \notin \mathbf{I}'_{n-1}$. Hence, we must have $\mathbf{J}_{n-1} \subset \mathbf{I}'_{n-1}, \forall n$.

The relation simply states that for any $\vec{i}_n \in \mathbf{I}'_n$, we must have $\vec{i}_{n-1} \in \mathbf{I}'_{n-1}$ with $\vec{i}_n = [\vec{i}_{n-1}, i_n]$. Hence, starting at $n = 1$ we get $\mathbf{I}'_1$ with $\vec{i}_1 = [i_1]$. The solution $\mathbf{I}'_2$ can then be obtained by the search of index i_2 in $\vec{i}_2 = [\vec{i}_1, i_2]$ for $\vec{i}_1 \in \mathbf{I}'_1$. In general, the solution $\mathbf{I}'_n$ is obtained by the search of index i_n in $\vec{i}_n = [\vec{i}_{n-1}, i_n]$ for every $\vec{i}_{n-1} \in \mathbf{I}'_{n-1}$.

Since we are interested in just getting one feasible solution in $\mathbf{I}'_m$, the ISA is best implemented by tree search. Initially, we have $\mathbf{I}_0 = \emptyset$ as the root of the tree. Every branch at level 1 is then formed by $\vec{i}_1 \in \mathbf{I}'_1$. From $\vec{i}_n = [\vec{i}_{n-1}, i_n]$, it is clear that every branch at level n is formed by $\vec{i}_n \in \mathbf{I}'_n$, which is built on top of a branch at level $n-1$. The branch $\vec{i}_{n-1}$ terminates at level $n-1$ if $\vec{i}_n = [\vec{i}_{n-1}, i_n] \notin \mathbf{I}'_n, \forall i_n$.

The search starts at the root to follow a certain path to climb up the tree. Before reaching a final solution on level m, if no i_n is identified at level n on a given branch $\vec{i}_{n-1}$ of level $n-1$, it will go back to select another branch $\vec{i}_{n-1}$ for continuation of identification of i_n on level n. Once such an i_n is identified, which creates a branch $\vec{i}_n$ on the next level n, the next search process will start for identification of i_{n+1} on level $n+1$, until reaching the level m. If no i_n is identified for all $\vec{i}_{n-1}$'s in $\mathbf{I}'_{n-1}$, the search process will go back to level $n-2$ to select another branch $\vec{i}_{n-2}$ for continuation of identification of i_{n-1}. Such a search process continues and terminates either when a feasible solution is reached at the final level m, or when the entire space is searched without finding a feasible solution.

The ISA can be stated as the following:

Input $\{m, N, \vec{\lambda}\}$;
count $= [0, 0, ..., 0]$;
$n = 0$;

for $i_n = count(n) + 1 : N - 1$

$\vec{i}_n$=$\{\vec{i}_{n-1}, i_n\}$;
Solve (11) by modified simplex (See Section 4.3) for $\vec{i}_n$;
if feasible solution is found
  if $n = m$, output:*solution found*! and exit
  else $count(n) = i_n$; $n = n + 1$;
else $count(n) = 0$; $n = n - 1$;

print: *No feasible solution found*!
end

The implementation of fast ISA algorithm is based on two principles: (a) to minimize the total number of searches and (b) to minimize the computation time of each search, as described in the next two subsections.

4.3 *On Reduction of Number of Searches*

In the tree search, $\mathbf{I}'_{n+1}$ tends to be small if $\mathbf{I}'_n$ is small, $\forall n$. This is because $\vec{i}_{n+1} = [\vec{i}_n, i_{n+1}]$ for $\vec{i}_{n+1} \in \mathbf{I}'_{n+1}$ and $\vec{i}_n \in \mathbf{I}'_n$. That is, the construction of $\mathbf{I}'_{n+1}$ is the search of feasible indexes i_{n+1} at each given $\vec{i}_n \in \mathbf{I}'_n$. In consequence, the ISA algorithm can be much accelerated if the original sets $\mathbf{I}'_n$ for small n are to be kept small.

Each time as n increases, a new eigenvalue will be selected from $\vec{\lambda}$. The size of $\mathbf{I}'_n$ is also directly associated with the selection of the n-th eigenvalue in $\vec{\lambda}$. From [5] we know that the eigenvalues of an N-state Markov chain must satisfy $\frac{-\mathrm{Re}\{\lambda_{i_n}\}}{\mathrm{Im}\{\lambda_{i_n}\}} \geq |\tan(\pi/N)|$ for $\lambda_{i_n} \in \vec{\lambda}$. In fact, the construction of an N-state Markov chain for given N is more difficult when the value of $\frac{-\mathrm{Re}\{\lambda_{i_n}\}}{\mathrm{Im}\{\lambda_{i_n}\}}$ is closer to the bound $|\tan(\pi/N)|$. In one extreme, one can always construct a 2-state Markov chain from any real eigenvalue (i.e., $\mathrm{Im}\{\lambda_{i_n}\} = 0$). In the other extreme, no Markov chain contains $\mathrm{Re}\{\lambda_{i_n}\} = 0$. Thus, all the eigenvalues in $\vec{\lambda}$ are presorted in the increasing order of the ratio $\frac{-\mathrm{Re}\{\lambda_{i_n}\}}{\mathrm{Im}\{\lambda_{i_n}\}}$, among which the smaller one is always first selected in the tree search as the level n increases. Our study shows that,on averge, a reduction of several orders of magnitude in the number of searches can be reached by the tree search algorithm to find a feasible solution, using the presorting as compared to random selection.

Another measure implemented to enhance the chance to find a feasible solution is, as an option, to allow a certain degree of tolerence of error for eigenvalues associated with the bells of power spectrum in the high frequency region. As the feasibility constraints for the N-state Markov chain imply, it is the narrow-band,high frequency bells in power spectrum which restrict the feasibility of the problem.

4.4 *On Reduction of Computation Time Per Search*

The computation time of each search on level n is to solve $\mathbf{A}(\vec{i}_n)\vec{a} = \vec{b}(\vec{i}_n)$ for $\vec{a} \geq 0$ in (11), which grows rapidly with n. Our design is based on the simplex algorithm, which can be divided into two phases. Phase I is to find a feasible solution; Phase II is to locate the optimal solution based on the feasible one. Since we are only interested in getting a feasible solution, Phase II and its associated objective function are eliminated. Such modification also removes a whole row in the tableau of the simplex algorithm, reducing the size of problem from $N \times n$ to $(N-1) \times (n-1)$.

The structure of the ISA is well suited for the so called sensitivity analysis [19]. Recall that the search of i_n in $\vec{i}_n = [\vec{i}_{n-1}, i_n]$ on level n is performed at each given $\vec{i}_{n-1} \in \mathbf{I}'_{n-1}$. For $i_n \in \{1, 2, ..., N-1\}$, each $\vec{i}_{n-1}$ is applied at most $(N-1)$ times in the search of i_n before the selection of the next $\vec{i}_{n-1}$. Further, $\mathbf{A}(\vec{i}_n)\vec{a} = \vec{b}(\vec{i}_n)$ is equivalent to $\mathbf{A}(\vec{i}_{n-1})\vec{a} = \vec{b}(\vec{i}_{n-1})$ except with the addition of the last two equations which are associated with a newly selected eigenvalue. It implies that the output tableau of the simplex obtained for $\vec{i}_{n-1}$ on level $n-1$ can be directly used by the simplex for $\vec{i}_n$ on level n, which reduces the time complexity of each search by a factor proportional to n. The following provides the detail formulation which is known as the sensitivity analysis in the linear programming field.

Suppose we want to find a feasible solution for the following constrained linear equations

$$\mathbf{A}_n\vec{x} = \vec{g}_n \qquad \text{s.t.} \qquad \vec{x} \geq 0. \tag{13}$$

where $\mathbf{A}_n$ is an $n \times N$ matrix $[a_{ij}]$, $\vec{x} = [x_1, x_2, ..., x_N]^T$, and $\vec{g}_n = [g_1, g_2, ..., g_n]^T$ with non-negative elements.

The standard simplex approach in phase I for a feasible solution is to introduce n artificial variables $\vec{z}_n = [z_1, z_2, ..., z_n]$ and change (13) into a linear programming problem:

$$\begin{array}{ll} \text{Maximize} & Z = -\sum_{k=1}^{n} z_k \\ \text{Subject to} & \\ & \vec{z}_n = \vec{g}_n - \mathbf{A}_n\vec{x}, \\ & \vec{x} \geq 0, \qquad \vec{z}_n \geq 0. \end{array} \tag{14}$$

The objective function Z is the so-called auxiliary objective function. The reason for introduction of n artificial variables is that the search for an optimal solution by simplex must start from a initial feasible solution. The above linear programming problem has an initial feasible solution simply by setting $\vec{x} = \vec{0}$ and $\vec{z}_n = \vec{g}_n$ where $\vec{0}$ is a zero vector. Obviously, the auxiliary objective function will be maximized for nonnegative $\vec{z}_n$ if all the $z'_i s$ are zero. To this end, the simplex has to swap all the left-hand z_i's to the right-hand of the equations while keeping the equations feasible at each swap. Once all the $z'_i s$ are swaped to the right hand side of the equations, a feasible solution for $\vec{x}$ is found and the maximum of Z is attained by letting all the right hand side variables including all $z'_i s$ equal to zero. The number of swaps required grows rapidly with the increase of n. An empirical complexity for the simplex is $O(N \times n)$.

In the sensitivity analysis of (13), assume that we have already found a feasible solution for the first $(n-1)$ equations in (13) and the output of the simplex takes the form:

$$\vec{x}_b = \vec{c}_{n-1} + \mathbf{G}_{n-1}\vec{x}_c \tag{15}$$

where $\vec{x}_b$ is the basic variable vector with $(n-1)$ elements, and $\vec{x}_c$ is the complementary (nonbasic) variable vector with $N-n+1$ elements. $\vec{c}_{n-1}$ is a constant column vector with all $(n-1)$ elements non-negative and $\mathbf{G}_{n-1}$ is a $(n-1) \times (N-n+1)$ constant matrix. Without loss of generality, assume $\vec{x}_b$ is a vector composed of the first $(n-1)$ elements in $\vec{x}$. Then the feasible solution can be immediately read off from the above equation as $\vec{x} = [\vec{c}^T_{n-1}, \vec{0}]$. This feasible solution may not satisfy the n-th equation which has not been included.

Since $\vec{x}_b$ has been expressed in terms of $\vec{x}_c$ in (15), the n-th equation can be cast in a form without having $\vec{x}_b$ in it by substituting the solution for $\vec{x}_b$, which has the following form,

$$\vec{u}\vec{x}_c = f \tag{16}$$

with some constant row vector $\vec{u}$ and a constant f. Without loss of generality we assume $f \geq 0$. The problem (13) is then equivalent to solving (15) and (16) simultaneously with non-negative constraints. To find a feasible solution by simplex, we need to find an initial feasible solution by introducing some artificial variables as well as an auxiliary objective function. Since (15) is already in a feasible format, all we need to do is to introduce a single artificial variable z for equation (16) and take the auxiliary objective function as the negative of z. Thus, finding a feasible solution for the original problem (13) with n constraint equations is now equivalent to solve for the following linear programming problem,

$$\begin{array}{ll} \text{Maximize} & -z \\ \text{Subject to} & \\ & \vec{x}_b = \vec{c}_{n-1} + \mathbf{G}_{n-1}\vec{x}_c, \\ & z = f - \vec{u}\vec{x}_c, \\ & \vec{x} \geq 0, \\ & z \geq 0. \end{array} \tag{17}$$

The above approach is generally known as the sensitivity analysis. Notice that only one artificial variable z has been added to the problem, whereas the start-from-scratch simplex (14) requires to add n artificial variables which is obviously undesirable. Further, the initial solution in (17) is very likely close to feasible $\vec{x}$ in the sense that there is only one artificial variable z to be swaped to the right hand side of the equations in (17). The sensitivity analysis requires a much smaller number of swaps than that of the start-from-scratch approach.

Given a feasible solution $\vec{i}_{n-1}$ on level $n-1$, one can readily apply the above algorithm to (11) for feasible solution $\vec{i}_n$ on level n. Note that two rows will be added in (11) when the nth eigenvalue is selected. The first row consists of cos coefficients; the second row consists of sin coefficients. The first row is first added and solved by the simplex with sensitivity analysis. If and only if its feasible solution exists, the second row will then be added. When no feasible solution is identified in either case, the search will go back to level $n-1$. The simplex is modified to creat a new output tableau each time when the simplex is called in such a way that it does not take extra running time while keeping input tableau intact, so that the input tableau can be reused. Our comprehensive case study further indicates that the number of swaps required for $n \leq 8$ and $N \sim 100$ is on average less than 30, which grows very slowly with n. The sensitivity analysis therefore reduces the time complexity of ISA significantly.

4.5 Complexity Issues

The space complexity of the ISA is mainly measured by the memory requirement for the tableaus of the simplex. Each tableau takes $(n+1) \times (N+n-1)$ unites of data with double precision on level n. Approximately about $2n$ tableaus need to be stored in the memory, which amounts to less than 1 megabytes for $n \leq 8$ at $N = 101$. The space complexity is therefore not a problem for the ISA design.

m	3	4	5	6	7	8
cpu(m)(min.)	0.027	0.084	0.59	1.57	3.26	5.10
cpu(m)/cpu(m-1)		3	7	3	2	1.5

Table 1: Average CPU times measured over 100 randomly generated cases at $N = 101$

Time complexity is highly dependent on the given eigenvalues and it is difficult to identify the average or worst-case scenarios. Instead of doing theoretical complexity analysis, a comprehensive case study was carried out for complexity measurement. To choose the range of eigenvalues large enough to cover diverse situations, we randomly generated $-\text{Re}\{\lambda_{i_n}\}$ and $\text{Im}\{\lambda_{i_n}\}$ in the range of [0, 100] with uniform distribution. In each case, m distinctive eigenvalues were randomly generated to form $\vec{\lambda}$ for $2 < m < 9$. While keeping N in the range of a few hundred, our study shows that ISA is a powerful and practically useful algorithm for the index search. For instance, the average computation time of ISA at $N = 101$ and $m = 7$ is less than two miniutes on a SUN workstation to find a feasible solution, providing the existence of feasible solutions. For cases without feasible solutions, the average computation time is found less than five minutes for the complete search. Such averages were taken over 100 randomly generated cases. Comparatively, the brute force approach would take years on average for complete searches for the same cases. Similarly for 100 randomly generated cases at state space $N = 401$ and $m = 5$, 90 of them found feasible solutions among which 90% finished the search within 5 minutes and 70% within one minute. There are only two cases which found feasible solution with CPU time greater than 20 minutes where the worst case is about 40 minutes. The cases without feasible solution took about an hour on average for complete search where the worst case is about 790 minutes.

To see the trend of CPU times as a function of m, we fixed state space at $N = 101$ and took 100 randomly generated cases for each m from 3 to 8 and measured the average CPU time per case. For a feasible case we measured the CPU time to find the first feasible solution; for an infeasible case we measured the total CPU time for complete search. The results are listed in Table 1. As m increases from 3 to 8, which is in the range of our practical interest, the average CPU time grows from 0.027 minute to 5.10 minutes. Note that the possiblity to find a feasible solution for $m \geq 8$ at $N = 101$ becomes much reduced (less than 10%). For comparison purposes, the ratio of the CPU time at adjacent m is also provided in Table 1. After reaching the peak at $m = 5$, the growth rate of the average CPU time declines significantly as m further increases.

5 Real Traffic Application

The proposed algorithms are applied to match the statistics of a wide spectrum of real traffic traces, ranging from Ethernet data, JPEG video, MPEG video to internet TCP data. We focus on the comparison of sequences generated from the matched CMPP with the original traces. The valid-

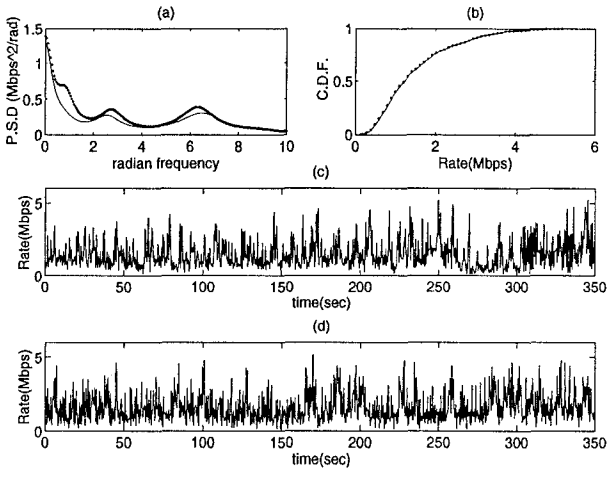

Figure 2: Comparison of Ethernet trace (dotted line) with matched CMPP (solid line) (a) power spectrum (b) rate distribution (c) original filtered trace (d) CMPP sequence

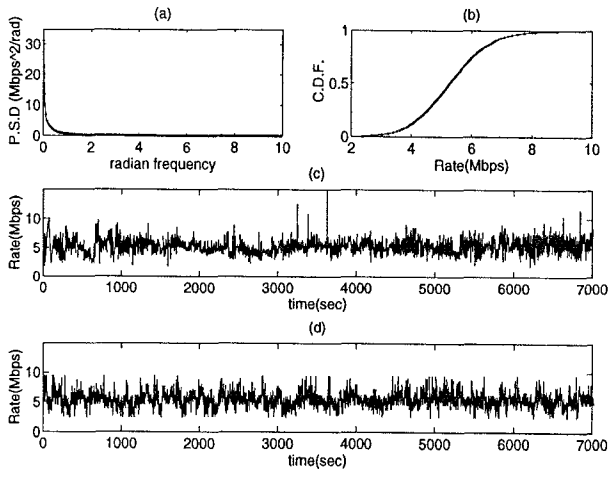

Figure 3: Comparison of a JPEG trace (dotted line) with matched CMPP (solid line) (a) power spectrum (b) rate distribution (c) original filtered trace (d) CMPP sequence

ity of the CMPP matching for queueing analysis was carefully examined in [25] by comparison of analytical queueing solutions of CMPP's to the simulated queuing solutions of original traces. Notice that each matched CMPP can be used as a traffic generator of certain type to load a network for testing and performance evaluation. One can also match the statistics of aggregate traces by a single CMPP which can be used as a traffic generator of aggregate sources. In practice, various traffic generators can be built from a libary, which consists of collective statistics of different traces. They are used not only for analytical modeling, but more importantly for generation of massive traffic flows within a large high-speed network for testing and simulation study. Each traffic generator is simply represented by two vectors $\vec{a}$ and $\vec{\gamma}$. In contrast, every real traffic trace can take a huge storage space to generate.

Since low-frequency traffic behavior (or, large time varying scales) has dominant impact on network performance as compared to high-frequency behavior [5], our statistical matching in this study are confined to the low-frequency range $\omega \in [0,\ 10]$ radian, which corresponds to the range of time scales longer than 0.6 second. In other words, all the traces are pre-filted by moving average operation for getting important low-frequency statistics.

We first take a 6-minute Ethernet data trace from the ftp site: thumper.bellcore.com. The results are displayed in Fig. 2. The dotted lines in Fig. 2a,b provide the collective power spectrum and rate cumulative distribution of the trace. The power spectrum, found by the proposed heuristic algorithm, consists of five distinctive complex eigenvalues ($m = 5$). But we cannot find a feasible CMPP for $N \leq 401$ unless with a certain degree of error tolerance as discussed in Section 4.3. Fig. 2 shows such a matched example by a CMPP at $N = 401$ with 30% error tolerance. While the matching error in power spectrum is relatively significant, its distribution matching is excellent due to the selection of a large N. The dynamics of the two traces in Fig. 2c,d are also found in a good agreement.

In the second example we take a 2-hour JPEG segment of the movie *Star wars* from the same ftp site at Bellcore. As one can see, the power spectrum highly concentrates on a low-frequency band, which is identified by four real eigenvalues plus one complex eigenvalue. A CMPP is matched at $N = 401$ with 0% error tolerance. The corresponding results are plotted in Fig. 3. Both power spectrum and distribution are matched very well except for a small tail portion of the distribution. This is why the peak rate of the CMPP sequence can only reach 9Mbps whereas the peak rate of the real trace can reach more than 10Mbps. Nevertheless, the CMPP sequence mainly captures the overall feature of the real JPEG trace as found in Fig. 3c,d.

Since the internet traffic constitutes 80% of the existing data traffic worldwide, it would be interesting to model the internet traffic. The data stream we used was gathered at the Lawrence Berkeley Laboratory's wide-area Internet gateway. For a comprehensive account of the TCP traffic traces and its modeling, readers should refer to the paper by V. Paxson and S. Floyd [3] and the references therein. The trace was collected from 2pm to 4pm on Jan. 20, 1994 and there are 1.3M packets collected. A time stamp was recorded for each packet arrived. The data file is lbl-pkt-4.tcp. The first 50,000 packets of the trace are used for our matching, which corresponds to 231.9 second aggregate TCP traffic going through the gateway. We need to convert the time stamp sequence into the packet rate sequence by counting the number of packets arrived in each fixed time slot (0.1 second). The results are presented in Fig. 4, where the traces are recorded by number of packets per second . There are four distinctive complex eigenvalues in the spectral matching. A feasible matched CMPP is found at $N = 301$ with 0% error tolerance.

We finally consider the MPEG video traffic whose behavior normally exihibits strong periodic nature due to its periodic IPB frame coding scheme. Fig. 5 shows an example of the power spectrum of an MPEG trace obtained from the ftp site:

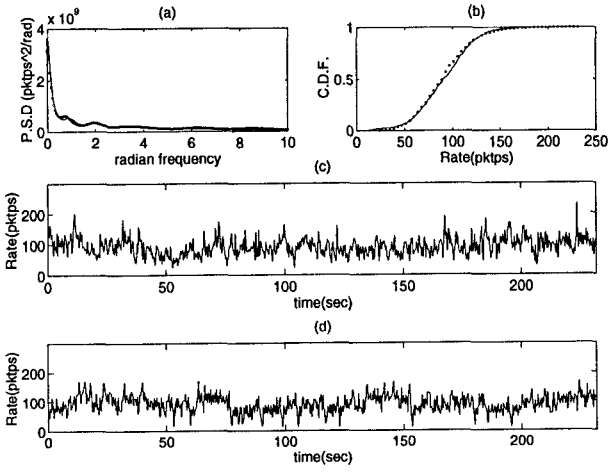

Figure 4: Comparison of a TCP trace (dotted line) with matched CMPP (solid line) (a) power spectrum (b) rate distribution (c) original filtered trace (d) CMPP sequence

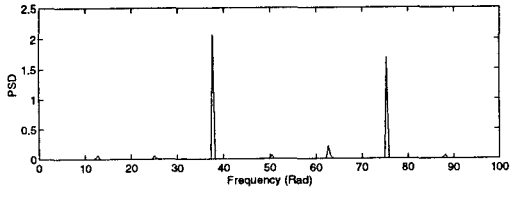

Figure 5: Power spectrum of a MPEG trace

tenet.berkeley.edu in the public domain. The sharp peaks are generated at the harmonics of frequency $\omega = \pi/T$ where T is the period of GOP. The trace is a 90-minute CNN news including commercials with the frame size 320x240 at the frame rate 30 fps. The GOP pattern is IBBPBBPBBPBBPBB and there are 15 frames per GOP. A so-called Futuretel hardware coder is used such that the picture quality is adaptively lowered during a high-action or colorful scene, in order to maintain its target coding rate. Such an adaptive variable encoding scheme is likely to be implemented for MPEG video transmission to comply with the agreement preset in the traffic descriptor during the connection setup in ATM networks. With the variable encoding scheme, the energies contributed by GOP harmonics become much more significant than the energies in the low frequency band contributed by scene changes.

The proposed ISA algrithm is unable to capture the GOP peaks in the video power spectrum as shown in Fig. 5. This is primarily due to the fundamental limit of Markov chains, whose eigenvalues cannot represent narrow-band high-frequency bells in power spectrum unless with a sufficiently large state space. The difficulty for circulant to capture a narrow-band high-frequency bell has been discussed in Section 4.3 on the subject of eigenvalue presorting.

The periodic burstiness nature of MPEG video has made its effective bandwidth allocation difficult in ATM networks. One possible solution, which currently is under intense study,

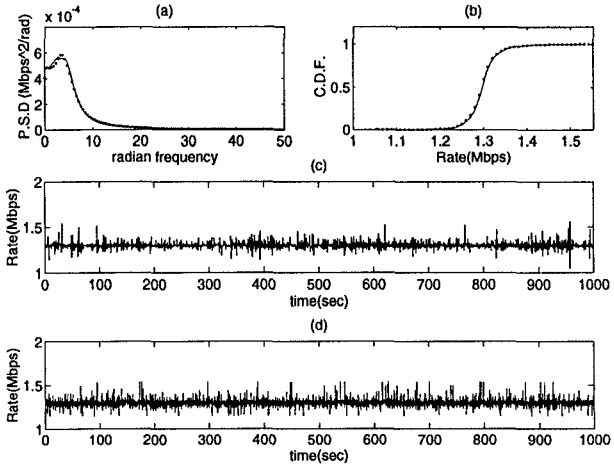

Figure 6: Comparison of a smoothed MPEG trace (dotted line) with matched CMPP (solid line) (a) power spectrum (b) rate distribution (c) original filtered trace (d) CMPP sequence

is to smooth each MPEG video stream before entering the network with its end-to-end delay constraint [24]. Hence, it would be interesting to match the smoothed MPEG traces for network traffic analysis. For example, such a smoothed trace can be generated from the original one by uniformly redistributing the bits of each GOP over its 15 frames, which yields a fixed smoothing delay of half second at the network entry point. The optimal design of video smoothing scheme is beyond the scope of the current paper. One may refer to [23, 24] for further information. The matching results of the smoothed video trace by CMPP with five complex eigenvalues at $N = 401$ are displayed in Fig. 6. As one can see, all the peaks of GOP harmonics in Fig. 5 have now been removed upon smoothing; only the power spectrum of low-frequency scene changes remain in Fig. 6. Note that the low-frequency scene change behavior of video has more significant impact on network delay performance than the relative high-frequency behavior of periodic GOPs [26].

6 Conclusion

This paper have developed algorithms to solve two most critical issues in measurement-based traffic modeling with the circulant structure. One is the parametric identification of the autocorrelation function $R(\tau)$. Because of the real non-negative constraint $\vec{\psi} \geq 0$, most existing techniques in the related signal processing field fail to apply. We have successfully developed a heuristic algorithm which is found fast and robust by numerical study based on real traces. The second issue is the construction of traffic model to match with complicated $R(\tau)$, which is originally an inverse eigenvalue problem but transformed to an index search problem for the circulant. Due to the large index space, however, any brute force search method fails to succeed especially if $R(\tau)$ is in complex form. The proposed fast index search algorithm has reduced the time complexity of the search by many orders of magnitude. For example, the typical CPU time for the

traffic modeling with $R(\tau)$ consisting of five or six complex exponential terms is found in the range of a few minutes by the proposed algorithms (which otherwise would take monthes by a brute force method). In real traffic application, our study shows that the sequences generated by the matched analytical model capture the characteristics of the real traffic traces very well.

Acknowledgement

One of the authors, H. Che, would like to thank Professor Patrick Jaillet for helpful discussion on design of the index search algorithm.

References

[1] W. E. Leland, M. Taqqu, W. Willinger, D. V. Wilson, "On the Self-Similar Nature of Ethernet Traffic," Proc. *SIGCOM93*, 1993, San Francisco, California, pp. 183-193.

[2] M. W. Garrett, W. Willinger, "Analysis, Modeling and Generation of Self-Similar VBR Video Traffic" Proc. ACM Sigcomm, London, September 1994, pp. 269-280.

[3] V. Paxson and S. Floyd, "Wide-area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Transactions on Networking*, pp.226-244, June 1995.

[4] S. Q. Li and C. Hwang, "On Input State Space Reduction and Buffer Noneffective Region," *Proc. IEEE Infocom'94 Conference*, June 1994, pp. 1018-1028.

[5] S. Q. Li and C. Hwang, "Queue Response to Input Correlation Functions: Continuous Spectral Analysis," *IEEE/ACM Trans. Networking*, Vol. 1, No. 6, Dec. 1993, pp. 678-692.

[6] C. L. Hwang and S. Q. Li, "On the Convergence of Traffic Measurement and Queueing Analysis: A Statistical-MAtch Queueing (SMAQ) Tool," *IEEE Infocom'95 Conference*, April 1995, pp. 602-613.

[7] L. A. Kulkarni and S. Q. Li, "Measurement-Based Traffic Modeling: Capture Important Statistics," submitted to *Journal of Stochastic Model* (also available on web page:http://mocha.ece.utexas.edu/ sanqi/papers.html).

[8] Steven M. Kay, *Modern Spectral Estimation: Theory & Application*, Prentice-Hall, 1988.

[9] R. Roy, A. Paulraj and T. Kailath, "ESPRIT - Estimation of Signal Parameters via Rational Invariance Technique," *IEEE Trans. Acoustics, Speech and Signal Processing*, Vol. 34, No. 5, Oct. 1986.

[10] P. Stoica and A. Nehorai, "Performance Comparison of Subspace Rotation and MUSIC Methods for Direction Estimation," *IEEE Trans. Signal Processing*, Vol. 39, No. 2, Feb. 1991.

[11] M. B. Priestley, *Spectral Analysis and Time Series*, Vol. 1, Academic Press, 1981.

[12] J. Ye and S. Q. Li, "Folding Algorithm: A Computational Method for Finite QBD Processes with Level-Dependent Transitions" *IEEE Trans. Commu.*, Vol. 42, No. 2, Feb. 1994, pp. 625-639.

[13] A. Elwalid, D. Heyman, T. V. Lakshman, D. Mitra, and A. Weiss, "Foundamental Bounds and Approximations for ATM Multiplexers with Applications to Video Teleconferencing," *IEEE J. Select. Areas Commun.*, vol. 13, No.6, pp. 1004-1016,1995.

[14] D. L. Jagerman and B. Melamed, "The Transition and Autocorrelation Structure of TES Processes; Part I: General Theory," *Stochastic Models*, Vol. 8, No. 2, 1992, pp. 193-219.

[15] V. S. Frost, B. Melamed, "Traffic Modeling for Telecommunications Networks," *IEEE Communications Magazine*, pp. 70-81, March 1994.

[16] J. D. Pruneski and S. Q. Li, "The Linearity of Low Frequency Traffic Flow: an Intrinsic I/O Property in Queueing System," *Proc. IEEE Infocom'95*, April 1995, pp. 613-623.

[17] Y. H. Kim and S. Q. Li, "Timescale of Interest in Traffic Measurement for Link Bandwidth Allocation Design," *Proc. IEEE Infocom'96 Conference*, April, 1996.

[18] S. Q. Li and C. L. Hwang, "Queue Response to Input Correlation Functions: Discrete Spectral Analysis," *IEEE/ACM Trans. Networking*, Vol. 1, No. 5, Oct. 1993, pp. 522(received the *IEEE Infocom'92* Conference Paper Award).

[19] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1993.

[20] S. K. Mitra, and J. F. Kaiser, ed., *Handbook for Digital Signal Processing*,John Wiley & Sons, 1993.

[21] O. Rose, "Statistical Properties of MPEG Video Traffic and Their Impact on Traffic Modeling in ATM systems," University of Wuerzburg. Institute of Computer Science Research Report Series. Report No. 101. Feb. 1995.

[22] P. Pancha and M. El. Zarki,"Bandwidth Requirements of Variable Bit Rate MPEG Sources in ATM Networks," *Proc. of Infocom'93*,pp. 902-909,Mar. 1993.

[23] S. S. Lam, S. Chow, and D. K. Y. Yau, "An Algorithm for Lossless Smoothing of MPEG Video," *Proc. of ACM SIGCOMM'94*, Aug. 1994.

[24] E. W. Knightly and P. Rossaro, "Improving QoS through Traffic Smoothing," *Proc. of IFIP IWQoS'96.*

[25] W. C. Lau and S. Q. Li, "Statistical Multiplexing and Buffer Sharing in Multimedia High-speed Networks:A Frequency Domain Perspective," *Proc. of GLOBECOM'95*, Dec. 1995.

[26] S. Q. Li, S. Chong, C. Hwang "Link Capacity Allocation and Network Control by Filtered Input Rate in High Speed Networks," *IEEE/ACM Trans. Networking*, Vol.3, No. 1, Feb. 1995, pp. 10-25.