Online Subspace Estimation and Tracking from Missing or Corrupted Data

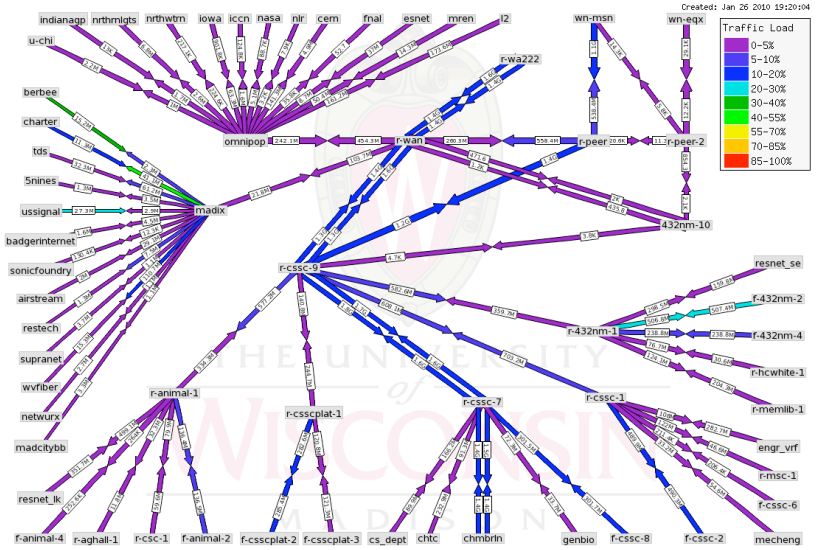
Laura Balzano
www.ece.wisc.edu/~sunbeam

THE UNIVERSITY of WISCONSIN MADISON

Work with **Benjamin Recht** and **Robert Nowak**

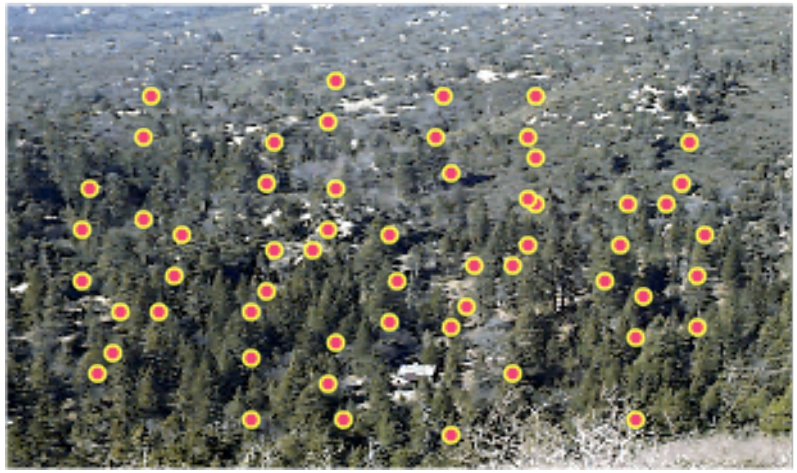
Subspace Representations Capture Dependencies

Monitor/sense with n nodes



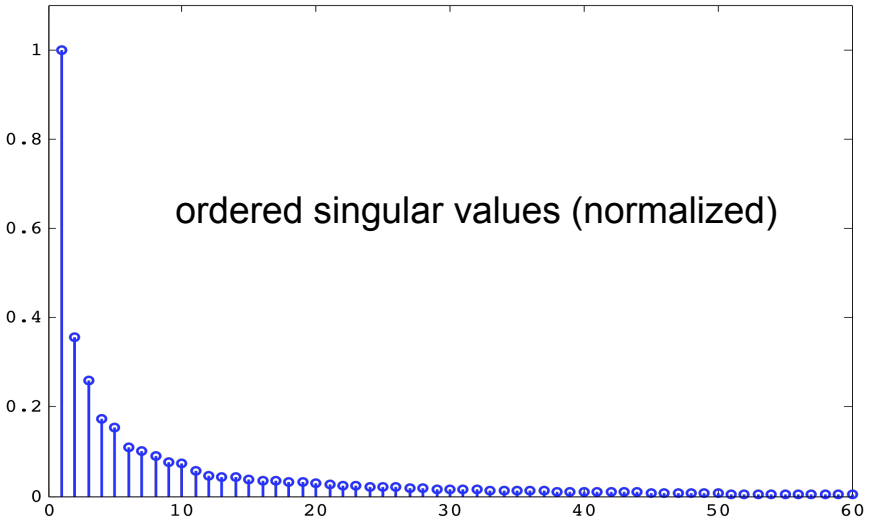
$x \in \mathbb{R}^n$ is a 'snapshot' of the network state (e.g., traffic rates at each monitor)

$x \in \mathbb{R}^n$ is a 'snapshot' of the system state (e.g., temperature at each node)



Subspace Representations Capture Dependencies

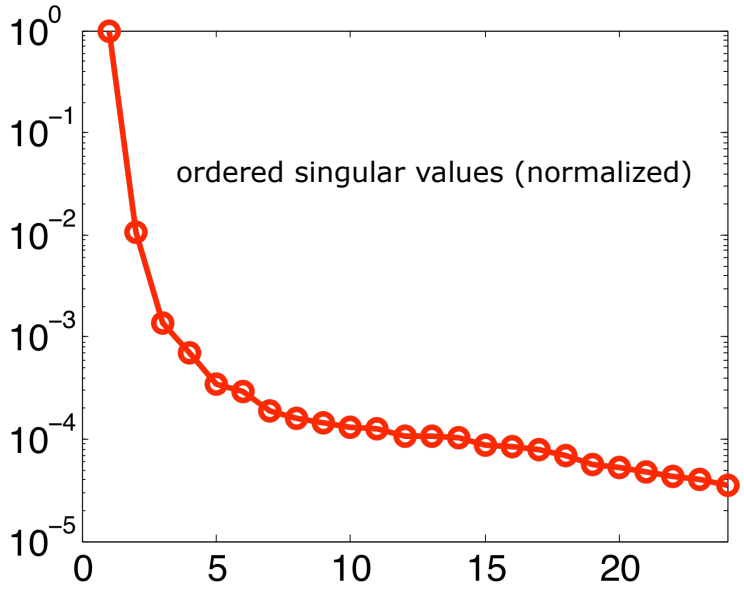
Monitor/sense with n nodes



Byte Count data from UW network

$x \in \mathbb{R}^n$ is a 'snapshot' of the network state (e.g., traffic rates at each monitor)

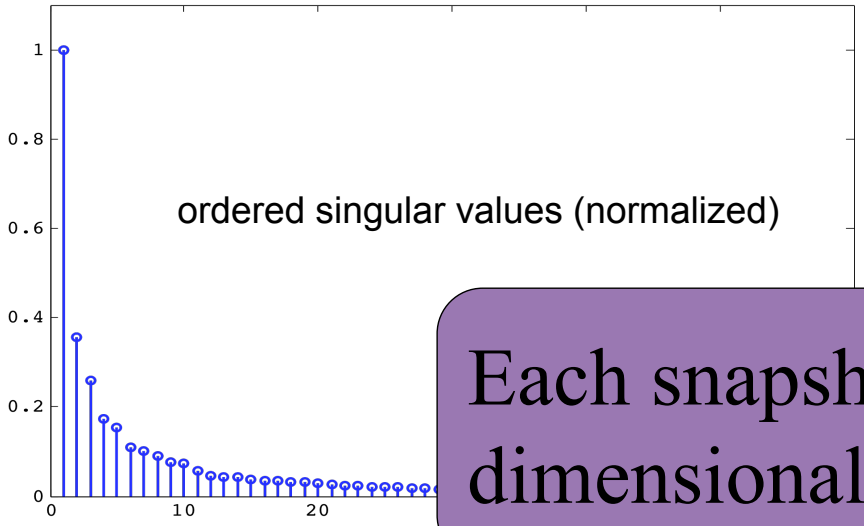
$x \in \mathbb{R}^n$ is a 'snapshot' of the system state (e.g., temperature at each node)



Temperature data from UCLA Sensornet

Subspace Representations Capture Dependencies

Monitor/sense with n nodes

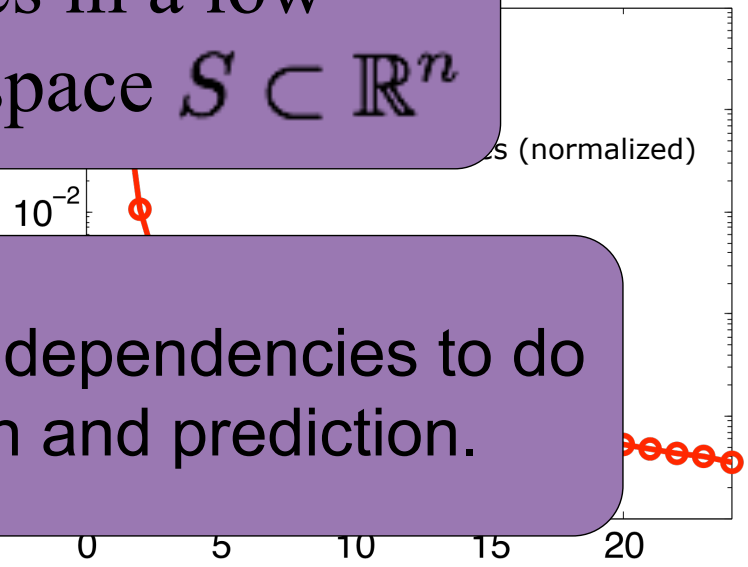


ordered singular values (normalized)

$x \in \mathbb{R}^n$ is a 'snapshot' of the network state
(e.g., traffic rates at each monitor)

Each snapshot lies in a low-dimensional subspace $S \subset \mathbb{R}^n$

Byte Count data from UW network

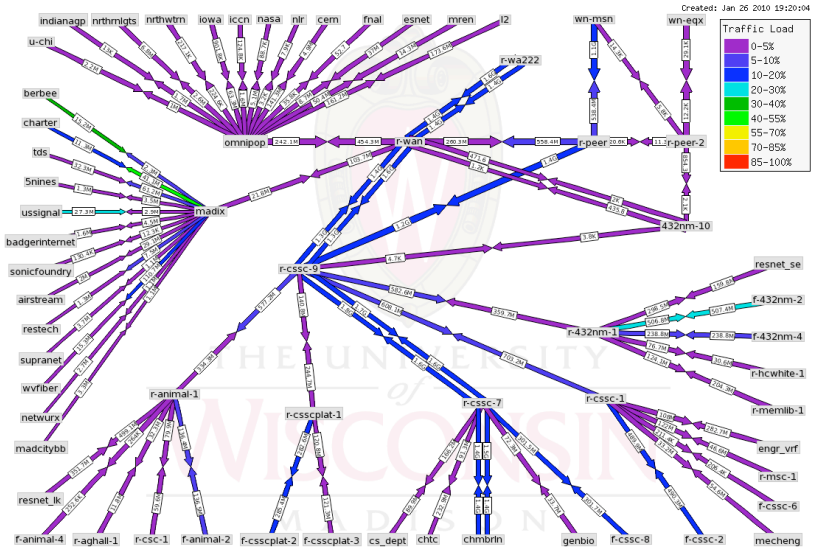


$x \in \mathbb{R}^n$ is a
(e.g., tempe

We can leverage these dependencies to do detection, estimation and prediction.

Temperature data from UCLA Sensornet

Subspace Representations May be Time-Varying



Slow changes in the subspace can indicate changes in usage patterns.

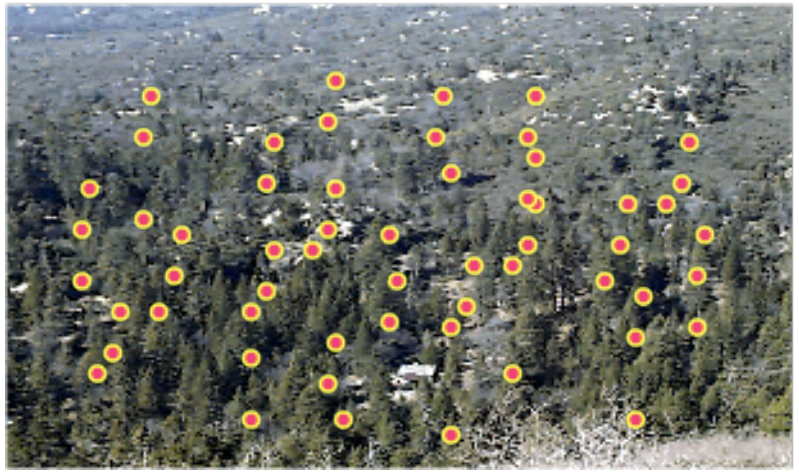
Sudden deviations from the subspace can indicate failures or attacks.

Lakhina, Crovella, Diot: Diagnosing Network-Wide Traffic Anomalies. Sigcomm 2004.

Slow changes in the subspace can indicate seasonal changes or calibration drift.

Sudden deviations from the subspace can indicate environmental anomalies.

Balzano and Nowak: Blind Calibration of Sensor Networks. Sigcomm 2004.

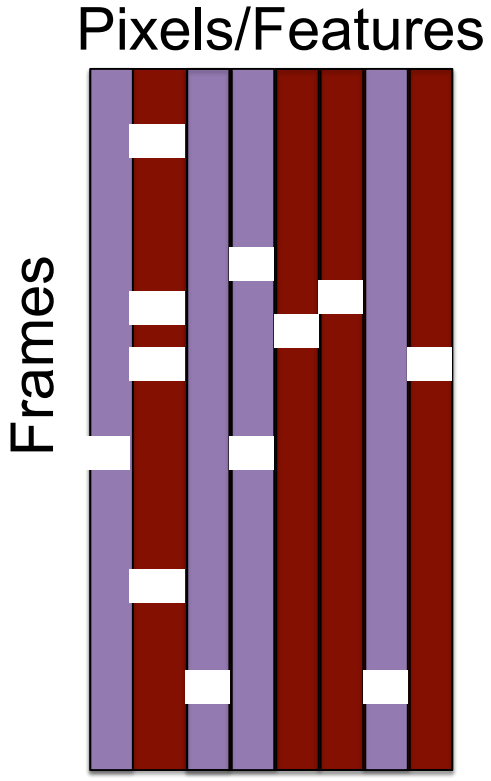


Online Subspace Tracking from Incomplete Data

- Data are incomplete:
  - In sensor networks, data go missing. In large complex systems, it is infeasible to collect all the data.
  - In recommender systems, there is no large overlapping set of experiences (i.e. not everyone has seen the same movie).
  - Data could even be too large to fit in memory
- Incremental updates are needed:
  - You don't want to update the model using the entire available dataset when a user adds one new movie rating.

Subspace Clustering from Incomplete Data

- Object tracking in video:
- First identify features of objects and matching them up across frames.
  - e.g. the SIFT algorithm-- scale invariant feature transform-- is used to stitch images on your camera
- Each tracked object's columns in the pixel-frame matrix lie in a subspace. This leads to the problem of "Subspace Clustering".
- Obstruction of objects means that some frames are missing information about some objects.



Zelnik-Manor, Machline, Irani: Multibody Factorization with Uncertainty: Revisiting Motion Consistency. Int'l Journal on Computer Vision. 2006

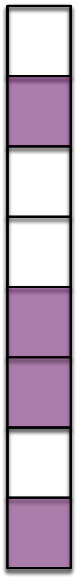
Problem Definition

Suppose we receive a sequence of length-n vectors that lie in a d-dimensional subspace S:

$$v_1, v_2, \ldots, v_t, \ldots, \in S \subset \mathbb{R}^n$$

but we only observe each vector on a particular subset of the n coordinates, $\Omega_t \subset \{1, \ldots, n\}$.

$$v_{\Omega_1}, v_{\Omega_2}, \ldots, v_{\Omega_t}, \ldots$$



We are interested in identifying that subspace S.

At first we consider the subspace S to be static, but we wish to develop an algorithm which operates *incrementally* one vector at a time, and thus is amenable also to *tracking* a possibly changing subspace.

Problem Definition: Full Data, Incremental Update

Suppose we receive a sequence of length-n vectors that lie in a d-dimensional subspace S:

$$v_1, v_2, \ldots, v_t, \ldots, \in S \subset \mathbb{R}^n$$

How do we generate a sequence of estimates S_t?

Given S_t and v_t, how do we generate S_{t+1}?

Solutions to the Full-Data Subspace Tracking Problem

S is a known $d < n$ dimensional subspace of $\mathbb{R}^n$.
Given observation $v \in \mathbb{R}^n$, update S to incorporate v.

Two techniques have been employed to solve this problem:

- Linear algebraic techniques.
- Gradient-based methods.

Linear Algebraic: Incremental SVD Update

Given matrix $X = USV^T$, form the SVD of $[X, v]$.

Estimate the weights: $w = \arg\min_a \|Ua - v\|_2^2$

Compute the residual: $v_\perp = v - Uw$.

Update the SVD:

$$[X, v] = \begin{bmatrix} U & v_\perp \end{bmatrix} \begin{bmatrix} S & w \\ 0 & \|v_\perp\| \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & 1 \end{bmatrix}^T$$

Then do a further eigen decomposition of this matrix to get the new U, S and V.

Subspace Tracking in the Signal Processing Community

- Canonical examples for subspace tracking exist in the sonar, radar, and communications systems literature.
  - Maintain a low-rank approximation of a covariance matrix.
  - Order of 10s of sensors.
  - Dimension of subspace corresponds to number of signal sources.
  - Pisarenko, MUSIC, ESPRIT

- LMS-style or Instantaneous gradient methods
  - Some have an orthogonalization step
  - Projection Approximation Subspace Tracking– does not need orthogonalization
  - Have not been extended for missing data
  - Can be sensitive to step size

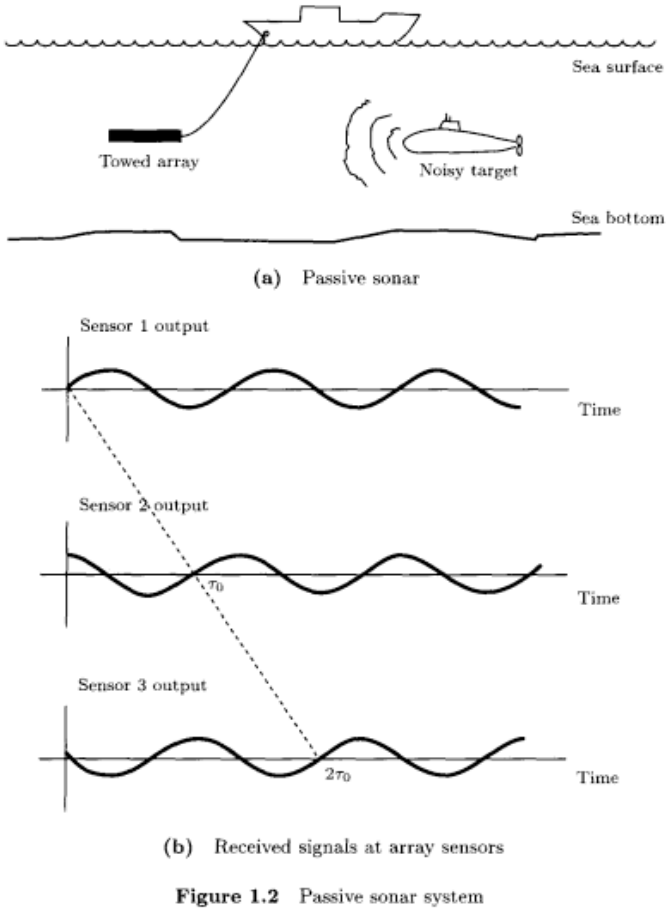


Figure 1.2 Passive sonar system

Figure from Stephen Kay, Fundamentals of Statistical Signal Processing Volume I: Estimation, p3.

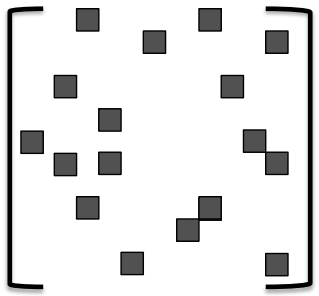
Incomplete Data, Batch Approach

Suppose we collect T incomplete vectors into a matrix, where all the completed vectors lie in the same d-dimensional subspace S:

$$\begin{bmatrix} | & | & & | \\ v_{\Omega_1} & v_{\Omega_2} & \cdots & v_{\Omega_T} \\ | & | & & | \end{bmatrix}$$

How do we use these data to estimate the underlying subspace S?

A Solution to Missing Data Subspace Estimation: Matrix Completion



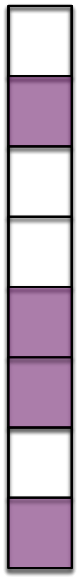
- Matrix Completion can be formulated as static subspace estimation.
  - Estimate the column space and then project to complete the columns.
  - Many algorithms have been proposed (Nuclear Norm minimization, OPT-Space, SVT, SET, FPCA) and the Nuclear Norm optimization is a tight relaxation to the rank minimization problem.

Problem Definition: Incomplete and Incremental

Suppose we receive a sequence of incomplete vectors that lie in a d-dimensional subspace S:

$$v_{\Omega_1}, v_{\Omega_2}, \ldots, v_{\Omega_t}, \ldots$$

Given S_t and v_{Ω_t}, how do we generate S_{t+1}?



The Incomplete Data Projection Residual

Full-data Residual

$$P_S = U(U^T U)^{-1} U^T:$$

$$v_\perp = v - P_S v$$

Incomplete-data Residual

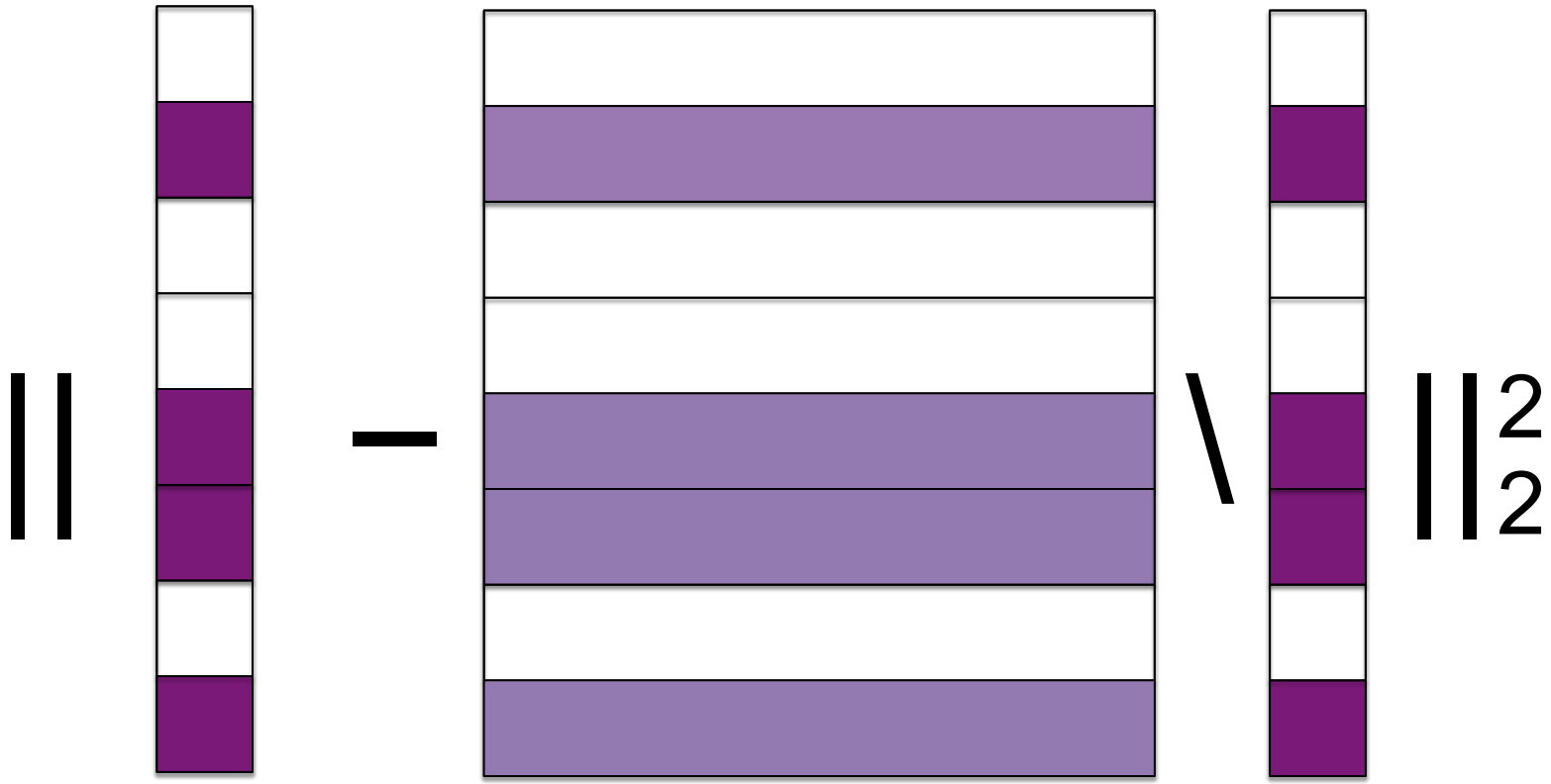
$$\text{Let } P_{S_\Omega} = U_\Omega \left(U_{\Omega_t}^T U_{\Omega_t}\right)^{-1} U_{\Omega_t}^T.$$

$$v_\perp = v_\Omega - P_{S_\Omega} v_\Omega$$

Norm of the Incomplete Data Residual

Incomplete Data Residual Norm

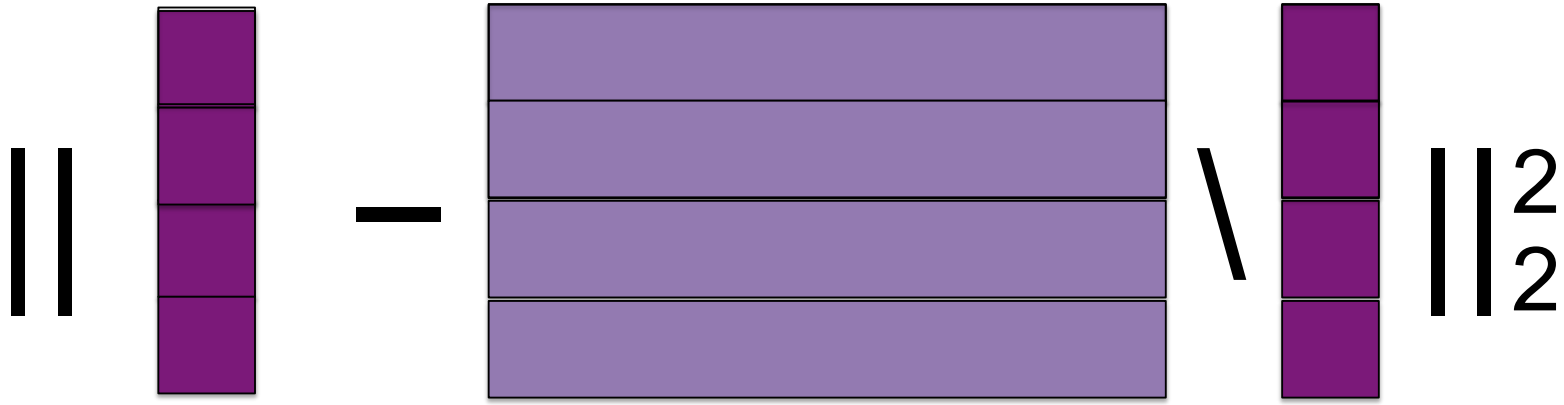
$$\|v_\Omega - P_{S_\Omega} v_\Omega\|_2^2$$



Norm of the Incomplete Data Residual

Incomplete Data Residual Norm

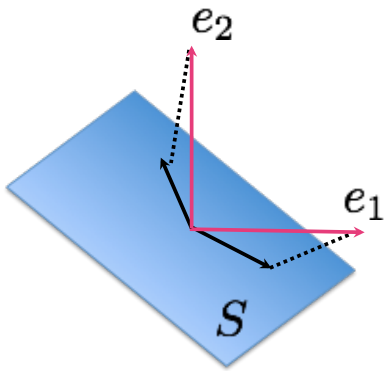
$$\|v_\Omega - P_{S_\Omega} v_\Omega\|_2^2$$



Coherence

A fundamental problem with subsampling is that we may miss the important information.

How aligned are the subspace S and the vector v to the canonical basis?



Examples of bases that form an Incoherent Subspace:

- Orthonormalize Gaussian random vectors.
- Fourier basis.

$$\begin{bmatrix} \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{n}} \end{bmatrix}$$

$$\mu := \frac{n}{d} \max_j \|P_S e_j\|_2^2$$

$$1 \leq \mu(v) \leq n$$

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Examples of bases that form a Coherent Subspace:

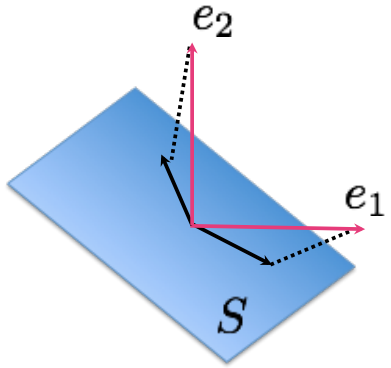
- Identity basis.
- Any basis where the vectors are very sparse.

Norm of the Incomplete Data Residual

Does random subsampling suffice?
How aligned is S to the canonical basis?

$$\mu := \frac{n}{d} \max_j \|P_S e_j\|_2^2$$

coherence (ideally close to 1)

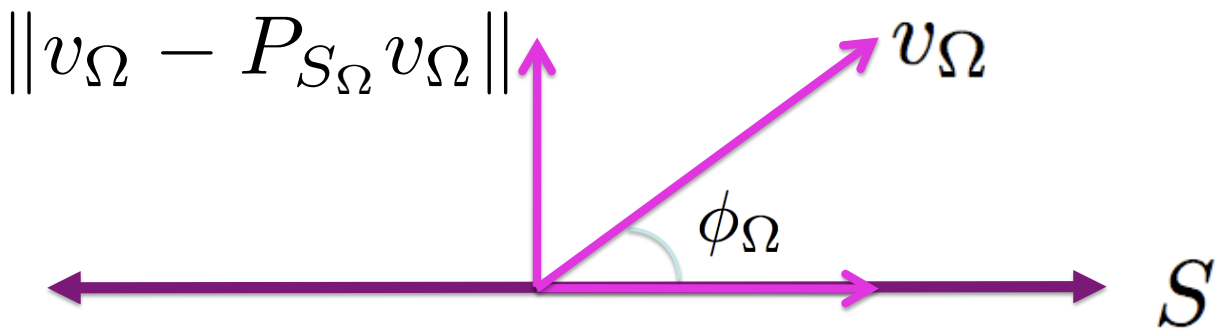


Theorem: If $|\Omega| = O(\mu d \log d)$ and Ω is chosen uniformly with replacement, then with high probability (and ignoring small constant factors)

$$\frac{|\Omega| - d\mu}{n} \|v - P_S v\|_2^2 \leq \|v_\Omega - P_{S_\Omega} v_\Omega\|_2^2 \leq \frac{|\Omega|}{n} \|v - P_S v\|_2^2$$

Random subsample of 8 sensors captures the residual norm from a 6-dimensional subspace of R^{23}

Angle of the Incomplete Data Residual



Theorem: If $|\Omega| = O(\mu d \log d)$ and Ω is chosen uniformly with replacement, then with high probability (and ignoring small constant factors)

$$\frac{|\Omega| - d\mu}{|\Omega|} \sin(\phi)^2 \leq \sin(\phi_\Omega)^2 \leq \sin(\phi)^2$$

Random subsample of 8 sensors captures the angle to a 6-dimensional subspace of R^{23}

Problem Definition (yet again)

Suppose we receive a sequence of incomplete vectors that lie in a d-dimensional subspace S:

$$v_{\Omega_1}, v_{\Omega_2}, \ldots, v_{\Omega_t}, \ldots$$

How do we generate a sequence of estimates S_t?

Given S_t and v_{Ω_t}, how do we generate S_{t+1}?

Subspace Tracking Problem

S is a known $d < n$ dimensional subspace of $\mathbb{R}^n$.
Given observation $v \in \mathbb{R}^n$, update S to incorporate v.

Given matrix $X = USV^T$, form the SVD of $[X, v]$.

Estimate the weights: $w = \arg\min_a \|Ua - v\|_2^2$

Compute the residual: $v_\perp = v - Uw$.

Update the SVD:

$$[X, v] = \begin{bmatrix} U & v_\perp \end{bmatrix} \begin{bmatrix} S & w \\ 0 & \|v_\perp\| \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & 1 \end{bmatrix}^T$$

Then do a further eigen decomposition of this matrix to get the new U, S and V.

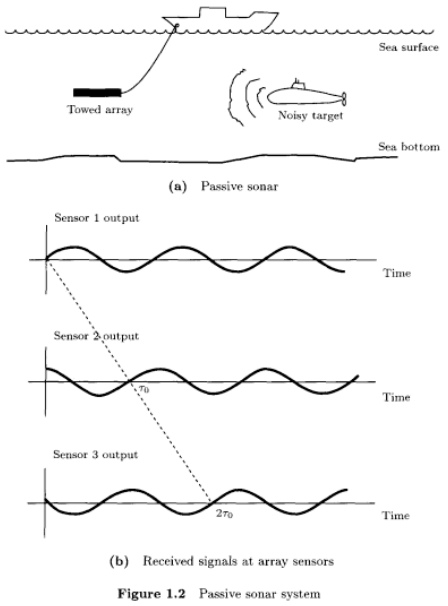


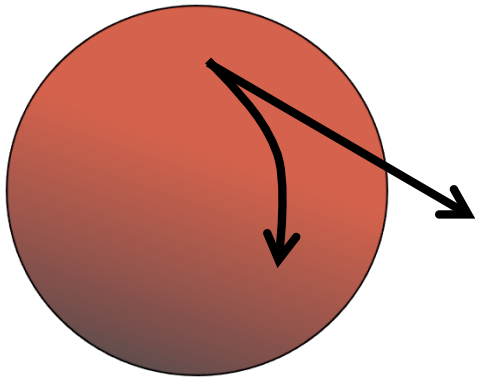
Figure 1.2 Passive sonar system

LMS-style or Instantaneous gradient methods,

some have an orthogonalization step, have not been adapted for missing data

Descent on the Grassmannian: Space of d-dim Subspaces

- Idea: Instead of projecting back to orthogonal columns and a d-dimensional subspace, follow the geodesic on the Grassman manifold.



- Basis of batch incomplete-data algorithms by Keshavan et al (2008) and Dai et al (2009).
- There are explicit formulas for a gradient descent step that follows the Grassmannian geodesic.

A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

Incremental Cost Function

U is an orthogonal matrix whose columns span S.
U_Ω denotes the submatrix with rows indicated by Ω,
where $\Omega \subset \{1, \ldots, n\}$ is the subset of indices observed.

$$F(S;t) = \min_a \|U_{\Omega_t} a - v_{\Omega_t}\|^2$$

Δ_{Ω_t} is the $n \times n$ diagonal matrix with j^{th} diagonal entry

$$= \begin{cases} 1 & \text{if} \quad j \in \Omega_t \\ 0 & \text{otherwise} \end{cases}$$

$$F(S;t) = \min_a \|\Delta_{\Omega_t}(Ua - v_t)\|^2$$

The sampling operator makes it so that our projection matrix is not simply UU^T

$$a = \left(U_{\Omega_t}^T U_{\Omega_t}\right)^{-1} U_{\Omega_t}^T v_{\Omega_t}$$

GROUSE: Grassman Rank-One Update Subspace Estimation

- Given step size η_t, subspace basis U_t, observations $\Delta_{\Omega_t}(v_t)$
- Calculate Weights: $w = \arg\min_a \|\Delta_{\Omega_t}(U_t a - v_t)\|_2^2$
- Predict full vector: $v_{\|} = U_t w$
- Compute Residual on observed entries: $v_{\perp} = \Delta_{\Omega_t}(v_t - v_{\|})$
- Update subspace:

$$U_{t+1} = U_t + \left(\sin(\sigma\eta_t)\frac{v_{\perp}}{\|v_{\perp}\|} + (\cos(\sigma\eta_t) - 1)\frac{v_{\|}}{\|v_{\|}\|}\right)\frac{w^T}{\|w\|}$$

$$\text{where } \sigma = \|v_{\perp}\|\|v_{\|}\|$$

- One iteration involves a projection and an outer product.
- The algorithm is *simple* and *fast*.

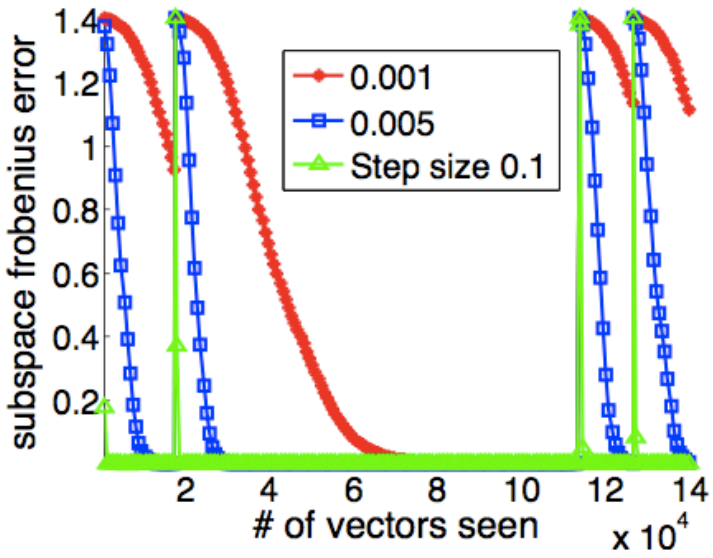
Performance of our Online Gradient Descent Algorithm

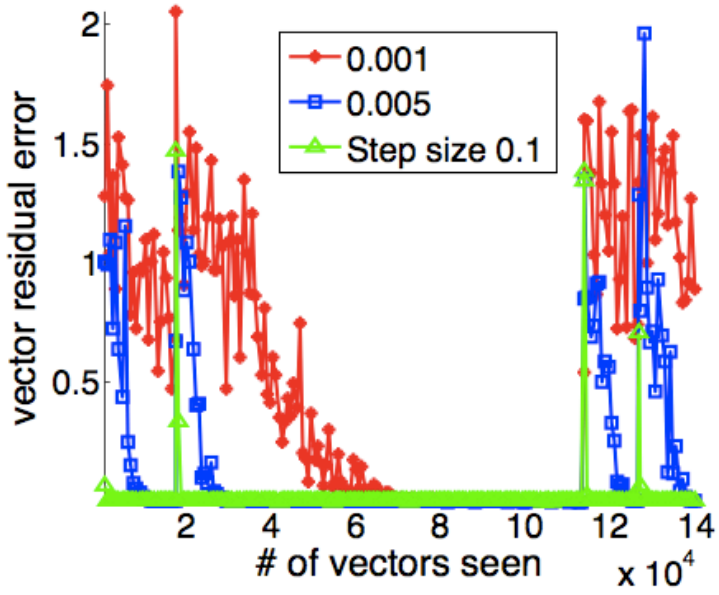
Let's see what it can do!

- Subspace Tracking
- Matrix Completion
- Subspace Clustering
- Robust Tracking

sunbeam.ece.wisc.edu/grouse

Performance of our On-line Gradient Descent Algorithm



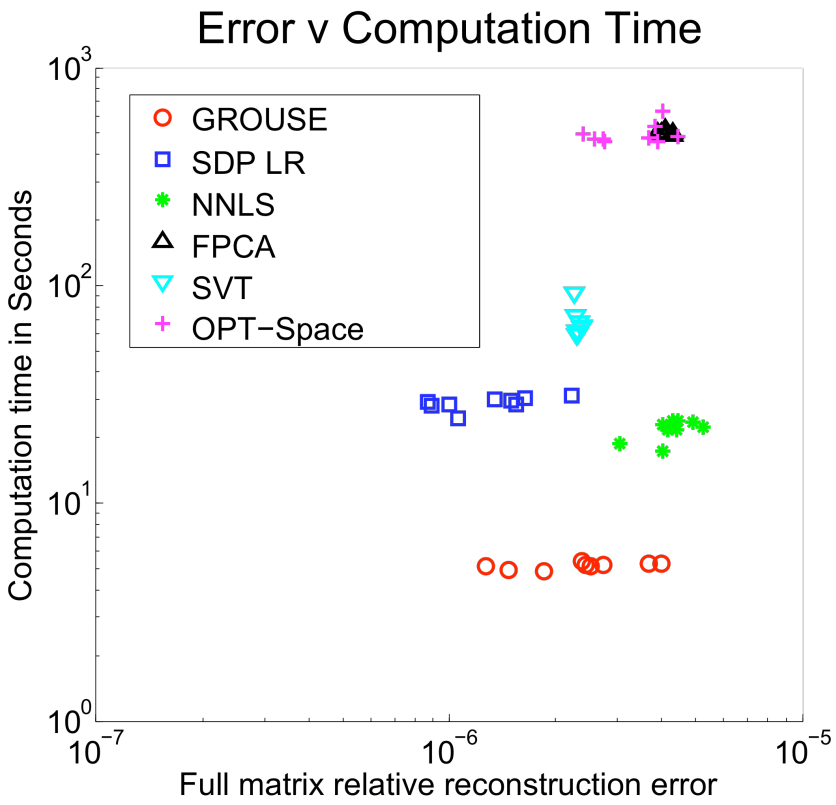


- For tracking a changing subspace, we first looked at sudden changes to new subspaces.
- The tracking was excellent again for a broad range of step sizes.

- We also noted that the vector residual (v perp) is an excellent proxy for error, which comes directly from the result in the first part of the talk.

Comparison Performance for Matrix Completion

- As we said before, Matrix completion is a type of static subspace estimation problem.
- To use GROUSE, we pass over the columns of the matrix in random order, doing an update for every column.
- We compared against other state of the art MC algorithms on reconstruction error and computation time.



Comparison Performance for Matrix Completion

- We compared against Accelerated Proximal Gradient algorithm for NNLS (Toh and Yun 2009) on very large problems. GROUSE is about 2-5 times faster.

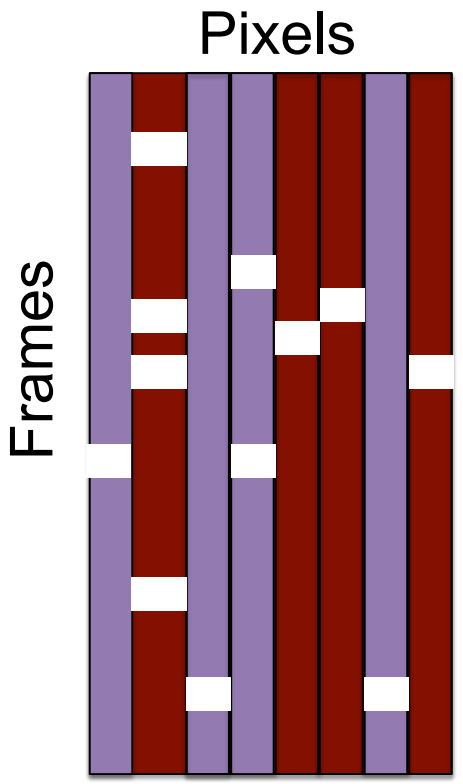
Problem parameters				GROUSE			NNLS	
n_r	n_c	r	dens	rel.err.	passes	time	rel. err.	time (s)
5000	20000	5	0.006	1.10e-4	2	14.8	4.12e-04	66.9
5000	20000	10	0.012	1.5e-3	2	21.1	1.79e-4	81.2
6000	18000	5	0.006	1.44e-5	3	21.3	3.17e-4	66.8
6000	18000	10	0.011	8.24e-5	3	31.1	2.58e-4	68.8
7500	15000	5	0.005	5.71e-4	4	36.0	3.09e-4	62.6
7500	15000	10	0.013	1.41e-5	4	38.3	1.67e-4	68.0

Performance for Subspace Clustering

with Arthur Szlam

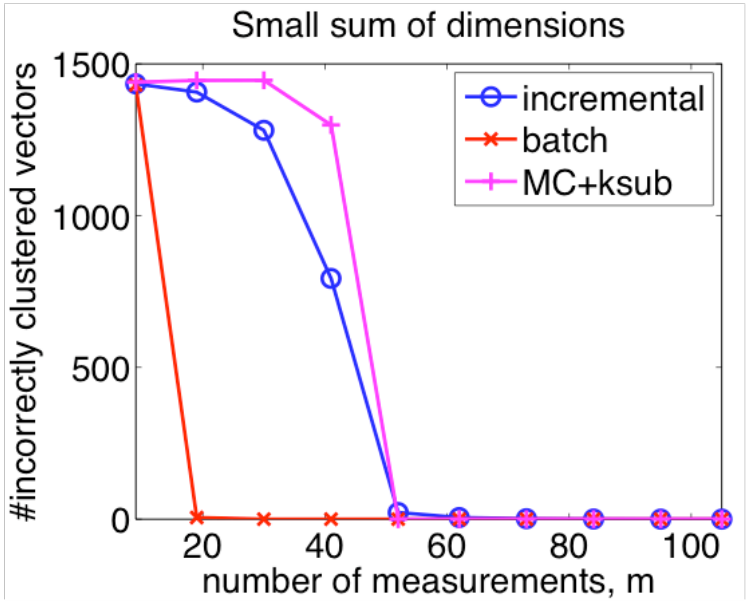
- We have several vectors that can be grouped into k subspaces. We want to group them and estimate the subspaces simultaneously.

- Batch Algorithm:
  - Estimate an initial clustering using a nearest-neighbor type approach.
  - Estimate best d-dimensional subspace for each cluster.
  - Re-cluster the data by finding the nearest subspace. Repeat.
- Incremental Algorithm:
  - Estimate an initial clustering using nn-type approach.
  - Estimate best d-dimensional subspace for each cluster.
  - Repeatedly choose a vector at random, find its nearest subspace, and do one GROUSE update of that subspace.



Performance for Subspace Clustering

with Arthur Szlam



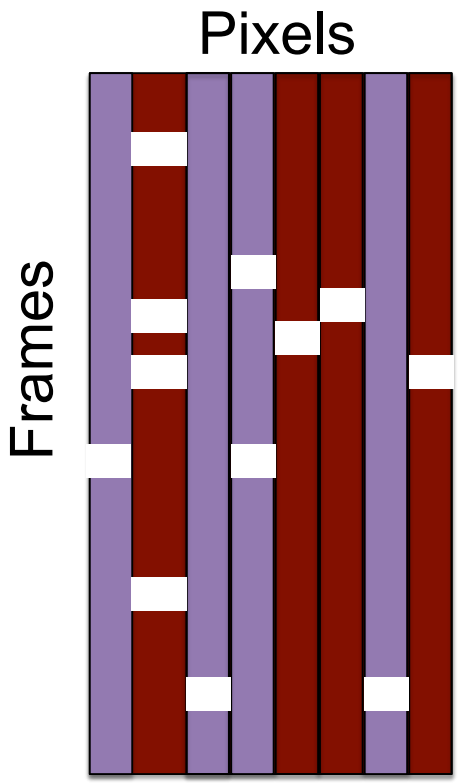
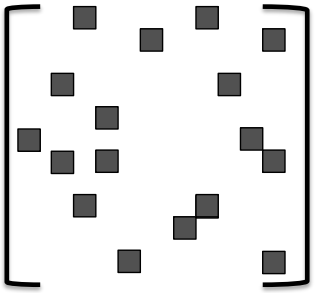


Fig. 5. For this simulation, $n = 150$, $d = 5$, $k = 6$, and the dimension of the union of subspaces is $D = 30$. We compare our missing data subspace clustering algorithms to a two-stage approach: matrix completion is first used to complete the data matrix of rank 30, and then k-planes clustering is used for clustering. Results are averaged over 10 random observation sets.

GROUSE for Robust Tracking

with Arthur Szlam



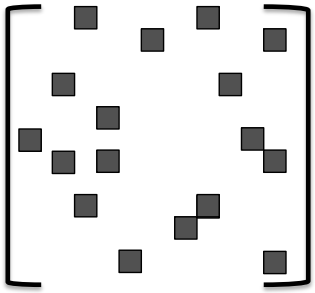
- What if data aren't missing but instead corrupted?

- If we can identify the corruptions, we can throw them out and use GROUSE for tracking.

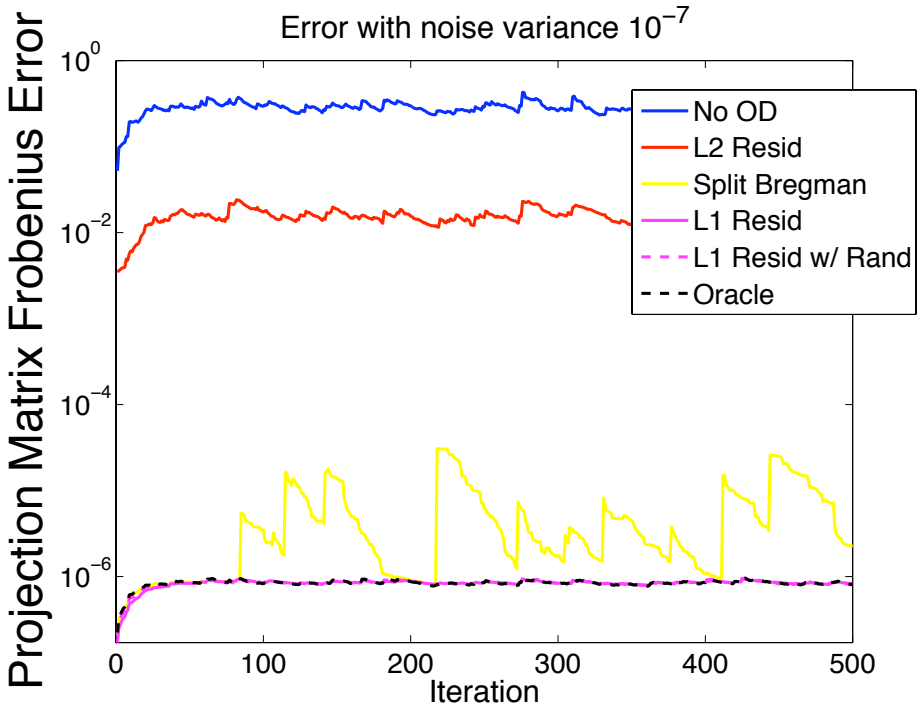
- So far we have only looked at the situation where we initialize with a good estimate of the subspace.
- We examined several methods of identifying the corruptions:
  - Throw out worst residual entries of the Min L1 norm of the residual
  - Randomizing the Min L1 norm index set.
  - Approximations to Min L1 norm.
  - Throw out worst residual entries of the L2 norm.

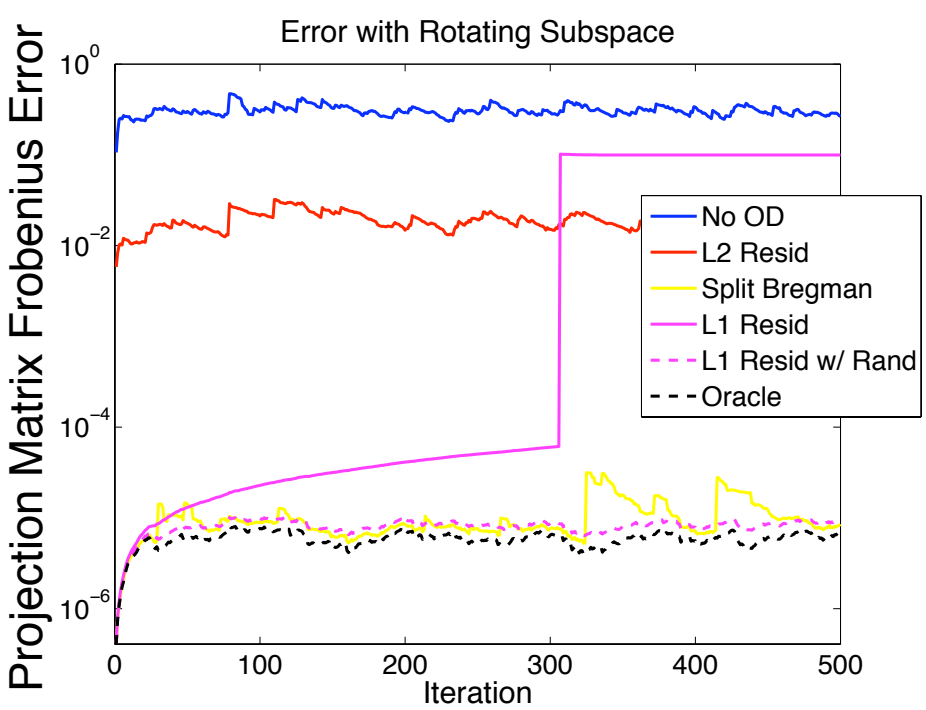
GROUSE for Robust Tracking

with Arthur Szlam



- Throw out worst residual entries of the Min L1 norm of the residual
- Randomizing the Min L1 norm index set.
- Approximations to Min L1 norm (Split Bregman).
- Throw out worst residual entries of the L2 norm.





Next Steps

Generalize the results to other models of correlation between the measurements– Updating nonlinear models, graphical models with incomplete data.

Study convergence of the Subspace Clustering algorithm and properties of the Robust Tracking algorithm.

Finish proof of GROUSE convergence for incomplete data: Will every subspace along the algorithm's path be incoherent with the canonical basis?


Thank you!
Questions?

GROUSE Update Step: Intuition Slides

GROUSE: Intuition. Look at Full-data algorithm.

- Given step size η_t, subspace basis U_t, FULL observations v_t
- Calculate Weights: $w = \arg\min_a \|U_t a - v_t\|_2^2$
- Predict full vector: $v_{\|} = U_t w$
- Compute Residual on observed entries: $v_{\perp} = v_t - v_{\|}$
- Update subspace:

$$U_{t+1} = U_t + \left(\sin(\sigma\eta_t) \frac{v_{\perp}}{\|v_{\perp}\|} + (\cos(\sigma\eta_t) - 1) \frac{v_{\|}}{\|v_{\|}\|} \right) \frac{w^T}{\|w\|}$$

$$\text{where } \sigma = \|v_{\perp}\| \|v_{\|}\|$$

GROUSE: Rewrite U_t

- Given step size η_t, subspace basis U_t, FULL observations v_t
- Calculate Weights: $w = \arg\min_a \|U_t a - v_t\|_2^2$
- Predict full vector: $v_{\|} = U_t w$
- Re-write U_t: Put $v_{\|}/\|v_{\|}\|$ in the first column and use Gram-Schmidt to populate the rest of the columns. Call this $\widetilde{U}_t$.
- New w: $\widetilde{w}(1) = \|w\|$, otherwise 0. Note $v_{\|} = U_t w = \widetilde{U}_t \widetilde{w}$, $\|w\| = \|\widetilde{w}\|$.
- Compute Residual on observed entries: $v_{\perp} = v_t - v_{\|}$
- Update subspace:

$$U_{t+1} = \widetilde{U}_t + \left(\sin(\sigma\eta_t) \frac{v_{\perp}}{\|v_{\perp}\|} + (\cos(\sigma\eta_t) - 1) \frac{v_{\|}}{\|v_{\|}\|} \right) \frac{\widetilde{w}^T}{\|w\|}$$

where $\sigma = \|v_{\perp}\| \|v_{\|}\|$

GROUSE: Rewrite U_t

- Given step size η_t, subspace basis U_t, FULL observations v_t
- Calculate Weights: $w = \arg\min_a \|U_t a - v_t\|_2^2$
- Predict full vector: $v_{\|} = U_t w$
- Re-write U_t: Put $v_{\|}/\|v_{\|}\|$ in the first column and use Gram-Schmidt to populate the rest of the columns. Call this $\widetilde{U}_t$.
- New w: $\widetilde{w}(1) = \|w\|$, otherwise 0. Note $v_{\|} = U_t w = \widetilde{U}_t \widetilde{w}$, $\|w\| = \|\widetilde{w}\|$.
- Compute Residual on observed entries: $v_\perp = v_t - v_{\|}$
- Update subspace:

$$U_{t+1} = \widetilde{U}_t + \left(\sin(\sigma\eta_t) \frac{v_\perp}{\|v_\perp\|} + (\cos(\sigma\eta_t) - 1) \frac{v_{\|}}{\|v_{\|}\|} \right) \frac{\widetilde{w}^T}{\|w\|}$$

where $\sigma = \|v_\perp\| \|v_{\|}\|$

GROUSE: Rewrite U_t

- Given step size η_t, subspace basis U_t, FULL observations v_t
- Calculate Weights: $w = \arg\min_a \|U_t a - v_t\|_2^2$
- Predict full vector: $v_{\|} = U_t w$
- Re-w… Schmidt to popu…
- New … $= \|\widetilde{w}\|$.
- Com…
- Update subspace:

This algorithm is equivalent to the given full-data algorithm, because U_t was arbitrarily chosen to span the subspace.

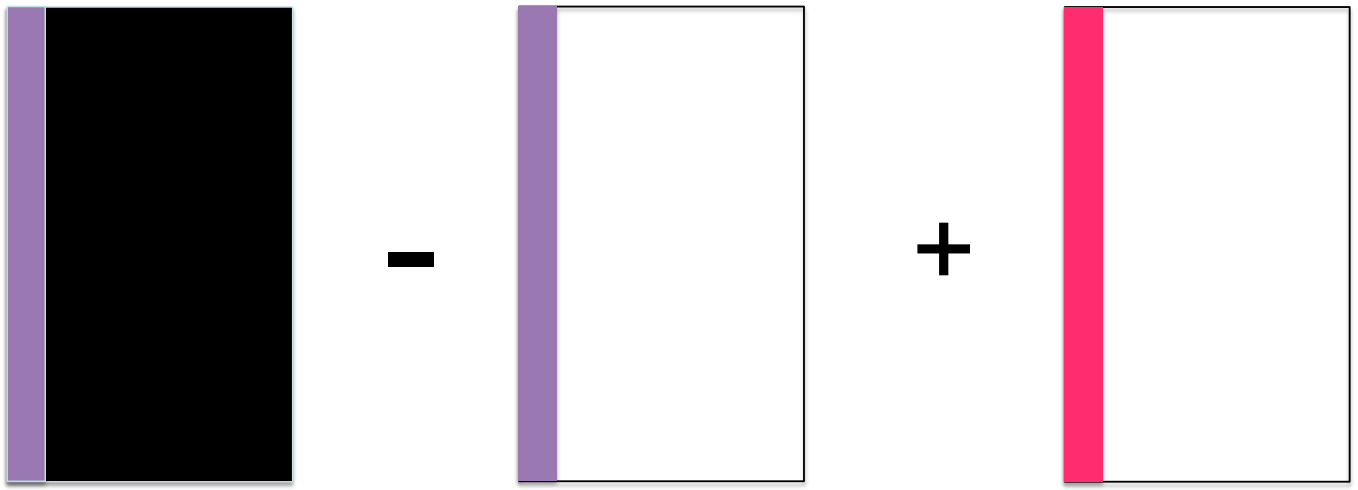
$$U_{t+1} = \widetilde{U}_t + \left(\sin(\sigma\eta_t) \frac{v_\perp}{\|v_\perp\|} + (\cos(\sigma\eta_t) - 1) \frac{v_{\|}}{\|v_{\|}\|} \right) \frac{\widetilde{w}^T}{\|w\|}$$

where $\sigma = \|v_\perp\| \|v_{\|}\|$

GROUSE: Replacing the component in the subspace with a component closer to v.

$$\sigma = \|v_\perp\| \|v_\| \|$$

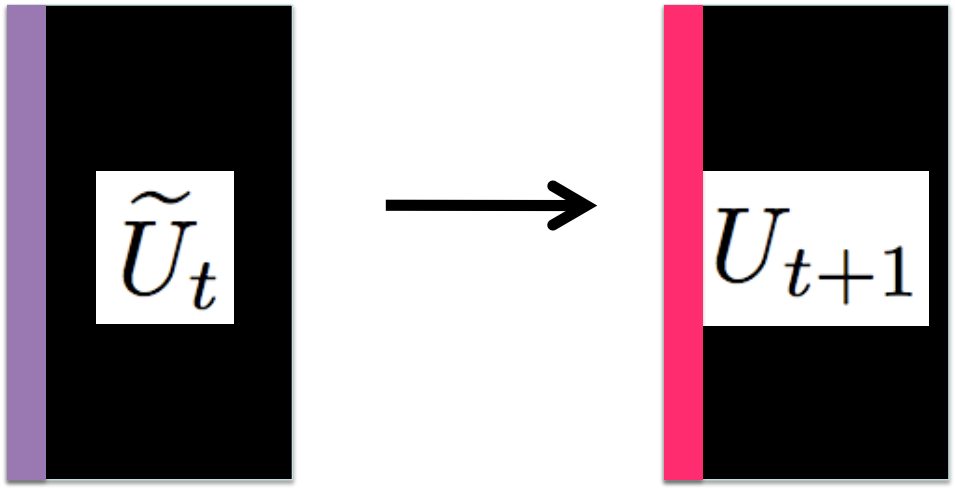
$$U_{t+1} = \widetilde{U}_t - \frac{v_\|}{\|v_\|\|}[1\ 0 \ldots 0] + \left(\sin(\sigma\eta_t)\frac{v_\perp}{\|v_\perp\|} + \cos(\sigma\eta_t)\frac{v_\|}{\|v_\|\|}\right)[1\ 0 \ldots 0]$$



GROUSE: Replacing the component in the subspace with a component closer to v.

$$\sigma = \|v_\perp\| \|v_\| \|$$

$$U_{t+1} = \widetilde{U}_t - \frac{v_\|}{\|v_\|\|}[1\ 0 \ldots 0] + \left(\sin(\sigma\eta_t)\frac{v_\perp}{\|v_\perp\|} + \cos(\sigma\eta_t)\frac{v_\|}{\|v_\|\|}\right)[1\ 0 \ldots 0]$$



GROUSE: Replacing the component in the subspace with a component closer to v.

$$\sigma = \|v_\perp\|\|v_\|\|$$

$$U_{t+1} = \widetilde{U}_t - \frac{v_\|}{\|v_\|\|}[1\ 0 \ldots 0] + \left(\sin(\sigma\eta_t)\frac{v_\perp}{\|v_\perp\|} + \cos(\sigma\eta_t)\frac{v_\|}{\|v_\|\|}\right)[1\ 0 \ldots 0]$$