Statistical and Visualization Techniques for Streaming Data

David J. Marchette

david.marchette@navy.mil

Naval Surface Warfare Center

Code B10

Streaming Data

Bill Szewczyk describes streaming data as satisfying the following:

- Order of the data is uncontroled. It isn't necessarily in time order.
- The data generally aren't stored, but collected in "real time".
- There is insufficient storage to store the data.
- Processing time is bounded by acquisition time.
- The data are (extremely) nonstationary.

I would add

- The data are complex. Often multivariate, mixed continuous and discrete, etc.

Network data satisfy all of these conditions. Trade-offs must be made between computational/model complexity and speed.

Streaming Internet Data

A few tasks one might perform on streaming data:

- Model packet interarrival times.
- Models for data transfers, session sizes, etc.
- Passive fingerprinting (used as a check for compliance with accreditation or to detect crafted packets).
- Statistics on backscatter packets to monitor the denial of service attacks on the Internet (Moore et al).
- Model server flows to detect Trojans, Worms, misuse, and to assess network utilization (Brodley and Early).
- Model flows in VPN's to assess "information leak" (Wright).
- Worm/virus activity models.
- Model user behavior (web browsing, etc). Detect interest shifts.

Statistics on Streaming Data

What is needed is methods for estimating statistics on streaming data.

Traditional Statistics starts with $\mathcal{X} = \{x_1, \ldots, x_n\}$ and computes:

- Sample mean, variance: $\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{X})^2$.
- Parametric probability density: $f(x;\hat{\theta}) = f(x;\hat{\theta}(x_1, \ldots, x_n))$.
- Nonparametric PDF: histogram, kernel estimator:
  $\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right)$.

With streaming data we see each observation as it arrives.

< > - +

Recursive Algorithms

We can compute the mean (and higher moments) recursively:

$$\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} x_i = \overline{X}_{n-1} + \frac{1}{n}\left(x_n - \overline{X}_{n-1}\right).$$

$$\sigma_n^2 = \sigma_{n-1}^2 + \frac{1}{n}\left((x_n - \overline{X}_{n-1})^2 - \sigma_{n-1}^2\right).$$

There are multivariate extensions to these, and various simplifying formulations. For streaming data, there is no n.

Exponential Windows

- We could use a sliding window, retaining the newest n points, and update/downdate each observations as it enters/leaves the window.
- Or we can apply an exponentially weighted window:

$$\widetilde{X}_t = \widetilde{X}_{t-1} + \frac{1}{N}\left(x_t - \widetilde{X}_{t-1}\right).$$

Or, more generally

$$\widetilde{X}_t = \widetilde{X}_{t-1} + \theta\left(x_t - \widetilde{X}_{t-1}\right) = (1-\theta)\widetilde{X}_{t-1} + \theta x_t,$$

for $0 < \theta < 1$.

< > - +

Nonparametric Density Estimation

What about density estimation? Note that the histogram is just an average, so can be put in the above framework.

The kernel estimator:

$$\hat{f}_n(x) = \frac{1}{nh_n}\sum_{i=1}^{n} K\left(\frac{x_i - x}{h_n}\right).$$

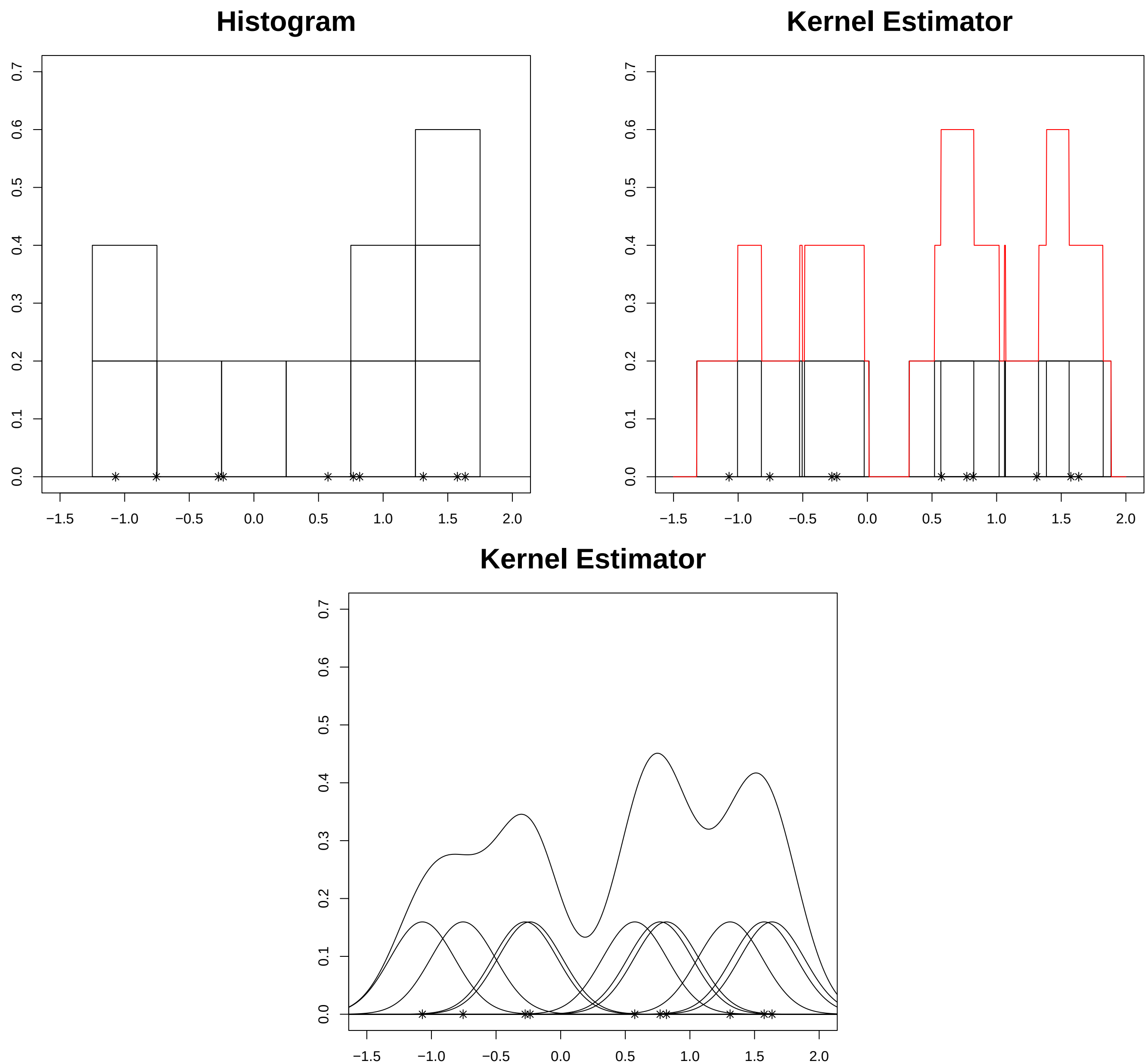
K is usually taken to be normal (Gaussian) density. The kernel estimator can be computed recursively at a preselected x:

$$\tilde{f}_n(x) = \theta\tilde{f}_{n-1}(x) + \frac{1-\theta}{h_n}K\left(\frac{x_n - x}{h_n}\right),$$

There are various other variations, as well as methods for adjusting h_n recursively.

< > - +

Kernel Estimator Picture



< > - +

Semiparametric Density Estimation

The kernel estimator and the normal density are extremes of a general approach called mixture modeling.

$$f(x) = \sum_{i=1}^{m} \pi_i f(x; \theta_i)$$

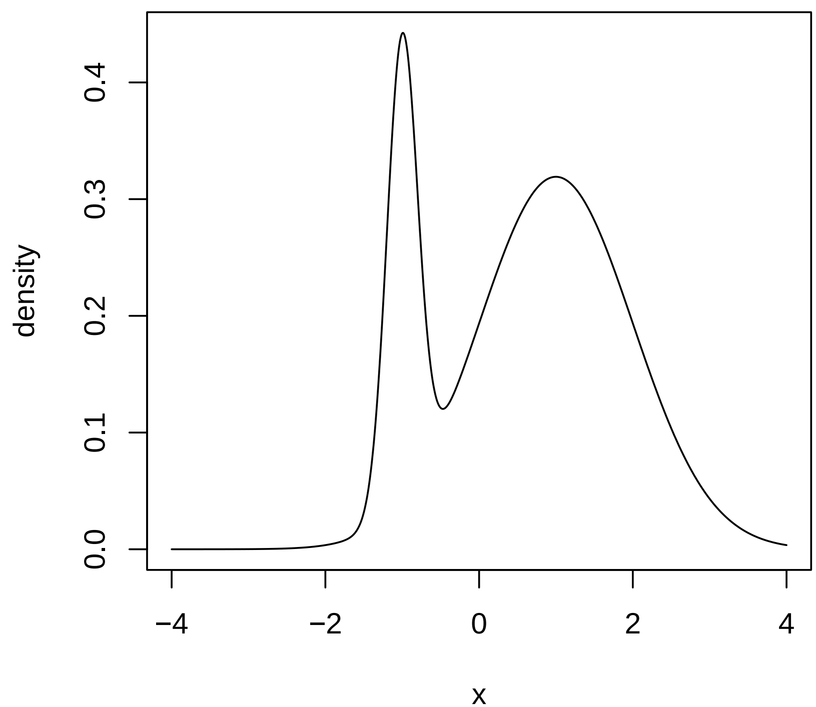
(think of $f(x; \theta_i)$ as being the normal density). This model is fit with an iterative algorithm, which can also be computed recursively, and thus modified as above.

A recursive algorithm called Adaptive Mixtures can be used to choose m (this tends to overfit).

Combining Kernels and Mixtures

The filtered kernel estimator combines these two ideas to allow different bandwidths in different regions:

$$f_t^{fke}(x) = \frac{1}{nh}\sum_{i=1}^{n}\sum_{j=1}^{m}\tau_j^i K\left(\frac{x - x_i}{h\sigma_j}\right).$$



Combining Kernels and Mixtures

A streaming data version of the FKE is:

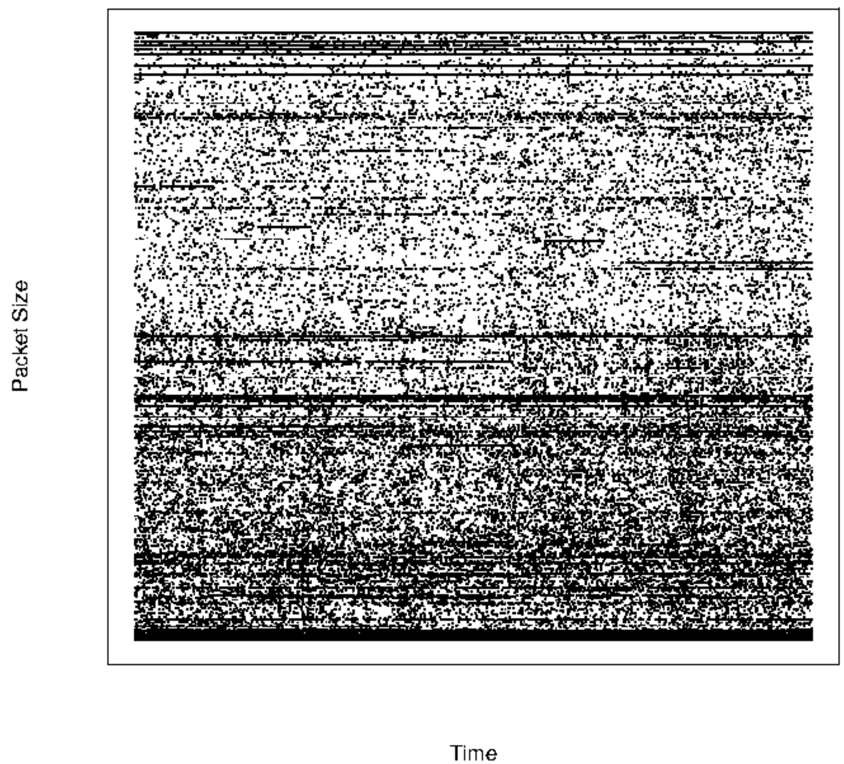
$$f_{t+1}^{fke}(x) = \theta f_t^{fke}(x) + \frac{1-\theta}{h_t} \sum_{j=1}^{m_t} \tau_j^{t+1} K\left(\frac{x - x_{t+1}}{h_t \sigma_j^{t+1}}\right),$$

where we can update the mixture using adaptive mixtures.

Note: We can use the adaptive mixtures estimate to select the “optimal” bandwidth h_t.

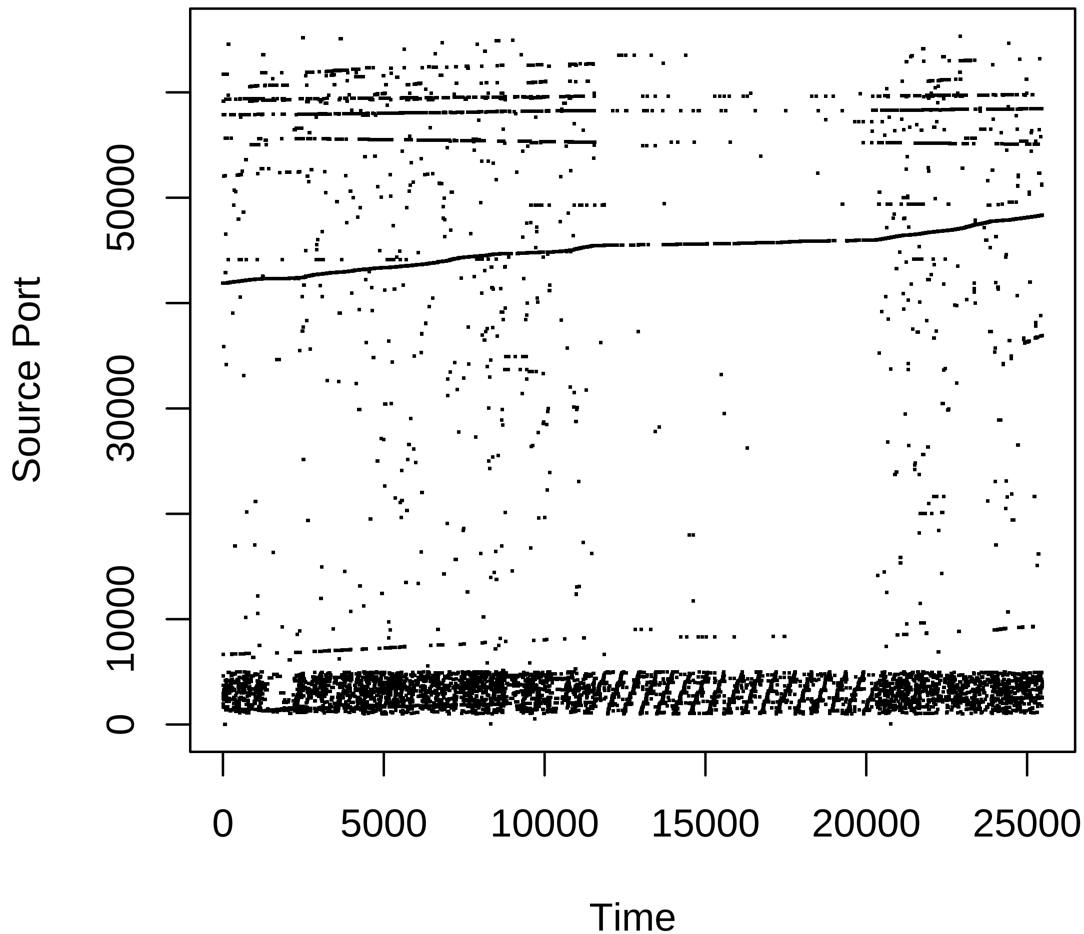
Visualization

- Waterfall plots: plot a variable each time step, shifting the plot in time (up/down or left/right).
- In general, simple is better: I like scatter plots.

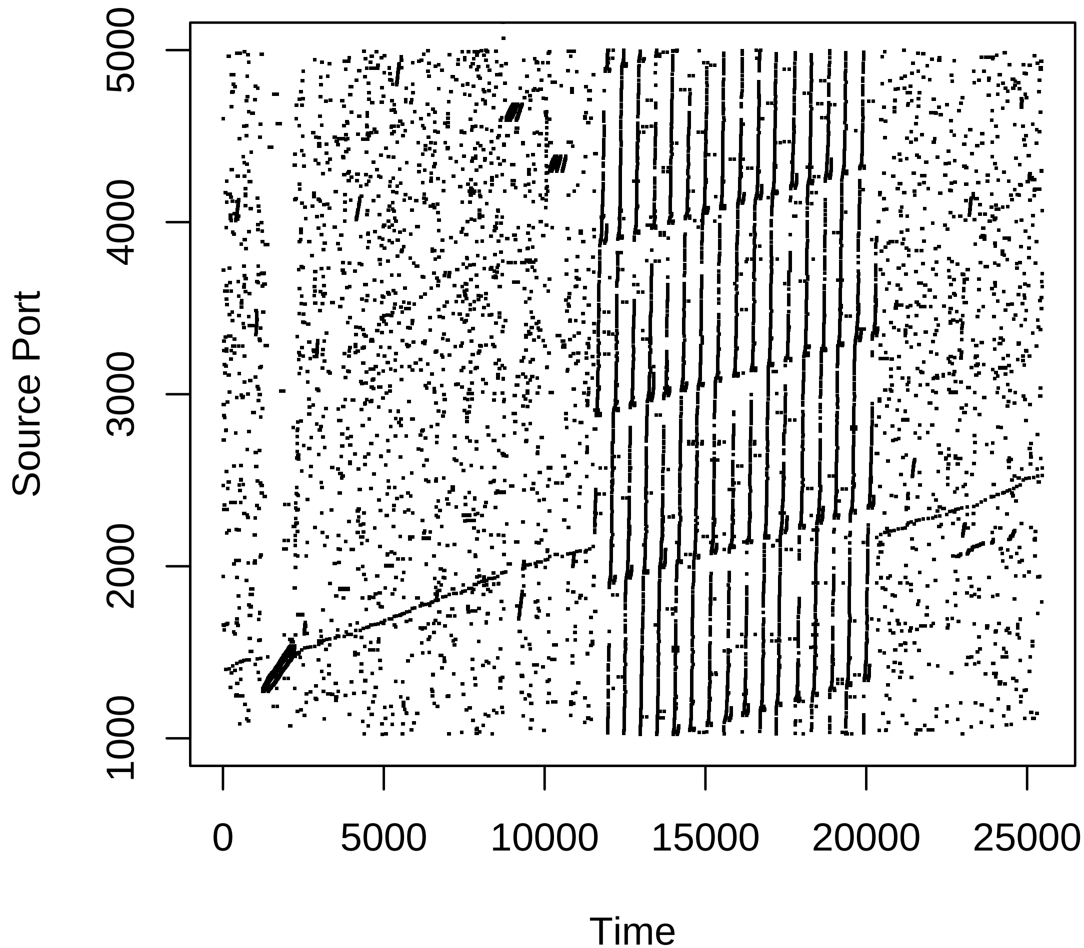


< > - +

Source Ports

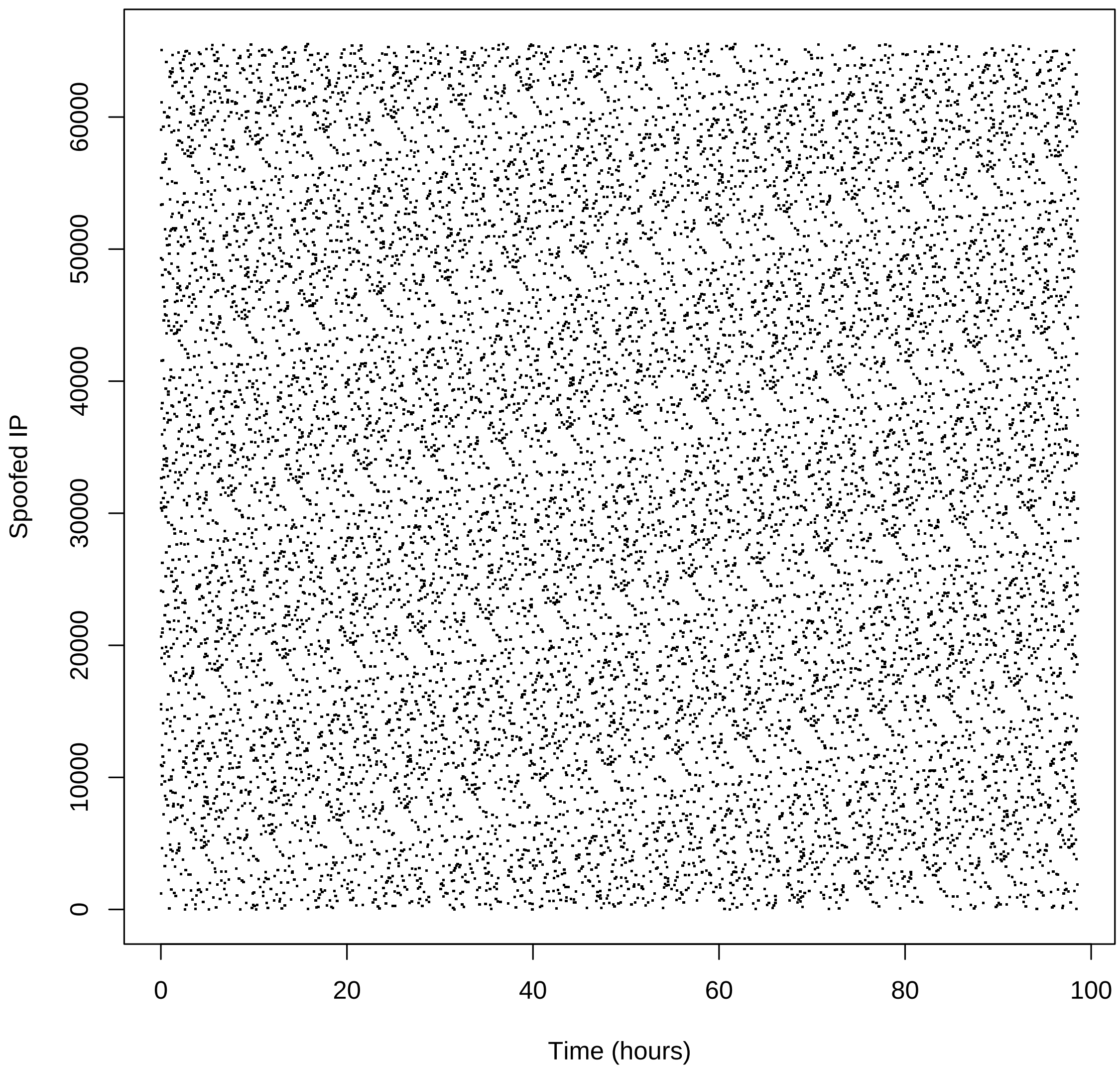


Source Ports Zoomed In



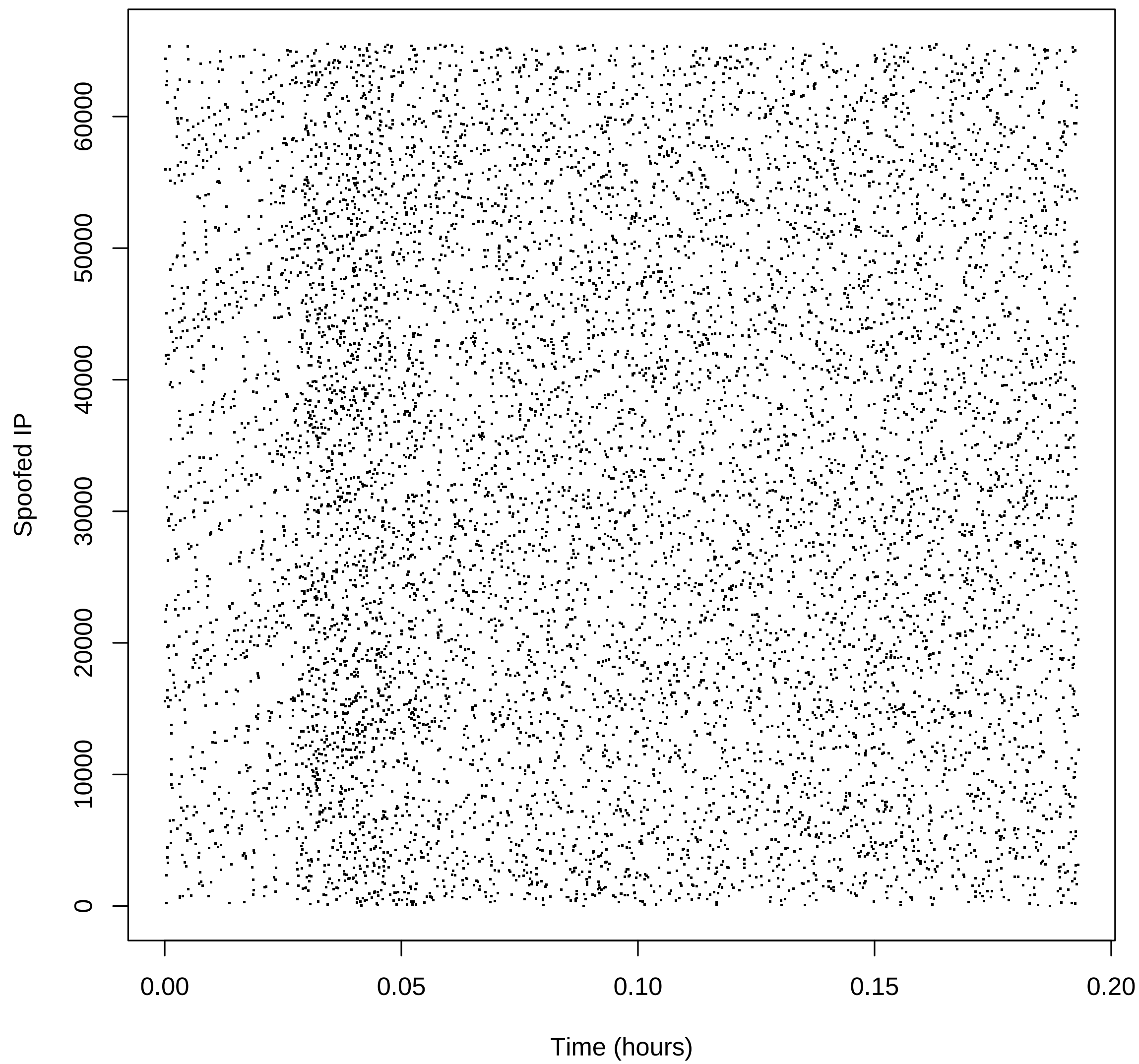
< > - +

Backscatter Structure



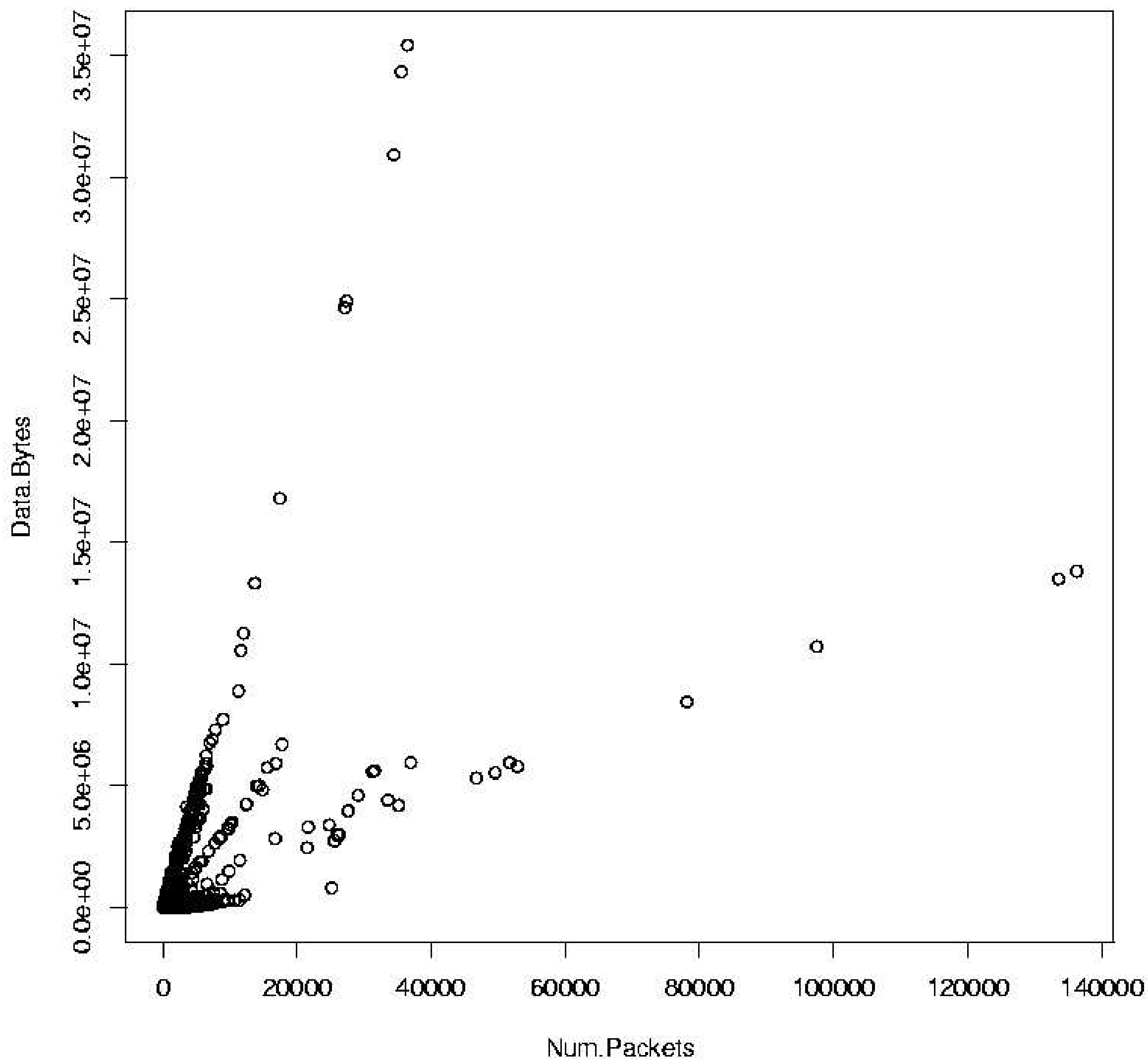
< > - +

More Backscatter Structure



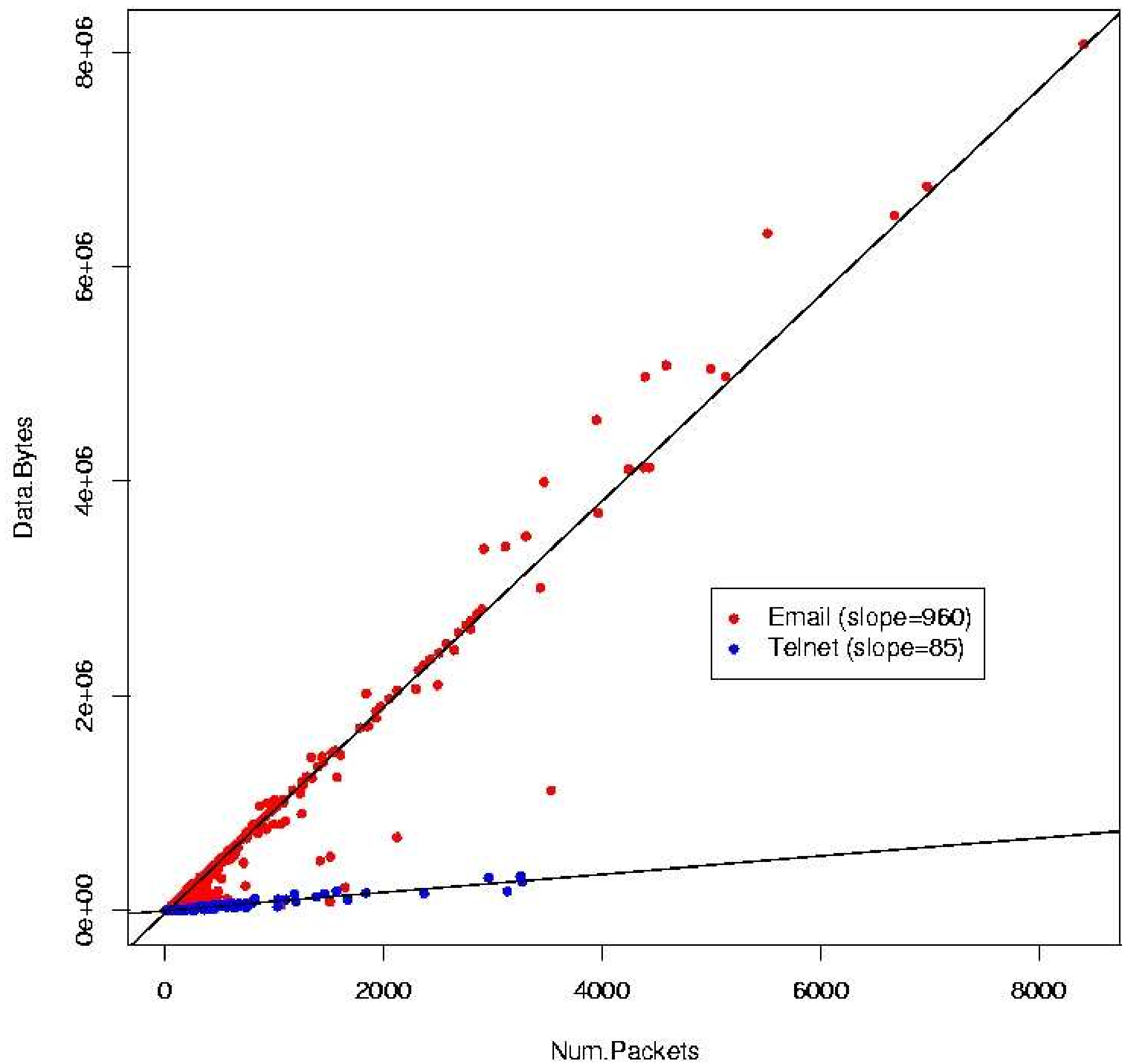
< > - +

Correlation: Data Transfer vs Number Packets



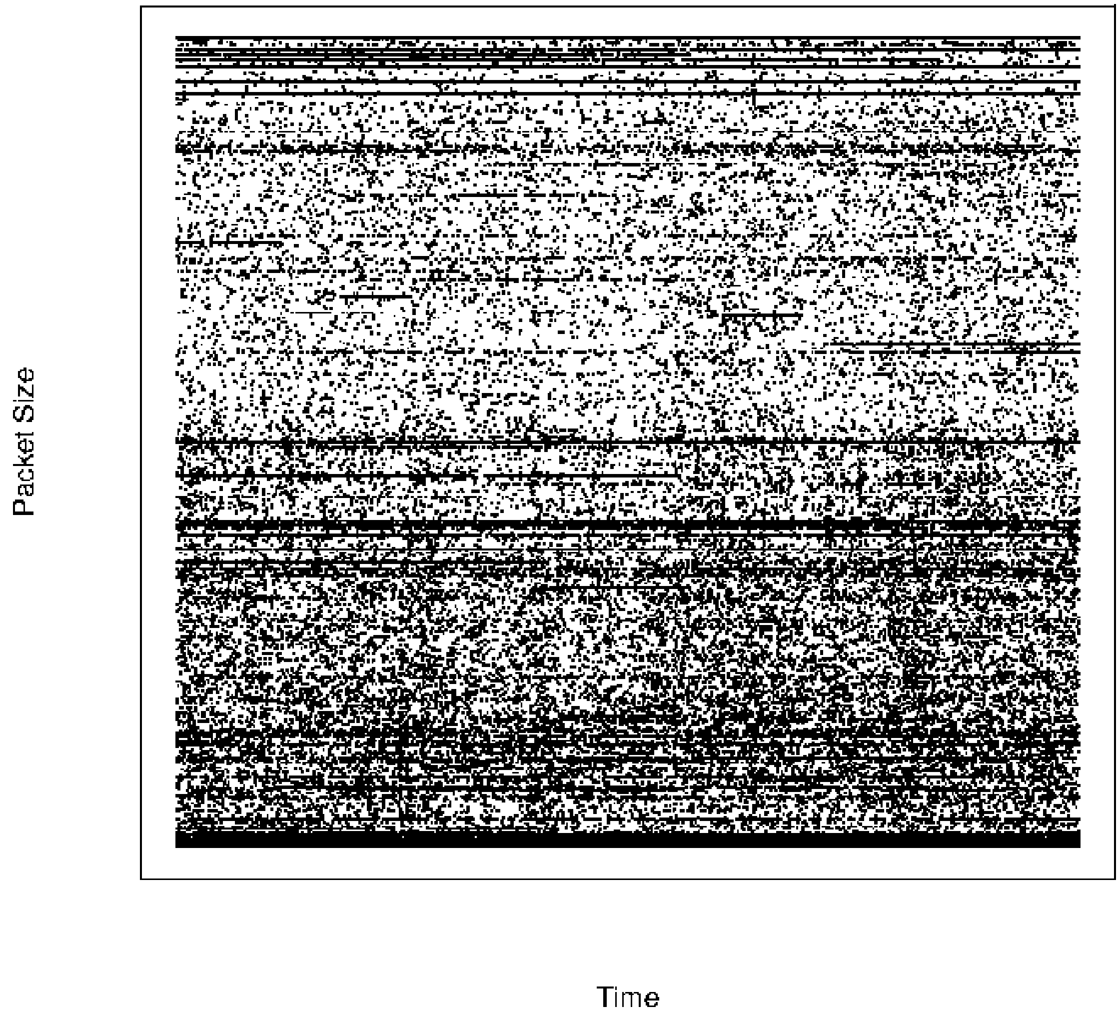
< > - +

Correlation: Packet Size vs Number Packets



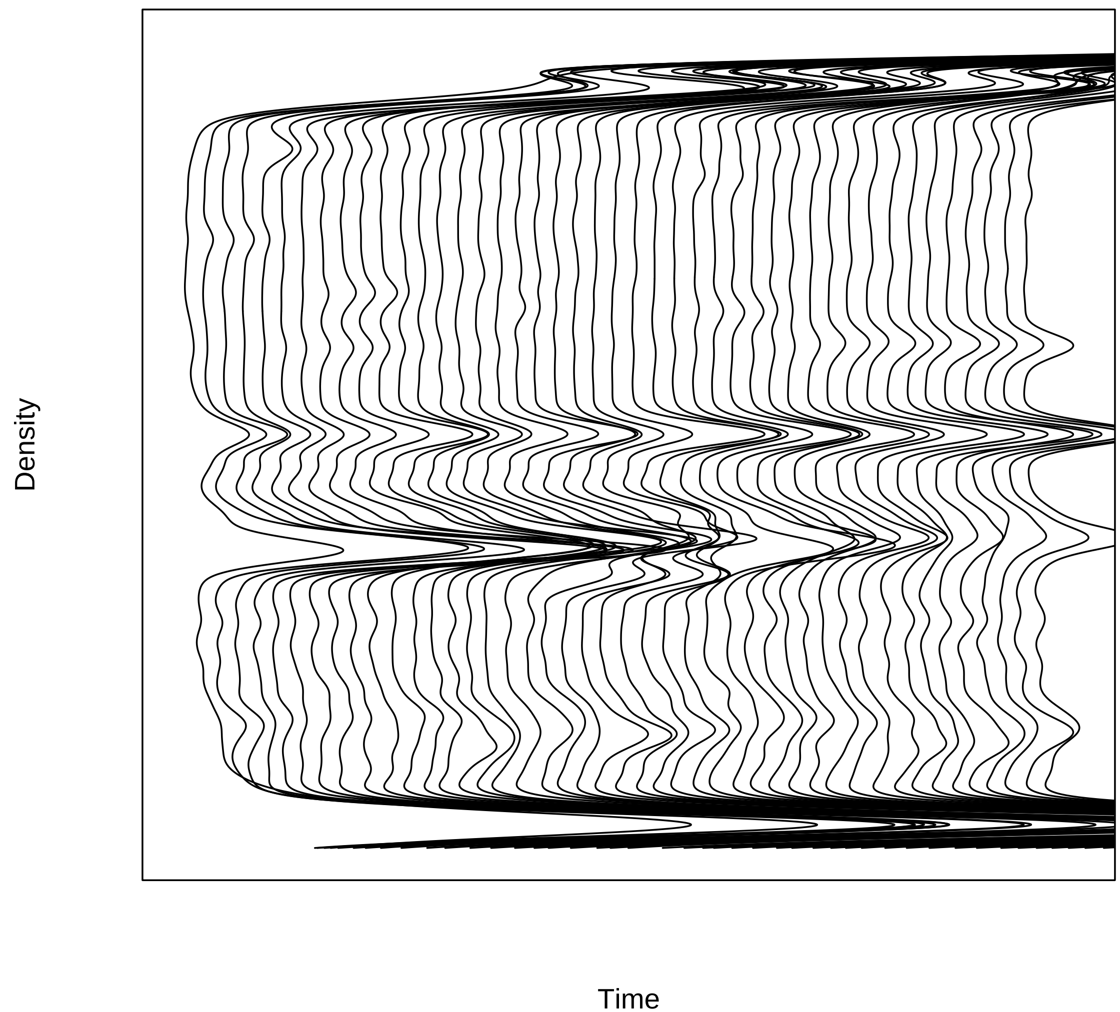
< > - +

Visualization of Densities



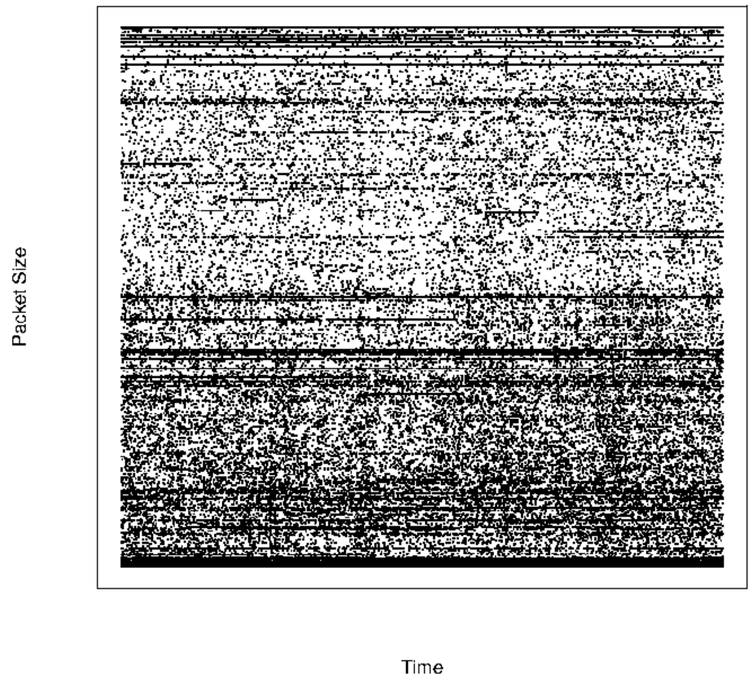
< > - +

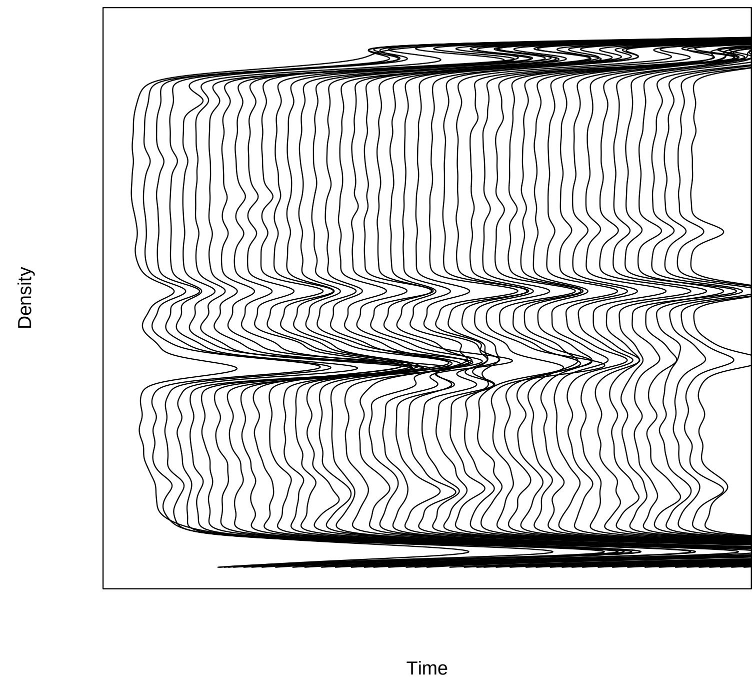
Visualization of Densities



< > - +

Visualization of Densities





< > - +

Dynamic Sensing

- The Conditionality Principle states (essentially) that inference should be made conditional on the experiment run (the current situation).
- Corollary: What you collect should depend on what you have collected (and what inferences you have made).
- Trivial Example: Backscatter packets are (a subset) of those that were not requested as a part of a legitimate session. Only collect packets (SYN/ACK, RST) if the destination IP is not in a session with the source. Similiarly for scans and probes.
- More generally, what you collect may depend on system load, situation assessment (attack or not), or many other ancillary statistics.

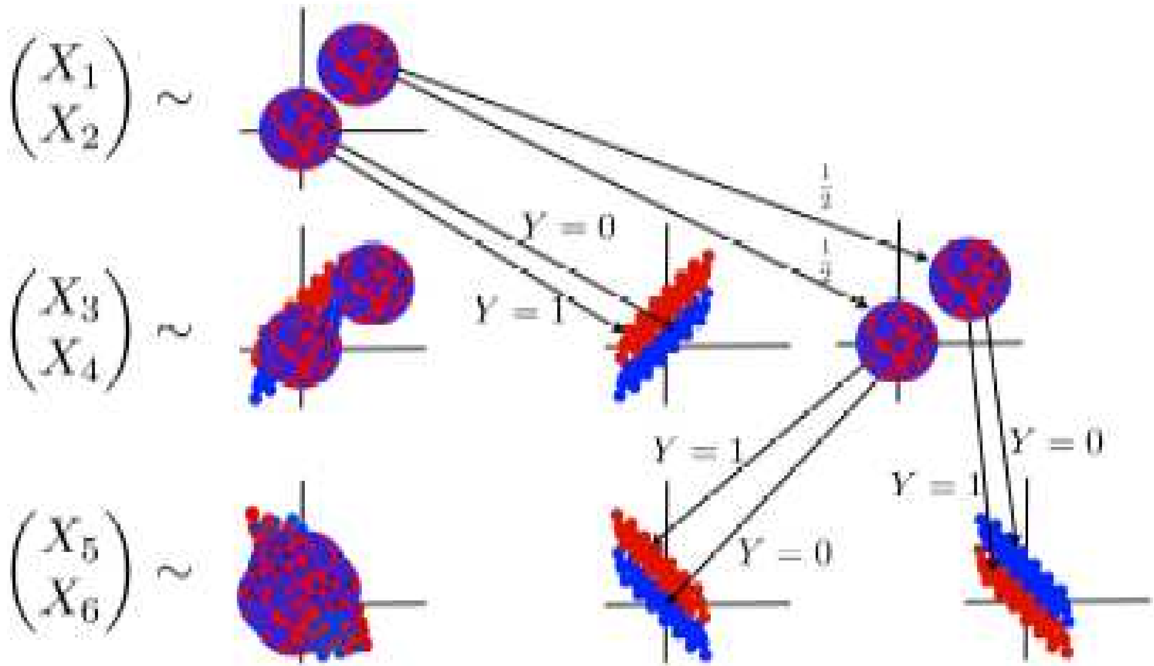
Integrated Sensing and Processing

- You want to determine what to collect next based on the situation.
- One way to acheive this is through ISP Decision Trees.
- Set up: you want to build a classifier that takes input and produces a class label (eg: attack, benign). There are many things you could measure off a packet, the more information you extract, the longer it takes to process it.
- The idea of ISPDT is to group data (cluster) according to measurements (independently of class) and use the groups to determine the next measurement to take.

ISPDT Example

Example

(from Priebe *et al.*, PAMI '04)



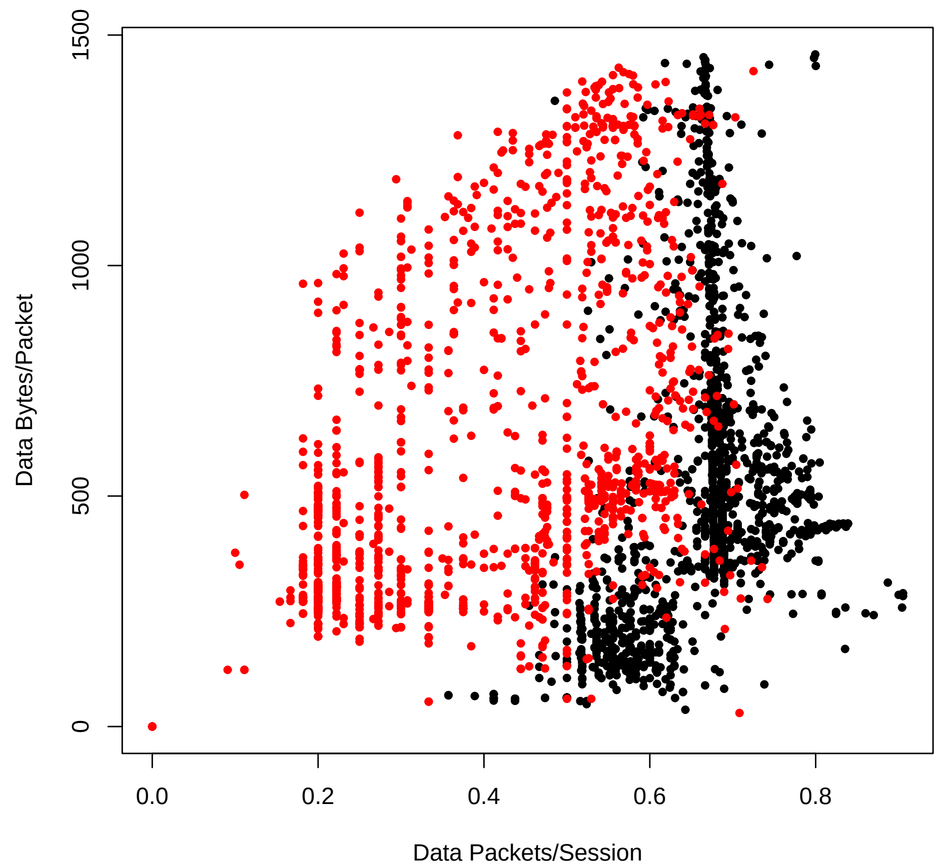
Network Data

- Some of this is obvious. You will collect different statistics for different:
  - protocols
  - applications
  - packet types
- You will also collect different information depending on the
  - purpose you wish to accomplish
  - the load on the network (sensor, analysis station)
  - whether you think you are under attack
  - memory/storage constraints.
- ISPDT is merely a framework within which to think about these issues and expand them to things you haven't considered.

Toy Example

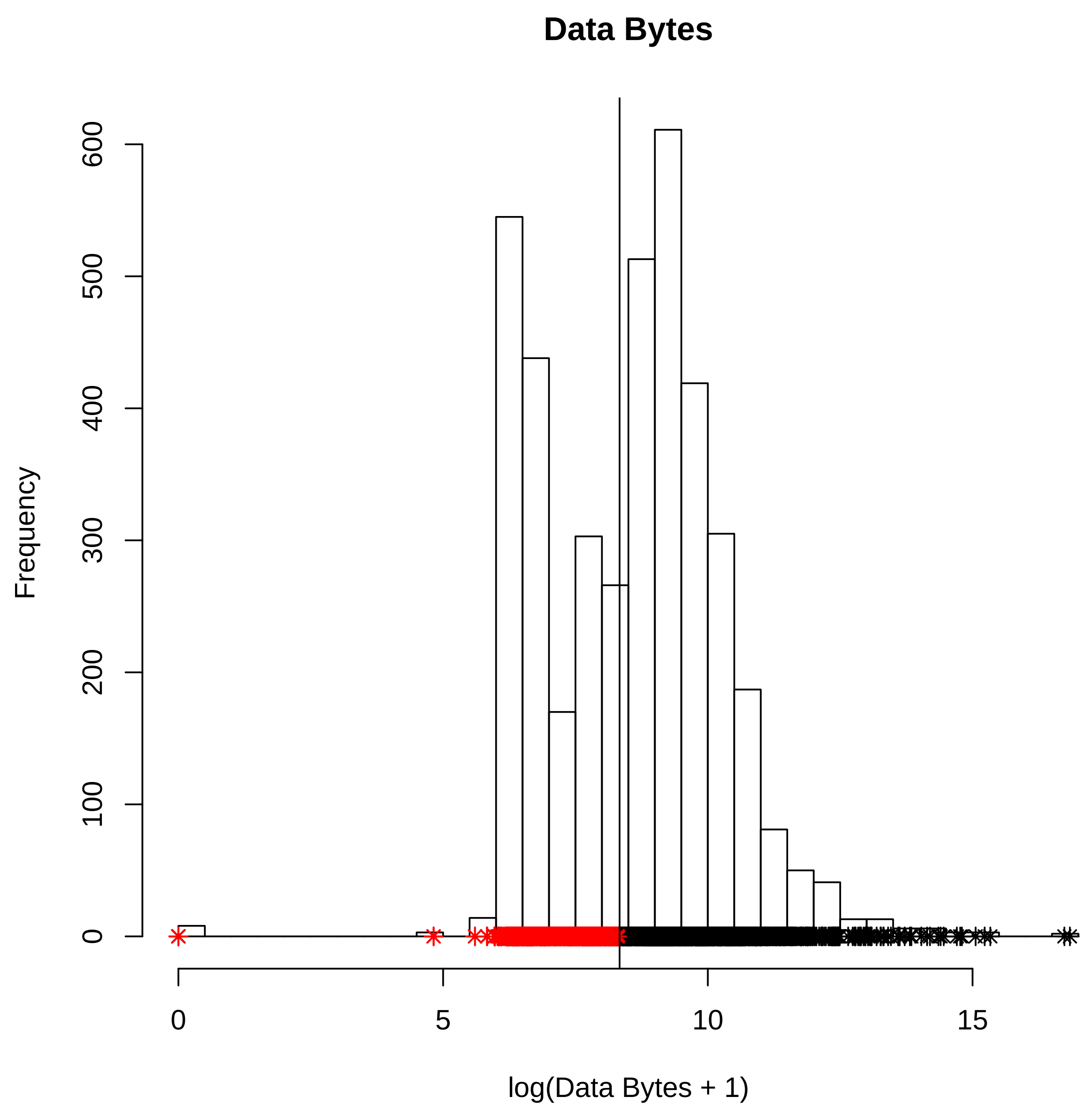
Distinguish between Web and Email sessions using:

- number of packets
- number of data packets
- number of data bytes



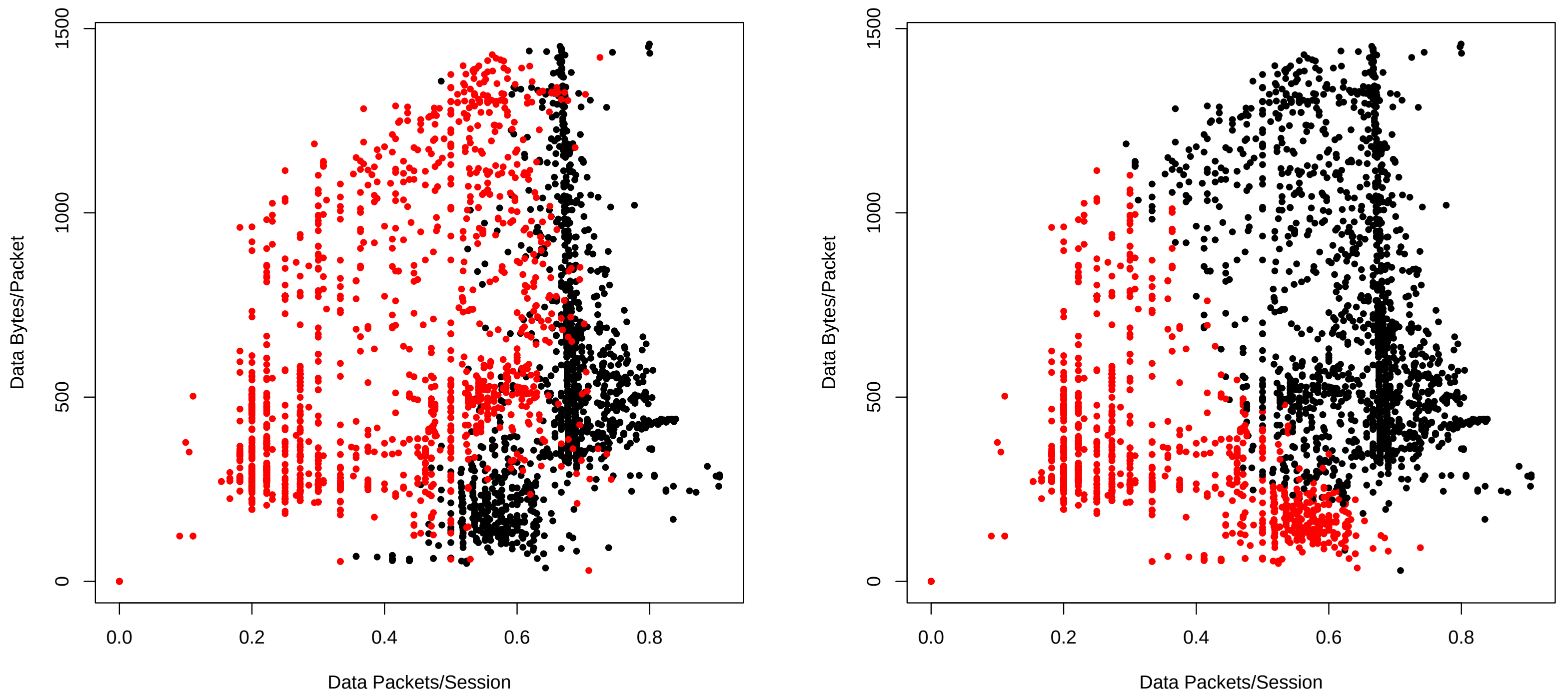
< > - +

Cluster On Data Bytes



< > - +

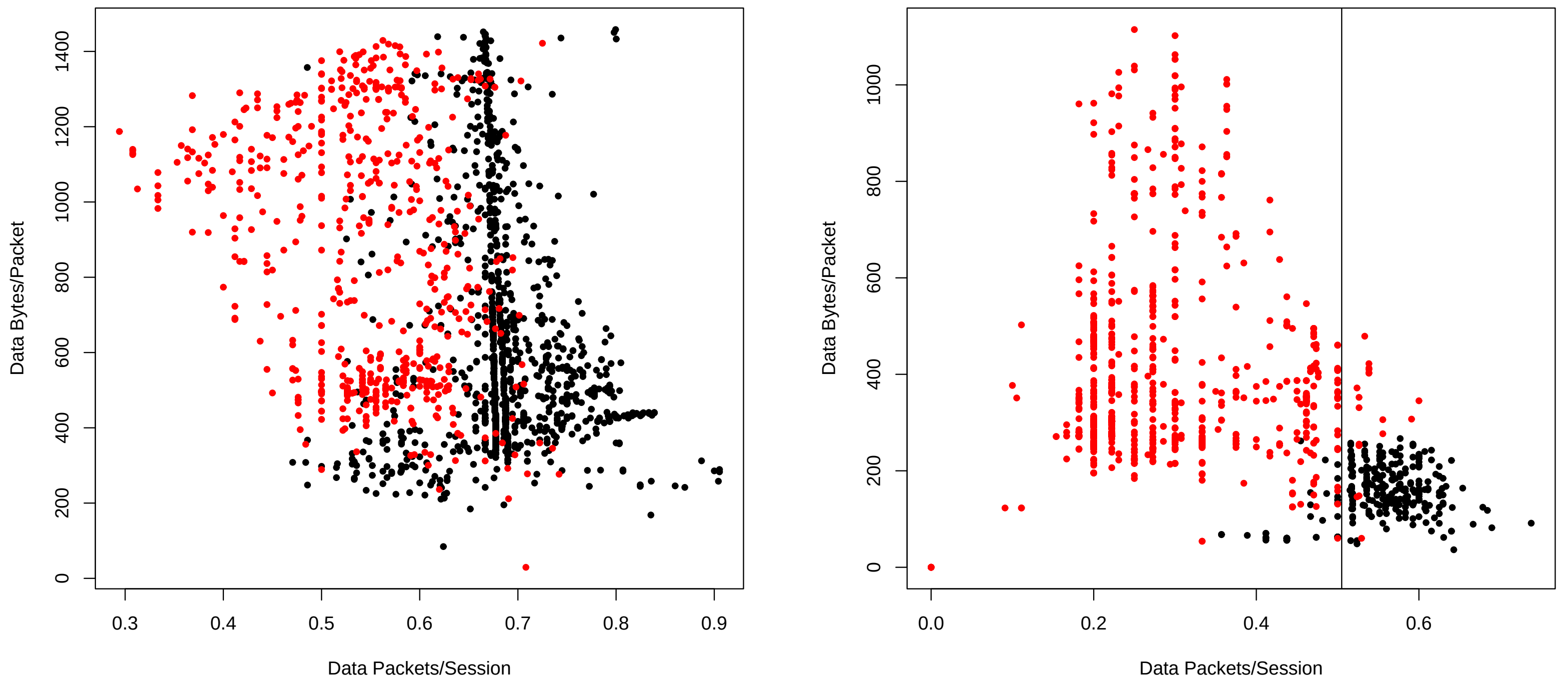
Cluster On Data Bytes



< > - +

ISPDT

Cluster on $log(\text{Data Bytes} + 1)$:



1-Nearest Neighbor on cluster 1, Data Packets/Packet > .5 on cluster 2.

ISPDT Results

Classifier 1: 1-nearest neighbor.

ISPDT: In cluster 1, 1-nearest neighbor, in cluster 2, only compute Data Pakets/Packet and use linear classifier.

Classifier	Error
1-NN	17.7%
ISPDT	13.5%

< > - +

References

J.P. Early and C.E. Brodley, “Behavioral authentication of server flows”, The 19th Annual Computer Security Applications Conference, 2003.

D.J. Marchette, C.E. Priebe, G.W. Rogers and J.L. Solka, “The Filtered Kernel Estimator,” Computational Statistics, Vol. 11, 95–112, 1996.

D. Moore, G.M. Voelker and S. Savage, “Infering Internet Denial-of-Service Activity”, USENIX Security 2001.

C.E. Priebe, “Adaptive mixtures”, Journal of the American Statistical Association, 89, 796–806, 1994.

C.E. Priebe, D.J. Marchette and D.M. Healy, “Integrated Sensing and Processing Decision Trees”, IEEE PAMI, 26, 699–708, 2004.

E.J. Wegman and D.J. Marchette, “On Some Techniques for Streaming Data: A Case Study of Internet Packet Headers”, JCGS, Vol. 12, No. 4, pp. 893–914, 2003.