

Traffic Matrix Tracking using Kalman Filters

Augustin Soule, Kavé Salamatian, Antonio Nucci, Nina Taft ¹

Abstract

In this work we develop a new approach to monitoring origin-destination flows in a large network. We start by building a state space model for OD flows that is rich enough to fully capture temporal and spatial correlations. We apply a kalman filter to our linear dynamic system that can be used for both estimation and prediction of traffic matrices. We call our system a traffic matrix tracker due to its lightweight mechanism for temporal updates that enables tracking traffic matrix dynamics at small time scales. Our Kalman filter approach allows us to go beyond traffic matrix estimation in that our single system can also carry out traffic prediction and yield confidence bounds on the estimates, the predictions and the residual error processes. We show that these elements provide key functionalities needed by monitoring systems of the future for carrying out anomaly detection. Using real data collected from a Tier-1 ISP, we validate our model, illustrate that it can achieve low errors, and that our method is adaptive on both short and long timescales.

1 Introduction

The complexity of managing large carrier networks is rising dramatically. The number of active elements in a network has increased steadily during the past few years, as well as the sophistication and functionality of these elements. At the same time traffic demand has evolved by becoming more complex and sometimes contains hostile components. Network operators rely on monitoring technologies to help them manage the vast system that constitutes each single administrative network domain. These two convergent phenomena of expanding network complexity and increased complexity of traffic demand, create an ever expanding network monitoring challenge.

Examples of how monitoring can enhance network management activities include the following. First, the management of a large network entails being able to identify failures and the extent of their impact. In addition, monitoring systems may be able to help identify the causes of failures if the right kind of monitoring data is collected, and an operator knows how to understand and inter-

pret such data. A second application is that of tracking traffic loads, traffic demands and how well capacity planning designs are performing, in terms of meeting the demands. Adjustments to capacity planning can be made when sustained changes to a set of link loads or traffic demands are uncovered. Third, the last few years have seen increasing emphasis on developing intrusion detection systems (IDS) to monitor the network, so that we may improve our ability to handle malicious traffic such as Distributed Denial of Service (DDOS) attacks, worms, flash crowds, etc. This field is considered young because the problems are very difficult and most commercial systems are still relatively simple; however it is well accepted that such types of monitoring are going to be a necessary and prevalent component of future network management systems.

Historically network monitoring has focused on surveilling individual elements. We believe this simplistic vision is no longer sufficient. Many of today's traffic monitors and IDS systems monitor a single link. Even when a network operator uses a set of monitors placed in strategic locations, the network operator is still faced with the task of correlating and understanding the monitored data from these multiple vantage points. Rather than being restricted to a subset of nodes or vantage points, why not monitor everywhere? In order to handle the types of management tasks above, we believe that having a full network-wide view of the traffic, along with any ability to monitor the global state of network traffic, would be a tremendous asset. However, monitoring everywhere can be extremely costly, and thus monitoring systems should aim to build network-wide views by collecting limited measurements and combining them with inference procedures.

A network-wide view of traffic in an administrative domain is often referred to as the traffic matrix. A traffic matrix is a representation of the volume of traffic that flows between origin-destination (OD) node pairs in a communications network. The choice of node type affects the granularity and type of the traffic matrix; one can define link-to-link, router-to-router and PoP-to-PoP (Point of Presence) traffic matrices. In this work we focus on PoP-to-PoP traffic matrices because carriers wanting to characterize the global state of their backbone, often aggregate traffic to this level. Moreover, the data set we have at our disposal for validation comes from a Tier-1 carrier backbone at this granularity level.

¹A. Soule and K. Salamatian are with the LIP6-UPMC Laboratory, France. A. Nucci is with Narus Inc., USA and N. Taft is with Intel Research Berkeley, USA

Because the traffic matrix captures the global state of network traffic, it is extremely challenging to monitor directly [5]. Since the traffic matrix is not directly observable, it typically has to be inferred from other data that is observable, such as the Simple Network Management Protocol (SNMP) that delivers total byte counts per link every five minutes. Traffic matrix inference from indirect observation has been an active research area in the last few years called Network Tomography [4, 9, 8]. The interest in monitoring traffic matrices is that they represent a global network state but do not require monitoring everywhere since they can be inferred from the data collected by existing monitoring protocols (SNMP).

Although monitoring has been used for quite some time in the application areas of failure detection and analysis, and capacity planning, the use of monitoring for anomaly detection has become popular only recently. We believe that monitoring traffic matrices for the purposes of facilitating anomaly detection is a potentially rich area that has barely been explored [3]. Being able to use traffic matrices for such an application imposes two requirements, an ability to track traffic in real-time (which limits algorithm complexity) and an ability to describe the statistical behavior of “normal” OD flows. Clearly a monitoring system cannot identify abnormal behavior without an a priori understanding of normal behavior. This means that a traffic matrix estimator must be built on the basis of a rich traffic model that can capture both temporal and spatial correlations among OD flows.

In this paper we propose to view traffic demands as the *state* of a system. The traffic matrix thus represents a global network state. We then build a model for OD flows that captures both the temporal and spatial correlations in flows. In our system the OD flows are not directly observable; instead only the link counts are directly observable. We define a linear dynamic state-space system that incorporates both the OD flow model and the observable link counts. The aim is thus to estimate the state from the observables. The classical tool for handling this kind of filtering problem is the well-known *Kalman filter*. This technique has been widely and successfully applied to monitoring of electric grids and chemical processes.

There are some interesting advantages to using Kalman filters in the context of traffic matrices. Not only can we do traffic matrix estimation, but we can also do traffic prediction. Moreover, we are able to obtain confidence bounds on our estimates, our predictions and a residual error process. We believe that using predictions, and confidence bounds, combined with estimates (that get updated on an on-going basis) can provide key functionality needed inside anomaly detection methodologies. We will illustrate these ideas in the paper. There are other advantages as well; it naturally incorporates both temporal and spatial correlations, it is adaptive because of the continuous adjustment to the Kalman gain factor

based on past errors, and it can meet real-time requirements (once the base model is calibrated). We can compute our estimates and predictions quickly because the update steps are lightweight; this makes our method a candidate for implementation.

The focus of the present paper is on our model, including the model calibration process and a validation of the model assumptions. Some performance results are provided herein. For a more complete evaluation, as well as a comparison to other methods, we refer the reader to [7].

Section 2, describes a general monitoring infrastructure and explains how our algorithms support such a system. Section 3 describes the methodology, including the Kalman filter equations, model calibration method and how we detect for underlying model changes. Section 4 illustrates the success of our traffic matrix tracking via Kalman filtering. We summarize our work and give some thoughts for future efforts in Section 5.

2 Basic Monitoring System Design

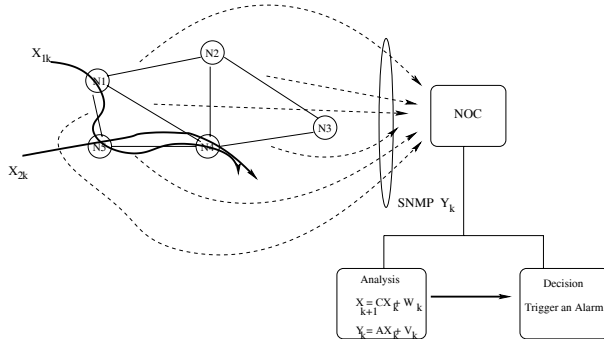


Figure 1: Example of Monitoring System

We now give a brief description of the components in a basic monitoring system. This description also helps to clarify the role that our work plays. Being able to efficiently monitor the network requires three functional blocks: the *data collection*, the *data analysis* and finally the *decision process* (see Figure 1). *Data collection* is challenging since measurements are typically scattered across the network, and collecting and exporting them to a single location, such as a *Network Operations Center* (NOC), may be a resource consuming process. In many ISP networks today, operators only collect SNMP data statistics and routing information. The dashed lines in Figure 1 indicate the collection of per-link statistics (e.g., SNMP), while the dark lines are examples of end-to-end flows (OD pairs) and is our target traffic granularity. The ensemble of all OD pairs is what constitutes a traffic matrix and what yields a network-wide view of traffic. Because collecting the entire traffic matrix is too

resource consuming [5], this often has to be inferred from limited data that is easier to collect.

The *data analysis* part refers to the processing and computations carried out on the data collected. Observing the value of a metric, even if interesting, can be of little use if an *a-priori* about this metric does not exist. These a-priori’s help an operator to understand if the observed metric reflects a normal (i.e., expected) network condition or not. An *a-priori* might come from the expertise of an on-duty network operator, or it might be captured using some kind of model. A good model could clarify normal ranges of behavior. A model that is powerful enough to capture the temporal evolution of the data, could even make a future prediction. Being able to compare predictions of a data stream with what actually happens (once the real data arrives via a collection procedure) can provide useful information to an operator. A discrepancy between the model’s prediction and what really happened could indicate that an unusual change happened somewhere. (The equations in the analysis block in Figure 1 are the modeling equations our analysis uses. These will be explained in the upcoming section.)

The *decision process* is the component that decides when to generate alarms and what action to take in response. Sometimes the data analysis part can yield, not only an estimate of the network state, but also some confidence intervals for that estimate. Having confidence intervals for an estimate facilitates decision making. The decision process can then compare an observed metric with the confidence interval around its estimated value; if the observation falls inside (*resp.* outside) the confidence interval the network is classified as working in “normal condition” (*resp.* “not-normal condition”).

In this work we focus on building a model for network-wide traffic that could be used by a monitoring system to both track normal behavior and to facilitate anomaly detection. Based on the above discussion, we set the following goals for such a model: (1) it should be able to make accurate estimates of the traffic; (2) it should make future predictions and thus capture temporal dynamics in order to do so; (3) it should yield confidence intervals for its traffic predictions and the associated errors. Our work thus falls primarily into the data analysis portion of a monitoring system. We are mindful of the data collection process as we seek to make traffic estimates and predictions using a small amount of collected data. We also seek to provide useful information to the decision process, which is what motivates our third requirement above.

Our intent is thus to go beyond traditional traffic matrix estimation by providing a single system that can do both estimation and prediction, as well as generate confidence intervals for our estimates and predictions.

3 Methodology

Before proposing a model, we first need to relate the observed data, i.e. SNMP, with the unobserved origin-destination (OD) flows. Second we seek a model that captures the evolution of OD flows in time (otherwise we could not do prediction). The natural relationship between SNMP data and OD flows can be expressed by the following equation

$$Y_t = A_t X_t + V_t \tag{1}$$

where Y_t represents the link count vector (SNMP data), X_t the OD flows organized as a vector (hidden network states). A_t denotes the routing matrix where each element a_{ij} is equal to 1 if the OD pair i is present on the link j . Because data collection techniques often incur errors, we let V_t denote a stochastic process capturing measurement errors. All these parameters are defined for a general discrete time t .

Providing an efficient model for X_t that captures traffic dynamics is not straightforward. It has been observed that traffic entering the network is characterized by highly variable behavior in time. Previous work [8] presented a deterministic model that assumes the traffic to be a combination of a deterministic diurnal pattern, and a zero-mean fluctuations process around it. The diurnal pattern is modeled through a Fourier series expansion. Although this simple model captures the periodic trend of the traffic well, it fails when changes in the traffic occur because it can neither allow for a little drift (small change), nor recalibrate the underlying trend (large or permanent change). Changes in traffic demands can arise due to different causes such as equipment failures, unusual customer behavior (e.g., flash crowds), the addition of new customers, or even attacks conducted against the network. In order to develop a more flexible model, we propose a general stochastic model where the network variables X_t at a generic discrete time t are described as a combination of three components $X_t = \hat{X}_t + \epsilon_t + \eta_t$ including a predictable component \hat{X}_t , a random noise component ϵ_t , and a third component η_t called the *innovation process*. The innovations account for the behaviors that we cannot predict. Later on we show how η_t surfaces in our method and how analysis of the η_t stochastic process can be used to detect unusual network dynamics.

In general the prediction model can have any structure and the noise process can also have any distribution. However linear stochastic predictive models combined with gaussian noise have a long record of successful application in a wide spectrum of monitoring problems. One of our ideas here is to view the OD flows as *network states*. (Since these variables are not directly observable from the network, we refer to them as *hidden network states*.) Viewing the variable we want to estimate (X_t)

as a *state* led us to think about using linear dynamic systems based on state-space models. We build a temporal model that relates X_{t+1} to X_t using the following linear equation,

$$X_{t+1} = C_t X_t + W_t \quad (2)$$

where the state transition matrix C_t captures the dynamic behavior of the system, while W_t is a process noise that accounts both for the randomness of the traffic and the unpredictable elements. The diagonal elements of the matrix C_t capture the temporal correlations within a single flow. The off-diagonal elements of C_t captures spatial correlations across different OD flows.

Combining equations 1 and 2 gives the full specification of our system:

$$\begin{cases} X_{t+1} &= C_t X_t + W_t \\ Y_t &= A_t X_t + V_t \end{cases} \quad (3)$$

This is the classic form for linear dynamic systems. In this model we assume both the state-noise W_t and the measurement-noise V_t to be uncorrelated, zero-mean gaussian white-noise processes and with covariance matrices Q_t and R_t .

Given the above model and system equations, the problem at hand is to find an optimal estimate (denoted by \hat{X}_{t+1}) of the true network state X_{t+1} , given a set of past and current observations $\{Y_1, \dots, Y_{t+1}\}$. Optimality is defined here in the sense of a Minimum Variance Error Estimator (MVUE), defined as $E[||X_{t+1} - \hat{X}_{t+1}||^2]$. The classical tool for dealing with this type of problem is the well-known Kalman Filter [2].

In modeling there is always a tradeoff between accuracy and complexity. Typically one seeks a model simple enough to be practical while also being accurate to within some target error rate. We will see that a simple linear model with dimensionality equal to the number of OD flows is good enough for tracking OD flow behavior. Clearly a more complex model (such as one incorporating second moments) might give better modeling properties. But this would most likely turn out to be significantly more complex and less practical. Besides, one crucial and well-known property of feedback systems is that the feedback can compensate for model uncertainty. This is one of the properties that explains the broad usage of Kalman filters in a wide range of applications.

3.1 Kalman Filter Equations

We present our kalman filter in terms of a time-varying system (e.g., A_t rather than A) to illustrate a general solution. (In our evaluations we find that parameters A , Q and R can be time-invariant for many consecutive days.) We refer to Y_t as the observation vector at discrete time t , and let $Y^t = \{Y_t\}$ denote the set of all observations up to (and including) time t . The state of the system is given by X_t , and we let $\hat{X}_{t|i}$ denote the

estimate of X_t using all the information available up to time i , i.e., $\forall \tau \leq i$. Thus $\hat{X}_{t+1|t}$ denotes the estimate of X_{t+1} given all the information available up to time t (this constitutes the *prediction*); while $\hat{X}_{t+1|t+1}$ denotes the *estimate* of X_{t+1} using all past information plus the newly arrived data collected at time $t+1$. We use $P_{t|t}$ to refer to the variance of the state estimate, and let $P_{t+1|t}$ indicate the variance of the state prediction.

The Kalman filter addresses the general problem of estimating a discrete state vector when the observations are only a linear combination of the underlying state vector. It estimates the system state using an iterative two step approach:

- **Prediction Step:** 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 $t+1$, $\hat{X}_{t+1|t}$. Since the evolution of the state is influenced by the noise W_t , we also compute the variance of this prediction $P_{t+1|t}$, based on the updated variance at time t , $P_{t|t}$, and the noise covariance at time t , Q_t .

$$\begin{cases} \hat{X}_{t+1|t} &= C_t \hat{X}_{t|t} \\ P_{t+1|t} &= C_t P_{t|t} C_t^T + Q_t \end{cases} \quad (4)$$

- **Update Step:** The second step updates the state and the variance of this estimate, using a combination of their predicted values (previous step) and the new observation Y_{t+1} . The update is based on the innovation process η_t defined as the difference between what would have been observed if the predicted value was correct, i.e. $A_{t+1} \hat{X}_{t+1|t}$ and what is effectively observed at time $t+1$, namely Y_{t+1} . (Hence $\eta_t = A_{t+1} \hat{X}_{t+1|t} - Y_{t+1}$)

$$\begin{cases} \hat{X}_{t+1|t+1} &= \hat{X}_{t+1|t} + K_{t+1} [Y_{t+1} - A_{t+1} \hat{X}_{t+1|t}] \\ P_{t+1|t+1} &= (I - K_{t+1} A_{t+1}) P_{t+1|t} (I - K_{t+1} A_{t+1})^T \\ &\quad + K_{t+1} R_{t+1} K_{t+1}^T \end{cases} \quad (5)$$

The matrix K_{t+1} is called the Kalman gain matrix. If we let the estimation error at time t be given by, $\tilde{X}_{t|t} = \hat{X}_{t|t} - X_t$, then we compute the gain by minimizing the conditional mean-squared estimation error $E[\tilde{X}_{t+1|t+1}^T \tilde{X}_{t+1|t+1} | Y^t]$. By applying some basic linear algebra, we can write it as:

$$K_{t+1} = P_{t+1|t} A_{t+1}^T [A_{t+1} P_{t+1|t} A_{t+1}^T + R_{t+1}]^{-1} \quad (6)$$

The above equations together with the initial conditions of the state of the system $\hat{X}_{0|0} = E[X_0]$ and the associated error covariance matrix $P_{0|0} = E[(\hat{X}_{0|0} - X_0)(\hat{X}_{0|0} - X_0)^T]$ define the discrete-time sequential, recursive algorithm for determining the linear minimum variance estimate known as the *Kalman Filter*.

3.2 Model Calibration

This previous section presented the Kalman filtering method in its general settings under non-stationary situation. In the following sections we will assume a stationary situation where the matrices A, C, Q and R are constant in time, making it possible to drop their subscripts. Although this can be seen as a strong assumption, we show that even with this simple model we can achieve excellent results in capturing the essential characteristics of normal traffic patterns. However the methodology presented in the rest of the paper can be easily generalized to consider time dependency.

In order to execute our kalman filter in Equations (2)-(4), we need the matrices A, Q, C and R . The matrix A is available given the routing scheme of a network. We thus only need to obtain Q, C and R . In our method, we assume that flow monitors (such as Netflow) are available in the network, but are expensive to turn on and use. It is thus desirable to use them only when necessary. We assume a system in which Netflow is turned on temporarily for limited amounts of time. (Justification for this approach can be found in [7].) We assume that flow monitors can be turned on for a period of 24 hours during which we capture the actual traffic matrix exactly (except for measurement errors). We use this rich data set to calibrate our model, namely to find C, Q and R . Through these partial TM measurements, we can obtain the temporal and spatial correlations for matrix parameter C .

Given these measurements, we thus have N OD flows and a set of T consecutive samples for each of them. As a consequence, the state of the system X_t at time t is represented by a N -dimensional vector, while the noise process W_t is assumed to be a N -dimensional zero mean white gaussian noise. For these steps, we need not assume the noise components are independent. We do our calibration using an *EM algorithm*. The EM algorithm is an iterative scheme for computing the maximum likelihood estimate of the system parameters, i.e. C and Q matrices, given the observed data Y . We refer to set of parameters to estimate as $\theta = \{C, Q\}$. Specifically, each iteration of the EM algorithm consists of an *expectation step* and a *maximization step*. The expectation step evaluates the conditional expectation of the log-likelihood function under the complete data. This can be expressed as:

$$\Gamma(\theta, \theta_l) = E_{\theta_l} \{ \log dP_{\theta} / \Gamma(\theta, \theta_l) \} = E_{\theta_l} \{ \log dP_{\theta} / dP_{\theta_l} | Y \}$$

The maximization step finds $\theta_{l+1} \in \arg \max_{\theta \in \Theta} \Gamma(\theta, \theta_l)$. The expectation and maximization steps are repeated until the convergence of the system parameters $\|\theta_l - \theta_{l-1}\| \leq \epsilon$. Details of the EM techniques can be found in [6].

3.3 Innovation process

It is possible that over time, the underlying traffic model will shift. It may thus become necessary to recalibrate C, Q and R using a new collection of 24 hours of OD flow measurements. We can make use of the innovation process, already inside the kalman filter, to determine if and when such a recalibration is necessary.

One of the kalman filters' direct goals is to estimate each OD pair i at time t , X_t^i . Note however that this can also be used to generate a prediction for each *link*, simply by multiplying the entire traffic matrix estimate by the routing matrix, namely $A_{t+1} \hat{X}_{t+1|t}$. Recall that the **innovation**, η_{t+1} , is defined as the difference between the observation (measurement) Y_{t+1} and its prediction $\hat{Y}_{t+1|t}$ (a link prediction now) made using the information available at time t . It is a measure of the new information provided by adding another measurement in the estimation process. Given that:

$$\hat{Y}_{t+1|t} = E[Y_{t+1}|Y^t] = A_{t+1} \hat{X}_{t+1|t}$$

The innovation η_{t+1} , expressed by $\eta_{t+1} = Y_{t+1} - A_{t+1} \hat{X}_{t+1|t}$, is a zero-mean process, whose variance S_{t+1} can easily derived as

$$S_{t+1} = E[\eta_{t+1} \eta_{t+1}^T] = R_{t+1} + A_{t+1} P_{t+1|t} A_{t+1}^T \quad (7)$$

The innovation process can be shown to be uncorrelated, i.e. $E[\eta_t^T \eta_l] = 0$, $k \neq l$. The dimensions of S_{t+1} are $J \times J$ (recall J is the number of network links). The diagonal terms S_t^{jj} give the error variance for the link j .

The innovation or residual is an important measure of how well an estimator is performing. Under Gaussian hypothesis for the process noise W_t and the measurement noise V_t , we can expect each component of the innovation vector to be inside a confidence interval of size $\pm 2\sqrt{S_t^{jj}}$ with a probability higher than 95%. If any component of the innovation exceeds this threshold too frequently we conclude that the model is no longer accurate and needs to be recalibrated. There is a tradeoff between the accuracy of the estimation and the number of recalibrations. If $\|\eta_t^j\| \geq 2\sqrt{S_t^{jj}}$ during more than 10 consecutive time slots, we trigger a model recalibration procedure.

4 Evaluation

4.1 Measurement Data Used

The data we use for both validation and evaluation comes from Sprint's European IP backbone, a network comprised of 12 PoPs and 27 routers. During three weeks in the summer of 2003, Sprint enabled Netflow on all incoming peering links and all the links going from access routers to backbone routers. This latter set of links captures nearly all customer traffic. Netflow v8 [1] employs periodic sampling in which one sample is collected every

250th packet. Each router ships the collected Netflow statistics to a centralized collection station every 5 minutes. We obtained the routing configuration for matrix A from the Sprint operators.

4.2 Model Validation

We now validate an assumption of the Kalman filter, namely that the noise process W_t follows a gaussian distribution. QQ plots of four of the OD flows obtained by applying the Kalman are plotted in Figure 2. The plots show that each of these distributions closely follows a Gaussian distribution. In addition, we also checked this assumption by applying the Lillie test which is a robust version of the well known Kolmogoroff-Smirnov goodness-of-fit test. The application of this test to the 132 OD flows, leads to only 3 Gaussian fitting rejections.¹ We conclude that this hypothesis of the Kalman model is valid.

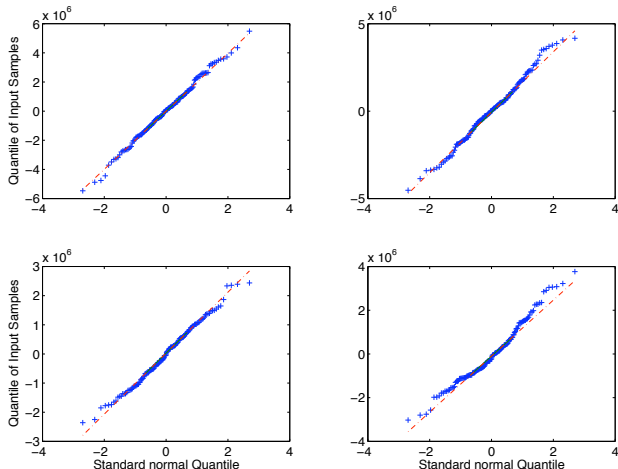


Figure 2: Quantile-Quantile plot of 4 innovation process.

4.3 Traffic Matrix Tracking

We now assess the ability of our model to track the behavior of OD flows over time. Not only do we need to estimate the OD flows, but our estimates need to adapt as the underlying traffic shifts around. In figure 3 we plot four sample OD flows; for each OD flow the real traffic is in grey, while the estimates are shown in black. The evolution of the traffic and estimates are shown for a 10 day period. In these figures, model calibration is applied only once, at the beginning of the experiment.

We see in OD#1 how well the Kalman tracks this OD flow, properly capturing the diurnal pattern. In OD#5, a good deal of fluctuations occur on days 3, 4 and 5. Our OD flow estimate does not get confused by this, and continues to uncover the proper trend. In OD#9 (bottom right), we see that our estimates successfully track the

¹We have 132 flows with 12 PoPs because we do not include self-loop traffic.

flow as it changes its trend in the latter half (right hand portion). However, the adjustment made to accommodate flow #9 has had an impact on flow #14 which is now no longer accurate after day 6. We examined our data and found that this was a case when the correlation between OD flows #9 and #14 changed. To accommodate this model change, a recalibration would have been necessary.

We then reran the same experiment allowing our method to recalibrate whenever our condition for launching a recalibration is triggered. The same OD flows are shown for this scenario in figure 4. We can now see that the model has been well adjusted and all four of these flows can now be properly tracked. This illustrates the adaptability of our method because OD#9 undergoes a large change; its mean trend more than doubles.

4.4 Errors

The error metric we use to illustrate the accuracy of our estimates, is the temporal relative L2-Norm (as in [7]):

$$RelL2_T(t) = \frac{\sqrt{\sum_{n=1}^N (x_t(n) - \hat{x}_t(n))^2}}{\sqrt{\sum_{n=1}^N x_t(n)^2}}$$

This yields one error measure for each moment in time, and each error gives a summary over all N OD flows. This is called a temporal error as it allows one to see how a summary error statistic evolves over time. Figure 5 shows the evolution of this error over time. As stated in [8] a target error rate of 10% has been loosely articulated for traffic matrix estimation tools. Here the error across all the OD pairs over the 10 days period is only 7.45%. As we can see, the error is pretty stable around this value for the first 8 days. Clearly something unusual happens on the ninth day. This kind of behavior, that is anomalous to the model, clearly shows that there is a need for a mechanism to detect model divergence.

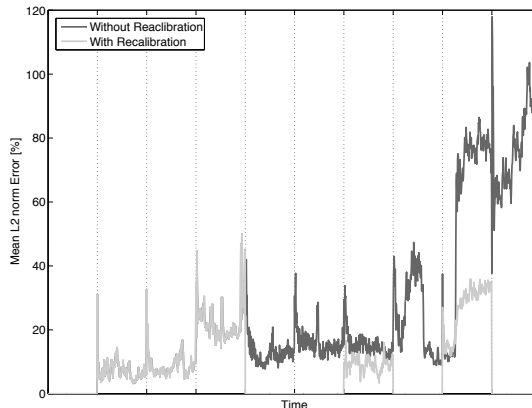


Figure 5: Relative Norm-2 error in the estimation of *all* the OD pairs over a 10 days period.

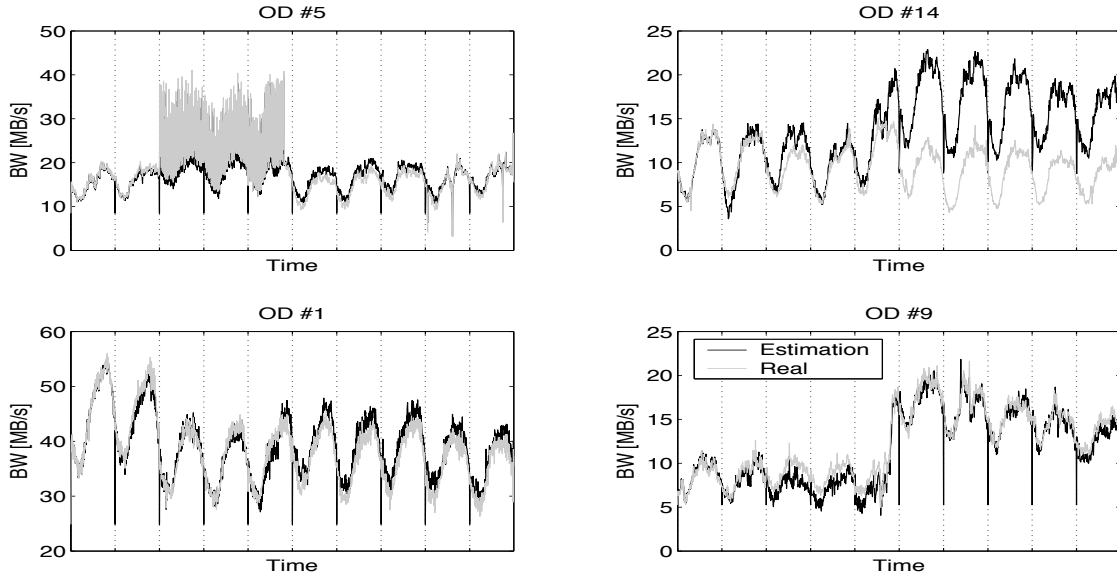


Figure 3: Real (light grey) and inferred OD (dark grey) flows obtained using Kalman Filter. The calibration is applied only once at the beginning of the experiment

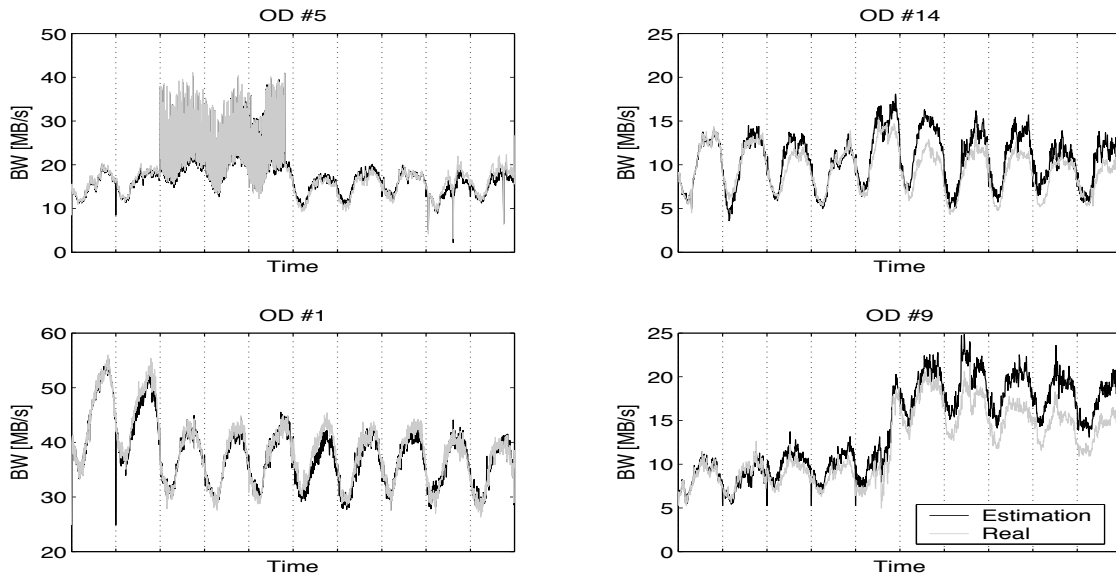


Figure 4: Real (light grey) and inferred OD (dark grey) flows obtained using Kalman Filter. The calibration is applied multiple times over the 10 days when a change in the link count behavior is observed. The period where the light grey line is equal to zero represents the recalibration days.

In Figure 6 we plot the CDF of the relative L2-Norm error over the 10 days. Using recalibration, 50% of OD flows are tracked with a relative L2-norm error less than 10%, and 80% can be tracked with errors under 20%. Recalibration appears to be important for roughly about 30% of the flows. These harder to estimate flows are either the volatile ones or the small ones which are well-known to be difficult to estimate. Sometimes a large error occurs due to a spike in the OD flows. Whenever there is a traffic spike, the filter corrects its estimate in

the following step thus introducing a lag of one time slot in the prediction. (Due to lack of space we do not present an analysis of the delay introduced by the filter.) Note that while this will show up in our error metric, a lag of one is pretty good for an estimation procedure.

In Figure 7 we show the innovation process in gray for 2 typical links, and the 95% error bounds in black. This is done for the case when recalibration occurs as needed. We see in link #3 that the innovation process fluctuates

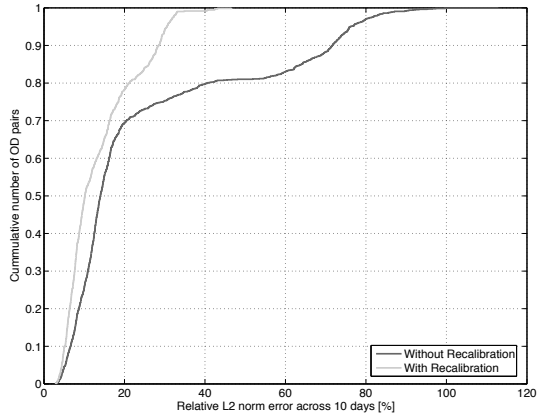


Figure 6: CDF of the Relative Norm-2 of *all* the OD pairs.

most of the time in the confidence bounds and remains largely a zero mean process. For link #6, we see the error variance adjusting as the OD flows on the link vary.

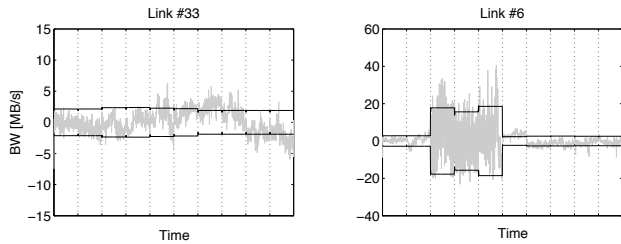


Figure 7: Innovation process of four typical links over a 10 day period. The calibration is done the first day and whenever a divergence occurs.

5 Conclusions

In this paper we proposed the use of Kalman filtering to track the behavior of a set of OD flows. By using Kalman filtering, we can go beyond traditional traffic matrix estimation because we can not only estimate the OD flows, but also predict them and provide confidence intervals for both our estimates and our predictions. The combination of predictions, confidence bounds and estimates that are continuously updated can form the basis of monitoring systems that have the potential to be very useful for anomaly detection tasks. We use data collected from a Tier-1 backbone, to validate some model assumptions and evaluate the accuracy and adaptability of our method. We illustrated that the model was able to adjust itself to small time scale changes, by using new inputs and adjusting the gain matrix. The model also adapts to longer time changes by detecting the change and triggering a model recalibration procedure.

We wish to point out that our kalman filtering framework

enables a deep analysis of the structure of OD flows in terms of their fundamental frequencies, the global traffic state's eigenvalues, and facilitates perturbation analysis. We intend to explore these in future work.

References

- [1] CISCO. Netflow services and applications, 2002.
- [2] T. Kailath, A. H. Sayed, B. Hassibi, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice Hall, 2000.
- [3] A. Lakhina, M. Crovella, and C. Diot. Diagnosing network-wide traffic anomalies. In *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM Press, 2004.
- [4] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot. Traffic matrix estimation: existing techniques and new directions. In *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 161–174. ACM Press, 2002.
- [5] K. Papagiannaki, N. Taft, and A. Lakhina. A Distributed Approach to Measure IP Traffic Matrices. In *ACM Sigcomm Internet Measurement Conference*, Taormina, ITALY, Oct. 2004.
- [6] R. Shumway and D. Stoffer. Dynamic linear models with switching. *Journal of the the American Statistical Association*, 86, 1991.
- [7] A. Soule, A. Lakhina, N. Taft, K. Papagiannaki, K. Salamatian, A. Nucci, M. Crovella, and C. Diot. Traffic matrices: Balancing measurements, inference and modeling. In *ACM Sigmetrics*, Banff, June 2005.
- [8] A. Soule, A. Nucci, R. Cruz, and E. L. and Nina Taft. How to identify and estimate the largest traffic matrix elements in a dynamic environment. In *ACM Sigmetrics*, New York, June 2004.
- [9] Y. Zhang, M. Roughan, C. Lund, and D. Donoho. An information-theoretic approach to traffic matrix estimation. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 301–312. ACM Press, 2003.