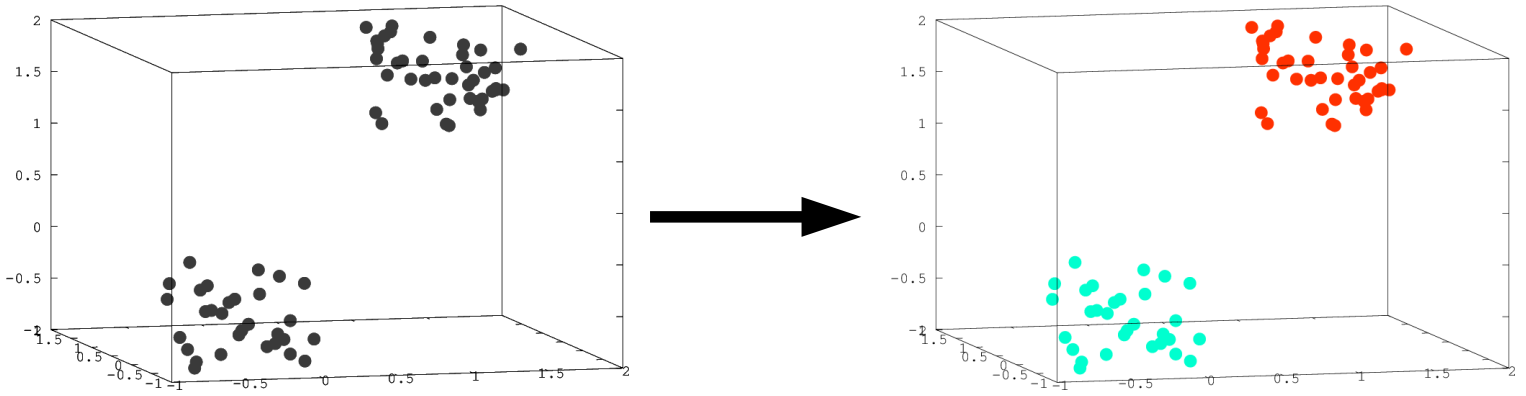
Subspace Clustering with Global Dimension Minimization And Application to Motion Segmentation

Bryan Poling
University of Minnesota

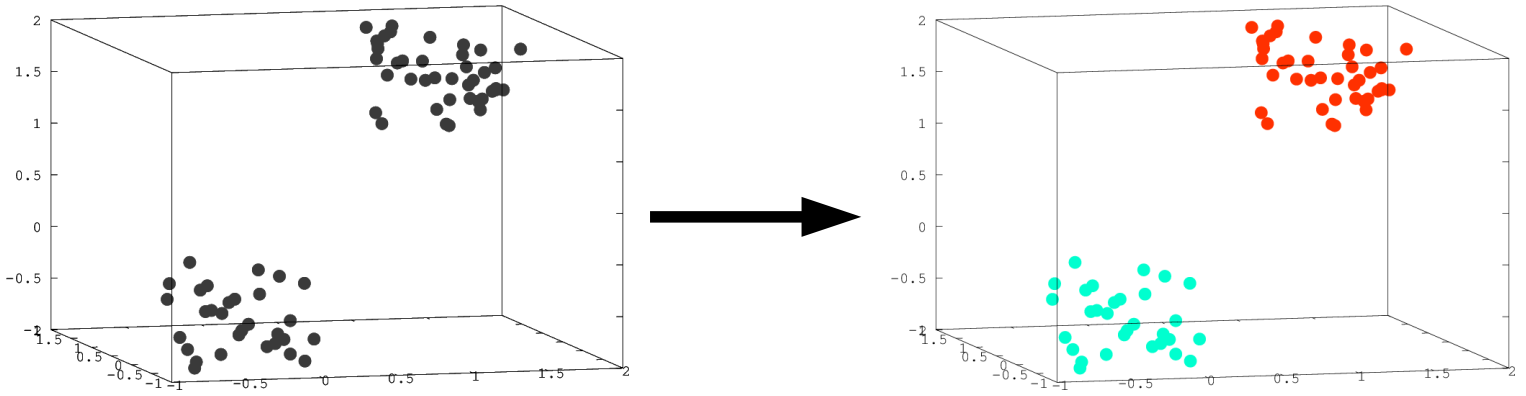
Joint work with Gilad Lerman – University of Minnesota

The Problem of Subspace Clustering

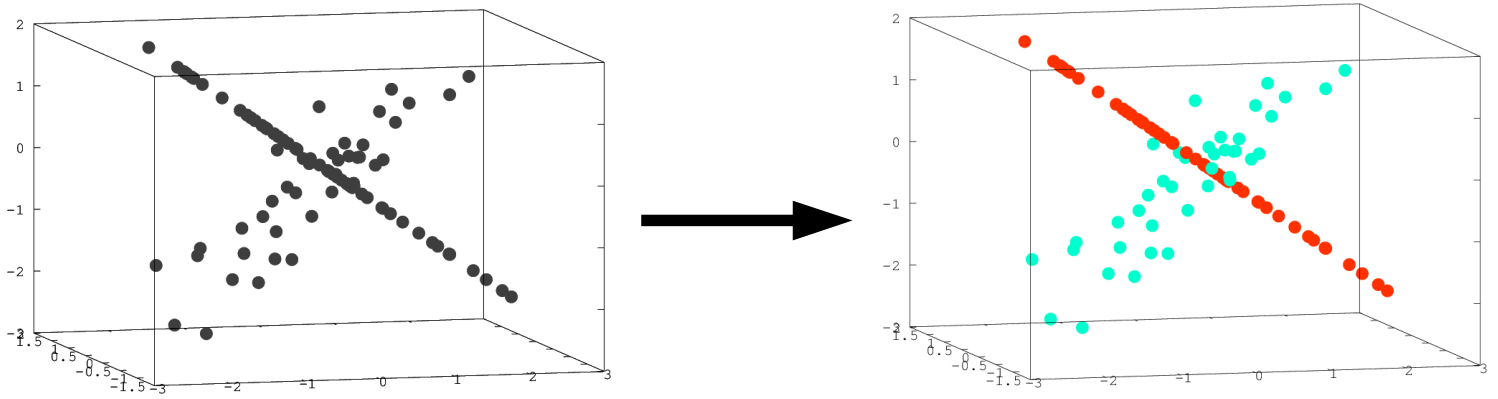


Conventional Clustering

The Problem of Subspace Clustering



Conventional Clustering



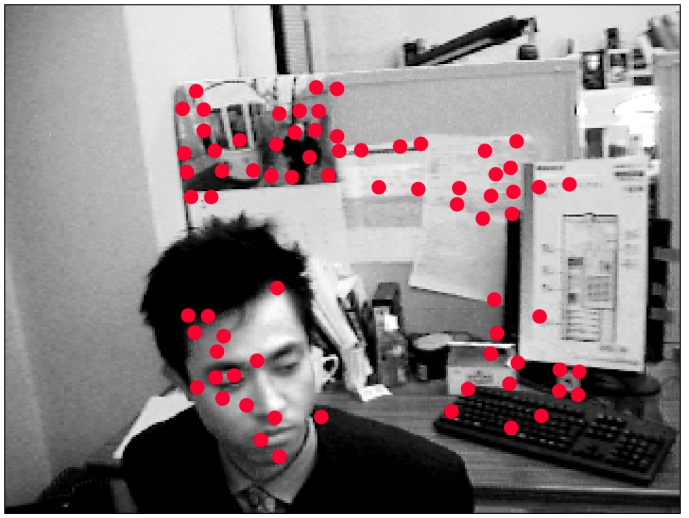
Subspace Clustering

Motivation For Subspace Clustering

- N-view feature-based motion segmentation
- 2-view feature-based motion segmentation
- Face clustering

2-View Motion Segmentation

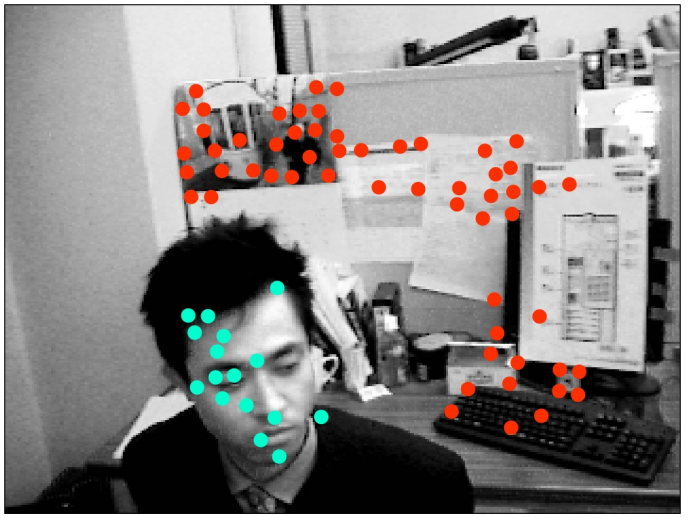
- In this problem, one has two images of a dynamic 3D scene, taken at different times.
- The scene has multiple rigid objects, moving independently.



- We are given the locations of certain “features” in each image. A pair of locations for a single feature is called a “point correspondence”.

2-View Motion Segmentation

- We want to cluster the point correspondences (or, equivalently, cluster the features) according to which rigid objects the are from.



A Few Proposed Solutions to this Problem

- Optical Flow Segmentation (starting around 1985)
- Expectation Maximization (algorithm formalized in 1970's)
- Multi-Body Fundamental Matrix (starting around 2002)
- Subspace Clustering Approaches
  - SSC (2009)
  - SCC (2009)
  - SLBF (2010)
  - Many Others ...

One Formulation Using Subspace Clustering

The Fundamental Matrix

- If we have a set of point correspondences from a single rigid object, there exists a matrix:

$$\mathbf{F} = (F_{i,j})_{i,j=1}^{3}$$

called the Fundamental Matrix, s.t. if $(\mathbf{x}, \mathbf{x}')$ is a point correspondence from the object, with $\mathbf{x}$ and $\mathbf{x}'$ represented in standard homogeneous coordinates, then:

$$\mathbf{x}'^{T}\mathbf{F}\mathbf{x} = 0$$

- This is equivalent to:

$$\mathrm{vec}(\mathbf{F}) \cdot \mathbf{v} = 0$$

where:

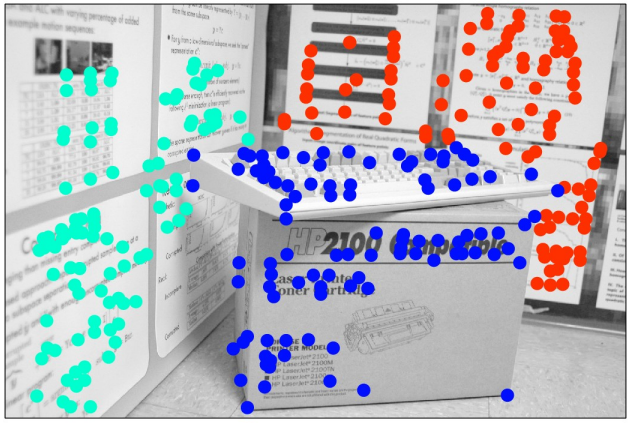
$$\mathrm{vec}(\mathbf{F}) = (F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33})^T$$
$$\mathbf{v} = (xx', x'y, x', xy', yy', y', x, y, 1)^T = (x, y, 1)^T \otimes (x', y', 1)^T$$

- The vector $\mathbf{v}$ is a function only of the point correspondence $(\mathbf{x}, \mathbf{x}')$ and is thought of as a nonlinear embedding of that pair into $\mathbb{R}^9$.

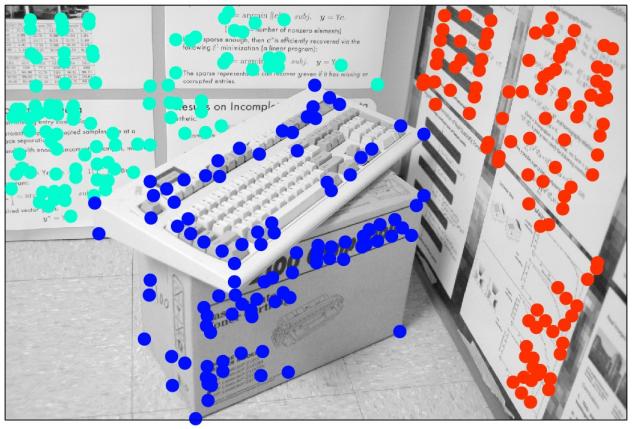
- If we perform this “Kronecker Embedding” on all point correspondences, then there is a single vector, $\mathrm{vec}(\mathbf{F})$, which is orthogonal to all embedded points. Thus, the embedded point correspondences lie in a subspace of dimension 8 or less.
- Subspaces can have dimension less than 8.
- If there are several objects, each gives rise to its own subspace.

Visualization of the Kronecker Map

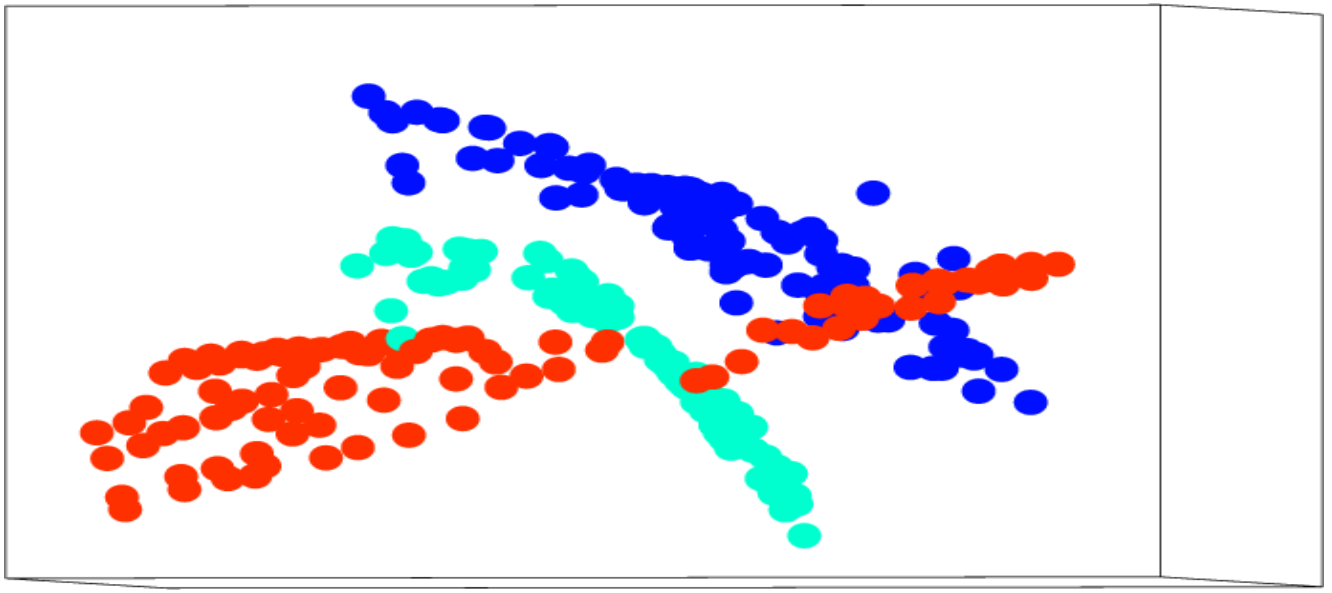
Frame 1



Frame 2



2 views of a dynamic 3D scene, with point correspondences.



Point correspondences mapped into $\mathbb{R}^9$ via the Kronecker embedding (shown projected onto 3rd, 4th, and 5th princ. comps)

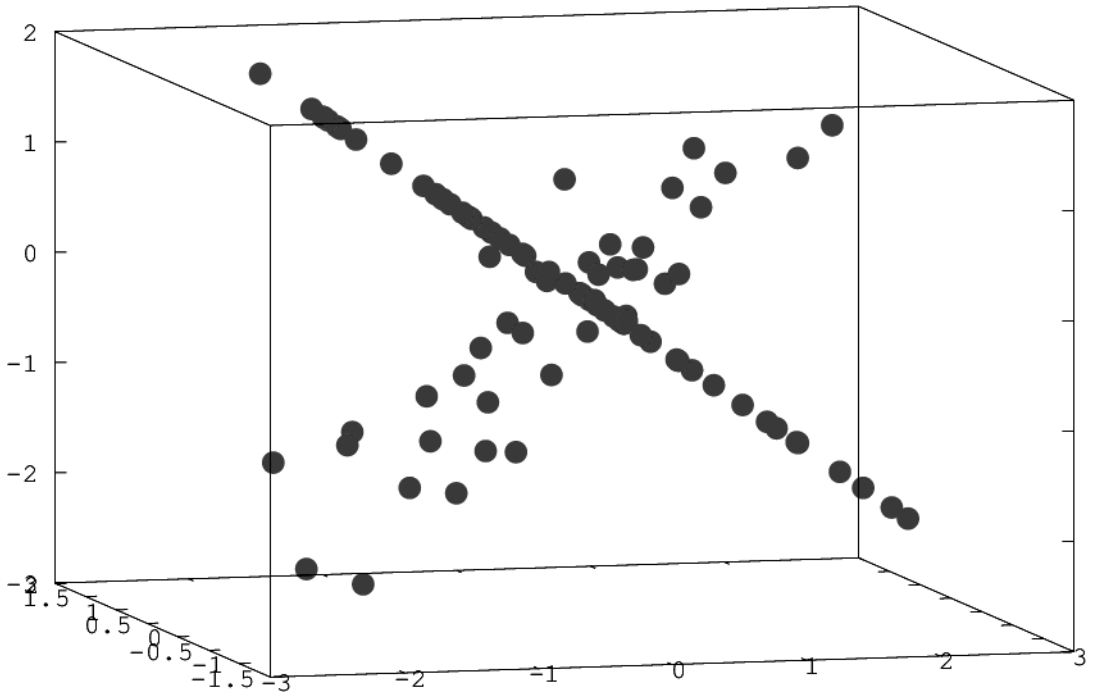
Issues for Subspace Clustering Solutions

- Subspaces of mixed dimensions.
- Subspaces of unknown dimensions.
- Unpleasantly distributed data within subspaces.
- Local lower-dimensional structure within subspaces.

Global Dimension Minimization (GDM)

- GDM is a new subspace clustering method.
- It is a “global” method. This makes it better at handling non-isotropic distributions of points, and at handling lower-dimensional manifold structure within subspaces.
- GDM can handle mixed and unknown dimensions.

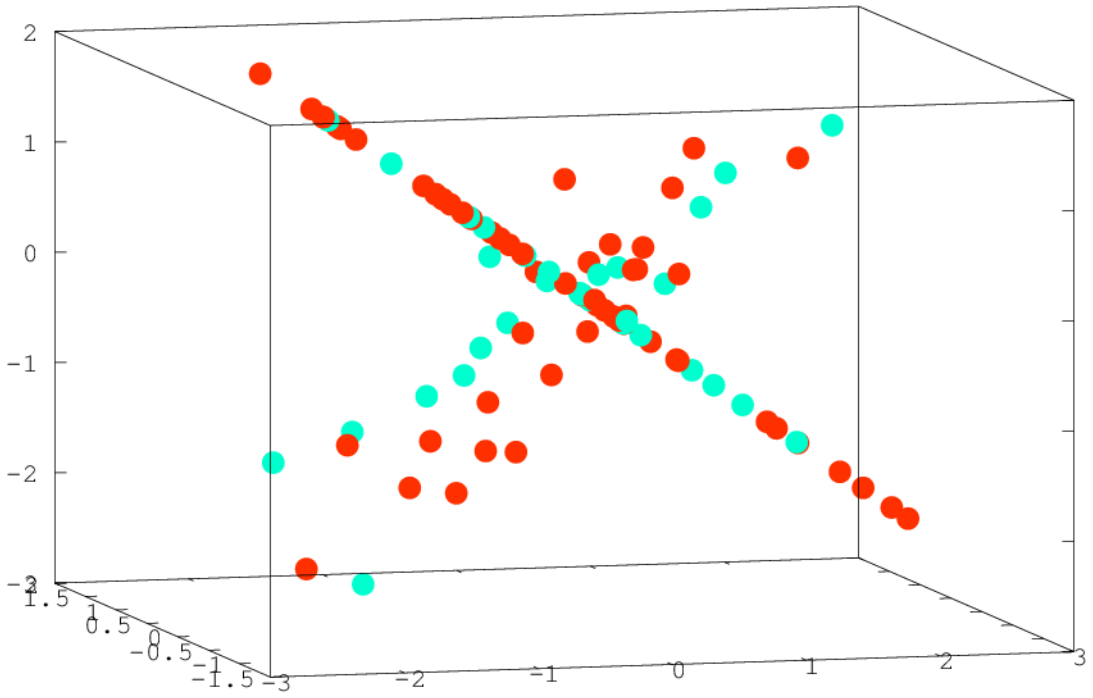
Intuition



- Consider a simple data set in $\mathbb{R}^3$, without any noise or outliers. Assume the data is contained in the union of a plane and a line

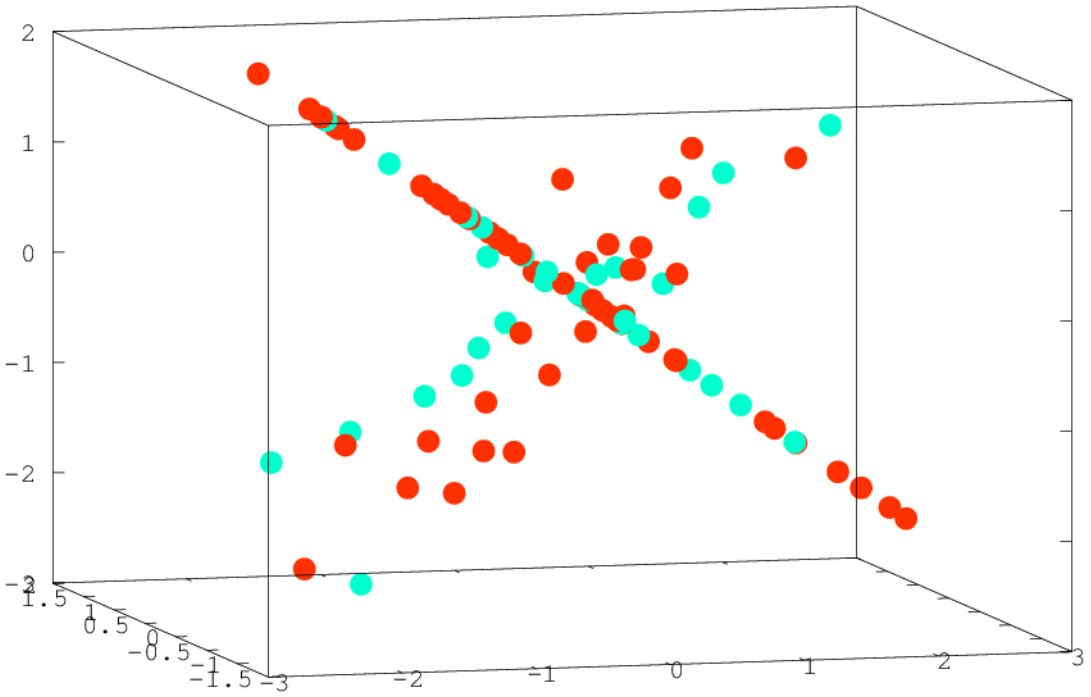
- Imagine we have access to an oracle, who will provide a reliable “dimension estimate” for any set of points.

Intuition



- Consider a random partition of the data into 2 sets (called the red set and the blue set).

Intuition



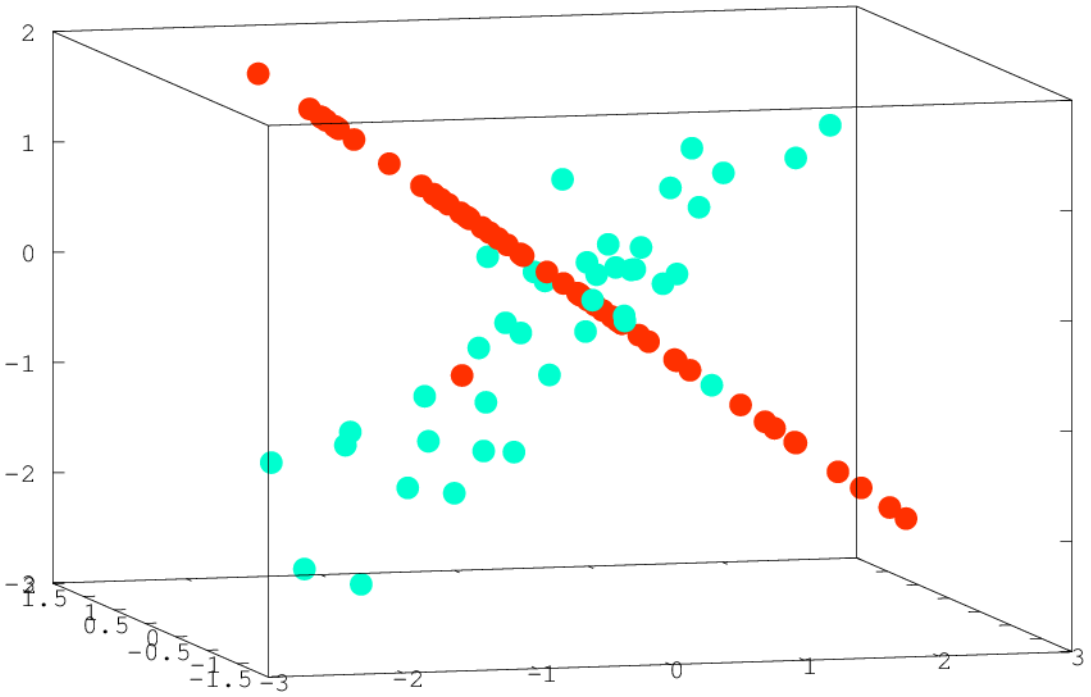
- Consider a random partition of the data into 2 sets (called the red set and the blue set).
- Ask the oracle to estimate the dimension of each set.

	Red Set	Blue Set
Dim Estimate	≈3	≈3

→

Total Dimension
≈6

Intuition



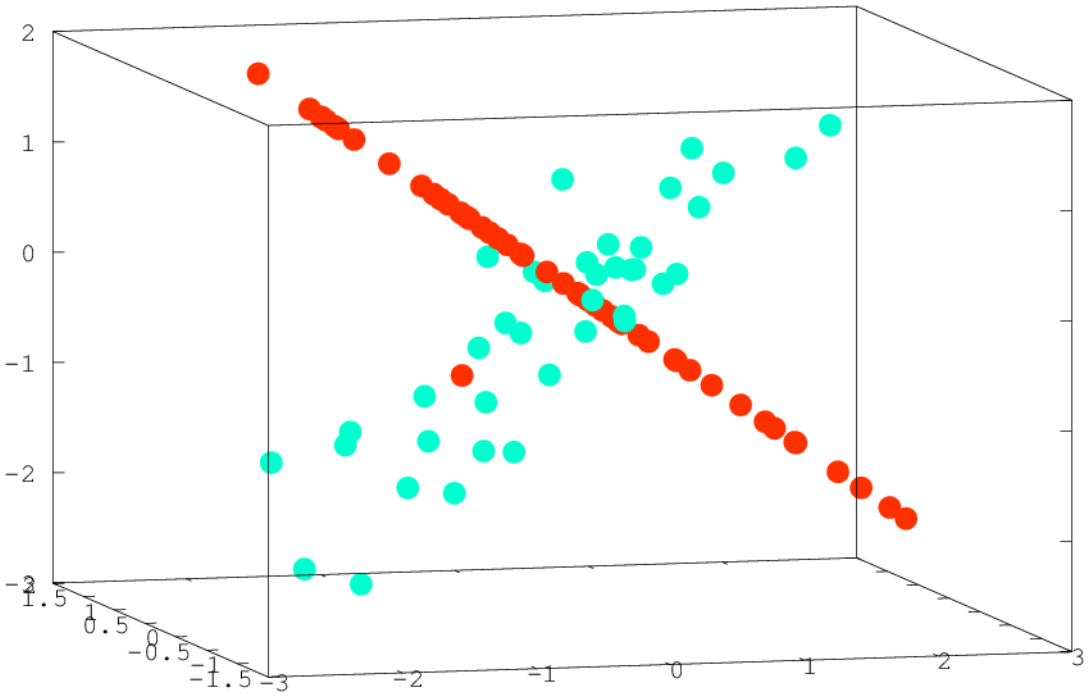
- Now consider a partition which is close to perfect, but with a few errors.

	Red Set	Blue Set
Dim Estimate	≈2	≈3

→

Total Dimension
≈5

Intuition



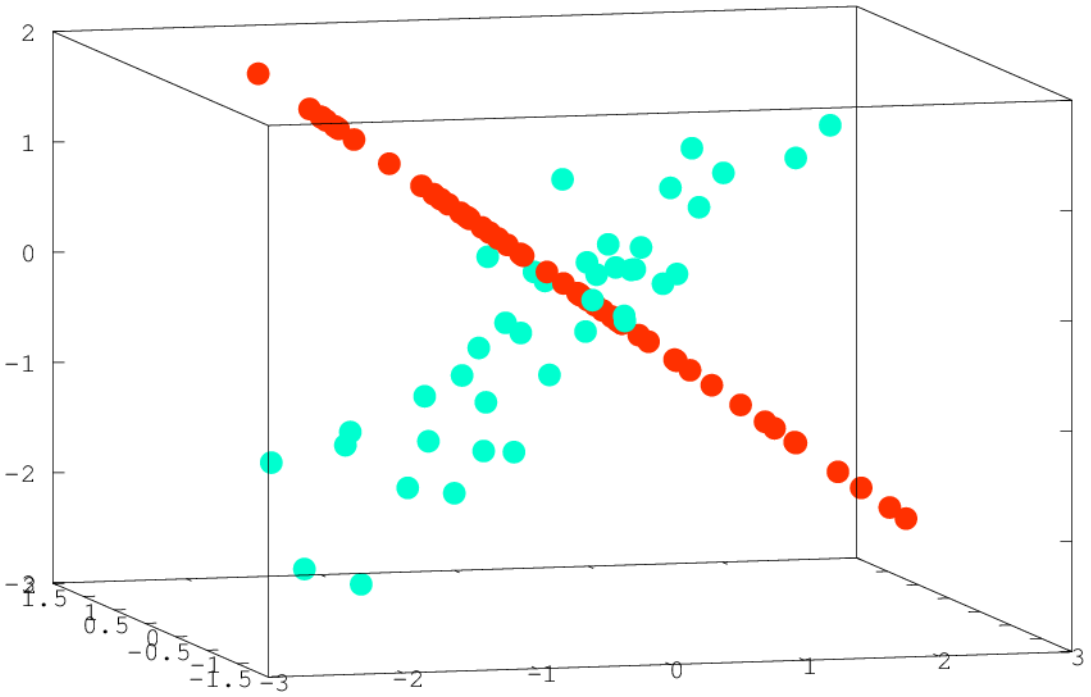
- Now consider a partition which is very close to perfect (1 error).

	Red Set	Blue Set
Dim Estimate	≈2	≈2

→

Total Dimension
≈4

Intuition



- Now consider the “correct partition”.

	Red Set	Blue Set
Dim Estimate	≈1	≈2

→

Total Dimension
≈3

Intuition

- “Total Dimension” is defined to equal the sum of the estimated dimensions of all sets in a partition.
- It seems that the best way to shrink total dimension is to group together points which come from the same underlying subspace.

A Theorem About Total Dimension

Theorem: *Let $\{L_i\}_{i=1}^K$ be independent linear subspaces in $\mathbb{R}^D$ of dimensions $\{d_i\}_{i=1}^K$ respectively, $\{\mu_i\}_{i=1}^K$ be probability measures supported on $\{L_i\}_{i=1}^K$ respectively such that for any linear subspace $L \subsetneq L_i$: $\mu_i(L) = 0$, N and $\{N_i\}_{1=1}^K$ be integers such that $N = \sum_{i=1}^K N_i$ as well as $N_i > d_i$, $i = 1, \ldots, K$ and X be a data set of N points with N_i i.i.d. samples from each μ_i. Then the natural partition of X, Π_{Nat}, is almost surely a global minimum of the total dimension function, and the only other minimizing partitions are those of which Π_{Nat} is a refinement. In particular, Π_{Nat} is the unique minimizer when considering partitions with exactly K sets.*

Subspaces are said to be “independent” if the sum of their dimensions equals the dimension of their union.

This theorem has little practical value – total dimension does not behave well when we have noisy data. We will see a better result later.

What is the next step to apply this idea?

What is the next step to apply this idea?

- Develop a way of reliably estimating the dimensionality of a set on our own.
- There are many solutions to dimension estimation. We will introduce a new concept: Empirical Dimension

Empirical Dimension

- Empirical Dimension is a way of estimating the dimensionality of a set based on singular values.
- Suppose we want to estimate the dimension of the set of vectors:

$$(\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N)$$

we stack these vectors into a matrix:

$$\mathbf{A} = \begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ \mathbf{v}_1 & \mathbf{v}_2 & \dots & \mathbf{v}_N \\ \downarrow & \downarrow & & \downarrow \end{pmatrix}$$

- Let σ denote the vector of singular values of $\mathbf{A}$.
- For a fixed $\epsilon \in (0, 1]$ define the "empirical dimension" of the set of vectors to be:

$$\hat{d}_\epsilon(\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N) := \frac{\|\sigma\|_\epsilon}{\|\sigma\|_{\left(\frac{\epsilon}{1-\epsilon}\right)}}$$

Empirical Dimension

$$\hat{d}_\epsilon(\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N) := \frac{\|\sigma\|_\epsilon}{\|\sigma\|_{\left(\frac{\epsilon}{1-\epsilon}\right)}}$$

- Empirical Dimension enjoys the following properties:
  - Scale invariance
  - Invariant under rotations of space
  - If $(\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N)$ are contained in a d-subspace then $d_\epsilon \leq d$
  - If $(\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N)$ are sampled from a spherically symmetric distribution in a d-subspace then $d_\epsilon \approx d$

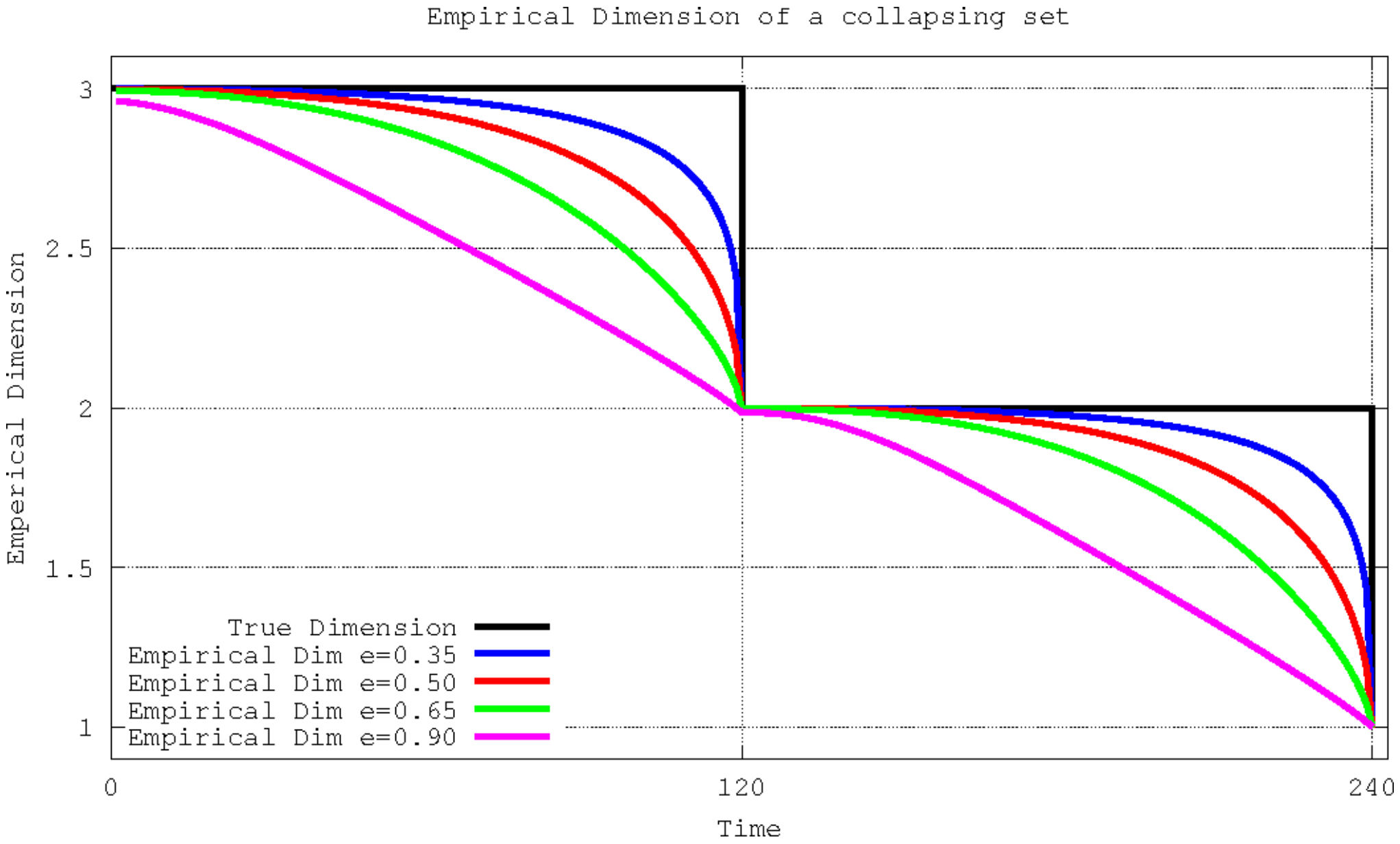
Insert Math

Empirical Dimension

- We defined an entire class of dimension estimators: one for each $\epsilon \in (0, 1]$
- What effect does ϵ have on the behavior of the estimator?

- It turns out that ϵ is a dial for tuning the “strictness” of the dimension estimator
- This is best demonstrated visually...

Empirical Dimension



Empirical Dimension - Summary

- Empirical dimension provides an estimate of the dimensionality of a set.
- This estimate is a continuous function of the input vectors.
- We can “tune” empirical dimension, via a single parameter (ϵ), so as to make it more or less “strict”.

Back to Total Dimension

- Now that we have a way of estimating the dimension of a set, we re-define total dimension:

$$\mathrm{TD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_1 = \sum_{i=1}^{K} \hat{d}_{\epsilon,i}$$

Here, $\hat{d}_{\epsilon,i}$ is the “empirical dimension” of the i'th set in the partition Π.

- To perform subspace clustering, one would want to find the partition with lowest total dimension.

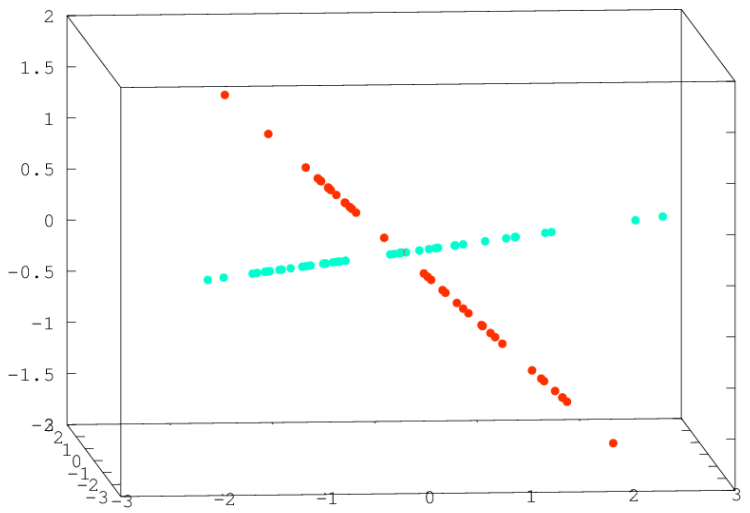
Does the “Natural Partition” Actually Minimize Total Dimension in the Real World?

- If the absence of noise, and when dealing with independent subspaces (i.e. high ambient dim.) then yes.
- If we have low ambient dimension, then our subspaces will not be independent.
- Noise will inflate the empirical dimension of any set. This can have undesirable effects on Total Dimension.
- Either of the above situations can result in degenerate partitions having lower total dimension than the natural partition.

Does the “Natural Partition” Actually Minimize Total Dimension in the Real World?

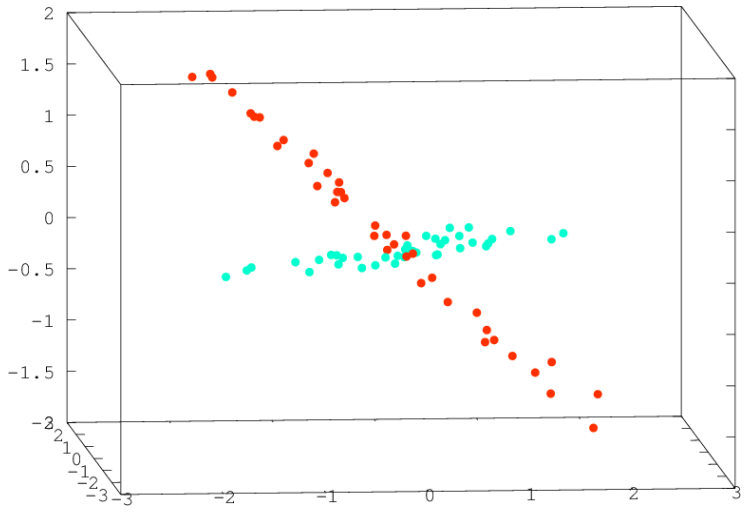
- If the absence of noise, and when dealing with independent subspaces (i.e. high ambient dim.) then yes.
- If we have low ambient dimension, then our subspaces will not be independent.
- Noise will inflate the empirical dimension of any set. This can have undesirable effects on Total Dimension.
- Either of the above situations can result in degenerate partitions having lower total dimension than the natural partition.

The Problem With Noise and Total Dimension



Noise Free Case:

	d_ϵ (set 1)	d_ϵ (set 2)	Total Dim
Natural Partition	1	1	2
Degenerate Partition	2		2



Noisy Case:

	d_ϵ (set 1)	d_ϵ (set 2)	Total Dim
Natural Partition	1.3	1.3	2.6
Degenerate Partition	2.3		2.3

Global Dimension

$$\mathrm{TD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_1$$

$$\mathrm{GD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_p \qquad p \geq 1$$

Global Dimension

$$\mathrm{TD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_1$$

$$\mathrm{GD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_p \qquad p \geq 1$$

Optimization objective for Total Dimension: Find a partition with K sets, such that each set has low empirical dimension. You may increase the dimension on one set, so long as your modification decreases the dimension of another set by at least that much.

Optimization objective for Global Dimension (p large): Find a partition with K sets, such that the maximum dimension of all sets in your partition is as low as possible. If there are decisions to make which don't effect the dimension of the “largest” set, then make those decisions so as to minimize the dimensions of the other sets.

Global Dimension

$$\mathrm{TD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_1$$

$$\mathrm{GD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_p \qquad p \geq 1$$

Theorem:

Consider a data set sampled from a mixture of K distinct, non-degenerate measures on d-subspaces, with each subspace sufficiently represented. Then, there exists p large enough that, amongst partitions of the data set into K or fewer sets, the natural partition is almost surely the unique minimizer of GD.

Global Dimension

$$\mathrm{GD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_p \qquad p \geq 1$$

- Global Dimension (large p) is far more robust to noise than Total Dimension.
- Global Dimension (large p) works when our subspaces are not independent.

- How large should we make p?
  - As large as we can without causing numerical problems.
  - In practice, p=15 seems large enough, and does not cause problems.

Minimizing Global Dimension

- Now that we have an objective function, how do we minimize it?
- The domain is the set of all partitions of a data set. This is a discrete domain.
- 0^{th}-order methods (Genetic Algorithms) can be applied, but these are very slow.

Global Dimension Minimization (GDM)

- GDM is a 1st-order method, based on the gradient projection algorithm. Several problems must be solved to make this approach possible:

- Step 1: Extend/Reformulate the problem so that GD is a smooth function, defined on a convex domain.
- Step 2: Find a fast way of computing the gradient of GD.
- Step 3: Find a way of projecting an arbitrary state into the domain of optimization.
- Step 4: Solve the step-size problem.
- Step 5: Establish Convergence Criteria.

Global Dimension Minimization (GDM)

- GDM is a 1st-order method, based on the gradient projection algorithm. Several problems must be solved to make this approach possible:

- Step 1: Extend/Reformulate the problem so that GD is a smooth function, defined on a convex domain.
- Step 2: Find a fast way of computing the gradient of GD.
- Step 3: Find a way of projecting an arbitrary state into the domain of optimization.
- Step 4: Solve the step-size problem.
- Step 5: Establish Convergence Criteria.

Extending Global Dimension

- This is accomplished through the use of “fuzzy assignment”
  - Instead of assigning each point a label, associating it with a single cluster, we assign each point a probability vector.
  - This vector holds the strength of the points affiliation with each cluster.
  - Each point belongs to each cluster, but in different amounts.
  - The fuzzy assignment is stored in a matrix $\mathbf{M}$, called the membership matrix. Column i of this matrix holds the assignment probability vector for point i.

Extending Global Dimension

$$\mathrm{GD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_p \qquad p \geq 1$$

$$\hat{d}_{\epsilon,k} = \hat{d}_{\epsilon} \left(\begin{array}{cccc} \uparrow & \uparrow & & \uparrow \\ \mathbf{M}_{k,1}\mathbf{v}_1 & \mathbf{M}_{k,2}\mathbf{v}_2 & ... & \mathbf{M}_{k,N}\mathbf{v}_N \\ \downarrow & \downarrow & & \downarrow \end{array} \right)$$

- In the original formulation, to estimate the dimension of the “k'th set in the partition”, we would select only those vectors which were assigned to set k and compute their empirical dimension.
- Now, we take every vector, and scale it by its membership strength to cluster k, and compute the empirical dimension of this set of scaled vectors.

Extending Global Dimension

$$\mathrm{GD}(\Pi) = \|(\hat{d}_{\epsilon,1}, \hat{d}_{\epsilon,2}, ..., \hat{d}_{\epsilon,K})^T\|_p \qquad p \geq 1$$

$$\hat{d}_{\epsilon,k} = \hat{d}_{\epsilon} \begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ \mathbf{M}_{k,1}\mathbf{v}_1 & \mathbf{M}_{k,2}\mathbf{v}_2 & ... & \mathbf{M}_{k,N}\mathbf{v}_N \\ \downarrow & \downarrow & & \downarrow \end{pmatrix}$$

- If we add 0 to a set of vectors, it does not effect the (true) empirical dimension of that set.
- If we consider a fuzzy partition where each point is fully assigned to a single set, then each point is fully expressed in the appropriate set, and it has the effect of appending 0 to all other sets.
- Thus, this extended definition agrees with our original definition when dealing with a “hard assignment” of the data set. This is therefore an extension of the first definition of global dimension.

Extending Global Dimension

- With our new definition, Global dimension is a (mostly) differentiable function of the members of the membership matrix.
- The domain of optimization is the set of all possible membership matrices. Since each column is a probability vector, this set can be viewed as the Cartesian product of N K-dimensional probability simplexes. This is a convex domain!
- The problem is now appropriate for gradient projection optimization.

Differentiating Global Dimension

- The gradient of Global Dimension can be computed (Theorem):

  Let $\mathbf{A}_k = \mathbf{U}_k \mathbf{\Sigma}_k (\mathbf{V}_k)^T$ be the thin SVD of $\mathbf{A}_k$, the data matrix scaled by each vectors association with set k. Let σ_k be the diagonal of $\mathbf{\Sigma}_k$.

- Let $\delta = \epsilon/(1-\epsilon)$ and define:

$$\mathbf{D}_k = \left(\frac{\|\sigma_k\|_\epsilon^{1-\epsilon} \|\sigma_k\|_\delta}{\|\sigma_k\|_\delta^2} \right) \cdot (\mathbf{\Sigma}_k)^{\epsilon-1} - \left(\frac{\|\sigma_k\|_\epsilon \|\sigma_k\|_\delta^{1-\delta}}{\|\sigma_k\|_\delta^2} \right) \cdot (\mathbf{\Sigma}_k)^{\delta-1}$$

- Then, the derivative of Global Dimension with respect to an arbitrary element of $\mathbf{M}$ is given by:

$$\frac{\partial G}{\partial \mathbf{M}_{(k,n)}} = \mathbf{V}_{k(n,:)} \left((\hat{d}_\epsilon^k)^{p-1} \, \| (\hat{d}_\epsilon^1, \hat{d}_\epsilon^2, ..., \hat{d}_\epsilon^K) \|_p^{1-p} \, \mathbf{D}_k \, (\mathbf{U}_k)^T \right) \mathbf{A}_{(:,n)}$$

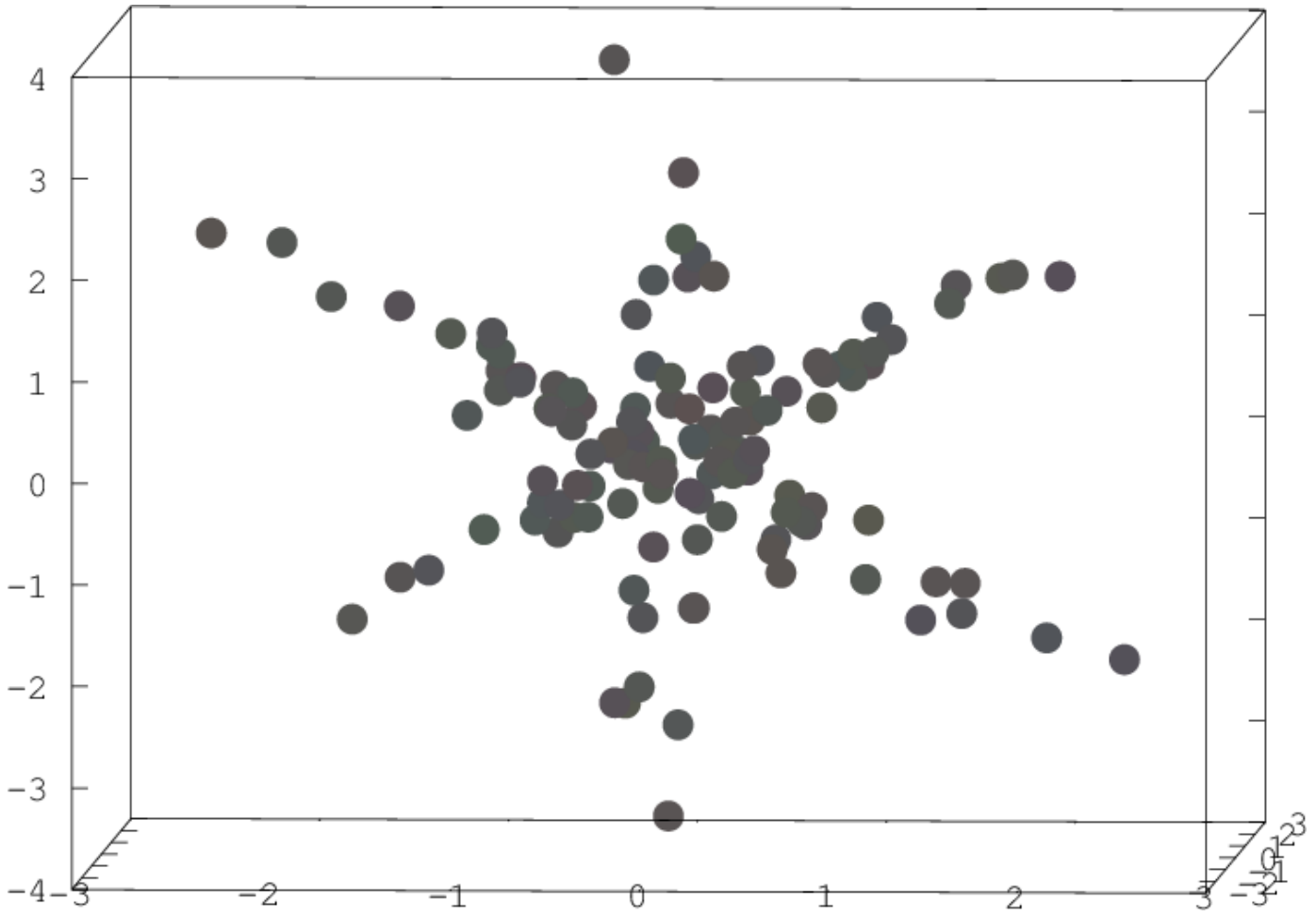
Overview of GDM

- Initialize the membership matrix (randomly or smartly)
- Do until convergence:
  - Compute gradient of GD.
  - Take a small step in direction $-1 * \nabla GD$.
  - Project new state back into domain of optimization.
- Threshold the membership matrix and extract “hard assignment”.

Global dimension is a non-convex function - we are not guaranteed to converge to the correct state. Therefore:

- Clever initialization can be very helpful.
- We can improve reliability by running the algorithm several times and returning the best of all runs (as determined by global dimension).

Example on Synthetic Data



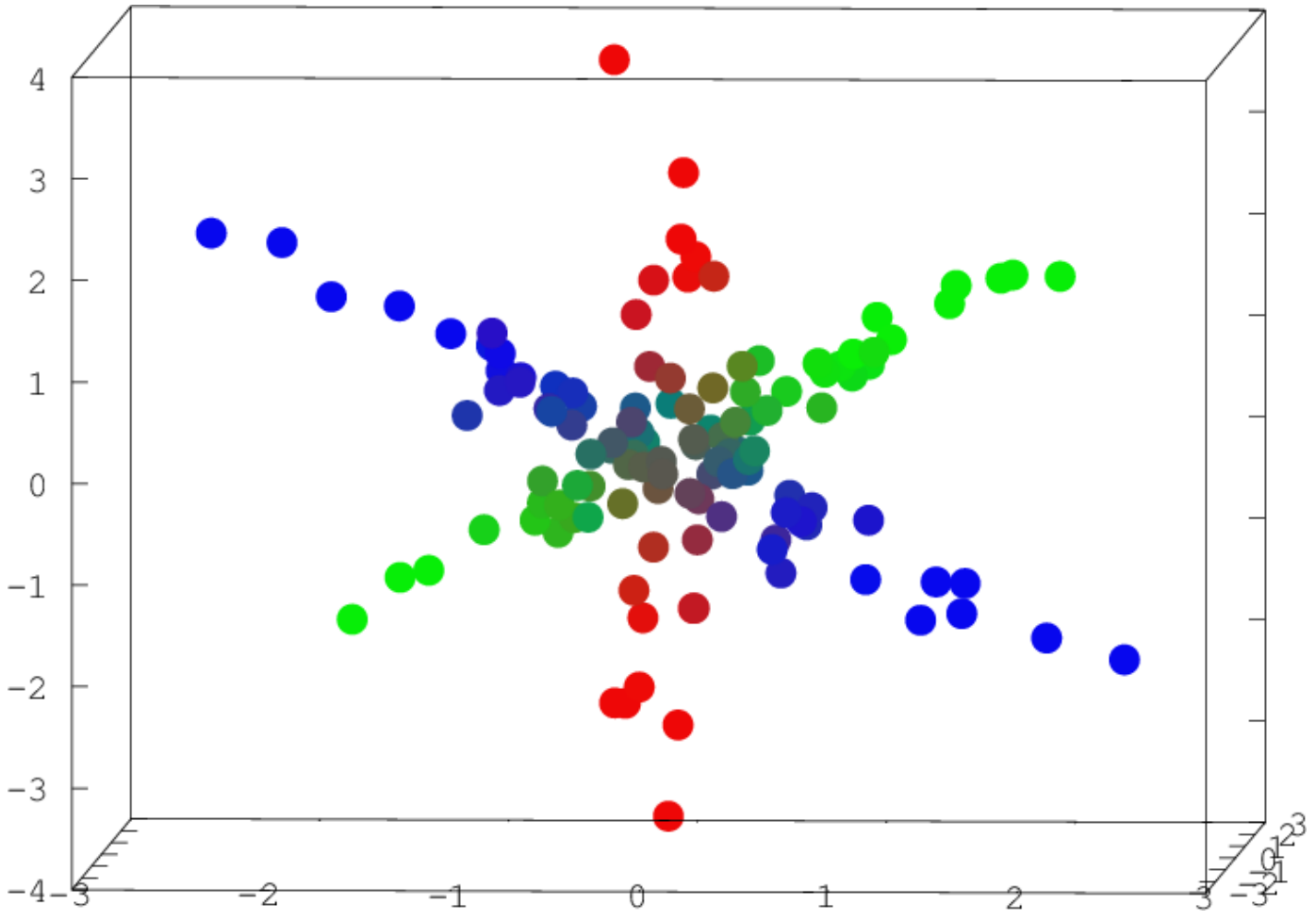
Date set sampled from a union of 3 subspaces:

- 1 line
- 2 planes

Heavy Gaussian noise added to every sample.

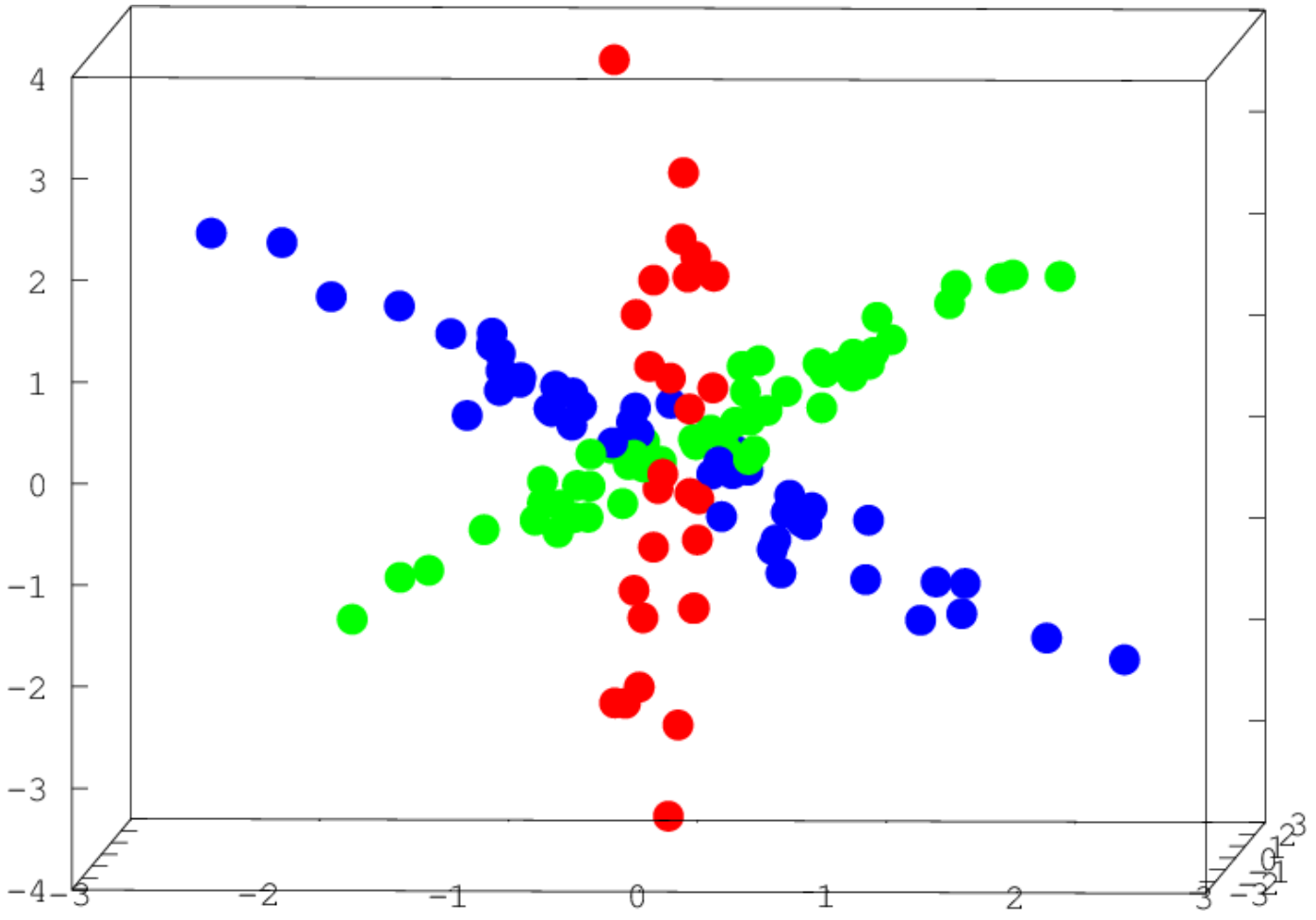
- See Video!

Example on Synthetic Data



Final State:
Before Thresholding

Example on Synthetic Data



Final State:
After Thresholding

Results on Real 2-View Data

The “RAS” dataset consists of 13 two-view sequences. Most scenes contain multiple rigid or approximately rigid objects, moving independently

The dataset provides pre-tracked point correspondences for each sequence. Outliers were removed for this experiment.

Each point correspondence was mapped into $\mathbb{R}^9$ via the Kronecker embedding. Various HLM techniques were applied to try and segment the embedded data. Other methods were also applied for comparison.

Results on File #6 of RAS Dataset

SCC

GDM

MAPA

Results on Real 2-View Data

Table 1: Misclassification Rates (given as % Error) on the outlier-free RAS database.

Method/Embedding	File Number												
	1	2	3	4	5	6	7	8	9	10	11	12	13
GDM – Nonlin	**0.85**	**0**	2.76	0.65	**0**	**0**	**0**	12.50	**0**	**0**	**0**	**0**	**0**
SCC – Linear	**0.85**	**0**	1.29	0.70	**0**	3.56	**0**	1.10	0.30	**0**	**0**	1.07	0.21
SCC – Nonlin	**0.85**	2.69	24.09	**0.06**	**0**	16.71	0.27	**0.02**	**0**	9.41	4.67	1.17	1.53
MAPA – Linear	**0.85**	3.65	1.18	0.64	**0**	13.69	15.97	1.28	**0**	**0**	**0**	0.67	3.29
MAPA – Nonlin	**0.85**	20.54	21.65	0.64	**0**	21.91	6.25	7.73	**0**	13.97	1.42	0.33	3.29
SSC – Linear	1.69	18.26	**0.78**	1.93	**0**	**0**	6.25	32.24	**0**	**0**	14.64	1.34	4.39
SSC – Nonlin	1.27	**0**	22.44	0.64	**0**	21.91	**0**	9.02	**0**	13.97	9.28	12.12	6.59
SLBF – Linear	**0.85**	0.45	1.18	0.64	**0**	**0**	**0**	0.25	**0**	**0**	**0**	0.67	4.39
SLBF – Nonlin	**0.85**	**0**	5.11	1.93	**0**	19.17	**0**	10.56	**0**	13.97	**0**	1.68	14.28
LRR – Linear	5.08	24.65	1.18	2.58	2.14	2.73	**0**	29.12	**0**	**0**	8.92	14.81	18.68
LRR – Nonlin	1.27	9.13	2.75	1.93	**0**	**0**	3.47	3.60	**0**	**0**	9.64	18.18	2.19
RAS	**0.85**	**0**	15.4	5.16	**0**	**0**	15.1	11.1	**0**	**0**	**0**	0.34	18.7
HOSC d=2	**0.85**	**0**	30.3	1.29	**0**	**0**	**0**	22.9	**0**	**0**	**0**	**0**	2.20
HOSC d=3	**0.85**	**0**	30.3	2.60	**0**	**0**	19.2	13.4	**0**	41.9	1.1	0.34	13.2

Thank You!

Bryan Poling