Multivariate Online Anomaly Detection Using Kernel Recursive Least Squares

Tarem Ahmed, Mark Coates and Anukool Lakhina*

tarem.ahmed@mail.mcgill.ca, coates@ece.mcgill.ca, anukool@cs.bu.edu

IEEE Infocom, Anchorage, AK

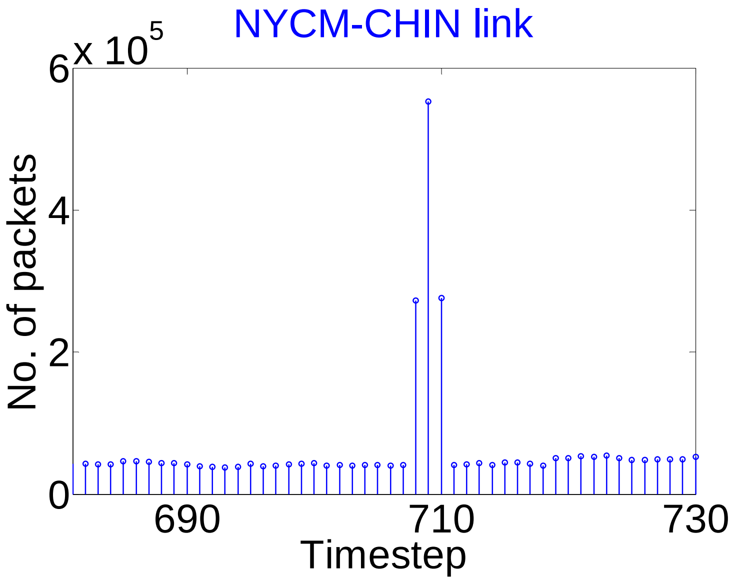
May 06-12, 2007

Research supported by Canadian National Science and Engineering Research Council (NSERC) through the Agile All-Photonics Research Network (AAPN) research network.

*Boston University

Introduction

- What is a network anomaly?
  - Deviation from normal trend of some traffic characteristic
  - Short-lived event
  - Rare event
- May be deliberate or accidental, harmful or innocuous
  - Examples: DoS, viruses, large data transfers, equipment failure
- **Objective**: Autonomously detect anomalies in real-time in multivariate, network-wide data

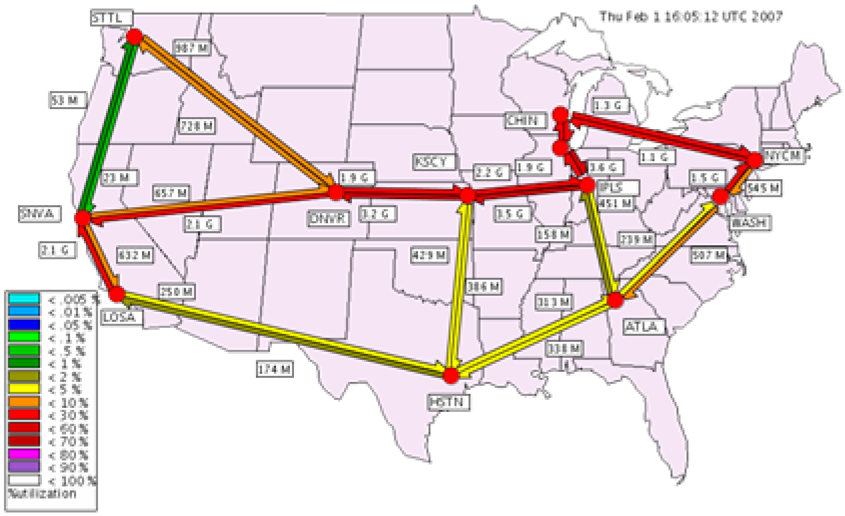


Network Traffic Characteristics

McGill

[Lakhina 05]

- Intrinsic low-dimensionality
- High spatial correlation
- Enables use of Principal Component Analysis (PCA)



Abilene weathermap.
Source: Indiana University

Existing Approach: PCA

McGill

- Determine PCs of traffic flow timeseries
- Assign
  - few highest PCs to normal subspace
  - remaining PCs to residual subspace
- Anomaly flagged when magnitude of projection onto residual subspace > threshold

- Online PCA:
  - project new arrival onto past PCs
- Problems:
  - covariance structure not stationary
  - too sensitive to threshold

Background: The 'Kernel Trick'

- Mapping from *input space* onto *feature space*:

$$\varphi : \mathbf{x}_i \in \mathrm{R}^d \rightarrow \varphi(\mathbf{x}_i) \in H$$

- Kernel computes inner product of feature vectors, without explicit knowledge of the feature vectors:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle$$

- *H* typically *much* higher dimensional than R^d
- Many algorithms only rely on inner products in *H*; hence employ *kernel trick*

Background: Kernel Recursive Least Squares (KRLS)

McGill

- Should be possible to describe *region of normality* in feature space using sparse *dictionary*, $D = \left\{ \tilde{\mathbf{x}}_j \right\}_{j=1}^{M}$

- Feature vector $\varphi\left(\mathbf{x}_t\right)$ is said to be *approximately linearly independent* on $\left\{ \varphi\left(\tilde{\mathbf{x}}_j\right) \right\}_{j=1}^{M}$ if [Engel 04]:

$$\delta_t = \min_{a} \left\| \sum_{j=1}^{m_{t-1}} a_j \phi(\tilde{\mathbf{x}}_j) - \phi(\mathbf{x}_t) \right\|^2 > \nu \qquad (1)$$

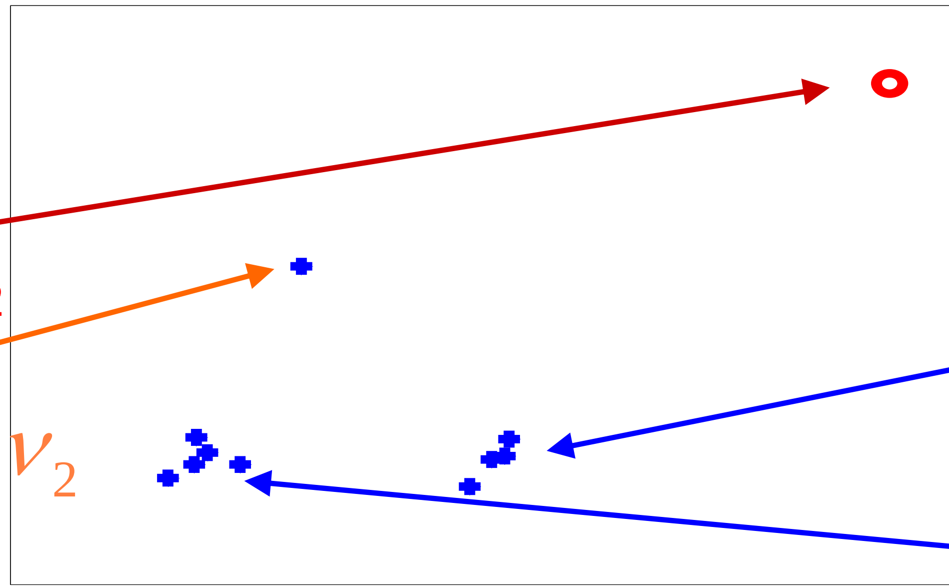
Dictionary approximation

Threshold

- Using (1), recursively construct $D = \left\{ \tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_m \right\}$ such that $\phi\left(D\right)$ *approximately* spans feature space

Kernel-based Online Anomaly Detection (KOAD): Key Idea

McGill



<u>Simplified 2-D depiction</u>

δ_t **: distance between new sample and span of Dictionary [Engel 04],** $\nu_1 < \nu_2$

KOAD: The Algorithm

1. Set thresholds ν_1 , ν_2

2. Evaluate current measurement

3. Process previous **Orange Alarm**

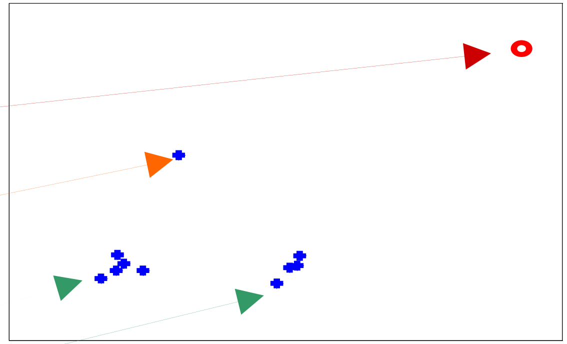
4. Remove any obsolete dictionary element

1. Set thresholds ν_1, ν_2

- ν_2 : upper threshold
  - controls immediate flagging (**Red1 Alarms**) of anomalies
- ν_1 : lower threshold
  - determines dictionary that is built
- Thresholds intertwined
  - together determine dictionary, space of normality
  - should be made adaptive!

2. Evaluate current measurement

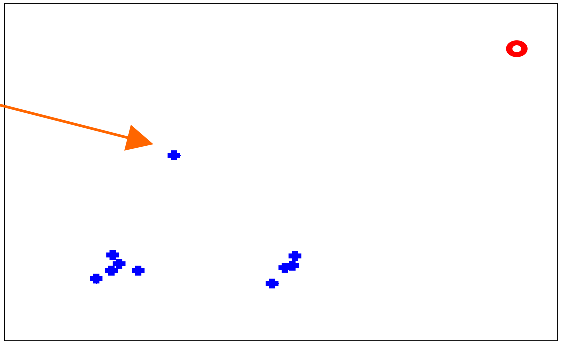
- At timestep t with arriving input vector $\mathbf{x}_t$:
  - Evaluate δ_t according to (1), compare with ν_1 and ν_2 where $\nu_1 < \nu_2$



  - If $\delta_t > \nu_2$, infer $\mathbf{x}_t$ far from normality: **Red1**
  - If $\delta_t > \nu_1$, raise **Orange**, resolve $\boldsymbol{l}$ timesteps later, after "*usefulness*" test
  - If $\delta_t < \nu_1$, infer $\mathbf{x}_t$ close to normality: **Green**

3. Resolving orange alarm

- An **Orange Alarm** may represent
  - a migration or expansion of region of normality: **Green**
  - an isolated incident: **Red2**



- Track contribution of $\mathbf{x}_t$ in explaining l subsequent arrivals
  - kernel of $\mathbf{x}_t$ with $\{\mathbf{x}_i\}_{i=t+1}^{t+l}$
  - perform secondary "***Usefulness Test***"

3. The "Usefulness Test"

- Define *closeness* threshold *d*

- Kernel of $\{\mathbf{x}_i\}_{i=t+1}^{t+l}$ with $\mathbf{x}_t$ *high*
  $\Rightarrow$ $\mathbf{x}_i$ close to $\mathbf{x}_t$

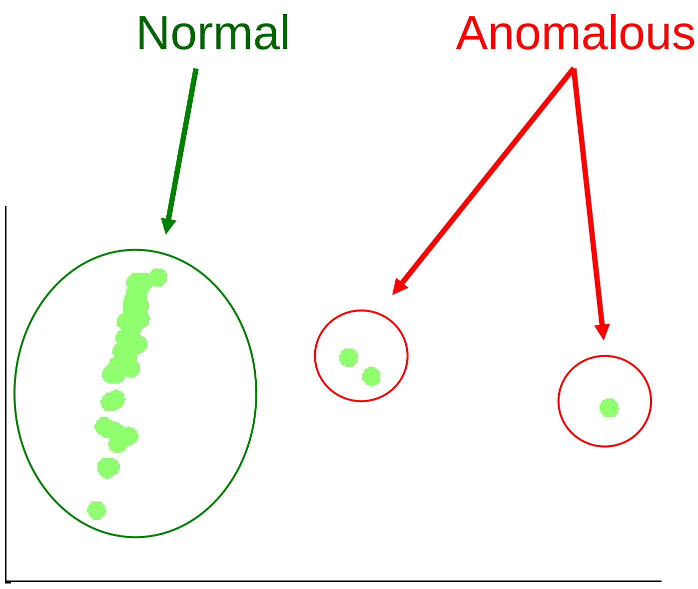
- *Most* (fraction ε) of l subsequent kernels high
  $\Rightarrow$ $\mathbf{x}_t$ *useful* as a ***D*** member

4. Remove any obsolete *D* element

- Test if kernel of arriving $\mathbf{x}_t$ with any ***D*** member remains consistently low
- If so, ***D*** element obsolete, must be deleted
- Dropping involves dimensionality reduction
  - Different from downdating
  - Difficult problem
- KOAD also incorporates exponential forgetting
  - impact of past observations gradually reduced

Relationship with MVS

- Region of normality should correspond to a Minimum Volume Set (MVS)
- One-Class Neighbor Machine (OCNM) for estimating MVS proposed in **[Muñoz 06]**

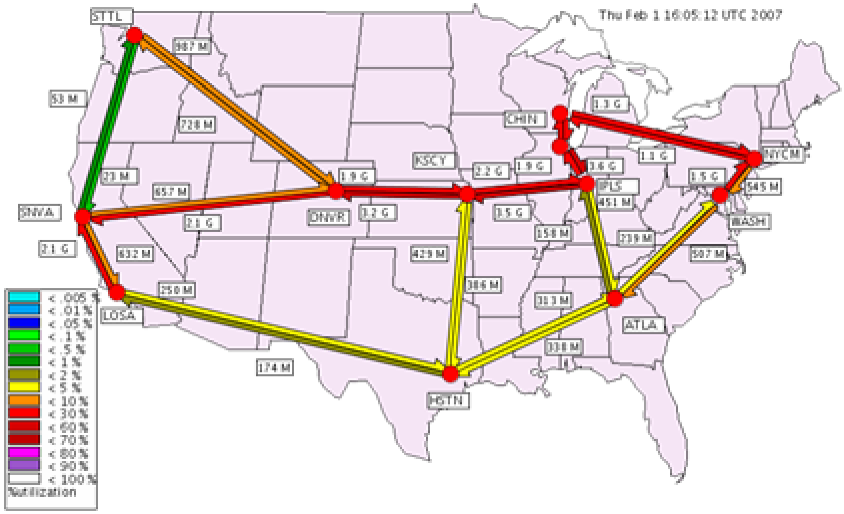


2-D isomap of number of packets in NYCN-CHIN backbone flow

- Requires choice of sparsity measure, *g*. Example: *k*-th nearest-neighbour distance
- Identifies fraction μ inside MVS

Experimental Data

- Stats collected at 11 backbone routers
- IP-space mapped to 121 backbone flows
- Obtain timeseries of backbone flow metrics:
  - number of packets
  - number of bytes
  - number of individual IP flows

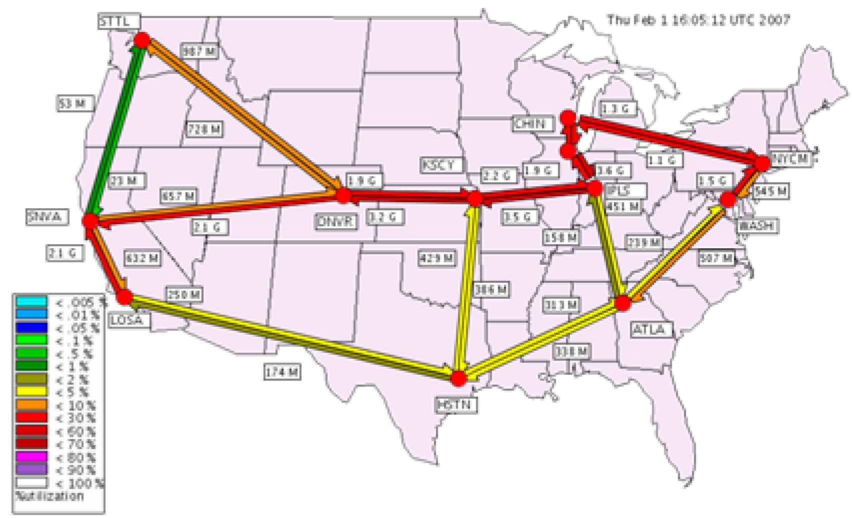


Abilene backbone network

Experimental Setup

McGill

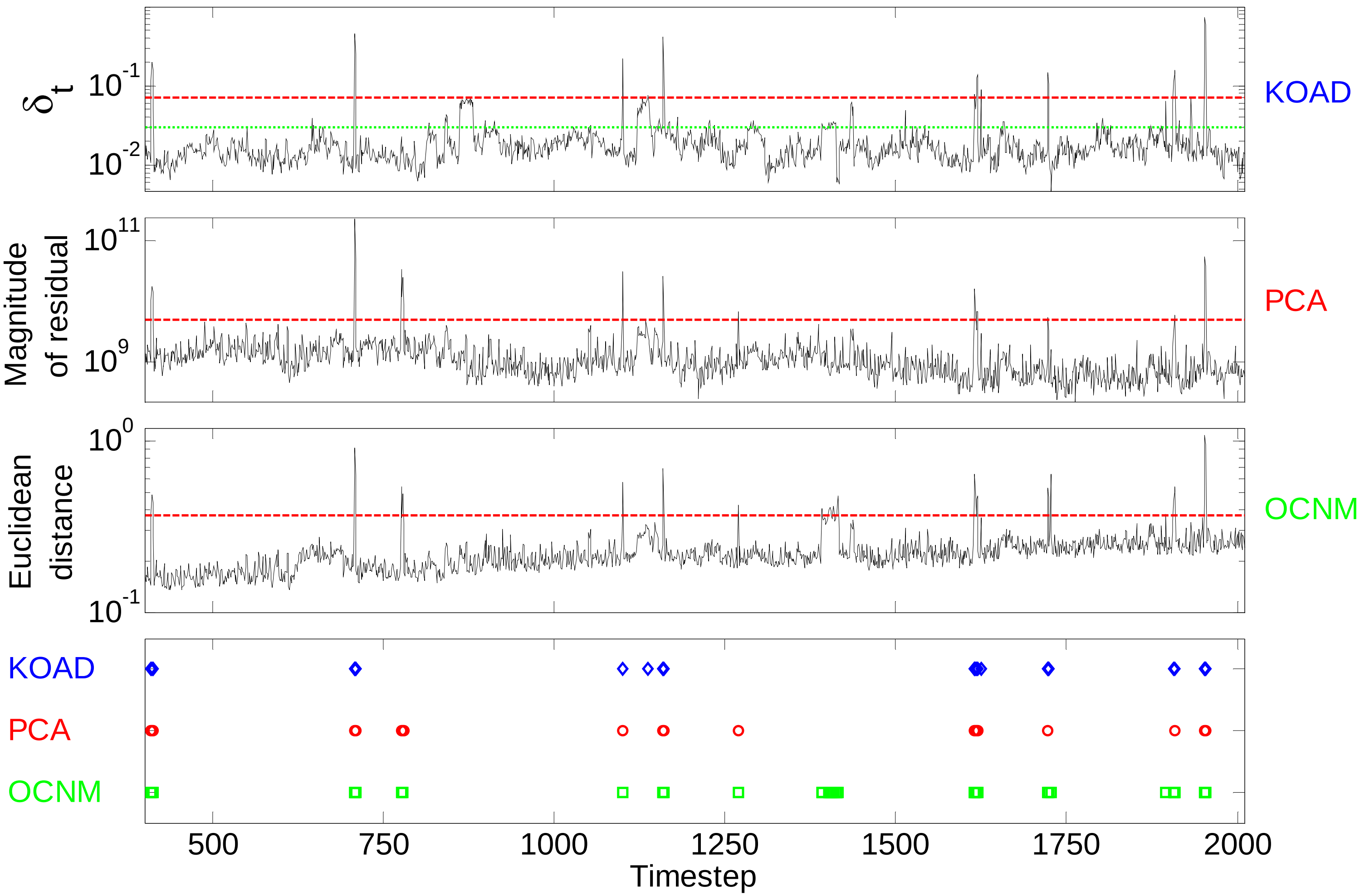
- **KOAD**
  - **$\mathbf{x}_t$** = *flow vector* (number of packets, bytes or individual IP flows, in each backbone flow during interval *t*
  - Linear kernel
- **PCA**
  - 4 PCs to normal subspace
- **OCNM**
  - 2% outliers



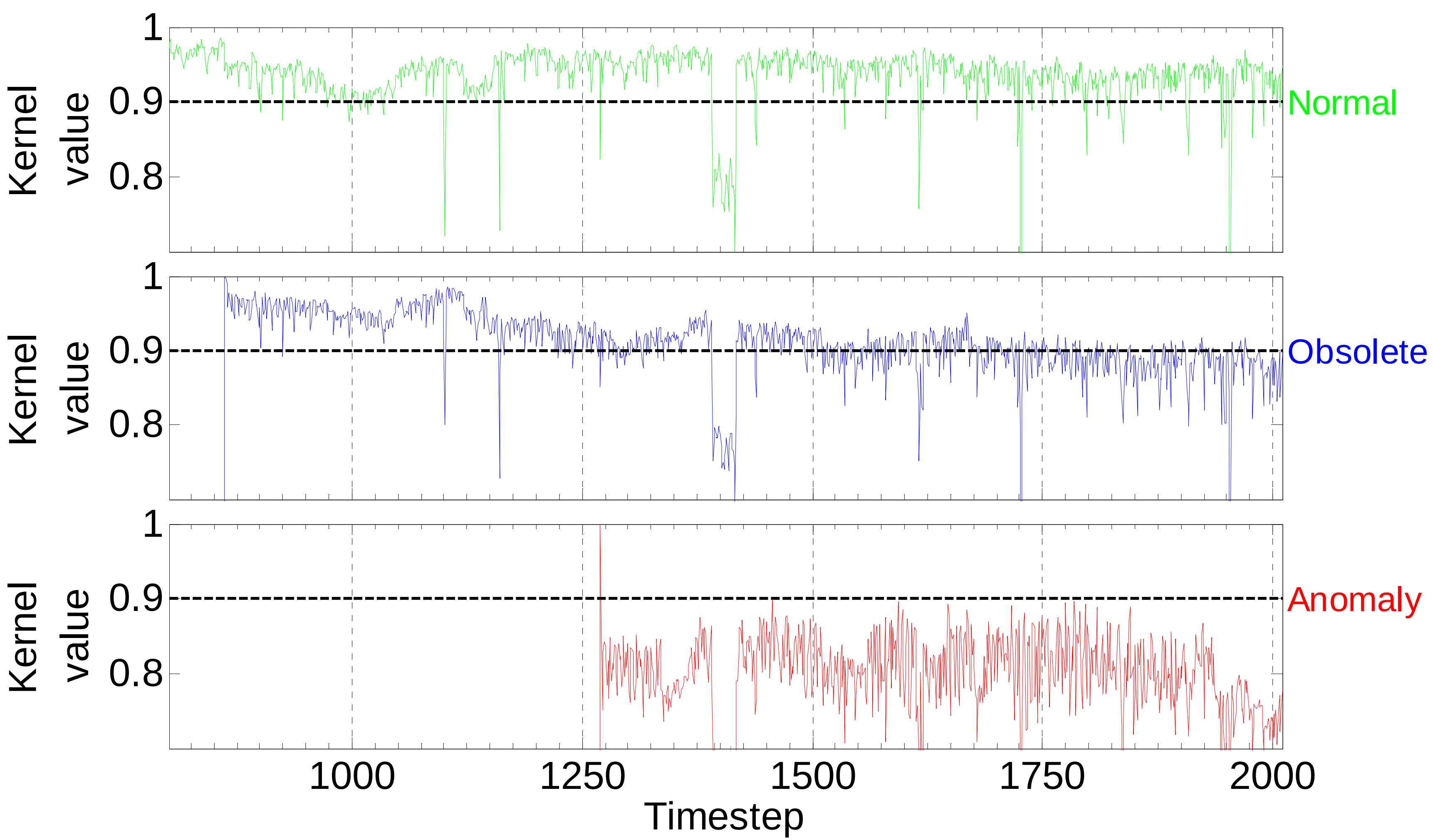
Abilene backbone network

- Code, instructions on replicating our experiments **[WebPage]**

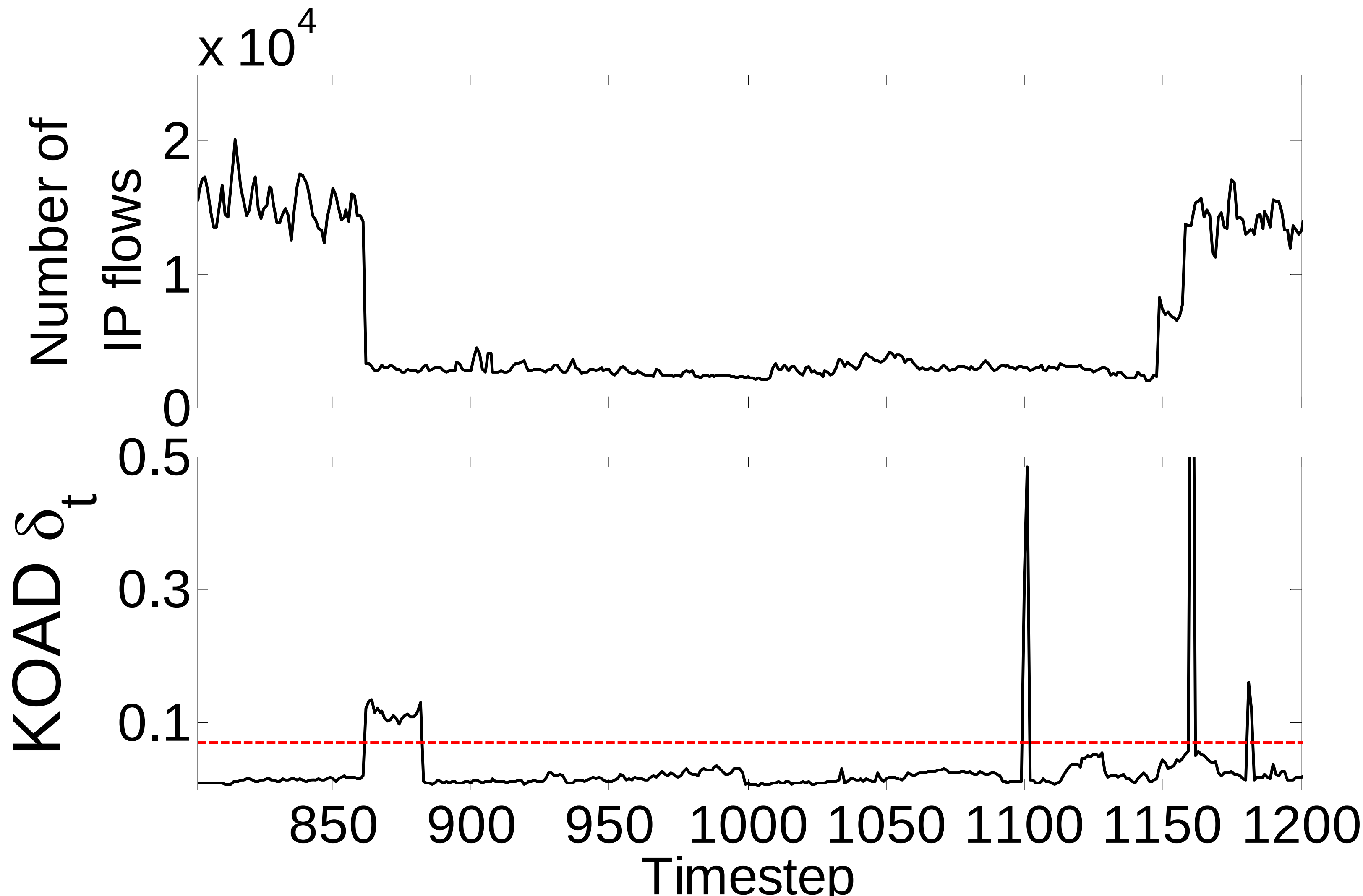
Results: Comparing Algorithms



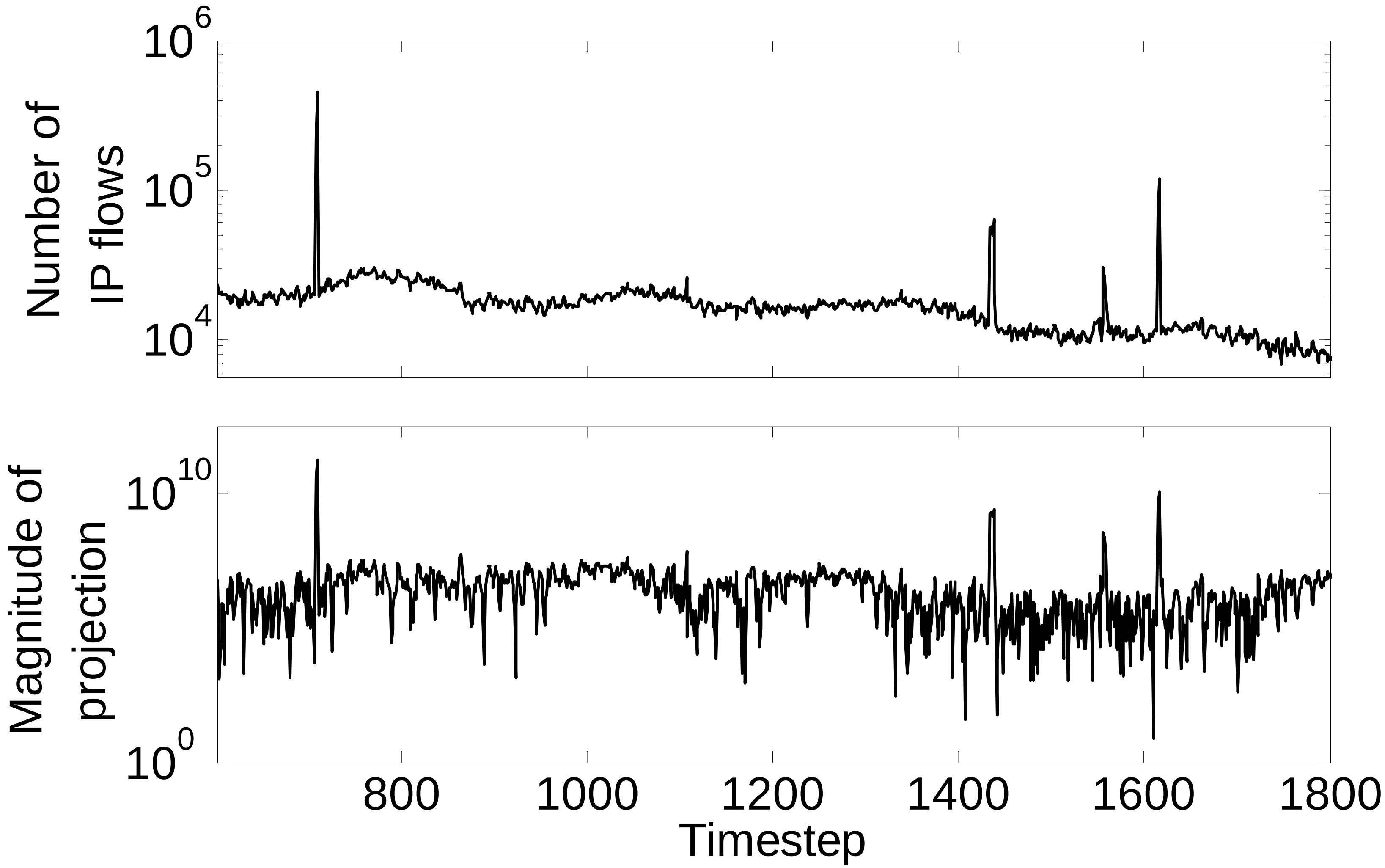
Results: Comparing *D* Elements



Results: Long-lived “Anomalies”



Results: PCA Missed Detections



Conclusions

- Anomaly detection important problem
- Proposed **KOAD** equally effective as **PCA**
- Faster time-to-detection (min vs hrs)

- **KOAD** Complexity
  - $\mathbf{O}(m^2)$ generally
  - $\mathbf{O}(m^3)$ when dropping occurs

- **PCA**
  - $\mathbf{O}(tR^2)$ with R PCs

Work-In-Progress

McGill

- Combinations of **PCA**, **OCNM**, **KOAD**
- Supervised learning, adaptively set parameters: ν_1, ν_2
- Distributed versions, incremental **OCNM**
- Other applications
  - Traffic Incident Detection **[Ahmed 07]**

References

[WebPage]
T. Ahmed and M. Coates. Online sequential diagnosis of network anomalies. Project Description. [Online]. Available: http://www.tsp.ece.mcgill.ca/Networks/projects/projdesc-monit-tarem.html

[Ahmed 07]
T. Ahmed, B. Oreshkin and M. Coates, "Machine learning approaches to network anomaly detection," *Proc. USENIX Workshop on Tackling Computer Systems Problems with Machine Learning Techniques (SysML)*, Cambridge, MA, Apr. 2007.

[Engel 04]
Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least squares algorithm," *IEEE Trans. Signal Proc.*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.

[Lakhina 05]
A. Lakhina, M. Crovella and C. Diot, "Mining anomalies using traffic feature distributions," in *Proc. ACM SIGCOMM*, Philadelphia, PA, Aug. 2005.

[Muñoz 06]
A. Muñoz and J. Moguerza, "Estimation of high-density regions using one-class neighbor machines," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, num 3, pp 476--480, Mar. 2006.