International Journal of Distributed Sensor Networks, 3: 311–330, 2007
Copyright © Taylor & Francis Group, LLC
ISSN: 1550-1329 print/1550-1477 online
DOI: 10.1080/15501320701259925

Estimation of the Hyperexponential Density with Applications in Sensor Networks

LARRY N. SINGH

School of Medicine, University of Pennsylvania, Philadelphia, Pennsylvania

G. R. DATTATREYA

Dept. of Computer Science, University of Texas at Dallas, Richardson, Texas

This paper solves the problem of estimation of the parameters of a hyperexponential density and presents a practical application of the solution in sensor networks. Two novel algorithms for estimating the parameters of the density are formulated. In the first algorithm, an objective function is constructed as a function of the unknown component means and an estimate of the cumulative distribution function (cdf) of the hyperexponential density. The component means are obtained by minimizing this objective function, using quasi-Newtonian techniques. The mixing probabilities are then computed using these known means and linear least squares analysis. In the second algorithm, an objective function of the unknown component means, mixing probabilities, and an estimate of the cdf is constructed. All the 2M *parameters are computed by minimizing this objective function, using quasi-Newtonian techniques. The developed algorithms are also compared to the basic EM algorithm, and their relative advantages over the EM algorithm are discussed. The algorithms developed are computationally efficient and easily implemented, and hence, are suitable for low-power and sensor nodes with limited storage and computational capacity. In particular, we demonstrate how the structure of these algorithms may be exploited to be effectively utilized in practical situations, and are hence ideal for sensor networks.*

Keywords Hyperexponential Density; Bursty Traffic Engineering; Parameter Estimation; Nonlinear Optimization; Sensor networks; Rainfall Prediction

1. Introduction

The advent of sensor networks has spurred research to develop new methods for distributed data collection and processing. Estimation of parameters of physical quantities from data collected over a geographic area is one such task. Mixture density is a powerful family of probability density functions and is useful in representing some physical quantities for such an application. The vast majority of the literature on the subject of mixture densities pertains to Gaussian mixture models (GMM). By comparison, mixtures of exponentials have received far less attention. Notwithstanding, mixtures of exponentials have significant applicability in a number of fields, including computer networks, biology, engineering, and medicine. The class of general mixtures of exponential distributions has relevance in linear systems theory and differential equations (Chauveau [1]). Cao *et al* [2]

Address correspondence to G. R. Dattatreya, Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083-0688, Tel.: 972-883-2189, Fax: 972-883-2349. E-mail: datta@utdallas.edu discuss modeling of rainfall and contract valuation with hyperexponential densities. Specifically, in the paper by Cao *et al* [2], the amount of rainfall in an area is modeled using a mixture of two exponential densities. A further exposition of generalized hyperexponential distribution functions may also be found in Botta *et al* [3]. The focus of this paper is on a subset of exponential probability densities, which will be henceforth referred to as *hyperexponential densities*. We develop two algorithms for estimating the parameters of hyperexponential densities. The strength of these algorithms lies in the fact that the methods presented require little computational power and that the computations can be decomposed efficiently. These features enable an effective solution that is appropriate for sensor networks as outlined in section I-A.

The probability distribution function (pdf) of an M-component hyperexponential density is

$$f(x) = \sum_{i=1}^{M} \pi_i \lambda_i e^{-\lambda_i x}, \tag{1}$$

where the mixing proportions are given by $\pi = [\pi_1, \ldots, \pi_M]^T$, and the means of each component are $\left[\frac{1}{\lambda_1}, \ldots, \frac{1}{\lambda_M}\right]$. Given N independent, identically distributed (iid) samples of data, the problem dealt with here is to estimate the parameters $\lambda = [\lambda_1 \ldots \lambda_M]^T$ and π of the hyperexponential density. In the following subsection, sensor networks are introduced and we demonstrate the applications of hyperexponential densities in a sensor network environment.

1.1. Sensor Networks

We now introduce applications in a sensor network environment, which can be modeled accurately by hyperexponential densities. Sensor networks (Akyildiz *et al* [4] and Rentala *et al* [5]) consist of many sensor nodes arranged in either an ordered or random fashion. Figure 1 shows such a sensor network. Data is transmitted via a multi-hop approach from the sensor nodes to special nodes called *base stations*. Sensor nodes are ideal for sensing environmental changes, especially in hostile or remote locations or over large geographical areas. One important category of environmental observation is rainfall observation in forestry and agriculture. The relevance of the statistical properties of rainfall is expounded

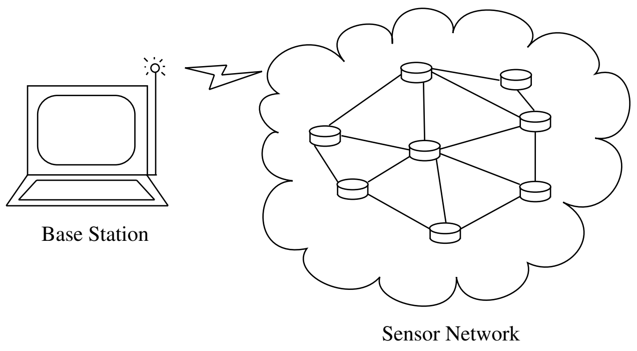

FIGURE 1 Sensor network.

in Hansen and Ines [6]. In particular, the quantity of rainfall in an area is decidedly important due to its effect on processes in nature, such as, solute leaching, soil erosion, and crop water stress response. Long-tailed distributions as discussed in the previous sections are useful for modeling the amount of rainfall in an area. Moreover, the amount of rainfall is commonly modeled as a two-component hyperexponential density with pdf

$$f(x) = \frac{\alpha}{\beta_1} e^{-\frac{x}{\beta_1}} + \frac{1-\alpha}{\beta_2} e^{-\frac{x}{\beta_2}}, \tag{2}$$

where α, β_1 and β_2 are estimated parameters ([2] and [6]).

Sensor nodes fitted with soil moisture sensors are ideal for measuring rainfall and collecting data that would allow for statistical modeling of rainfall. In large, remote, forested areas or large agricultural fields or both, sensor networks would be valuable for collecting rainfall data. Typically, many sensor nodes are scattered or positioned over a large area and these nodes function until their batteries are expended. Due to the limited battery supply in each sensor node, transmissions by sensor nodes should be kept to a minimum. Furthermore, each sensor node has a relatively meager amount of memory (usually about 4K) and processing ability. Therefore, it is also desirable to reduce the complexity of calculations performed by a sensor node.

In the following sections, we develop algorithms for estimating the parameters of a hyperexponential density, which can also be used to obtain the parameters of equation 2 in our rainfall example. The algorithms are also designed so as to work well in a sensor network environment with energy and computational power constraints. That is to say, the data communication and computational power requirements of the sensor nodes are very minimal. In the following subsection, additional motivation for studying hyperexponential densities is presented, as we look at applications in modeling long-tailed distributions.

1.2. Long-tailed Distributions

In recent years, the volume of traffic on large-scaled networks and across the Internet has increased tremendously, necessitating serviceable statistical models for traffic flow and analysis. Statistical models are required for the design of underlying IP-based transport layer protocols, efficient data gathering and evaluation, and statistical analysis of corresponding random processes and variables (Markovitch and Krieger [7]). The work of Leland *et al* [8] demonstrates that both LANs and WANs exhibit long range dependencies, and hence, classical Poisson-based methods are insufficient for such large-scale networks. Typical Poisson methods predict early fluctuations in network traffic and hinge on the assumption that these anomalies will smooth out over a long period of time. In reality, these fluctuations occur over a wide range of time scales generating *high variability* and *self-similar* behavior (Schwefel [9]). The notion of self-similarity and fractal geometry was introduced by Mandelbrot [10] to describe naturally occurring processes using mathematical formulae. Self-similar processes are structurally similar over many different time scales. This phenomenon leads to long range dependencies in network traffic. Indeed, the concept of self-similarity and long range dependencies go hand-in-hand.

Ostensibly, an alternative to classical Poisson methods is sought. To this end, *long-tailed distributions* have shown much success. A distribution is a long-tailed distribution (also referred to as a heavy-tailed distribution) if its complementary cumulative distribution function (ccdf), F^c, decays slower than exponentially, i.e. if

$$\lim_{x\to\infty} e^{\alpha x} F^c(x) = \infty, \tag{3}$$

for all $\alpha > 0$ (Feldmann and Whitt [11]). The ccdf is defined as the complement of the cumulative distribution function (cdf), as follows

$$F^c(x) = 1 - F(x), \tag{4}$$

where $F(x)$ is the cdf. Conversely, a distribution is a *short-tailed distribution* if its ccdf decays exponentially, i.e. if

$$\lim_{x\to\infty} e^{\alpha x} F^c(x) = 0. \tag{5}$$

for some $\alpha > 0$. Of note is a special case of long-tailed distributions called the *power-tail* distribution. A distribution is said to be power-tail distribution if

$$F^c(x) \sim \alpha x^{-\beta} \text{ as } x \to \infty, \tag{6}$$

where α and β are positive constants and the operator ~ is defined such that $f(x) \sim g(x)$ implies that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = 1. \tag{7}$$

Two common examples of long-tailed distributions that are used widely in network performance analysis are the *Pareto* and *Weibull* distributions. In addition, the Pareto distribution is a power-tailed distribution; however, the Weibull distribution is not.

Recent studies have demonstrated that long-tailed distributions model many network characteristics aptly. For example, long-tailed distributions have been valuable in modeling World Wide Web (WWW) traffic, since the said traffic often originates from heterogeneous sources [7]. Long-tailed distributions have also been shown to be successful in modeling file transfer protocol (FTP) connections and intervals between connection requests [11]. As effective as the long-tailed distribution has been in capturing the statistical characteristics of large scale networks, there are some deficiencies of this model. Most notably, long-tailed distributions are generally very difficult to analyze. An example is the task of analyzing the performance figures of the basic M/G/1 queue which becomes quite involved if the service-time distribution is Pareto. Furthermore, unlike many short-tailed distributions, expressions for the Laplace transforms of long-tailed distributions are quite complex. Laplace transforms are generally useful for analyzing the distributions by numerical transform inversion. Standard non-parametric estimators such as the histogram, projections, and kernel estimators are not suitable for long-tailed distributions and often exhibit poor estimates or "spikes" in the tail region. It is well-known that kernel estimators suffer from spurious noise appearing in the tail of the estimator [12]. Markovitch and Krieger [7] have explored some non-parametric procedures for estimating and approximating long-tailed distributions. Two of these procedures include transformation functions that map a long-tailed density into a pdf with compact support, and *polygrams*. Polygrams are histograms with variable bin widths.

The aforementioned problems associated with long-tailed distributions do not appear in short-tailed distributions. Accordingly, a natural proposal for alleviating the problems of long-tailed distributions is to approximate the said long-tailed distributions with corresponding short-tailed distributions. This is justifiable by the fact that although long-tailed distributions do not have compact support, for practical purposes there is only a finite interval for which the distribution in question takes on meaningful values. Therefore, one approach for approximating long-tailed distributions is to effectively truncate the long-tailed distribution, confining it to a finite interval. Intuitively, this method encompasses the essential features of the long-tailed distribution. In spite of this, the truncation approach may not be a suitable technique since the resulting function has discontinuities for values in the real, positive domain. An alternative to approximating the long-tailed distributions by truncating is to use hyperexponential densities. Hyperexponential densities have exhibited much success in approximating long-tailed distributions and for constructing network performance models ([11] and [7]). Moreover, the analysis of hyperexponential densities is tractable and the Laplace transforms are simple expressions.

1.3. Organization

Two computationally efficient, tractable, and easily implemented algorithm for estimating the parameters of a hyperexponential density are developed and tested. The first algorithm — hereafter, referred to as *Algorithm 1* — is a two-step procedure. The first step involves estimating the component rates λ of the hyperexponential distribution. The approach here is to develop equations that express π in terms of λ. These expressions are then substituted into an objective function which is a function of λ and an estimate for the cdf of the hyperexponential distribution. Minimizing this objective function yields the required estimates of λ. The second step estimates the steady state probabilities π given that the component means are known by making use of linear least squares analysis. The second algorithm — *Algorithm 2* — estimates π and λ in one phase by minimizing an objective function that is a function of π, λ and an estimate for the cdf of the hyperexponential distribution. The relative merits of each algorithm are also compared and discussed.

Our algorithms are also compared to maximum-likelihood techniques and the expectation-maximization (EM) algorithm (Dempster *et al* [13]), in particular. Finally, we demonstrate how our algorithms are adapted to work well in a sensor network environment.

2. Algorithm 1

Most **ML** techniques for computing the parameters of hyperexponential densities intrinsically require estimation of $2M$ parameters. Likewise, any optimization procedure that uses or evaluates the hyperexponential density directly, also involves the estimation of $2M$ parameters. In this section, equations expressing π in terms of λ and functions of the samples of data are derived, thus reducing the number of unknown parameters to M. Making use of these equations, an objective function is formulated and the values of λ are calculated by minimizing this objective function. It is assumed that each mixing proportion is strictly positive, ensuring that each component plays a role in influencing the resulting pdf. Algorithm 1 essentially reduces the number of unknown parameters by incorporating additional information from the structure of the mixture pdf and from samples of data.

2.1. Expressions for π

Nonlinear transforms are employed in order to reduce the number of unknown parameters. Specifically, the Laplace transform of the exponential density is exploited. Define

$\alpha = [a_1 \ldots\ldots \alpha_M]^T$ where each α_i is a distinct, real, positive value. The Laplace transform of the pdf in equation (1) scaled by a factor of $\frac{1}{\alpha_i}$ is

$$E\left[\frac{1}{\alpha_i} e^{-\frac{x}{\alpha_i}}\right] = \frac{1}{\alpha_i}\sum_{j=1}^{M} \pi_j \int_0^{\infty} e^{-\frac{x}{\alpha_i}} \lambda_i e^{-\lambda_i x} dx \tag{8}$$

$$= \sum_{j=1}^{M} \pi_j \frac{1}{\frac{1}{\lambda_j} + \alpha_i} \tag{9}$$

Let **A** be the matrix with elements $\frac{1}{\frac{1}{x_j} + \alpha_i}$ at row i and column j. Also, let $\boldsymbol{a} = [a_1. \ldots,$ $a_M]^T$ be the vector of expectations where $\alpha_i = E\left[\frac{1}{a_i} e^{-\frac{x}{\alpha_i}}\right]$. Therefore, in matrix notation, equation (9) is

$$\mathbf{A}\pi = \boldsymbol{a} \quad \text{and} \tag{10}$$

$$\pi = \mathbf{A}^{-1}\boldsymbol{a}, \tag{11}$$

provided that **A** is nonsingular. Matrix **A** is a Cauchy matrix and hence, has certain nice properties (Boras [14]). For instance, the inverse of a Cauchy matrix can be represented as an explicit expression (Knuth [15]) and thus, $\mathbf{B} = \mathbf{A}^{-1}$ is an $M \times M$ matrix with elements

$$b_{ij} = \frac{\prod_{1\le k\le M}\left(\frac{1}{\lambda_j} + \alpha_k\right)\left(\frac{1}{\lambda_k} + \alpha_i\right)}{\left(\frac{1}{\lambda_j} + \alpha_i\right)\left(\prod_{\substack{1\le k\le M \\ k\ne j}}\left(\frac{1}{\lambda_j} - \frac{1}{\lambda_k}\right)\right)\left(\prod_{\substack{1\le k\le M \\ k\ne i}}(\alpha_i - \alpha_k)\right)}, \tag{12}$$

at row i and column j and where $\lambda_1 \ne \lambda_2 \ne \ldots \ne \lambda_M \ne 0$ and $\alpha_1 \ne \alpha_2 \ne \ldots \ne \alpha_M$. Clearly, each λ_i is distinct by the assumption that the mixing proportions for each component are all non-zero, and each α_i is distinct by assumption. Hence, **A** is clearly invertible. From equations (11) and (12) the steady state probabilities may be expressed as

$$\pi_i(\lambda) = \sum_{j=1}^{M} b_{ij} a_j. \tag{13}$$

This gives a means of computing π if the component means and expectations in equation (9) are provided.

2.2. Determination of λ

The expressions for the mixing probabilities developed in the previous section afford a means of reducing the number of unknown variables to just *M*. In this section, an algorithm to obtain this *M* component mean vector is formulated by fitting a candidate cdf to the given exact cdf. The approach iteratively attempts to improve the fit of a real or estimated cdf to that of a cdf constructed from successive approximations of the parameter set. The cdf of a hyperexponential density is

$$F(x) = 1 - \sum_{k=1}^{M} \pi_k e^{-\lambda_k x}. \tag{14}$$

Let $\tilde{\lambda} = [\tilde{\lambda}_1, \ldots, \tilde{\lambda}_M]^T$ and $\tilde{\pi}(\tilde{\lambda}) = [\tilde{\pi}_1(\tilde{\lambda}), \ldots, \tilde{\pi}_M(\tilde{\lambda})]^T$ be the current approximations for λ and π, respectively. Hence, an approximate or candidate cdf is given as

$$\tilde{F}(x, \tilde{\lambda}) = 1 - \sum_{k=1}^{M} \tilde{\pi}_k(\tilde{\lambda}) e^{-\tilde{\lambda}_k x}. \tag{15}$$

Notice that the only unknown in this candidate cdf is $\tilde{\lambda}$. The error of the fit of this candidate cdf at a single point *x* is defined to be

$$(F(x) - \tilde{F}(x, \tilde{\lambda}))^2. \tag{16}$$

This result can be extended over the entire domain of *x* — i.e. the set of all real numbers greater than or equal to zero ($\mathbf{R}^+$) — to give the total error by integrating equation (16) for all $x \in \mathbf{R}^+$. Unfortunately, this integral does not furnish a simple, tractable expression for the total error. Thus in practice, the integral would have to be numerically evaluated which is a somewhat expensive task. Realize that for practical purposes, the entire domain of $\mathbf{R}^+$ need not be considered. A viable approximation of the total error may be obtained by computing the error at a finite number of arbitrary points, $[x_1, \ldots, x_m]$ over the region of interest and summing up the error as follows

$$d(\tilde{\lambda}) = \sum_{k=1}^{m} (F(x_k) - \tilde{F}(x_k, \tilde{\lambda}))^2. \tag{17}$$

Observe that $d(\tilde{\lambda}) \geq 0$ for all $\tilde{\lambda}_i > 0$ and $1 \leq i \leq M$. Therefore, $d(\tilde{\lambda})$ is bounded from below and has a global minimizer (Borwein and Lewis [16]). Moreover, this minimum is known to be zero and for an ideal candidate cdf, $d(\tilde{\lambda}) = 0$. Obviously a better approximation of the total error is obtained if *m* is made large. The component means are obtained by minimizing (17) with respect to $\tilde{\lambda}$. However, the objective function in question is not a convex function of $\tilde{\lambda}$ in general. Therefore, the Newtonian methods for minimization cannot be directly applied. Nonetheless, the problem of finding $\tilde{\lambda}$ is posed as a constrained nonlinear optimization problem. The first set of constraints ensure that $\lambda_k > 0$ for all $1 \le k \le M$. The second set are as a result of ensuring that π are valid, non-zero probabilities, i.e.

$$\sum_{k=1}^{M} \pi_k(\tilde{\lambda}) = 1 \text{ and} \tag{18}$$

$$0 < \pi_k(\tilde{\lambda}) < 1 \text{ for all } 1 \le k \le M. \tag{19}$$

2.3. Estimation of λ from Statistical Data

In order to minimize the objective function developed, there are several quantities that need to be estimated from the samples of data. In this section, estimators for these quantities are devised. Given n samples of data $\boldsymbol{w} = \{w_1. \ldots, w_n\}$, and assumed distinct constants α_i, an estimator for a_i is

$$a_i = \frac{1}{n} \sum_{k=1}^{n} e^{-\alpha_i w_k}. \tag{20}$$

Define $\hat{\boldsymbol{a}}$ to be the estimate of $\boldsymbol{a}$ from the samples of data.

The equations derived previously implicitly assume that the cdf $F(x)$ is exact and available. Of course this is not the case, and an estimate for the cdf must be obtained from the samples of data. The cdf is a good choice for function-fitting and estimation in general for the hyperexponential density for a number of reasons. First, the cdf is a smooth, monotonically increasing function. Second, the cdf can be estimated easily and accurately from a finite number of samples of data. Third, given the chosen implementation for the cdf, a lookup for a value runs in $O(\log n)$ time which is quite fast. A piecewise continuous estimate of the cdf is obtained as follows. Sort the observations of data $\{w_1, \ldots, w_n\}$ to produce the values $\{y_1, \ldots, y_n\}$ such that $y_1 \le y_2 \le \ldots \le y_n$ and $\{y_1, \ldots, y_n\}$ is a permutation of $\{x_1, \ldots, x_n\}$. Hence, the estimate of the cdf is defined as,

$$\hat{F}(x) = \begin{cases} 0, & x < y_1 \\ \dfrac{i-1}{n-1}, & x = y_i \\ \dfrac{i-1}{n-1} + \dfrac{x - y_i}{(y_i + 1 - y_i)(n-1)}, & y_i < x < y_{i+1} \\ 1, & x > y_n. \end{cases} \tag{21}$$

Define $\tilde{\lambda}$ as the estimates of $\boldsymbol{\lambda}$ and substitute equation (21) into (17) giving the new objective function

$$\hat{d}(\hat{\lambda}) = \sum_{k=1}^{m} (\hat{F}(z_k) - \hat{F}(z_k, \hat{\lambda}))^2. \tag{22}$$

Minimizing this objective function produces the estimates $\tilde{\lambda}$.

2.4. Minimization Procedure

The current minimization problem involves finding the minimum of a nonlinear objective function subject to nonlinear constraints, and so the problem is a nonlinear programming task. *Sequential quadratic programming* methods are generally accepted as being the best nonlinear programming techniques (Palambros and Wilde [17]) and thus, are our chosen methods for minimizing (22). An abridged overview of sequential quadratic programming (SQP) methods is presented here. A further account on the details of SQP procedures and their derivations may be found in [17], Nocedal and Wright [18] and Fletcher [19]. Let $\{x_1, x_2, \ldots\}$ be a sequence of iterates of the minimization procedure which approach a solution x^* that is a minimum. SQP algorithms function by approximating each iterate x_k by a quadratic programming subproblem. The next iterate is computed by minimizing this subproblem. Quadratic programming involves minimizing objective functions that are quadratic multivariate functions with linear constraints. Inequality constraints posed as nonlinear functions need to be linearized and approximated. The main difficulty is to formulate the quadratic subproblems in such a manner that good steps are made for each iterate and the overall SQP algorithm has good convergence properties and is efficient.

For the process of finding stationary points of a multivariate objective function subject to one or more constraints, it is common to use *Lagrange multipliers* [20]. Denote the vector of Lagrange multipliers as $\boldsymbol{\xi}$. For convenience, the stationarity conditions are usually expressed in terms of the *Lagrangian function* which is defined in the most general case, as

$$\mathrm{L}(\mathbf{x}, \xi) = \mathbf{f}(\mathbf{x}) + \xi^{\mathrm{T}} \mathbf{g}(\mathbf{x}). \tag{23}$$

where $\boldsymbol{g}(\boldsymbol{x})$ represents the constraints. The Lagrangian function contains the first-order and second-order conditions for a point to be a local minimizer of the objective function subject to the given constraints. In our chosen implementation of an SQP algorithm, the Hessian of the Lagrangian function needs to be updated with each iteration. The chosen method for doing so is the BFGS update method. The next major step involves solving the quadratic problem for each iteration. To do so, an *active set* strategy [17] is employed. An active set strategy activates and deactivates constraints during each iteration as the algorithm progresses towards the minimum. The third and final major step is to form a new iterate

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \beta_k \mathbf{s}_k. \tag{24}$$

where $\mathbf{s}_k$ is the search direction vector. The step size β_k is chosen by minimizing an appropriate function called the *merit* function. The merit function measures how appropriate and feasible each $\mathbf{x}_k$ and ξ_k are, and the function should have a local minimum at the solution. In our chosen implementation for minimization, we have chosen to enforce constraints for the probabilities in the second phase of computing the probabilities by the linear least squares analysis. In the first stage, $\hat{\lambda}$ are determined by unconstrained minimization.

2.5. Estimation of π from Statistical Data

In the preceding section, a method was developed for determining estimates of $\boldsymbol{\lambda}$ given an estimate for the cdf. A naive approach to attaining values for π is to exploit equation (13).

However, due to the nature of the hyperexponential density and the equations involved, the values for the vector ***a*** cannot be estimated accurately enough from data for computing π. The values attained for π from equation (13) are very sensitive to the values derived for $\tilde{\lambda}$. This means that small inaccuracies in $\tilde{\lambda}$ translate into large errors in π. This is mainly due to the inability of $\hat{\boldsymbol{a}}$ to estimate ***a*** with sufficient accuracy. In contrast, the estimate for the cdf $\hat{F}(x)$ from data provides a very accurate representation of the exact cdf. This is yet another advantage of using the estimate of the cdf for function-fitting.

The cdf of the hyperexponential density is linear in π. In addition, the estimated cdf can be expressed in terms of the estimates for $\hat{\lambda}$ and the unknown parameters π as follows,

$$\sum_{k=1}^{M} \tilde{\pi}_k e^{-\hat{\lambda}_k x} = 1 - \hat{F}(x). \tag{25}$$

for all $0 < x < \infty$. Let $\mathbf{z} = \{z_1, \ldots, z_S\}$ be an arbitrary set of positive, real constants such that

$$\inf_{1\le i\le S} z_i = \inf_{1\le j\le M} w_j \text{ and} \tag{26}$$

$$\sup_{1\le i\le S} z_i = \sup_{1\le j\le M} w_j. \tag{27}$$

Define $\mathbf{F}(\mathbf{z}) = [F(z_1),\ldots,F(z_S)]^T$ and $\hat{\pi} = [\hat{\pi}_1,\ldots,\hat{\pi}_M]^T$. Let $\hat{C}$ be the $S \times M$ matrix with $\hat{C}_{ij} = e^{-\lambda_j z_i}$ being the element at the i^{th} row and j^{th} column. From these definitions, equation (25) can be written using matrix notation as

$$\mathrm{C}\hat{\pi} = 1 - \mathrm{F(z)} \tag{28}$$

leading to the following theorem.

Theorem 1. Equation (28) has a unique solution for the mixing proportions, given that the component means are known.

Proof. Equation (28) can be solved using linear least squares regression analysis, and is also a convex function (Boyd [21]). Therefore, this function has a global minimizer and has a unique solution for π.

From this theorem, $\tilde{\pi}$ is obtained by solving equation (28) using linear least squares regression analysis.

2.6. Summary of Algorithm 1

The following presents a summary of Algorithm 1, for obtaining estimates of the parameters for the hyperexponential density, given *n* samples of data.

1. Choose values for α such that each α_i is distinct and positive.
2. Obtain an initial estimate for λ.

3. Compute $\hat{\boldsymbol{a}}$ using α and the samples of data.
4. Minimize $\hat{d}(\hat{\lambda})$ to obtain a new estimate for $\hat{\lambda}$.
5. Using the new estimate of $\hat{\lambda}$, compute **C** and using linear least squares regression, obtain an estimate for π.

3. Algorithm 2

The second algorithm developed is similar to Algorithm 1 of the previous section. The essential difference in Algorithm 2 is that $2M$ parameters are estimated from the developed objective function. In the following subsection, the required objective function is developed.

3.1. Development of the Objective Function

As in the previous section, an approximate cdf is constructed as follows

$$\tilde{F}(x,\tilde{\lambda},\tilde{\pi}) = 1 - \sum_{k=1}^{M} \tilde{\pi}_k e^{-\tilde{\lambda}_k x}. \tag{29}$$

Notice that this cdf has three arguments as opposed to two, and that $\tilde{\pi}$ is no longer described as a function of $\tilde{\lambda}$. Following the procedure of the previous section, the error of fit of this candidate function is expressed as

$$(F(x) - \tilde{F}(x,\tilde{\lambda},\tilde{\pi}))^2. \tag{30}$$

Similarly, the approximate total error of the fit is denoted by

$$d(\tilde{\lambda},\tilde{\pi}) = \sum_{k=1}^{m} (F(x_k) - \tilde{F}(x_k,\tilde{\lambda},\tilde{\pi}))^2\, dx, \tag{31}$$

where $m \geq M$. The function $d(\hat{\lambda}, \tilde{\pi})$ has similar properties as $d(\tilde{\lambda})$, in that $d(\tilde{\lambda}, \tilde{\pi}) \geq 0$ for all $\hat{\lambda}_i > 0$ and $1 \leq i \leq M$. Hence, $d(\tilde{\lambda}, \tilde{\pi})$ is also bounded from below and has a global minimizer.

The required parameters are obtained by minimizing the objective function in equation (31). As in the previous section, this is a constrained nonlinear optimization problem. The constraints are the same, i.e. that the component means are positive and that the mixing proportions are valid probabilities. The latter constraint can, however, be relaxed by introducing the *softmax* function (Bishop [20]). Let $\gamma = \{\gamma_1, \ldots, \gamma_M\}$ be a vector of real constants, and hence,

$$\pi_i(\gamma) = \frac{e^{\gamma_i}}{\sum_{j=1}^{M} e^{\gamma_j}} \tag{32}$$

for all $1 \leq i \leq M$. Fix one of γ, γ_M (say) to be 0, which ensures that the transformation is one-to-one. Using this definition for π_i ensures that the constraints of the mixing proportions are met. Thus, the only applicable constraint is that $\lambda_i > 0$, for all $1 \le i \le M$. However, since the hyperexponential cdf is strictly monotonic, this constraint can effectively be disregarded, reducing the problem to a nonlinear unconstrained optimization problem. Let $\hat{\gamma}$ be the estimates of γ and introduce the new objective function defined as

$$d_u(\hat{\lambda},\hat{\gamma}) = \sum_{k=1}^{m} (F(x_k) - F(x_k, \tilde{\lambda}, \tilde{\pi}(\hat{\gamma})))^2 \, dx, \tag{33}$$

It is worth noticing that (33) is not a convex function of $\hat{\gamma}$ and $\hat{\lambda}$, so the Newtonian method for optimization cannot be applied. Instead, the minimization should be performed using either quasi-Newtonian methods with the BFGS update method and inexact line searches, or using the Levenberg-Marquardt method for nonlinear regression analysis.

For Algorithm 2, the only quantity that needs to be estimated is the cdf. The approach for estimating the cdf is the same in both algorithms 1 and 2. In the next subsection, a summary of the algorithm is presented.

3.2. Summary of Algorithm 2

The following presents a summary of of Algorithm 2. Estimates of the parameters of the hyperexponential density are obtained, given n samples of data.

1. Obtain an initial estimate for $\hat{\lambda}$ and $\hat{\pi}$.
2. Minimize $d_u(\hat{\lambda}, \hat{\pi})$ to obtain new estimates for $\hat{\lambda}$ and $\hat{\pi}$.

4. Discussion

In this section, we discuss the expediency of the algorithms developed to sensor networks. An account of the simulation experiments executed to demonstrate the algorithms is also presented, along with a discussion of the results of these experiments.

4.1. Applicability of Algorithms to Sensor Networks

The primary strength of Algorithms 1 and 2 is that they are well-suited to sensor networks. In order to illustrate how these algorithms are applicable to sensor networks, we return to the rainfall example in section I-A. Each sensor node collects rainfall data from its immediate environment. Hansen and lnes [6] discuss the difficulties in measuring rainfall. Taking soil moisture measurements, for example, is one suggested approach for determining the amount of rainfall. However, the exact details of measuring rainfall with sensor networks are beyond the scope of this paper. The individual sensor nodes do not possess the computing power to perform the complete parameter estimation procedure. In addition, transmission of all the data values collected at all sensor nodes to the base station is impractical, as the large number of transmissions needed would quickly drain the batteries of the sensor nodes. Therefore, an algorithm such as the EM algorithm which requires all the data values would not be applicable in this scenario. Nowak [22] introduces a distribued EM algorithm approach for performing density estimation in sensor networks. In [22], it is assumed that the local processing is much less expensive than communication. We do not compare explicitly our algorithms with those in [22], since Nowak performs the density estimation of Gaussian mixtures. However, the following observations are made. In our algorithms, there is very little processing to be done at individual sensor nodes, and the number of communications for each sensor node is on par with that presented in [22]. Hence, it is expected that our algorithms will use less battery power than that of [22].

Recall that in both Algorithms 1 and 2, an estimate of the cdf of the hyperexponential density, $\hat{F}(x)$ is required. However, the domain of $\hat{F}(x)$ is a set of fixed, pre-determined values of x. Computing an estimate of $\hat{F}(x)$ as in equation (21) would require each node to store all data points collected, which is infeasible. Instead, the estimate of the cdf is calculated as follows. Given n observations of data $\{w_1 \dots .w_n\}$, the *empirical cdf* is defined as

$$F_n(x) = \frac{1}{n} S(x) \tag{34}$$

where

$$S(x) = \sum_{i=1}^{n} u(x - w_i). \tag{35}$$

and u is the unit step function defined as

$$u(y) = \begin{cases} 0 & \text{if } y < 0 \\ 1 & \text{if } y \geq 0 \end{cases} \tag{36}$$

Only the values of $F_n(x)$ for a fixed domain of x are required, so let the required values of x be $\{x_1, \dots, x_m\}$. Hence, the necessary values that need to be recorded and updated by each sensor node are n and $S(x_j)$ for each $1 \leq j \leq m$. For a specific sensor node k, denote S_k to be the value of S for node k, and similarly n_k denotes the value of n for node k. Each sensor node k collects data and increments its own value of n_k while updating S_k. At pre-defined epochs, for instance when n_k goes beyond a certain threshold value, the recorded values of S_k are transmitted to the base station. With this approach, each sensor node is required to record at most m+1 values. Updating and recording these values require very little computational power.

Upon transmitting the values of $S_k(x_j)$ to the base station, the sensor node resets the values of $S_k(x_j)$ for all j to zero and resets n_k to zero. Each sensor node transmits m values of $S_k(x_j)$, but some values of $S_k(x_j)$ may change more often and by greater amounts than others. For instance, after a period of collecting rainfall observations we might find that $S_k(x_1)$ is large while $S_k(x_2)$ and $S_k(x_3)$ are still zero. Sending the values of just $S_k(x_1)$ would reduce the size of the transmission for node k significantly. Therefore, we propose a format for the transmission packet as in Fig. 2. The bit vector at the head of the packet identifies which of the values of x are being sent in the packet. For instance, if bits number 1, 2, and 5 are set in the bit vector in the transmission from sensor node k, and all other bits are

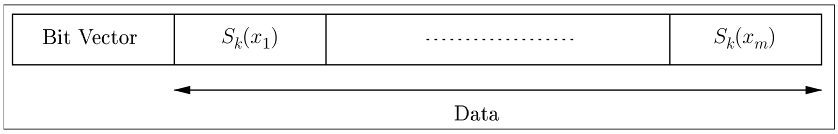

FIGURE 2 Data packet format for transmission from sensor nodes.

unset, then there are 3 data values to follow in the packet. The data values are in consecutive order and are correspondingly $S_k(x_1)$, $S_k(x_2)$ and $S_k(x_5)$. The proposed approach ensures that only values that are non-zero will be transmitted, thus minimizing the size of the transmissions.

The values of $S_k(x_j)$ for each sensor node k and all $1 \le j \le m$, are transmitted to the base station along with the corresponding values of n_k. Let $\hat{F}^{new}(x_j)$ and $\hat{F}^{old}(x_j)$ be the new and current estimates of the cdf, respectively. Similarly, let n^{new} and n^{old} be the new and current values of n. Assuming that the base station receives values from r sensor nodes the estimate of the cdf is updated as follows,

$$\hat{F}^{new}(x_j) = \frac{n^{old}}{n^{new}}\hat{F}^{old} + \frac{1}{n^{new}}\sum_{k=1}^{r} S_k(x_j), \tag{37}$$

and where

$$n^{new} = n^{old} + \sum_{k=1}^{r} n_k. \tag{38}$$

If a value of $S_k(x_j)$ is not received from a sensor node, assume that it has a value of zero in equation (37).

Equation (37) is used instead of equation (21) and thus, provides the necessary estimate of the cdf in the objective functions of Algorithms 1 and 2. Once an estimate of the cdf is obtained, the base station then performs the appropriate minimization procedure to estimate the parameters of the hyperexponential density. The problem presented here of estimating the amount of rainfall in an area is ideal for sensor networks, and the approach to solving this problem is well-suited to sensor networks. Each sensor node requires only a small amount of computational power and memory, while keeping the number of network transmissions to a minimum.

4.2. Error Analysis of Empirical cdf

The empirical cdf is commonly used as an estimator for the cdf in many applications including bootstrapping techniques and various EDF statistics such as the Cramer-von Mises and Anderson-Darling tests [23]. The virtues of the empirical cdf as an estimator stem from the fact that the empirical cdf converges uniformly to the actual cdf by the Gilvenko-Cantelli theorem [24]. Further properties of the empirical cdf are discussed extensively in [23] and [24]. The accuracy of the empirical cdf hinges on obtaining data points that range between the tails of the cdf. Extreme values which fall in the tails of the cdf will not produce good results as is the case with other estimators.

The sensor network implementation of our estimator requires the data to be discretized into several bins. The number of bins can be increased for increased accuracy. However, the final bin represents all data falling in the range $[x_{\max}, \infty)$. We can derive an upper bound for $x_{\max}$ to ensure that the probability of a random sample falling in $[x_{\max}, \infty)$ is below a threshold, say 1% or 0.01. Let $\beta_{\max}$ be the largest mean of all the exponential components of the mixture density. The highest probability of samples having a very large value is obtained when all other exponential components of the mixture occur with zero mixing probabilities. Therefore, we have

$$P[X > x_{\max}] < \exp\left(-\frac{x_{\max}}{\beta_{\max}}\right). \tag{39}$$

If we prefer this probability to be less than 0.01, we have

$$P[X > x_{\max}] < \exp\left(-\frac{x_{\max}}{\beta_{\max}}\right) < 0.01 \tag{40}$$

or, equivalently, $x_{\max} \geq 4.7\beta_{\max}$. Therefore, a good guideline is that the largest threshold in the specification of intervals for the sensor network implementation, should be about 5 times the largest anticipated mean value of the exponential probability density components of the mixture.

The Gilvenko-Cantelli theorem states that if the data samples are truly iid and the size of the data sample is sufficiently large, a good estimate of the cdf will be obtained. Since the empirical cdf is not a point estimator, deriving the standard error of the estimate is not trivial. The question of how large a sample is required for a good estimate can be answered by the Dvoretzky, Kiefer, Wolfowitz inequality [23], which is stated as follows. If $X_1, \ldots, X_n$ are iid real-valued random variables with cdf F, and F_n is the empirical cdf, then for any $d > 0$ and any positive integer n.

$$P\left[d_K\left(F_n, F\right) > d\right] \leq C\exp(-2nd^2), \tag{41}$$

where d_K is the well-known Kolmogorov-Smirnov statistic, and C is a universal constant. It can be shown that $C = 2$ is a good choice [23]. For illustrative purposes, if we restricted the probability of having a discrepancy of no more than 0.01 at a single point between F_n and F to be less than or equal to 0.01, then we would require approximately 200 data samples or more.

4.3. Summary of Simulation Experiments and Results

The algorithms discussed in the previous section were implemented and tested through simulation using different values for M, λ, and π. A subset of the simulation trials is discussed here. Both algorithms were tested on hyperexponential density using synthetically generated iid mixture samples. In addition, the results of both algorithms were compared to the basic implementation of the EM algorithm. Evaluation of the objective function $d(\lambda)$ was performed by evaluating the approximate total error for equally spaced points over a meaningful domain. In Algorithm 1, the first phase of estimating λ is accomplished through the use of quasi-Newtonian methods with the BFGS method for the Hessian update and a safeguarded mixed quadratic and cubic polynomial interpolation and extrapolation method for line scarches. The implementation chosen is the *fminunc* MATLAB function [25]. For the second phase of Algorithm 1, constrained linear regression was performed using SQP methods and specifically the *lsqlin* MATLAB function. Algorithm 2 was implemented using quasi-Newtonian methods with the BFGS update formula, and a safeguarded mixed quadratic and cubic polynomial interpolation and extrapolation method for line searches. Once again, the *fminunc* MATLAB function [25] was employed.

The following is an outline of the simulation strategy utilized. In the first class of experiments, 50 experiments were performed using simulation data generated from a four component hyperexponential density having the following characteristics: $\boldsymbol{\lambda}$ = [1.0, 2.0, 3.0, 4.0] and **P** = [0.28, 0.14, 0.38, 0.2]. Also, for each of the 50 experiments, the same data samples were applied, but each experiment was executed using a random initial guess for $\boldsymbol{\lambda}$ from the range (0.5, 4.5). A total of 100 data samples was chosen for each experiment and each of the three algorithms was executed on these data samples for each experiment. The initial values for **P** were chosen to be $\frac{1}{M}$, where applicable. For the second class of experiments, 50 experiments were performed repeating all the steps and characteristics of the first class of experiments, except that for each of the 50 experiments different synthetic data was generated. Once again, the values for $\boldsymbol{\lambda}$ were chosen randomly from the range (0, 10). Finally, for the fourth and the final class of experiments, the same steps were followed as in the third class, however, this time for a six component hyperexponential density. The values for each of the six rates were chosen randomly in the range (0, 10). A total of 200 experiments were performed for the simulation exercise.

Sample results of the three algorithms are given in Figs. 3, 4, and 5. Each figure compares the generated pdf of the corresponding algorithm to the actual pdf. Each of the algorithms terminated in 1–10 iterations. From the simulation results, all of the three algorithms appear to produce similar results. In most cases, Algorithm 1 and the EM algorithm give similar results, and Algorithm 2 gives slightly better results in some cases. There is no simulation evidence to suggest that there are operating regions in which one algorithm is superior to the other, given the current approaches taken for optimization. Extensive simulation evidence indicates that Algorithm 2 gives the best results. in terms of accuracy of results, amongst the three algorithms compared.

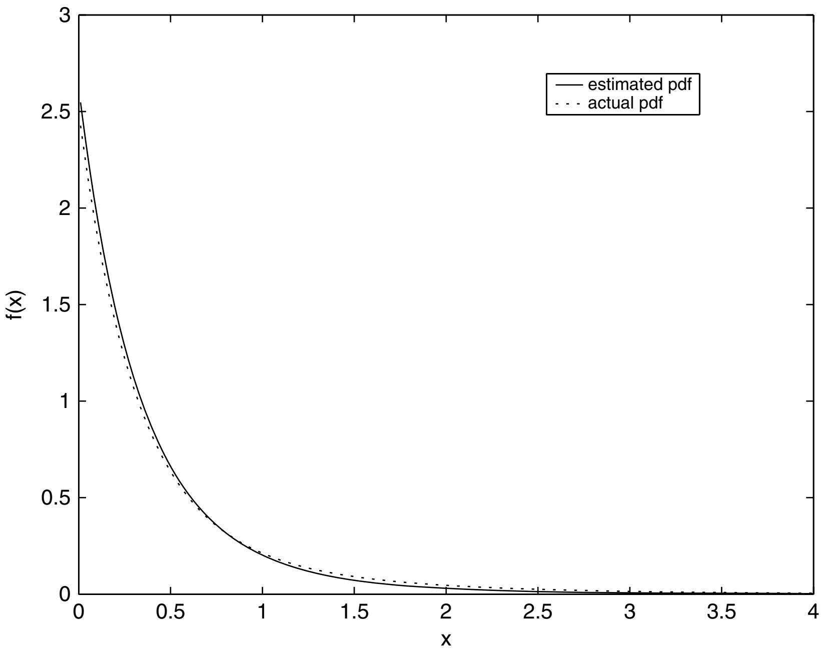

FIGURE 3 Sample plot of pdf generated from Algorithm 1 using 100 samples of data.

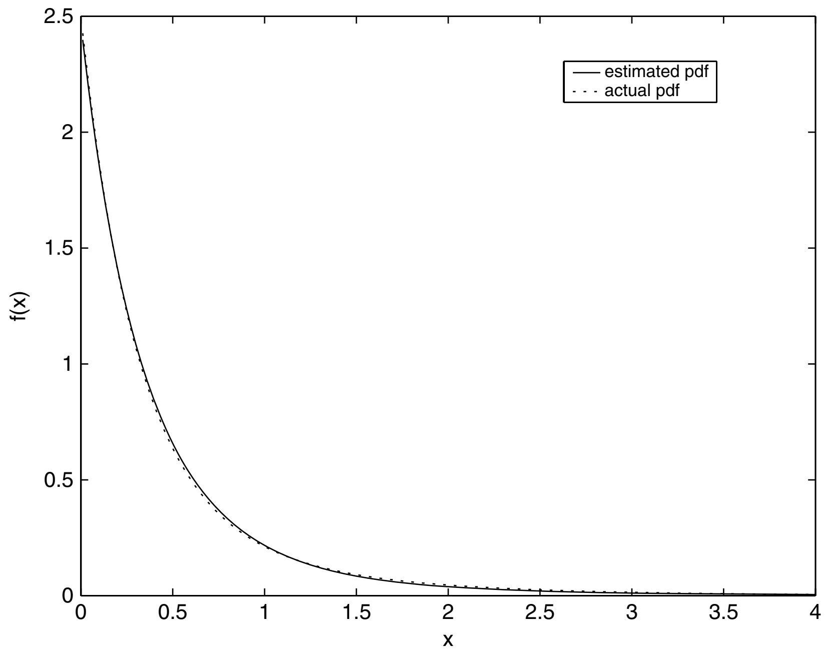

FIGURE 4 Sample plot of pdf generated from Algorithm 2 using 100 samples of data.

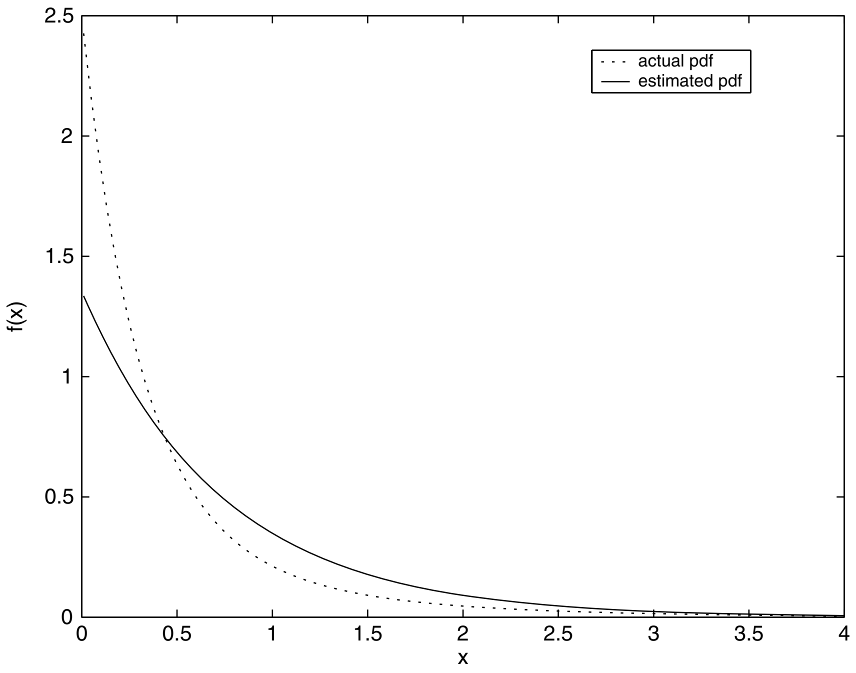

FIGURE 5 Sample plot of pdf generated from the EM algorithm using 100 samples of data.

In terms of computation speed. Algorithms 1 and 2 offer significant advantages over the EM algorithm. The EM algorithm involves computations that include all N data samples per iteration. As a consequence, as the number of data samples increases, the computation time increases dramatically. Compare this to Algorithms 1 and 2, in which the computation time of each iteration increases as a function of the number of unknown parameters only. For Algorithms 1 and 2, the values obtained from the estimated cdf function are at predefined points, and need only be looked up once, and not for every iteration. This lookup is very cheap as well, since a binary search of running time $O(\log N)$ is all that is required. For these reasons, the computation times for both Algorithms 1 and 2 are significantly improved versus the EM algorithm, particularly when N is large and the algorithms converage slowly.

The EM algorithm is generally regarded as the standard technique for estimating the parameters of mixtures of probability densities. Ruhe [26], Hasselblad [27], and Jewell [28] present solutions for finding the parameters of positive sums of exponentials, by maximum-likelihood techniques. Gruet *et al* [29] present a technique for estimation of mixtures of exponentials based on MCMC methods. It is well-known that the EM algorithm has a slow rate of convergence and may converge to values on the boundary of the parameter space. Moreover, the EM algorithm is not well-suited to an application in sensor networks, as the algorithm would require a high amount of network traffic to transfer all the data.

5. Conclusion

The major contributions presented here are algorithms for computing the parameters of a hyperexponential density. Our algorithms are easily implemented yet computationally very efficient. There are numerous potential applications of this algorithm particularly in the areas of network traffic modeling and queuing theory; however, the algorithms presented here are particularly applicable in sensor network situations. In particular, due to the formulation of Algorithms 1 and 2, accurate estimates of the hyperexponential pdf are obtained while keeping computation and network transmissions at the sensor nodes to a minimum. The two developed algorithms as well as the EM algorithm all give very good estimates of the hyperexponential pdf. However, extensive simulation evidence demonstrates that Algorithm 2 gives the best numerical results of the three algorithms compared, in terms of the quality of the estimate of the hyperexponential pdf produced, which is what our goal is. In addition, evidence is presented to suggest that the algorithms developed are superior to the EM algorithm in terms of computation speed in cases where a large number of data samples are present or if the convergence speed of the algorithm declines. Both algorithms developed are efficient and easily implemented using standard tools of numerical optimization.

About the Authors

Larry N. Singh graduated from the University of Toronto with an Hons. B.Sc. degree in Computer Science in 1997. From 1996 to 1997, he held positions as a Software Engineer and Consultant to companies such as Europay International and American Expression International. He completed his M.S. degree in Computer Science at the University of Texas at Dallas in 1999, and his Ph.D. in Computer Science in 2004. Currently, Dr. Singh is a Postdoctoral Researcher in the Penn Center for Bioinformatics at the University of Pennsylvania, Philadelphia.

G. R. Dattatreya received the B.Tech. degree in Electrical Engineering from the Indian Institute of Technology, Madras, M.E. in Electrical Communication Engineering, and Ph.D. from the Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India. He is currently an Associate Professor in the Department of Computer Science, University of Texas at Dallas. He has also held the following positions. (1) Consultant at Rockwell Collins, Inc. (2) Visiting Faculty in the Department of Computer Science, University of Maryland, College Park, (3) Visiting Faculty in the Center for Artificial Intelligence, ITESM, Monterrey, Mexico, (4) Consultant in the Purdue University's Malaysia Polytechnic Development Project, and (5) Senior Scientist with the Scientific Analysis Group, Defense Research and Deve1opment Organizations, Delhi, India. His research interests are (1) Performance modeling and optimization in telecommunication networks including ad hoc networks and sensor networks. (2) Signal processing for cognitive and software defined radio, and (3) Performance modeling and analysis of embedded systems and IT middleware.

References

1. D. Chauveau, C. F. Martin, A. C. M. van Rooij, and F. H. Ruymgaart, "Discrete signed mixtures of exponentials." *Communications in Statistics: Stochastic Models*. vol. 12, no. 2, pp. 245–263, 1996.
2. M. Cao, A. Li, and J. Wei, "Precipitation modeling and contract valuation: A frontier in weather derivatives." *The Journal of Alternative Investments*. vol. 7, no. 2, pp. 93–99, 2004.
3. R. F. Botta, C. M. Harris, and W. G. Marchal, "Characterizations of generalized hyperexponential distribution functions." *Communications in Statistics: Stochastic Models*, vol. 3, no. 1, pp. 115–148, 1987.
4. I. F. Akyildiz., W. Su, Y. Sankarasubramanian, and E. Cayirei, "A survey on sensor networks," *Computer Networks*. vol. 38, no. 4, pp. 393–422, 2002.
5. P. Rentala, R. Musunuri, S. Gandham, and U. Saxena. "Survey on sensor networks," Tech. Rep. UTDCS-33-02. The University of Texas at Dallas, TX, U.S.A., 2002.
6. J. W. Hansen and A. V. M. Ines. "Stochastic disaggregation of monthly rainfall data for crop simulation studies," *Agricultural and Forest Methodology*, vol. 131, pp. 233–246, 2005.
7. N. M. Markovitech and U. R. Krieger. "Nonparametric estimation of long-tailed density functions and its applications to the analysis of World Wide Web Traffic," *Performance Evaluation*, vol. 42, pp. 205–222, 2002.
8. W. E. Leland, M. S. Taqqu, W. Willinger, and V. Wilson. "On the self-similar nature of ethernet traffic (Extended Version." *IEEE/ACM Trans, on Networking*, vol. 2. no. 1, pp. 1–15, 1994.
9. H. Schwefel. "Modeling of packet arrivals using markov-modulated poisson processes with power-tail bursts." Master's thesis. Technische Universität Mnchen München, Garching, Germany, August 1997.
10. B. B. Mandelbrot. *The Fractal Geometry of Nature*. NY.: W. H. Freeman, 1982.
11. A. Feldmann and W. Whitt, "Fitting mixtures of exponentials to long-tail distributions to analyze network performance models," *performance Evaluation*, vol. 31, pp. 245–279, 1998.
12. B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. FL.: Chapman & Hall, 1986.
13. A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum-likelihood from incomplete data via the EM algorithm," *J. Royal Statist. Soc., Ser. B.* vol. 39, pp. 1–38, 1977.
14. T. Boras. *Studies in Displacement Structure Theory*. Ph.D. dissertation, Stanford University, CA, June 1996.
15. D. E. Knuth, *Fundamental Algorithms - The Art of Computer Programming: Vol. 1*. MA.: Addison-Wesley, 1973.
16. J. M. Borwein and A. S. Lewis, *Convex Analysis and Nonlinear Optimization: Theory and Examples*, NY: Springer-Verlag, 2000.

17. P. Y. Papalambros and D. J. Wilde, *Principles of Optimal Design*. NY: Cambridge University Press, 1988.
18. J. Nocedal and S. J. Wright, *Numerical Optimization*. NY: Springer-Verlag, 1999.
19. R. Fletcher, *Practical Methods of Optimization*. NY: John Wiley & Sons, 1987.
20. C. M. Bishop. *Neural Networks for Pattern Recognition*. NY: Oxford University Press, 1998.
21. S. Boyd and L. Vandenberghe, *Convex Optimization*. NY: Cambridge University Press, 2004.
22. R. D. Nowak, "Distributed EM algorithms for density estimation and clustering in sensor network," *IEEE Trans. Sig. Proc.*, vol. 51, no. 8, pp. 2245–2253, 2003.
23. E. L. Lehmann and J. P. Romano, *Testing Statistical llypotheses: 3rd edition*. NY: Springer-Verlag, 2005.
24. A. W. van der Vaart, *Asymptotic Statistics*. NY: Cambridge University Press 1998.
25. "MATLAB Software." http://mathworks.com.
26. A. Ruhe, "Fitting empirical data by positive sums of exponentials," *SIAM Journal Sci. Stat. Comput.*, vol. 1, no. 4, pp. 481–497, 1980.
27. V. Hasselblad. "Estimation of finite mixtures of distributions from the exponential family," *Journal of the American Statistical Association*, vol. 64, pp. 1459–1471, 1969.
28. N. P. Jewell, "Mixtures of Exponential Distributions," *The Annals of Statisties*, vol. 10, no. 2, pp. 479–484, 1982.
29. M. A. Gruet, A. Philippe, and C. P. Robert, "Estimation of exponential mixtures," in *Discretization and MCMC Convergence Assessment* (C. P. Robert ed.), ch. 8, pp. 161–173, NY: Springer-Verlag, 1998.