Dimension Reduction Techniques for Identifying Relevant Data Streams

Chandrika Kamath
Lawrence Livermore National Laboratory
Livermore, CA, USA
kamath2@llnl.gov

2011 DOE Applied Mathematics Program Meeting

October 2011

LLNL-PRES-504391 This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Goal: find useful information in multi-variate data streams

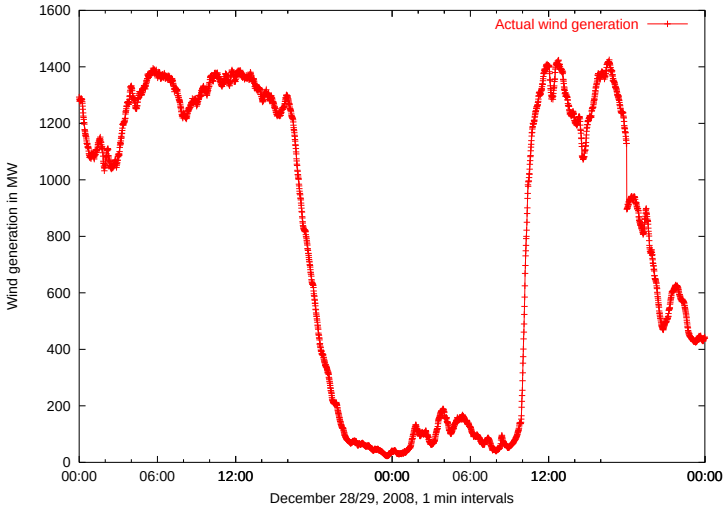
- A data stream is a semi-infinite sequence
- Using only the most recent data, we want to identify
  - Concept drift in real time: for appropriate control of the system
  - Anomalies in real time: for corrective action
  - Periods of interesting behavior in near real time: for rapid analysis

Challenges

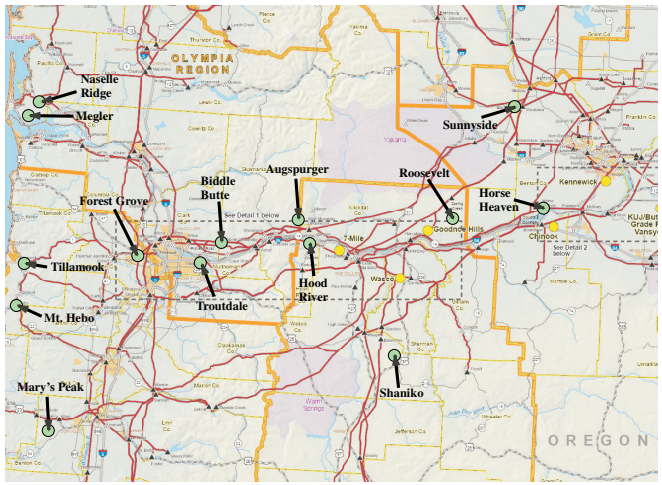
- Data are high-dimensional, heterogeneous, sampled at different rates, of low quality, massive, and with time-varying statistics
- Real-time response is often required
- "Anomalies" and "interesting events" are poorly defined
- Need to minimize false positives

Motivation: incorporating wind energy into the power grid†

As the percentage of wind energy on the grid increases, we need improved forecasts of ramp events, where there is a large change in the generation over a short time.



Ramp events in BPA wind power generation

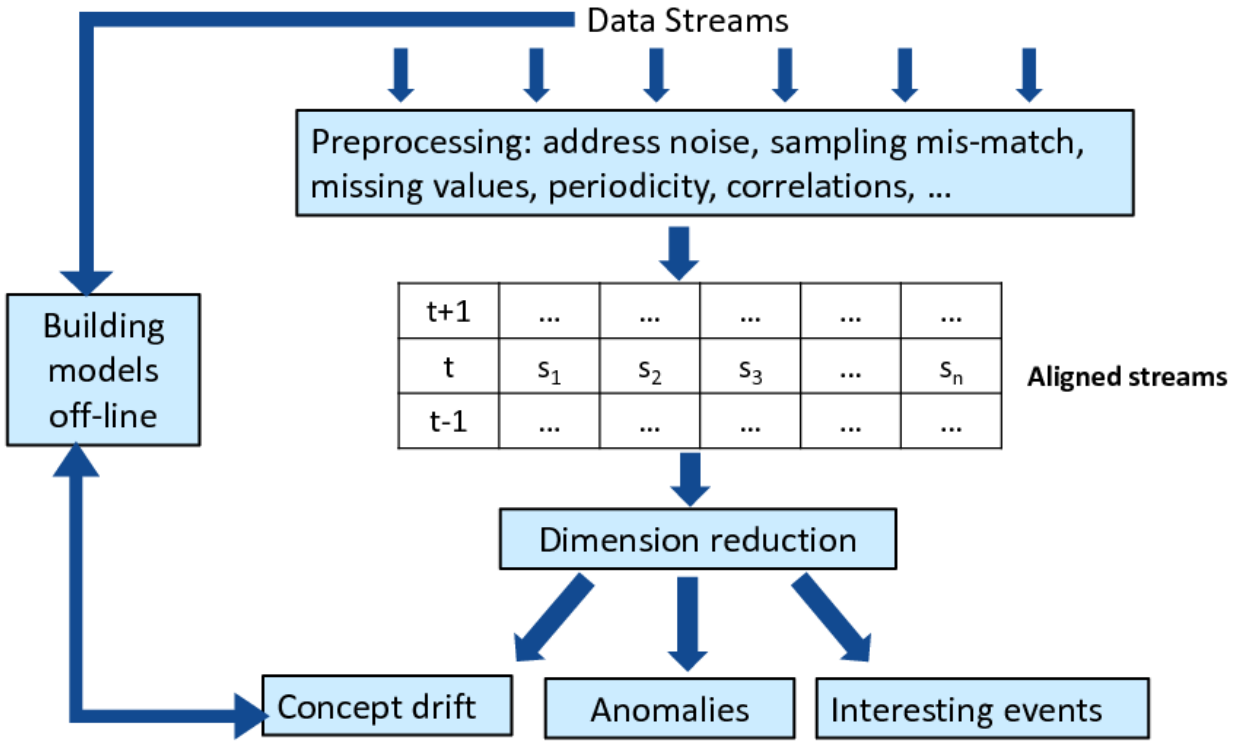


Locations of meteorological towers in Columbia Basin

Can we predict ramp events using data from the meteorological towers?

† WindSENSE for control room integration, https://computation.llnl.gov/casc/StarSapphire/WindSENSE.html

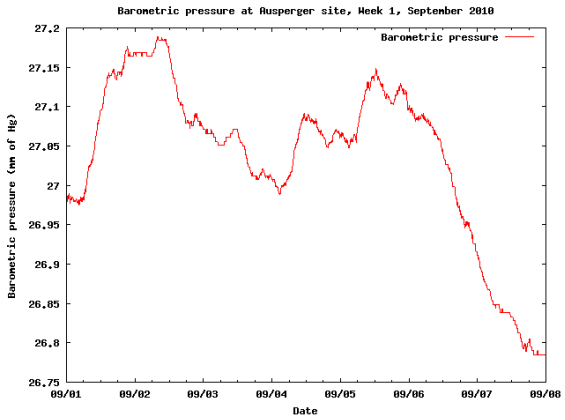
Our end-to-end solution approach

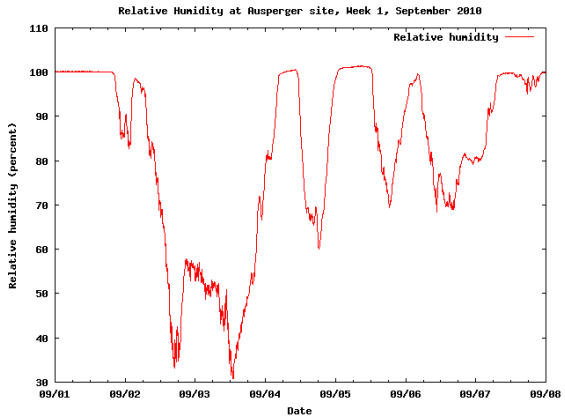


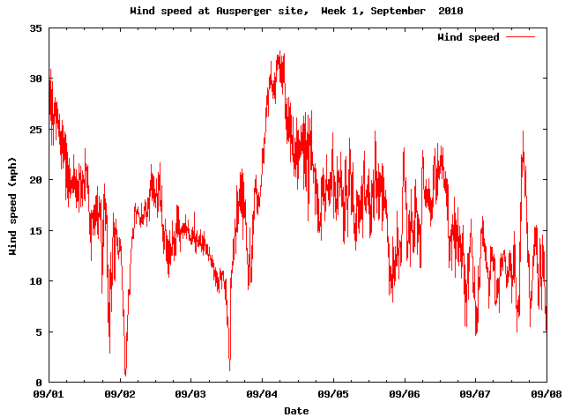
Our recent focus is in three areas

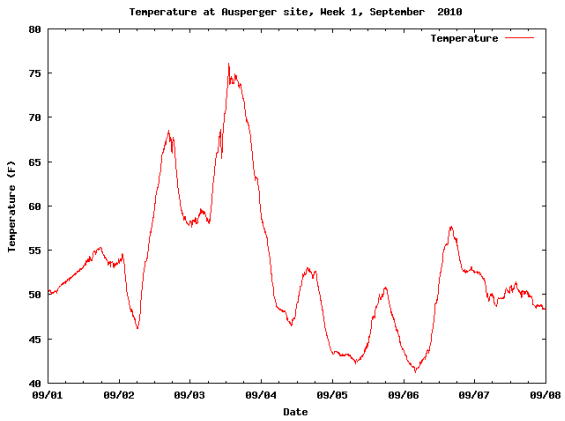
- Adaptive noise reduction
- Fast subspace trackers for dimension reduction
- Effects of spatial distribution of sensors on the predictive models

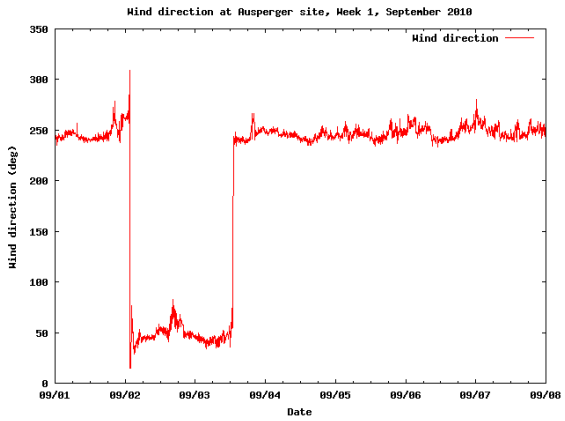
Focus area 1: Adaptive noise reduction in data streams

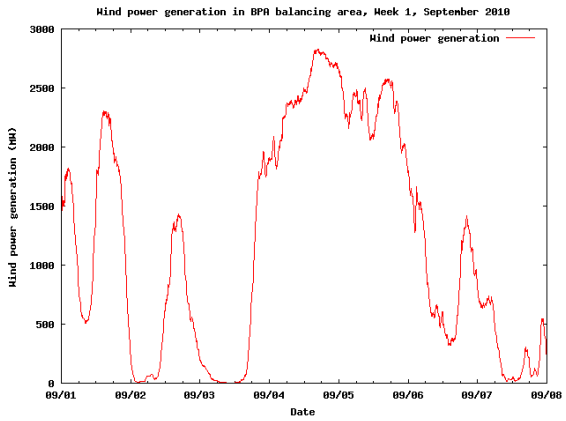












For each data stream:

- How do we select the parameters for a denoising algorithm?
- How do we adapt the parameters with time?

Find the optimal parameters using Monte Carlo SURE†

$y = x + b$

$x, y \in \mathbb{R}^N$

x : noise free signal

y : noisy data

b : zero mean, white Gaussian noise of variance σ^2

Denoising algorithm : $\hat{x} = f_\lambda(y)$

λ : set of parameters for the algorithm

Find parameters by estimating MSE using Stein's Unbiased Risk Estimate

SURE: $$\eta(f_\lambda(y)) = \frac{1}{N}\|y - f_\lambda(y)\|^2 - \sigma^2 + \frac{2\sigma^2}{N}\text{div}_y\{f_\lambda(y)\}$$

where $$\text{div}_y\{f_\lambda(y)\} = \sum_{k=1}^{N} \frac{\partial f_{\lambda k}(y)}{\partial y_k}$$

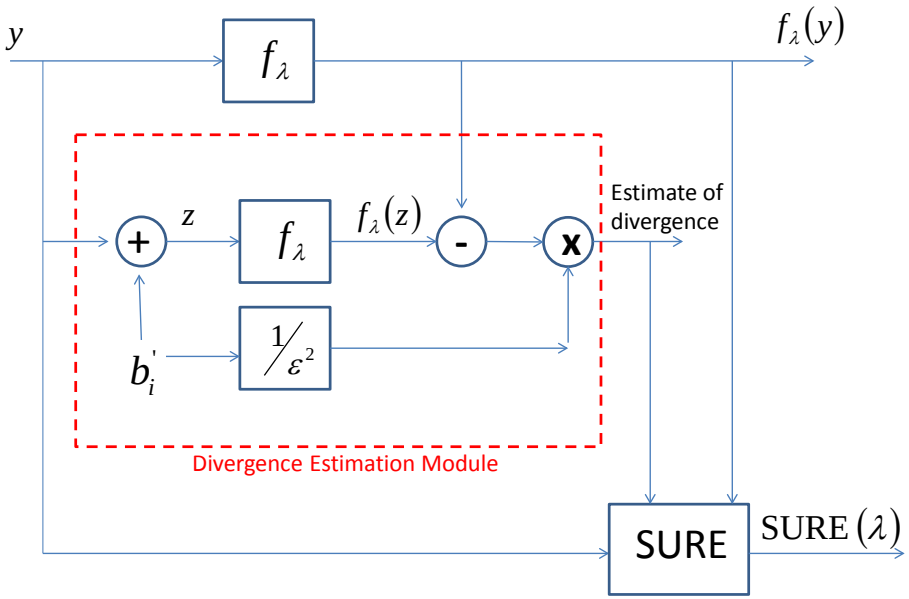
To determine SURE, we need to estimate σ and $\text{div}_y\{f_\lambda(y)\}$.

† S. Ramani, T. Blu, and M. Unser, "Monte-Carlo SURE: A black-box optimization of regularization parameters for general denoising algorithms", IEEE Trans. on Image Processing, Vol.17, No. 9, September 2008.

Estimate divergence by probing the algorithm with noise

$$\mathrm{div}_y\{f_\lambda(y)\} \approx \frac{1}{\epsilon^2} b'^T (f_\lambda(y + b') - f_\lambda(y))$$

where b' is a zero mean i.i.d. random vector with covariance $\epsilon^2 I$, provided that f_λ has a well-defined second-order Taylor expansion.

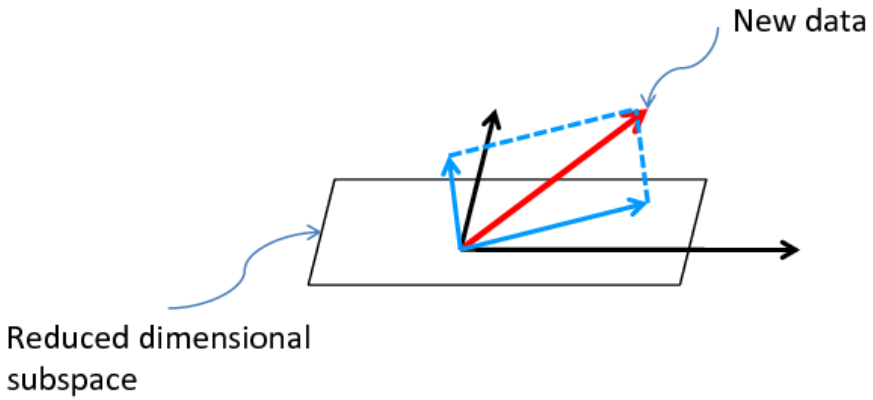


We can then vary λ, find the corresponding SURE, and use the λ corresponding to the minimum SURE.

We are extending this idea of adaptivity to streaming data and improved sampling of λ space.

Focus area 2: Incremental dimension reduction

- Find correlated features - useful when sensors spatially close
  - Windowing approach: Incremental and non-incremental versions
  - Forgetting factor approach
- Random projections - useful for very high dimensions; with ensembles
- Projection approximate subspace trackers - $O(dk)$, based on a forgetting factor, uses matrix inversion lemma (stability issues)
- Incremental SVD - general approach[†], a sliding window approach (FAST)[‡]



† M. Brand, "Incremental singular value decomposition of uncertain data with missing values", Proceedings, ECCV, pp. 707-720, 2002.

‡ E. C. Real, D. W. Tufts, and J. W. Cooley, "Two algorithms for fast approximate subspace tracking," IEEE Transactions of Signal Processing, Vol. 47, No. 7, July 1999.

Fast Approximate Subspace Tracking (FAST)

$$M_{old} = S_{old} + N_{old}$$
$$= \begin{bmatrix} m_1 & m_2 & \dots & m_c \end{bmatrix}_{(r \times c)}$$

M_{old} : matrix representing the data

S_{old} : reduced-rank signal matrix of rank k

N_{old} : full-rank noise matrix

Let the k orthonormal approximate left singular vectors of M_{old} be

$$U_{old} = \begin{bmatrix} u_1 & u_2 & \dots & u_k \end{bmatrix}_{(r \times k)}$$

The error in reconstruction using only the largest k singular values/vectors:

$$\epsilon_{old} = \| M_{old} - U_{old} U_{old}^T M_{old} \|_F^2$$

When new data arrive, $M_{new} = \begin{bmatrix} m_2 & m_2 & \dots & m_{(c+1)} \end{bmatrix}_{(r \times c)}$

Goal: track the k singular values/vectors of the signal subspace as the data transition from M_{old} to M_{new}.

Step 1: create a low-rank approximation $A_{(r\times c)}$ to M_{new}

$$\begin{aligned}\text{Let } M_{old} &\approx U_{old}\,U_{old}^T M_{old}\\ &= U_{old}\begin{bmatrix}a_1 & a_2 & \dots & a_c\end{bmatrix}\\ &= \begin{bmatrix}g_1 & g_2 & \dots & g_c\end{bmatrix}\end{aligned}$$

$$\begin{aligned}A &\triangleq \begin{bmatrix}U_{old} & q\end{bmatrix}\begin{bmatrix}a_2 & \dots & a_c & a_{(c+1)}\\ 0 & \dots & 0 & b\end{bmatrix}\\ &= \begin{bmatrix}U_{old} & q\end{bmatrix}_{r\times(k+1)} E_{(k+1)\times c}\end{aligned}$$

where

$$\begin{aligned}a_{(c+1)} &= U_{old}^T m_{(c+1)}\\ z &= m_{(c+1)} - U_{old}\,a_{(c+1)}\\ b &= \|z\|\\ q &= \frac{z}{b}\end{aligned}$$

$$\begin{aligned}\text{Therefore, } \|M_{new} - A\|_F^2 &= \sum_{i=2}^{c}\left\{\|m_i - g_i\|^2\right\} + \|m_{(c+1)} - m_{(c+1)}\|^2\\ &\leq \epsilon_{old}\end{aligned}$$

New error is no greater than ϵ_{old}; however, A is the same size as M_{new}.

Step 2: update U_{old} but work with a matrix smaller than A

$$A = \underbrace{\begin{bmatrix} U_{old} & q \end{bmatrix}_{r \times (k+1)}}_{(k+1) \text{ orthonormal columns}} \quad E_{(k+1) \times c}$$

$$\text{Obtain} \quad E = U_E \Sigma_E V_E^T$$

$$\text{Then,} \quad A = \left(\begin{bmatrix} U_{old} & q \end{bmatrix} U_E \right) \Sigma_E V_E^T \quad \text{where} \quad \begin{aligned} U_A &= \begin{bmatrix} U_{old} & q \end{bmatrix} U_E \\ \Sigma_A &= \Sigma_E \\ V_A &= V_E. \end{aligned}$$

$$= U_A \Sigma_A V_A^T$$

$$\text{Or, we can obtain} \quad F_{(k+1)\times(k+1)} = EE^T$$

$$= \left(U_E \Sigma_E V_E^T\right)\left(V_E \Sigma_E U_E^T\right)$$

$$= U_E \Sigma_E \Sigma_E U_E^T$$

$$= U_F \Sigma_F V_F^T.$$

By calculating the SVD of E or F we can obtain the left singular values and vectors of A, which is an approximation to M_{new}.

Use the energy of the data to determine the number of singular vectors to keep

The dimension of the signal subspace: the top k singular values/vectors that explain most of the energy in the data

$$\text{Energy in the data} = \sum_i \sigma_i^2 = \|M\|_F^2$$

Update the Frobenius norm from the previous iteration:

$$\|M_{new}\|_F^2 = \|M_{old}\|_F^2 - \|m_1\|_2^2 + \|m_{(c+1)}\|_2^2$$

Choose a percent of energy to keep. Use the singular values to either increase the dimension by 1 or reduce by any amount if fewer singular values/vectors can explain the energy in the new data.

Focus area 3: the effects of spatial distribution of sensors

One approach to identifying "anomalies" such as ramp events:

- determine if each time interval is part of a ramp event or not
- describe each time interval by the sensor values

Time interval	Sensor 1	...	Sensor n	Ramp
1	...	...	...	1
2	...	...	...	0
...	...	...	...	...
m	...	...	...	1

- build a predictive model based on this training set

Should the sensor values be at the same time instant as the output?

- Some sensors may be "upstream" of an event - a lag correlation
- The lag could vary with time

Use lag correlations between the sensors and the output

Pearson's coefficient, with a lag l: $\frac{\sum_{t=l+1}^{n}(x_t-\bar{x})(y_t-\bar{y})}{\sqrt{\sum_{t=l+1}^{n}(x_t-\bar{x})^2}\sqrt{\sum_{t=1}^{n-l}(y_t-\bar{y})^2}}$

where

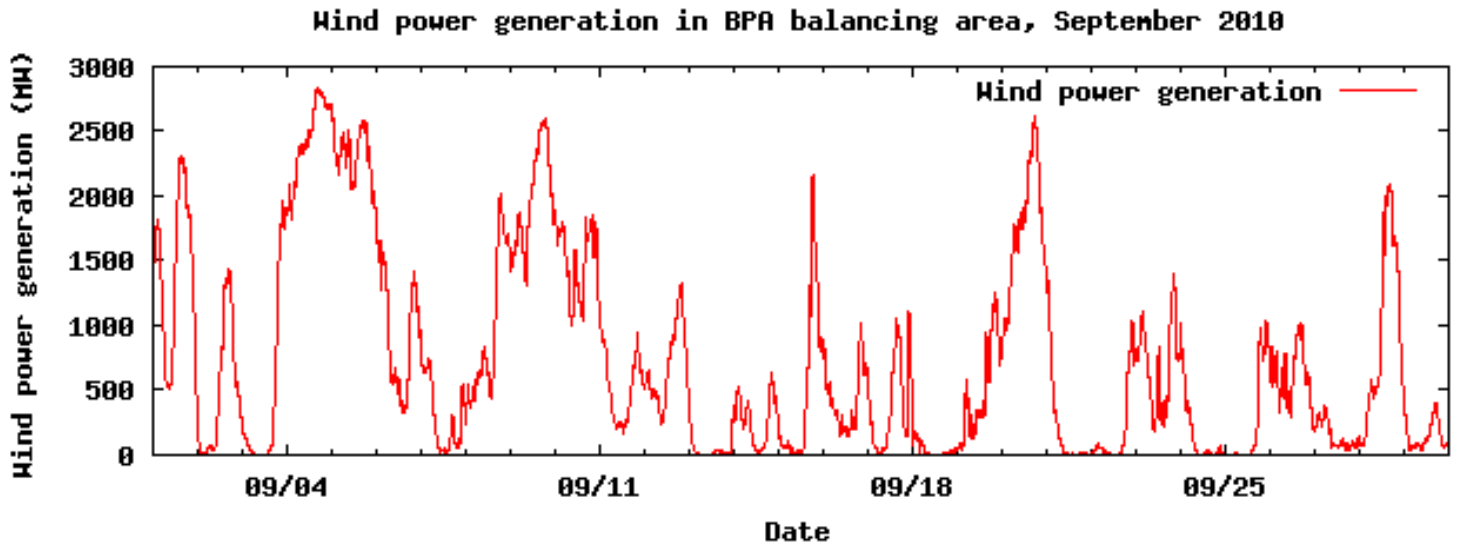
$$\bar{x} = \frac{1}{n-l}\sum_{t=l+1}^{n} x_t \quad , \quad \bar{y} = \frac{1}{n-l}\sum_{t=1}^{n-l} y_t$$

- The two series are lag correlated if the correlation coefficient is higher than a threshold
- The lag is the value of l at the highest correlation coefficient
- Can implement this using a windowing approach or a forgetting factor approach by keeping the sufficient statistics
- Can use historical data to calculate the lag, or calculate online using fast methods†

† Y. Sakurai, S. Papadimitriou, and C. Faloutsos, "BRAID: Stream mining through group lag correlations," Proceedings, SIGMOD, pp 599-610, 2005.

Experimental results using data from wind farms

- Bonneville Power Administration balancing area - mid-Columbia Basin
- Wind power generation available at 5 min intervals
- Weather data available at 14 sites at 5 minute intervals
  - pressure, relative humidity, temperature, wind speed and direction, peak wind speed and direction
  - several missing values
- Data from Sept 2010: use 13 sites; exclude Forest Grove which has missing values $\Rightarrow$ 8640 instants, 91 variables



Reducing number of streams by using correlations

Compared the windowing approach (incremental and non-incremental versions) and the forgetting factor approach:

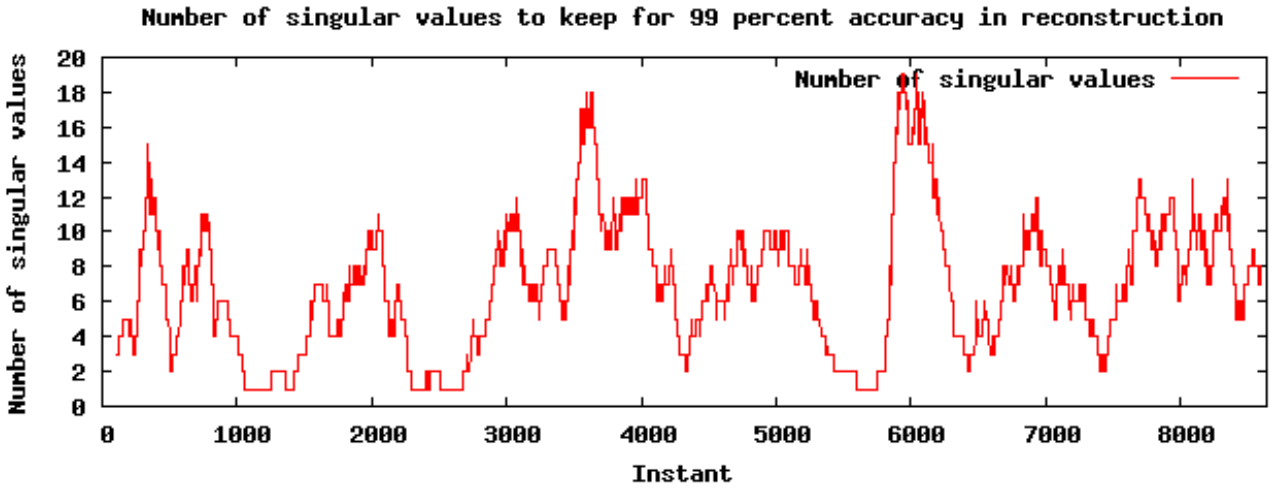
- Implement using sufficient statistics
- Results are similar, though forgetting factor approach gives results closer to the non-incremental windowing approach
- Windowing approach requires more memory to store the data in the window

Results for the wind generation weather data:

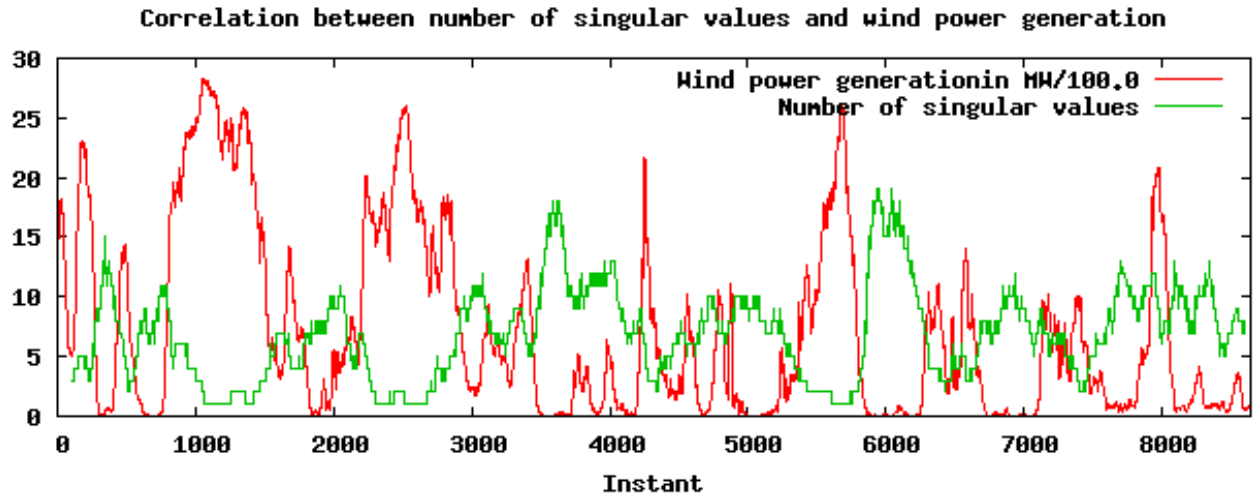
- Barometric pressure at all 13 locations are correlated.
- Relative humidity at the 13 locations tends to be correlated.
- Wind direction correlated to peak wind direction at the same location.
- Wind speed correlated to peak wind speed at the same location.

Application of the FAST algorithm

Consider a window of 100 and a threshold of 99% for the number of singular values to keep.

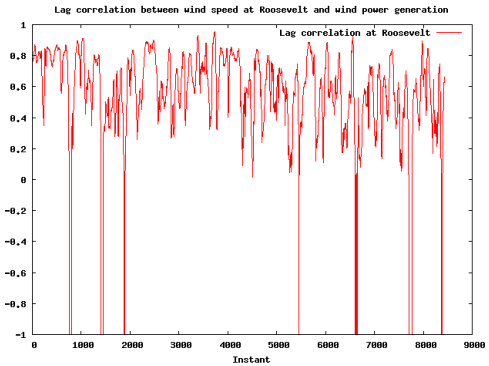


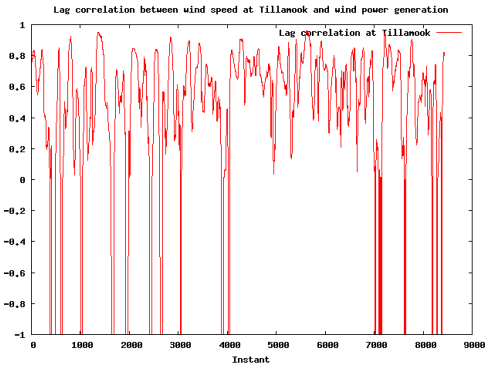
Possible correlation between number of singular values and generation?

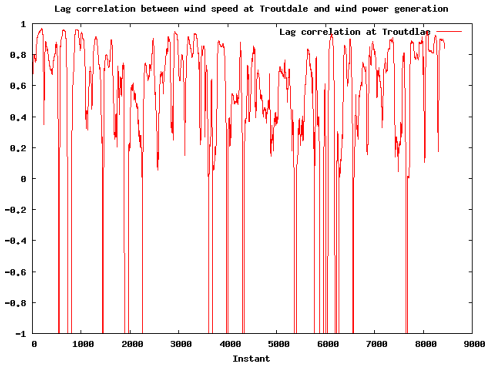


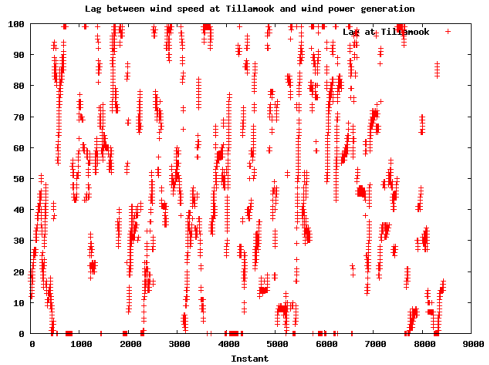
The lag correlations are harder to interpret

Lag correlations between wind speed and wind power generation using a window of size 200 and a maximum lag of 100.









- Values of -1 indicate a negative correlation
- The lag correlation changes with time: how do we interpret this?
- Should negative correlations be considered?
- The lag changes with time. How do we exploit this?

Summary and conclusions

- We are investigating ways to pre-process multiple data streams, with time varying statistics.
- We are implementing a Monte-Carlo approach to adaptively identify parameters for algorithms which reduce noise in the data.
- A fast implementation of an incremental singular value decomposition shows promise in identifying ramp events.
- It is unclear if lag correlations will help in creating an appropriate training data set.
- Our focus is on techniques with low memory requirements; we use single precision where adequate and double precision as needed.

Publications and presentations:

- C. Kamath, "Subspace tracking for dimension reduction in streaming data," SIAM Conference on Computational Science and Engineering, 2011.
- C. Kamath, "Dimension reduction for streaming data," book chapter in Data Intensive Computing: Architectures, Algorithms, and Applications, Ian Gorton and Deb Gracio, editors. To be published by Cambridge University Press, 2011.

Plans for future work

Algorithm research:

- apply the adaptive denoising prior to the dimension reduction
- investigate how to process data streams with vastly different sampling rates
- understand what the changing singular values indicate about the data
- investigate ways of creating the training data to build predictive models to detect anomalies

Applications:

- For the wind generation weather data: look at other months/years to see if a seasonal or yearly variation
- For the fusion data set from DIII-D: apply the ideas developed thus far

https://computation.llnl.gov/casc/StarSapphire/SensorStreams.html