A Robust Anomaly Detection Technique Using Combined Statistical Methods

Joseph Ndong, Kavé Salamatian

A Robust Anomaly Detection Technique using Combined Statistical Methods

Joseph Ndong
Université Pierre et Marie Curie
LIP6-CNRS, Paris, France
joseph.ndong@lip6.fr

Kavé Salamatian
Université de Savoie Chambery Annecy
LISTIC PolyTech
kave.salamatian@univ-savoie.fr

Abstract

Parametric anomaly detection is generally a three steps process where, in the first step a model of normal behavior is calibrated and thereafter, the obtained model is used in order to reduce the entropy of the observation. The second step generates an innovation process that is used in the third step to make a decision on the existence or not of an anomaly in the observed data. Under favorable conditions the innovation process is expected to be a Gaussian white noise. However, in practice, this is hardly the case as frequently the observed signals are not gaussian themselves. Moreover long range dependencies, as well as heavy tail in the observation can lead to important deviation from the normality and the independence in the innovation processes. This, results in the frequent observation that the decisions made assuming that the innovation process is a white and Gaussian results in a large false positive rate. In this paper we deal with the above issue. Our approach consists of not assuming anymore that the innovation process is Gaussian and white. In place we are assuming that the real distribution of the process is a mixture of Gaussian and that there are some time dependency in the innovation that we will capture by using a Hidden Markov Model. We therefore derive a new decision process and we show that this approach results into an important decrease of false alarm rates. We validate this approach over realistic traces.

Keywords: Anomaly Detection, System Monitors, Kalman filter, GMM, HMM

1 Introduction

In the Internet, anomalous traffic behavior such as attacks, configuration changes, flash crowds, large transfer file and outages occur frequently. Large enterprise networks often have a security operations center where operators continuously monitor the network traffic hoping to detect, identify and treat anomalous events. The *detection process* of these events can then be used to trigger alarms to the network management system, which, in turn, trigger recovery mechanisms. Despite the recent growth in monitoring technology and in intrusion detection systems, correctly detecting anomalies in a timely fashion remains a challenging task. One of the reasons for this is that, many today's security solutions yield equipments that collect and analyze traffic from one link at a time. Similarly many research efforts consider anomaly detection on a per link basis [13],[11],[12].

To detect traffic anomalies, one typically seeks to characterize or build a model of *normal* behavior. In this paper we use signal processing techniques to track anomalies in situations where *normal* network conditions occur. The approaches used to address the anomaly detection problem are dependent on the nature of the data that is available for analysis. Network data can be obtained at multiple levels of granularity such as end-user-based (TCP/UDP data) or network-based. Traffic counts obtained from both types of data can be used to generate a time series to which statistical signal processing techniques can be applied [15], [16]. In this work, we focus our attention on network-based (at the IP *flow* level) anomaly detection. After filtering out normal looking traffic, anomaly detection methods analyze the residual traffic pattern for deviations. Taking account only one link per unit time is limiting, since any flow will traverse multiple links along its path. And it is intuitive that, a flow carrying an anomalous event will appear in these links, thus increasing the evidence to detect it.

In this paper we will show that, it is more convenient to track anomalies by running the detection mechanism in all links at the same time, than in the case of per-link detection. To do this, we focus on using data from all links in an enterprise or ISP network simultaneously. Since any anomaly has to traverse several links on route to its destination, an anomaly has the potential to be visible in any of the links it traverses. Since we cannot know in advance where anomalies will originate, nor the path they will take, it is advantageous to consider the behavior of all links in the same time, when developing both a model of "normal" traffic and a method for analyzing ”residuals”.

Our study is built on traffic matrix scheme [2],[4]. Each traffic matrix entries describes the average volume of traffic, in a given time interval, that originates at a given source node (routers) and is headed towards a particular destination node. In this paper we propose to use predictions of traffic matrix behavior for the purpose of anomaly detection. Since the traffic matrix is a representation of traffic volume, the types of anomalies we might be able to detect via analysis of the traffic matrix are **volume anomalies**,[8]. In [4], we used traffic matrix at the granularity of OD flows to detect, identify and track anomalies, by comparing four schemes (a basic analysis using variance, the CUSUM and Generalized Likelihood Ration test combined, Wavelets and a fourth method based multi scale variance shift). We had shown that the basic analysis using variance performs the best above the four methods, in the case of real network data (Abilene). In this work, we show that our method perform better that that analysis using variance, with the same network data. Our study differs, to the previously cited work, in numerous ways: i) instead of using OD flows, we used granularity of link counts ii)we do not assume that the Kalman innovation process does strictly remain a zero mean gaussian process,iii) instead, we assume that the real distribution of the process is a mixture of gaussians, making sense the use of gaussian mixture modeller to show that residuals can be split in different families (where anomalies might or might not happen), iv) we use also hidden markov model in order to capture time dependencies over the different mixture components, v) we use the Viterbi algorithm to select the potential best sates path where anomalies might come from v) we use finally variance shift analysis to track the anomalies.

To start our study, first a linear dynamical system is needed to model the link counts. Thus, a model is built, where the Kalman filter algorithm is derived to estimate the state of the system, for denoising the observed data. Thereafter, we use a gaussian mixture modeller above the innovation process (obtained from the Kalman filter phase), for clustering. This makes our monitoring technique robust toward gaussianity, since we do not assume that the residual remains typically a zero mean gaussian process. These two steps constitute our learning phase. In the second step, we perform an operational phase where, first, we build a discrete sequence of finite alphabet of mixture memberships IDs (based on the residual clustered in the GMM-learning phase), using a maximum a posteriori (MAP) criteria. This discrete sequence is then used to check the sequence of hidden states (each state containing a part of our alphabet) of the HMM, using again a MAP criteria. We perform anomaly detection on this results, combining finally the use of the Viterbi algorithm (in order to find the only best sequence of our hidden states) and a variance shift test.

The organization of this paper is as follows. Section 2 describes the system monitoring design we used. Section 3 deals with the methodology we adopt in our anomaly detection scheme. In section 4, we detail our calibration method and we validate our approach by showing efficient results. Section 5 concludes our work and fixes some ideas for future works.

2 Description of the Monitoring System

In our paper referred in [2], a simple basic system was designed to monitor OD flows traffic coming from network backbone. This system was produced with three functional blocks: data collection (by a NOC-Network Operations Center), a data analysis block and a decision process phase. The data analysis center is simply an operational phase built on Kalman filtering. We extend this anomaly detection system, to take account to three new components shown as part of our learning phase. In addition, our decision process is based on a combination of two underlying materials: the Viterbi algorithm and a variance test based data-point analysis. In Figure 2, the data analysis block contains a gaussian mixture modeller, a hidden markov modeller and a maximum a posteriori criteria used twice. Recall that we have made no assumptions that the residual process (on which we perform our anomaly detection scheme) is a zero mean gaussian process; instead the kalman innovation is supposed to be driven by a family of normal distributions. One can think that, this idea can be a good decision, if we know that the real distribution of the original data themselves is not purely gaussian. Based on this belief, we have introduced the use of a GMM, which is a good paradigm for modelling processes in situations where data might come from different families of exponential distributions. So, the calibration of a gaussian mixture model will help us for data clustering, then making possible to classify each residual data point into a gaussian component. After building a finite set of classes of residuals, all elements belonging to the same class might appear with the same identifier. The motivation to do so, is based on the fact that, one should easily identify a cluster (with label 1,2,3,...) and at the other hand, as we will see later, the use of a *hidden markov model* will be adequate to study the temporal dependencies between all the clusters. To perform this task, the *gaussian mixture model* is followed by a phase where we run, using a *Maximum A Posteriori (MAP)*criteria, an operation making us possible to form a discrete finite alphabet (sequence of symbols), using all the GMM components. At this step, all the continuous observation sequences are transformed into a discrete observation sequence, which can be easily re-used to perform a monitoring method for the purpose of anomaly detection.

The GMM phase for clustering and the MAP criteria operation for discrimination, are followed by the use of an hid-

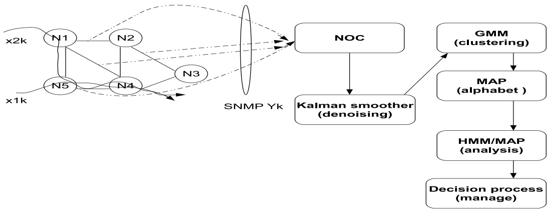


Figure 1. Architecture of the monitoring system.

den markov model, for time dependencies tracking, in the innovation process. One can think that time dependencies might involve in the innovation process, and these temporal correlation might appear between the different clusters we have already built. This means that the different families of discrete alphabet can be re-organized into different other groups, each of them containing a subset of the GMM components. Since these groups are a priori unknown, we denote them as *hidden states*. The use of hidden markov model will be a good paradigm to achieve this aim. Here, we perform a maximum a posteriori criteria to obtain the sequence of the hidden states of our HMM. Finally, the Figure 2 shows that, to track anomalies buried in the link counts, the use of a decision process is necessary, for management and decisions issue. In our anomaly detection philosophy, since anomalous events might be rare, we believe that it is possible to find one typically unique path, where these anomalies might be identified and detected. So, to perform the management task, we propose in this work a decision process based on two routines: the use of the Viterbi algorithm, in order to find all the variations in the residual process and, as a final step, the use of variance test for each data-point in the residual. Clearly, we will build a GMM with N components, then we use these N components to form an HMM with P states(each state containing part of the N components), and at last the Viterbi algorithm will find one unique path with a sequence of K(¡P) states and finally we use these K states to perform anomaly detection. Based on the discrete innovation sequence, the Viterbi algorithm finds the only best sequence that capture all the variations in the continuous form of the innovation process. In the case of per-link analysis, we build a Viterbi path for each data observation sequence, but in the case where we analyze all the links simultaneously, we find one single best Viterbi sequence. Our study shows thereafter that, a simple basic variance test based point-analysis suffices to identify and track the anomalous events.

3 Methodology

The first step in our study is to seek for a model which can monitor the system traffic dynamics. We believe that observing and estimated the real values of these metrics can help networks administrators to understand normal or potential abnormal conditions in their environment. A powerful model might identify the behaviour of normal traffic, by capturing the temporal evolution of the data, and at least can help operators to do future prediction. The full model is represented by the two linear equations, combined to form the complete specification of our system:

$$\begin{cases} X_{t+1} = C_t X_t + W_t \\ Y_t \ = A_t X_t + V_t \end{cases} \tag{1}$$

This block equations is the classic form for linear dynamical system (without inputs). In this model we assume that the state-noise W_t and the measurement-noise V_t are uncorrelated zero-mean gaussian white-noise processes with covariance matrices Q_t and R_t, respectively. For a full understanding of these system, we refer the reader to our work in [4]. Since we have found a model and system equations for our system, next we need to deal properly with the different steps of our optimization algorithm for anomaly decision issue.

3.1 Equations of the Kalman filter

The first problem to solve after building a simple model to monitor link counts, is to find an optimal estimate ($\hat{X}_t$) of our unobservable network states X_t, given a set of measurements $\{Y_1, ..., Y_t\}$. A well-known and robust method to achieve this aim is the Kalman filtering algorithm. In our dynamical linear system, we refer to Y_t as the observation vector at a specific time t. And the state of the system at time t is given by X_t, let also $\hat{X}_{t|k}$ denotes the estimate of X_t using all the information available up to time k, i.e, $\forall \tau < k$. $\hat{X}_{t+1}$ denotes the estimate of X_{t+1} using all the information up to time t, (this constitutes the *phase predictor*). The quantity $\hat{X}_{t+1|t+1}$ denotes the estimate of X_{t+1} using all past information and the recently arrived data point at time t+1. In the other hand, $P_{t|t}$ denotes the covariance of the *state estimate* and $P_{t+1|t}$ indicates the covariance of *the state prediction*. As it is shown in its earlier elaboration, the Kalman filter addresses the problem of estimating a discrete state vector when the observations are only a linear combination of the underlying state vector. The filter runs as a *predictor-corrector* algorithm. As an iterative algorithm, it estimates the system state using two steps: *prediction* comes in the *time update* phase, and *correction* in the *measurement update* phase.

- **Prediction step** (*time update equations*):
  In this step, the estimated state of the system at time

t, $\hat{X}_{t|t}$, is used to predict the state at next time t+1, $\hat{X}_{t+1|t}$. And, as we know that the noise W_t influences the evolution of the system at each time t, we compute only the covariance of the prediction, $P_{t+1|t}$ based on the updated covariance at the previous time t, $P_{t|t}$, and the noise covariance at the same time, Q. The error covariance $P_{t+1|t}$ provides an indication of the uncertainty associated with the state estimate.

$$\begin{cases} \hat{X}_{t+1|t} = \hat{X}_{t|t} \\ P_{t+1|t} = P_{t|t} + Q \end{cases} \quad (2)$$

- **Correction step** (*measurement update equations*): This step updates (corrects) the state and the variance of the estimate in the previous step, using a combination of their predicted values and the new observations Y_{t+1}. The correctness of this update depends on the Kalman innovation $Y_{t+1} - \hat{X}_{t+1|t}$.

$$\begin{cases} \hat{X}_{t+1|t+1} = \hat{X}_{t+1|t} + K_{t+1}(Y_{t+1} - \hat{X}_{t+1|t}), \\ P_{t+1} = (I - K_{t+1})P_{t+1|t}(I - K_{t+1})^T + K_{t+1}RK_{t+1}^T \end{cases} \quad (3)$$

In the measurement equations, K_{t+1} denotes the Kalman gain. For more details in linear dynamical system, estimation and Kalman filter techniques, we refer the reader to: [22],[20],[3] and [1]. The above equations with initial conditions of the state of the system $\hat{X}_{0|0} = E[X_0]$ and the associated error covariance matrix $P_{0|0} = E[(\hat{X}_0 - X_0)(\hat{X}_0 - X_0)^T]$ define the discrete-time sequential recursive algorithm for determining the linear *minimum variance* estimate known as the *Kalman filter*.

3.2 Gaussian Mixture Model

In Kalman filtering philosophy it is more generally assumed that the residual remains a zero mean gaussian process, however this assertion is not always true in practice. Long range dependences and heavy tails distribution can lead to non negligible deviation from the normality and the independence of the innovation process. Thus our motivation to use gaussian mixture model is based on our belief that the real distribution of the process is an ensemble (mixture) of gaussians and there is some time dependency in the innovation (which we will later study with the aid of an hidden markov model). This assertion allows us to build a method which has the ability to find anomalies in different families built on the same residual sequence. The calibration of a GMM helps us, taking as input the Kalman innovation, to build K families (clusters) and in the next step we propose the use of an hidden markov model (HMM) to classify these groups into P states, each state being formed with part of the K families. This operation have the advantage to discover the potential temporal dependencies in the innovation process.

To find the values of the GMM model parameters μ_m (mean)and Σ_m (variance), as well as the prior probability vector π, we are interested in maximizing the *likelihood* $\pounds(\theta|\mathbf{X}) = p(\mathbf{X};\theta)$ of generating the known observed data (X) given the model parameters $\theta = \{\mu_m, \Sigma_m, \pi_m\}$, 1≤m≤M. **X** denotes all the observation while θ contains all the parameters of the mixture. In other words, we hope to find $\hat{\theta}_{ML}$=argmax $p(x|\theta)$. This approach is called the Maximum Likelihood (ML) framework since it finds the parameter settings that maximize the likelihood of observing the data sets.
To find the best parameters of the features of θ, the Expectation Maximization (EM) iterative algorithm can be used to simplify the math considerably and numerically compute the unknown parameters. For more details about mathematical routines for EM, see [10],[17],[21].

3.3 Hidden Markov Model

From the above learning phase, each GMM component (cluster) is transformed into a sequence of a finite set of alphabet (symbols as 1,2,3,...), using a *maximum a posteriori* criteria. This discrimination phase will help for plugging the above clusters into different a priori unknown states, using hidden markov model. Our results will show us that, clusters with anomalies appear on some states and clusters with no anomalies remain in an other states. We represent these families of states by the following collection of unknown random variables $\{Q_1, Q_2, ...Q_T\}$ (where Q_t is a constant value with values in $\{1, 2, ..., K\}$). We also represent our alphabet by the known vector $\{O_1, O_2, ..., O_N\}$. Now, the problem is resumed to find a model to produce the states and to determine the probability of each symbol being in a state. A well known model-based approach to tackle this problem, is the discrete *hidden markov model* (HMM) which is an exponential family of mixture model. Our choice of using HMM is based on the fact that: i) potential time dependencies in the innovation process can be modelled and captured using a finite set of a *priori hidden states*, each of them containing a subset of gaussian components ii) relatively efficient algorithm can be derived to solve the problems related to them, [17],[10],[9]. The full HMM model we used is defined by the quantity $\lambda = (A, B, \pi)$ (where A is the *transition matrix*, B is the *emission probabilities* matrix and pi the *prior* probabilities). To find and estimate the best parameters of our model, we use the well-known *forward-backward* algorithm parameter estimation (or Baum-Welch algorithm). For more details for EM techniques related to hidden markov model, see, [10], [9]. Thereafter, we reuse the model to find the optimal state sequence associated with the given observation sequence. We believe that this final step of our approach will allow us to capture all the variations in the innovation process. An optimal criteria we have chosen here is to find the single best state sequence (path), $Q = \{q_1, q_2, ..., q_T\}$ for the given observation sequence $O = \{O_1, O_2, ..., O_T\}$,i.e., we aim to maximize $P(Q|O, \lambda)$. A formal technique for finding this unique best state sequence is the Viterbi algorithm.

4 Model Evaluation

4.1 Data collection : Abilene network

The Abilene backbone has 11 Points of Presence(PoP) and spans the continental US. The data from this network was collected from every PoP at the granularity of IP level flows. The Abilene backbone is composed of Juniper routers whose traffic sampling feature was enabled. Of all the packets entering a router, 1% are sampled at random. Sampled packets are aggregated at the 5-tuple IP-flow level and aggregated into intervals of 10 minute bins. This thus dictates the underlying time unit of all of our estimations and detections. The raw IP flow level data is converted into a PoP-to-PoP level matrix using the procedure described in [5]. Since the Abilene backbone has 11 PoPs, this yields a traffic matrix with 121 OD flows. Note that each traffic matrix element corresponds to a single OD flow, however, for each OD flow we have a seven week long time series depicting the evolution (in 10 minute bin increments) of that flow over the measurement period. All the OD flows have traversed 41 links.

4.2 Model Validation

The anomalies injected in the Abilene data are small and high *synthetic volume anomalies*. We used exactly the same Abilene data as in [8]. So for a full understanding on how the **ground-truth** is obtained (based on EWMA and Fourier algorithms) , we refer the reader to [8].

4.2.1 System parameter identification.

Our method begins with a learning phase where we calibrate a Kalman filter for denoising, using our linear dynamical system, and the collection of observation data. In order to run the Kalman filter, we need the matrices A, Q, C et R. The matrix A is available given the routing scheme of a network. We thus only need to obtain Q, C and R. Recall that one of our aims is to prove that, learning all the links simultaneously is better than using one link at a time. Calibrating the system for one link at a time is not very straightforward. Here to estimate the parameters θ=$\{C, Q, R\}$, we use an autoregressive (AR) model of order p. The method is based on a stepwise least squares algorithm which uses a QR factorization of a data matrix to evaluate, for a sequence of successive orders, a criterion (here Schwarz's Bayesian Criterion and Akaike's Final Prediction Error) for the selection of the model order p, and to compute the parameters of the AR model of the optimum order. It is sufficient to learn the model's parameters using a sample of one week measurements from the data links studied. We run this method based on the notes described in [6], and developed in [7]. We form the matrix C using the p parameters of the AR(p). The noise variance of the model is used to fix Q. The matrix R is built as a function of Q. In practice, R is obtained by dividing the measurement noise by some constant. In our experiments we have found this constant in the interval [2:4]. In the case where we take account of all links at the same time, we find the same parameters θ, using N consecutive samples of link counts, and we do our calibration using the *EM algorithm*. In practice, in order to apply the EM procedure, initial values are required for the parameters and these were simply guessed by examining portions of the completely observed series. It is a convenient idea to examine several different sets of starting values, since the EM algorithm may reach different kinds of stationary values corresponding to local rather than global maxima. For more details of the EM techniques, see [18],[19].

4.2.2 Summary of the Results.

The first result we want to show is the ability of our method to track the behavior of link counts (total byte per unit time) over time. In Figure 2, we show the real and inferred link counts for our model. The evolution of the traffic and estimates are shown for a seven weeks duration for each observation vector. The calibration is applied only once, for the case where we analyze all the links at the same time (we have shown only the figures for this case). For the case of per-ink analysis, we have performed the calibration for each sequence separately. Now we are looking at the performance of our two methods, in the Abilene network. To validate our GMM model, we calibrate one GMM with a set of *r* components (r=2,3,4,5...) using the EM algorithm, and the decision to select the best model is done by analyzing the variance performed for each component in the mixture. The model with the lowest variance is chosen. We find in our computations, that the data residual can be organized into three ($r = 3$) distinct clusters. Thereafter A maximum a posteriori criteria is used to build, over the clusters, a finite alphabet of symbols, where we run the *hidden markov model*. To train the *hidden markov model*, one must ensure that that the different *hidden states* are well separated. This means that one should have a transition matrix with *higher probabilities in its main diagonal*. In our study we obtain two (2) states with the transition and observation matrices shown as follow (case where the

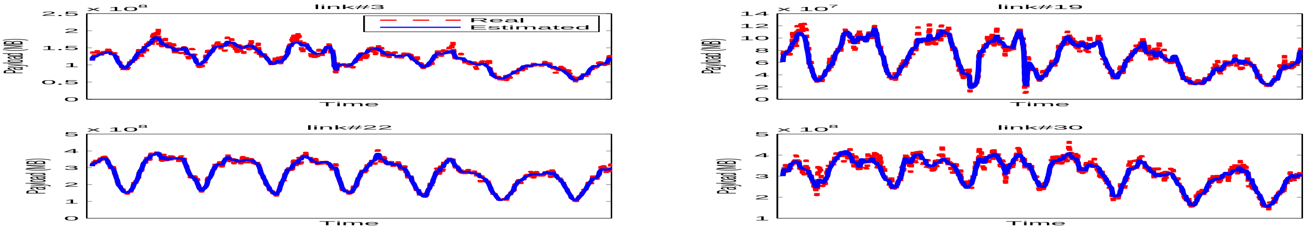


Figure 2. Real(red) and estimated (blue) links obtained using Kalman filter.

analysis is performed in all links.)[1]:

$$transmat = \begin{bmatrix} 0.9862 & 0.0138 \\ 0.0162 & 0.9838 \end{bmatrix}$$

$$obsmat = \begin{bmatrix} 0.0013 & 0.0020 & 0.9967 \\ 0.4892 & 0.5104 & 0.0004 \end{bmatrix}.$$

The HMM results show clearly that, the state 1 is composed with almost entirely the symbol$\#3$ and the state 2 is a mixture of two symbols (1 and 2) with 49% of probability of presence of symbol$\#1$ and 51% for symbol$\#2$. In the philosophy of anomaly detection theory, generally it is assumed, that anomalies might be rare; base on this assumption we can ask this question: if anomalous events occur, do they might come from state $\#1$ or state $\#2$ or both? To answer this question, we believe that all the changes in the mean of the residual (abrupt change, slow or high variations) can be tracked by a combination of symbols yielding in one state. In other words, one can think that one state must be classify as normal state and the other as abnormal state. Clearly, we believe that the state 1 containing the symbol $\#3$ corresponding to the component with **mean closely equal to zero** might be labelled as the "normal state" and the state 2 as the "abnormal state". To confirm our intuition, we run the Viterbi algorithm for all the sequences of discrete alphabet and we discover one unique sequence composed only by the symbols in the state $\#2$. In Figure 3, we show how the lower and higher variations in the residual are well tracked by the Viterbi path (composed only by symbol$\#1$ and symbol$\#2$). For the purpose of plotting, we multiply each symbol by a same constant). At this time, we can argue that, if anomalies exist they might be caught either by the two symbols simultaneously, either by one symbol only. In addition, in our study we discover that anomalous events never happen in the cluster ($\#3$) with *mean closely equal to zero*. Combining this theory with a basic variance test performed for each data point, allows us to detect, identify and track anomalies in the different links. Among the 41 links, only 13 have drawn anomalies. In the Figure 5 we plot the tracking operation for some links. For all the figures, the red points corresponds to the symbol$\#1$ and the blue ones to the symbol$\#2$. The solid blue line is the threshold for the part of residual corresponding to the symbol$\#1$, and the dashed red line is the threshold for the part of residual corresponding to the symbol$\#2$. We run the HMM more than one time to show that in some cases, the Viterbi path can be a sequence with only the presence of one symbol. And all anomalies yielding in the link in question are tracked inside the residual corresponding to that symbol. We can see some examples in Figure 4. In these cases, we can plot the results in continuous time, what we can not do in Figure 5. In our running scheme, we found that the coefficient that we multiply the standard deviation (to fix the threshold) of the residual variance lies in the interval $[2:6]$. Another result is about the performance obtained in analyzing the trade-off between false positive and false negative rates. We examine the entire traffic for each method at two levels: in level 1 we run the methods using one link at a time and in level 2 we consider all the links at the same time. We then can compute one false positive percentage and one false negative percentage for each threshold configuration scheme. The performance of the two methods on the Abilene data is depicted in the ROC (Receiver Operating Characteristic) curve of Figure 6. In Figure 6(a), we can see clearly that: i) when the analysis is done on a per-link basis, the hmm-variance-based anomaly detection performs better than a basic variance analysis. We can see that with a false positive rate (FPR) of 1.3% the HMM misses no anomalies (100% detection rate), while the variance test catches about 80% of the anomalies for the same FPR. ii) in the Figure 6(b), we show that analyzing all the links count simultaneously is more advantageous than taking account for one link at a time. One can observe that the HMM catches all the anomalies (0% of false negative) with a false positive rate of 0.006, while in the same time the variance test catches 85% of anomalies with the same FPR.

5 Conclusion

Our work has shown the robustness of combining mathematical tools for Anomaly detection issue. We begin by performing a Kalman filter on data selected in granularity of link counts. Then a GMM is used to organize the residual into different gaussian components, since we have as-

[1] In the case of per-link analysis we obtain an transition matrix and an observation matrix for each link

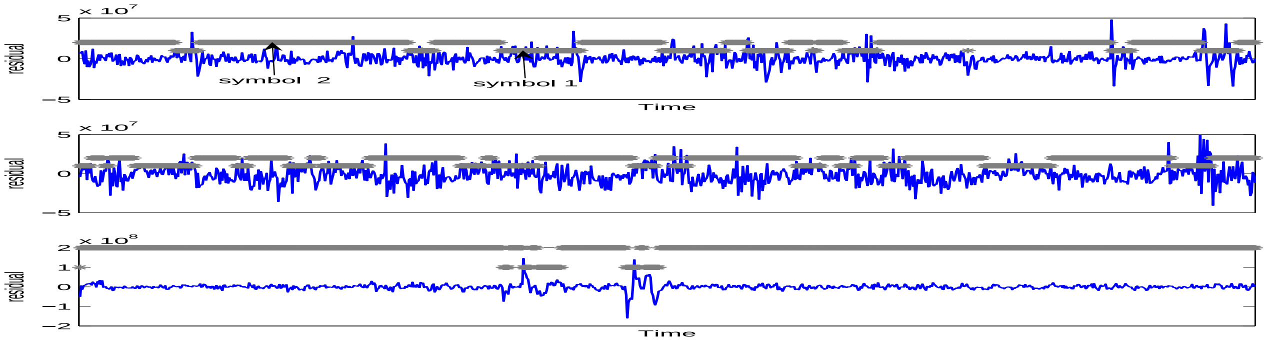


Figure 3. Track of the variations in the residual with the Viterbi path.

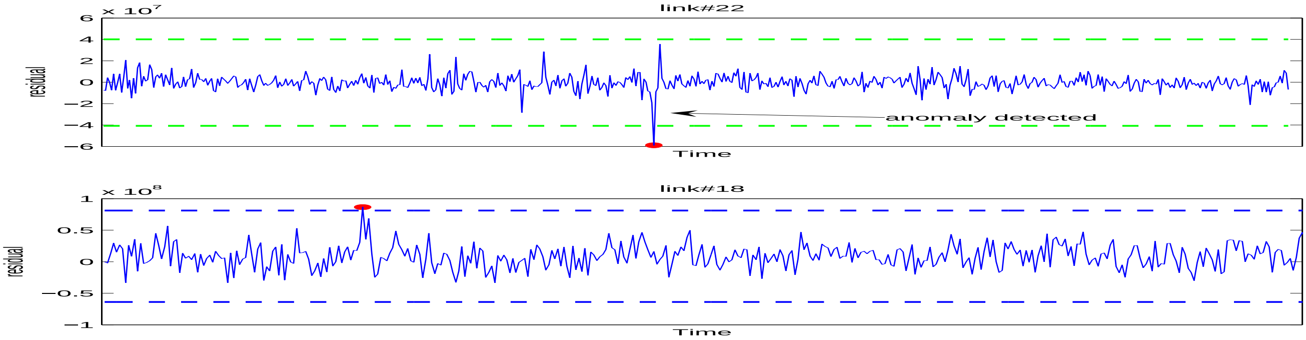


Figure 4. Anomalies detection in 2 links using only symbol 1.

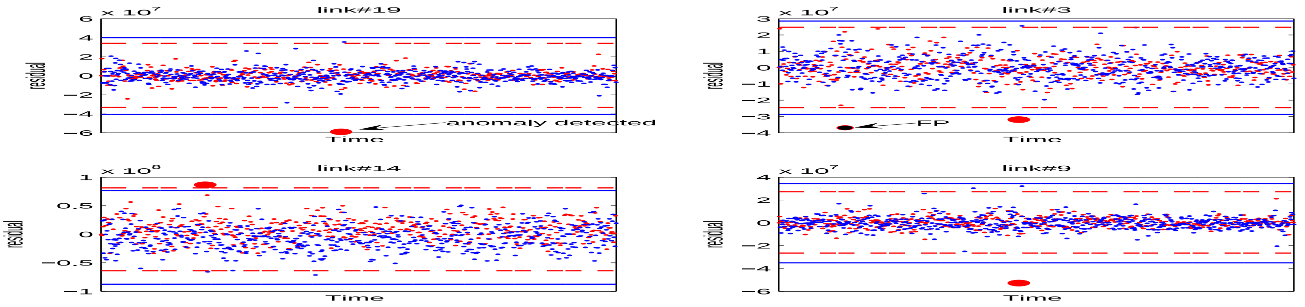
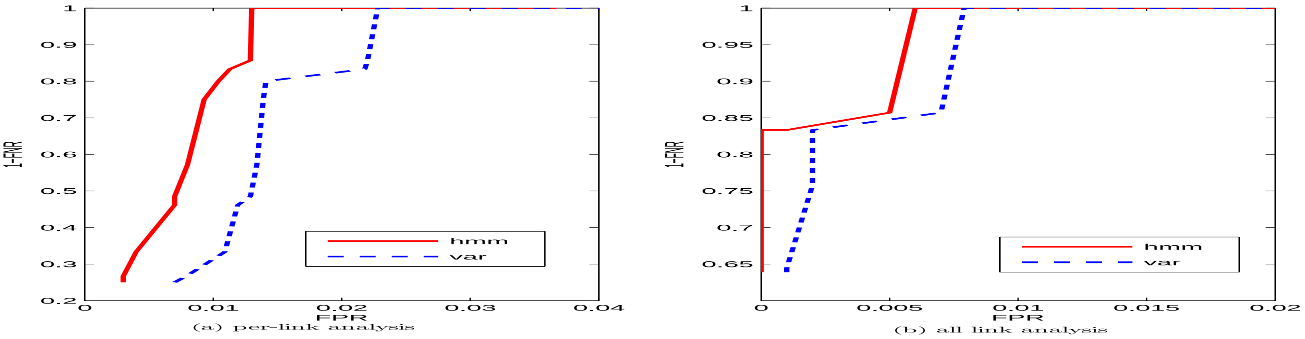


Figure 5. Anomalies detection in 4 links using symbols 1 and 2.



(a) per-link analysis

(b) all link analysis

Figure 6. ROC curve using Abilene data.

sumed that the real distribution of the innovation process is a mixture of normal densities. Thereafter a MAP criteria is used to build an alphabet in which we perform a hidden markov model to find the temporal dependencies above the different gaussian components. Finally, we use the Viterbi algorithm to show that anomalous events can be tracked using a single sequence of our HMM states where we perform a variance test for each data point. In our study, we show that our anomaly detection methodology can provide appropriate management tools and can help network administrators to surveying their environment, since it can achieve higher detection rate with few false positive and the results are more convenient in the case of all-links analysis than in per-link based analysis. We have found that anomalies are capture by a part of the alphabet and one can ask this question: what about the remaining symbol (here symbol #3) ? Is it uncertain that anomalies may never happen in the cluster with mean closely equal to zero ? At the other hand, we must notice that the determination of the constant used to fix the variance threshold is heuristic and not straightforward. One would prefer the fixation of this value in an obvious and automatic manner. We will point out these ideas and questions in future works.

References

[1] Raugh, H. E.: Solutions to the linear smoothing problem. IEEE Trans. Automatic Control. Octobre 1963.

[2] Soule, A., Salamatian, K.,Taft, N.: Traffic Matrix Tracking using Kalman Filters. ACM LSNI Workshop (2005).

[3] Wolverton, C.: On the Linear Smoothing Problem. IEEE Transactions on Automatic Control, February 1969.

[4] Soule, A., Salamatian, K.,Taft, N.: Combining Filtering and Statistical Methods for Anomaly Detection. USENIX Association, Internet Measurement Conference (2005).

[5] Lakhina, A., Crovella, M.,Diot, C.: Characterization of network-wide traffic anomalies. In ACM Sigmetrics (2004).

[6] Neumaier, A. and Schneider,T.: Estimation of parameters and eigenmodes of multivariate autoregressive models. ACM Trans. Math. Softw., 27, 2757. (2001).

[7] Schneider, T. and Neumaier A.: Algorithm 808: ARfit - A Matlab package for the estimation of parameters and eigenmodes of multivariate autoregressive models. ACM Trans. Math. Softw., 27, 5865. (2001).

[8] Lakhina, A., Crovella, M.,Diot, C.: Diagnosing network-wide traffic anomalies. In ACM Sigcomm (2004).

[9] Rabiner, L.,: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition.

[10] Bilmes, Jeff A.:A Gentle Tutorial of the EM algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models, International Computer Science Institute,Berkeley CA,April 1998.

[11] Barford, P., Kline, J.,Plonka, D.,and Ron, A.: A signal analysis of network traffic anomalies. ACM Sigcomm IMW (2002).

[12] Barford, P.and Plonka, D.: Characteristics of network flow anomalies. In ACM IMW (Nov. 2001).

[13] Hussain, A.: Measurement and Spectral Analysis of Denial of Service Attacks. PhD thesis, USC, May 2005.

[14] Thottan, M. and Ji, C.:Anomaly Detection in IP Networks. IEEE Transactions on Signa Processing, vol. 51, NO. 8. August 2003.

[15] Thottan, M. Fault detection in ip networks, PhD dissertation, Rensselaer Polytech. Inst.,Troy, NY, 2000. under patent with RPI.

[16] Wang, H., Zhang, D., and Shin, K. G.:Detecting syn flooding attacks, in Proc. IEEE INFOCOM 2002.

[17] McLachlan, G. and Krishnan, T.: The EM Algorithm and Extensions. John Wiley and Sons, New York, 1996.

[18] Shumway, R. H. and Stoffer, D. S.: An Approach to Time Series Smoothing And Forecasting Using the EM Algorithm. Journal of Time Series Analysis, vol.3, No 4.

[19] Shumway, R. H. and Stoffer, D. S.: Dynamic Linear Model With Switching. Journal of the American Statistical Association, 86, 1991.

[20] Kailath, T., Sayed, A. H. and Hassibi B.: Linear Estimation. Prentice Hall, 2000.

[21] Dempster, A. P., Laird N. M., and Rubin D. B.: Maximum likelihood from in-complete data via the em algorithm. Journal of the Royal Statistical Society: Series B, 39(1):138, November 1977.

[22] Kalman, R. E. and Bucy R. S.: New results in linear filtering and predictions. Trans. ASM E., March 1961.