THE UNIVERSITY of TENNESSEE KNOXVILLE

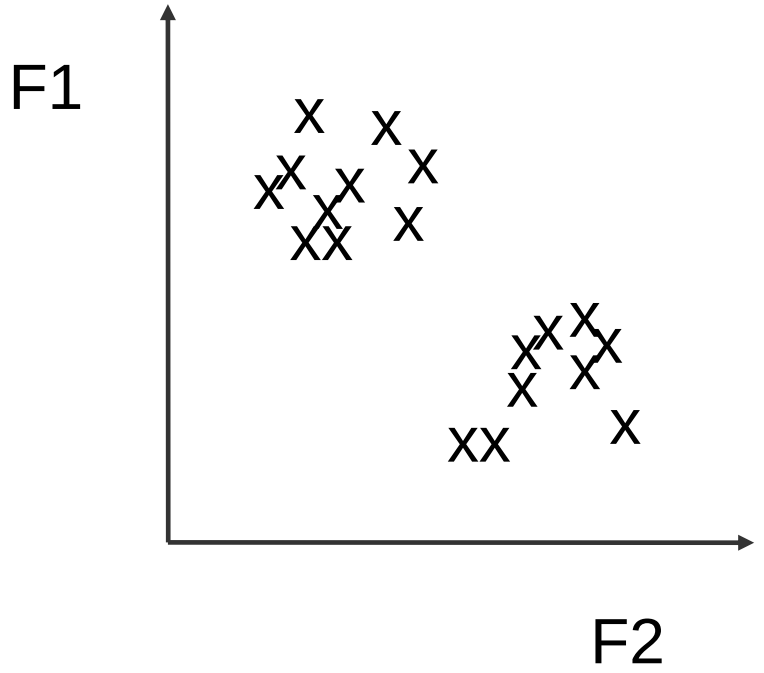
Day 3: Clustering and Linear Methods

Instructor: Sergei V. Kalinin

Clustering

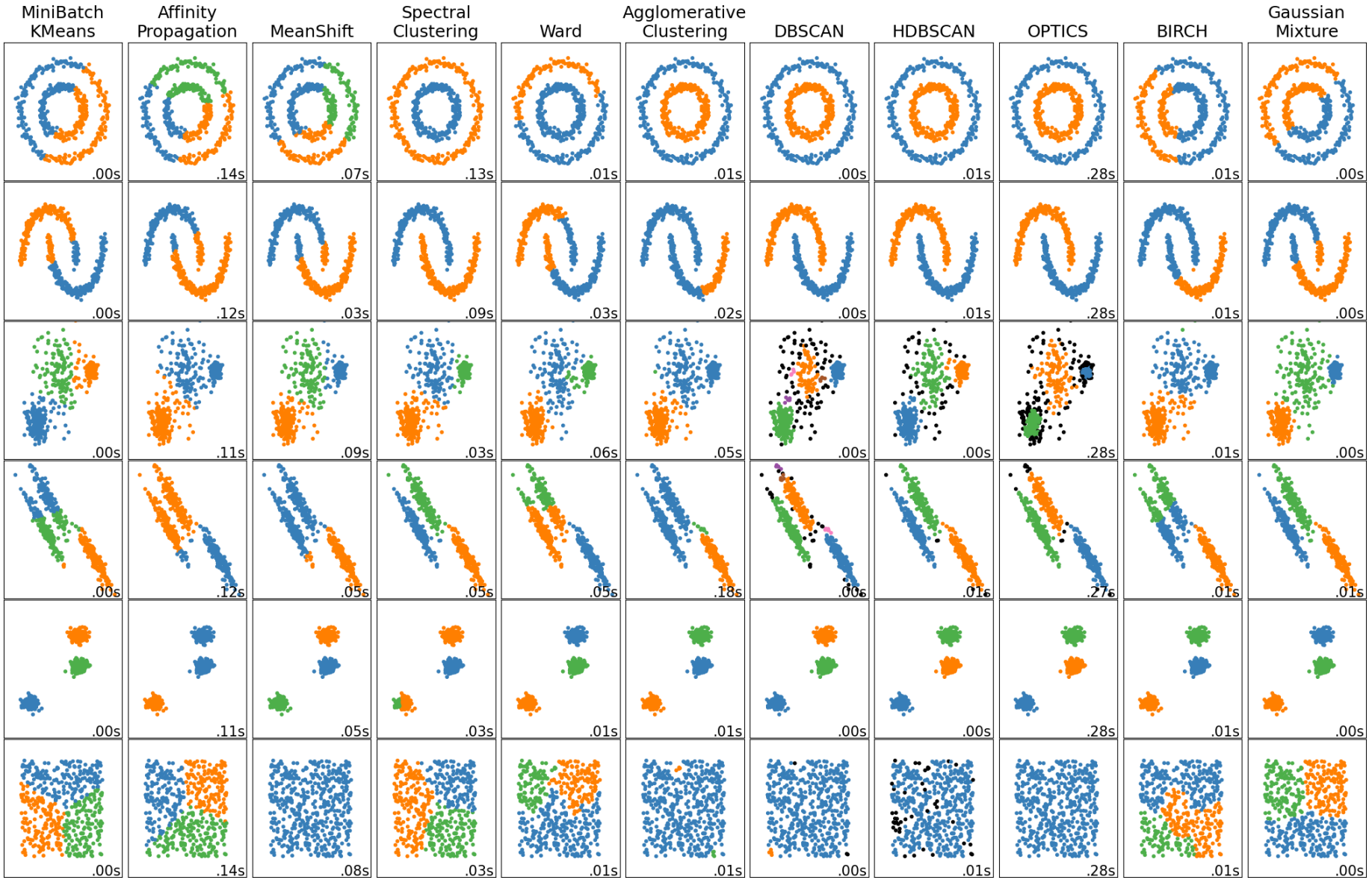
- The process of grouping a set of objects into classes of similar objects
  - Objects within a cluster should be similar.
  - Objects from different clusters should be dissimilar.
- The most common form of *unsupervised learning*

- Given a set of data points, each described by a set of attributes, find clusters such that:
  - Inter-cluster similarity is maximized
  - Intra-cluster similarity is minimized
- Requires the definition of a similarity measure



https://en.wikipedia.org/wiki/Cluster_analysis

Clustering: good news and bad news



https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

Clustering: Data Structures

- Hierarchical clustering
- K-means clustering
- Gaussian Mixture Models
- Density-based clustering
- Spectral clustering

Data matrix (two modes)

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

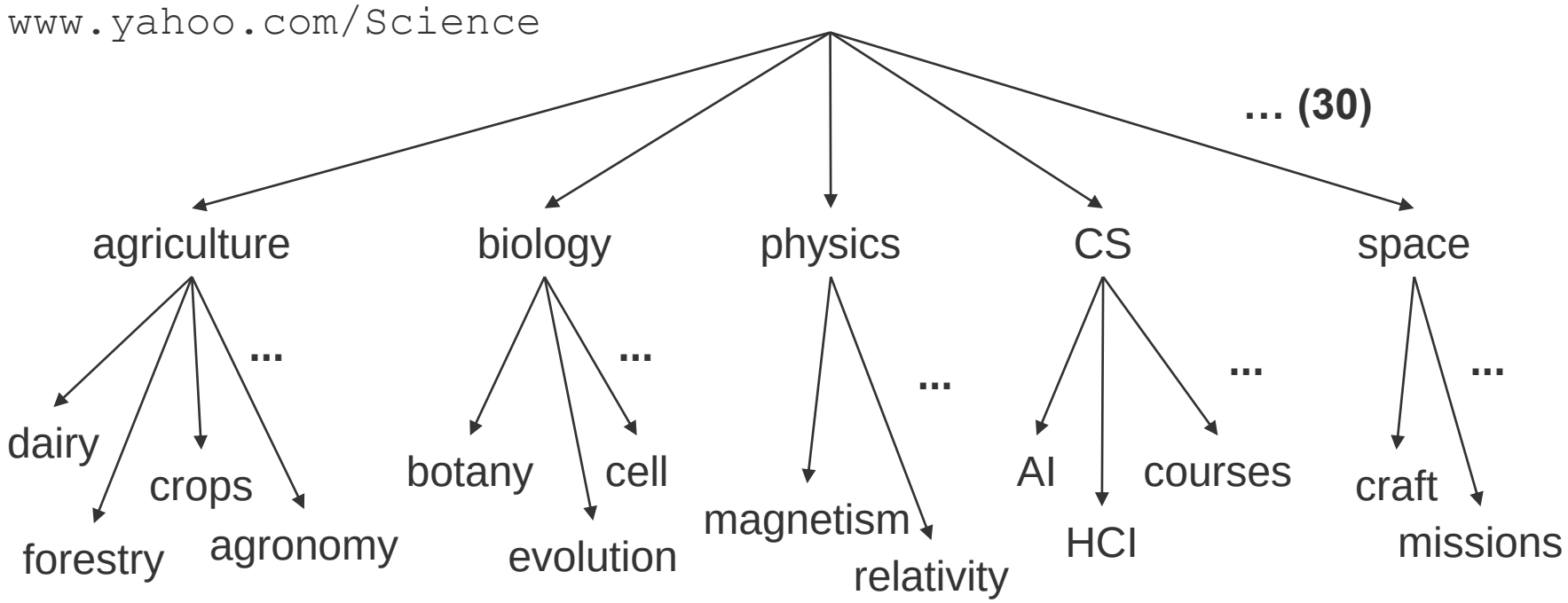
Dissimilarity matrix (one mode)

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

From: Han and Kamber, Data Mining: Concepts and Techniques

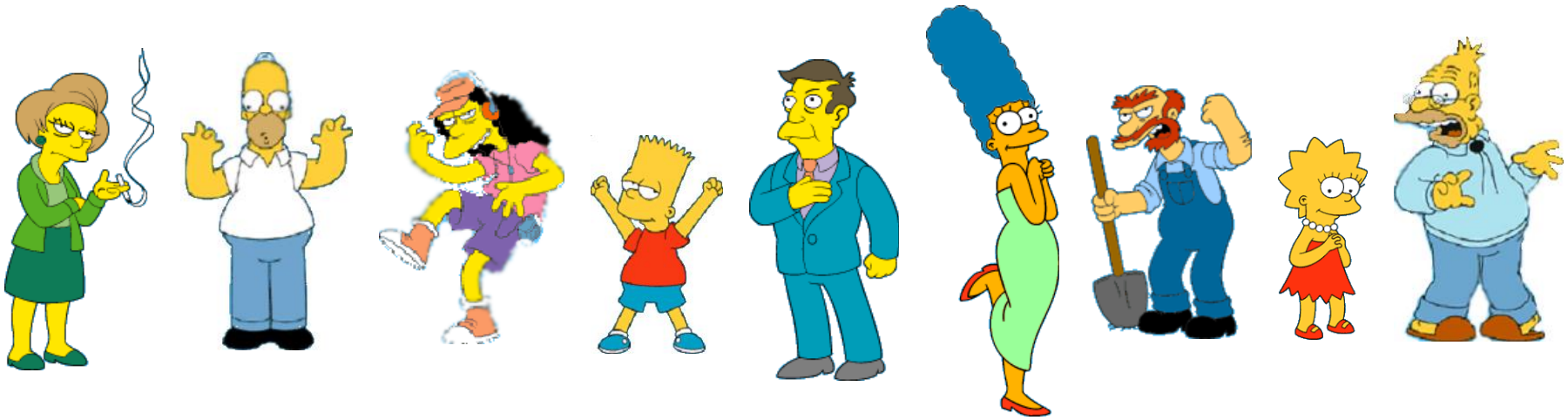
Clustering

Yahoo! Hierarchy *isn't* clustering but *is* the kind of output you want from clustering



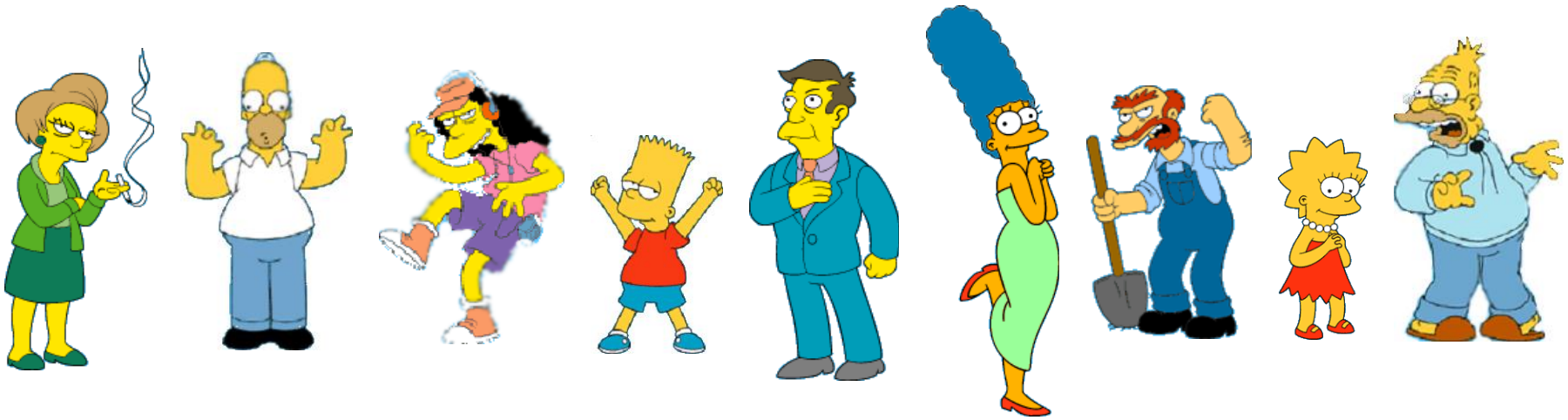
What is a natural grouping of these objects?

Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

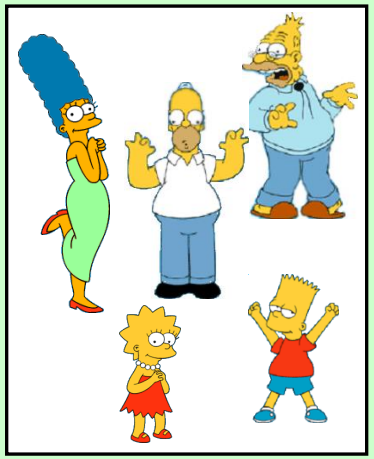


What is a natural grouping of these objects?

Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

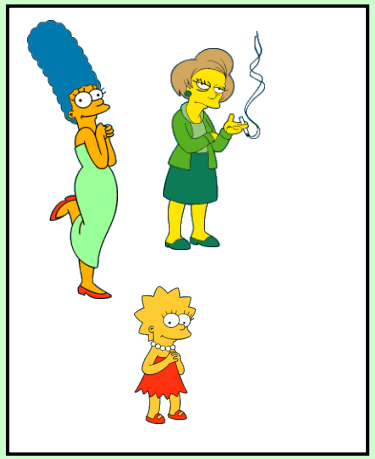


Clustering is subjective



Simpson's Family

School Employees



Females

Males

What is Similarity?

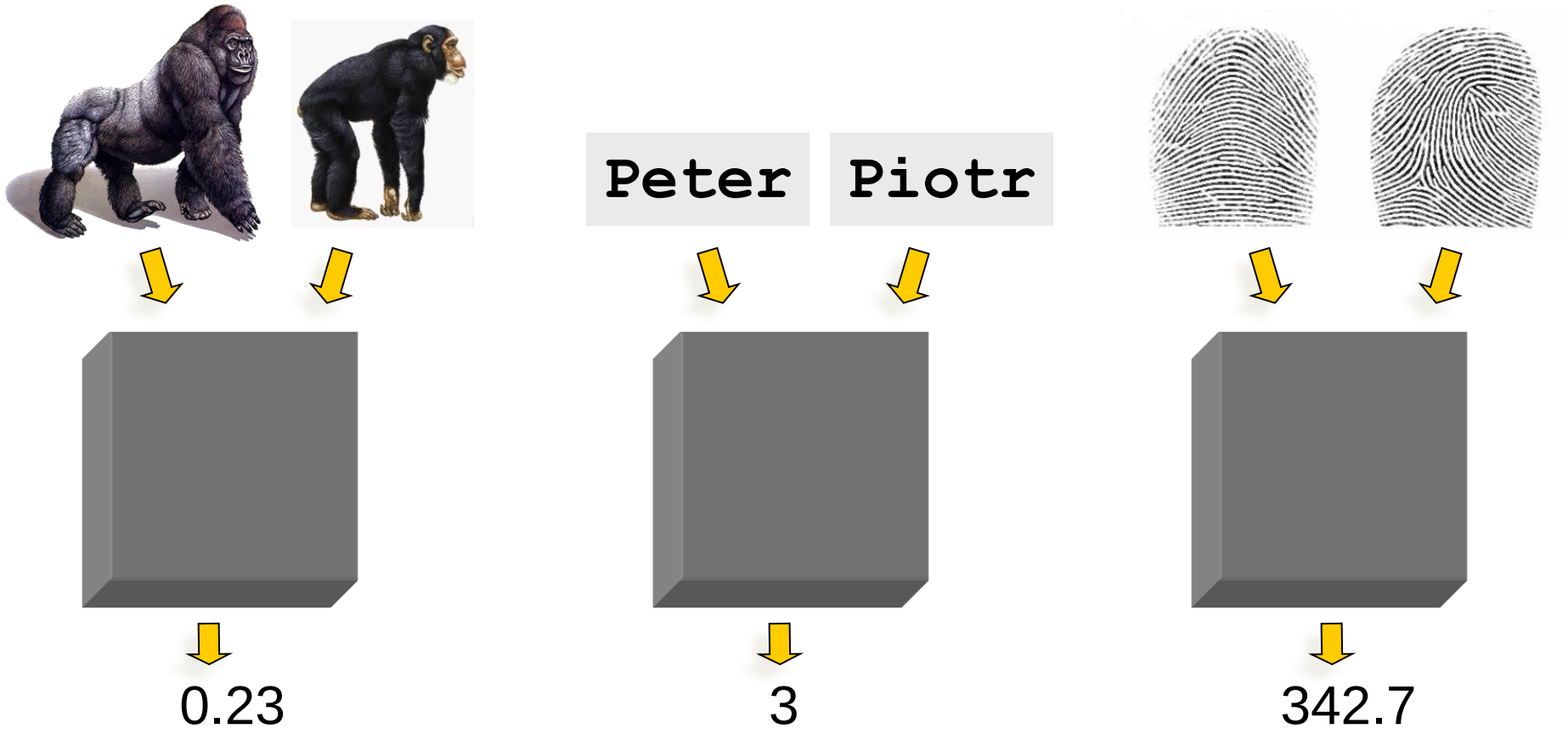
Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

Similarity is hard to define, but…
"*We know it when we see it*"

Defining Distance Measures

Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

Definition: Let O_1 and O_2 be two objects from the universe of possible objects. The distance (dissimilarity) between O_1 and O_2 is a real number denoted by $D(O_1,O_2)$



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

What properties should a distance measure have?

- $D(A,B) = D(B,A)$ *Symmetry*
- $D(A,A) = 0$ *Constancy of Self-Similarity*
- $D(A,B) = 0$ iif $A = B$ *Positivity (Separation)*
- $D(A,B) \leq D(A,C) + D(B,C)$ *Triangular Inequality*

Intuitions behind desirable distance measure properties

$D(A,B) = D(B,A)$

Otherwise you could claim "Alex looks like Bob, but Bob looks nothing like Alex."

$D(A,A) = 0$

Otherwise you could claim "Alex looks more like Bob, than Bob does."

Intuitions behind desirable distance measure properties (continued)

$D(A,B) = 0$ IIf $A=B$

Otherwise there are objects in your world that are different, but you cannot tell apart.

$D(A,B) \leq D(A,C) + D(B,C)$

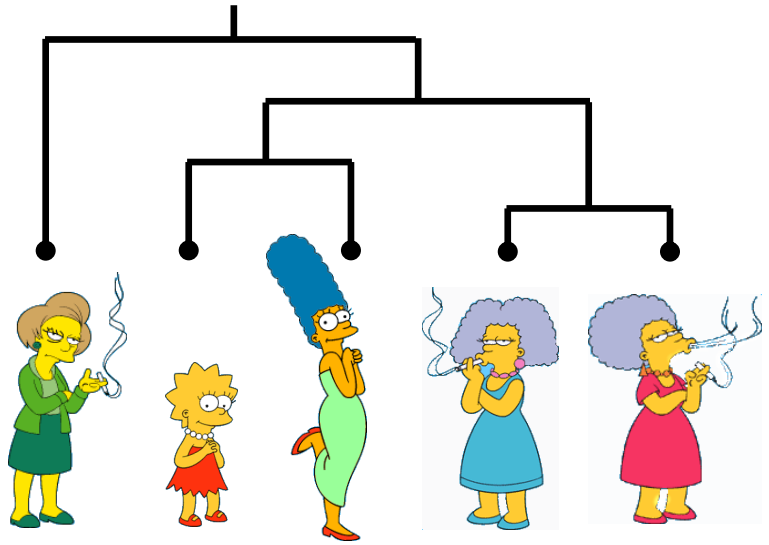
Otherwise you could claim "Alex is very like Bob, and Alex is very like Carl, but Bob is very unlike Carl."

Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

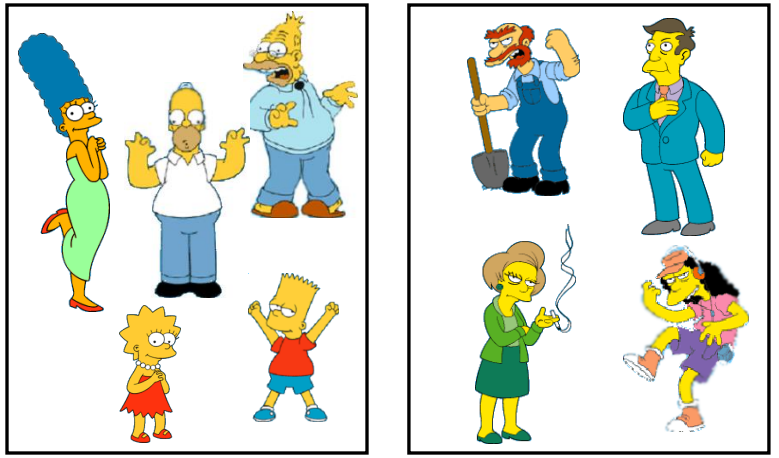
Two Types of Clustering

- **Partitional algorithms:** Construct various partitions and then evaluate them by some criterion
- **Hierarchical algorithms:** Create a hierarchical decomposition of the set of objects using some criterion

Hierarchical

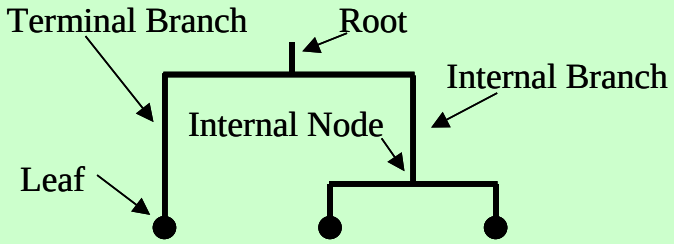


Partitional

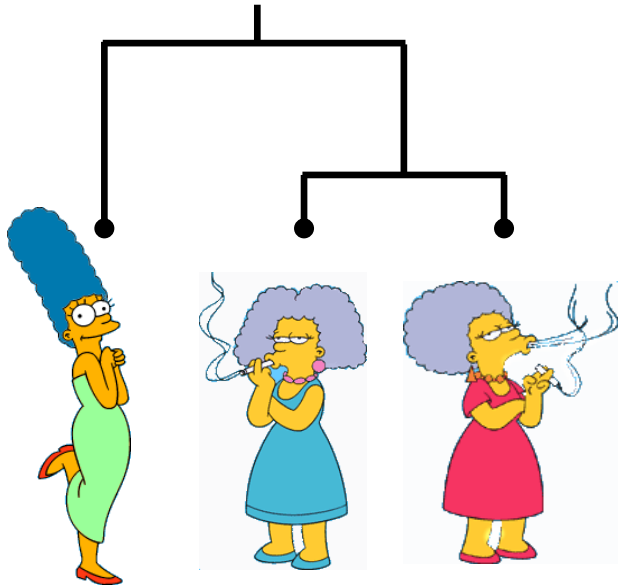


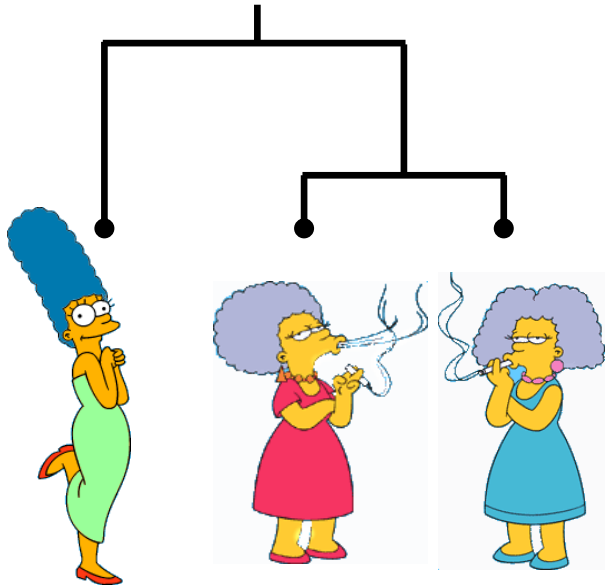
Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

Dendogram: A Useful Tool for Summarizing Similarity Measurements



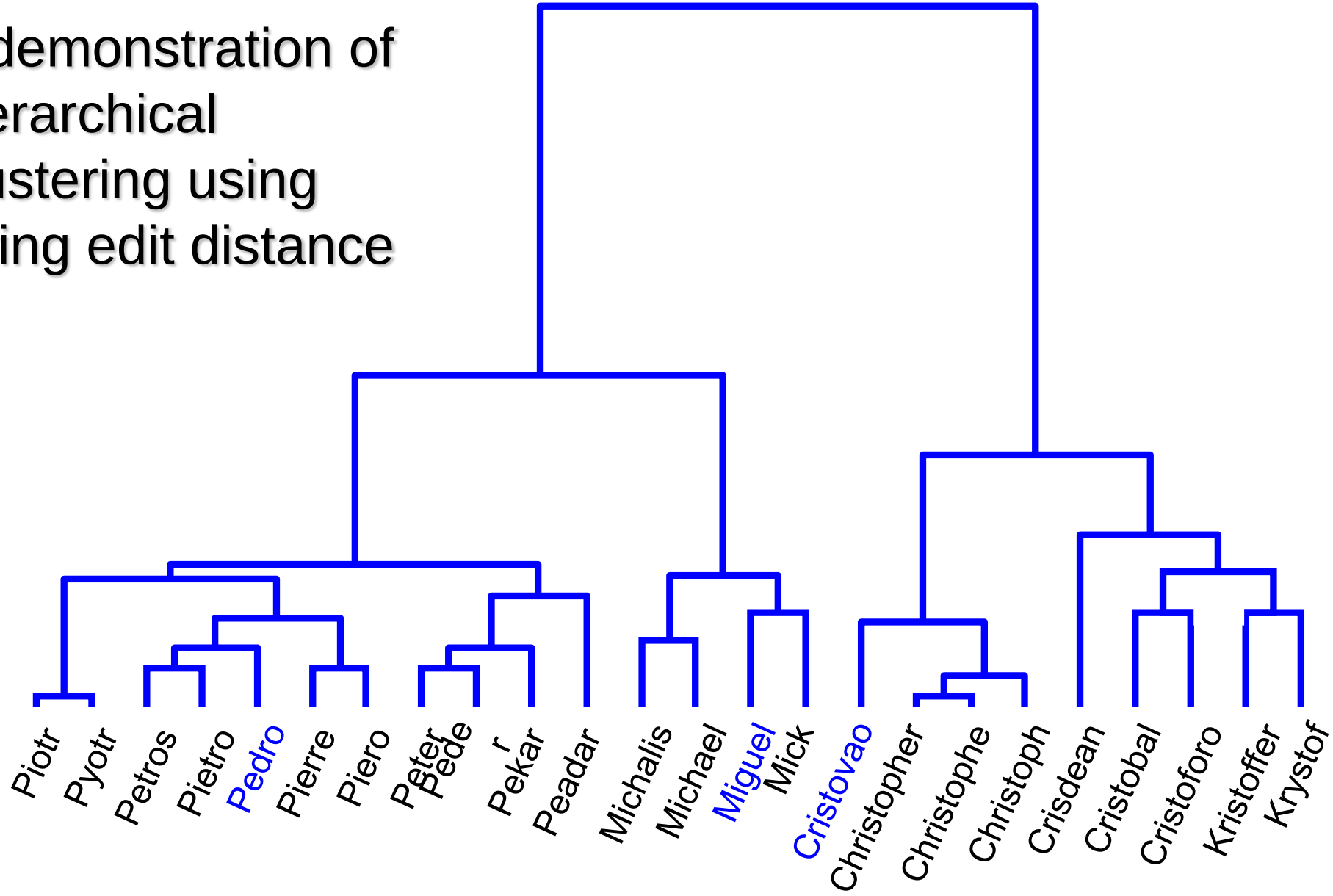
The similarity between two objects in a dendrogram is represented as the height of the lowest internal node they share.





Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

A demonstration of hierarchical clustering using string edit distance

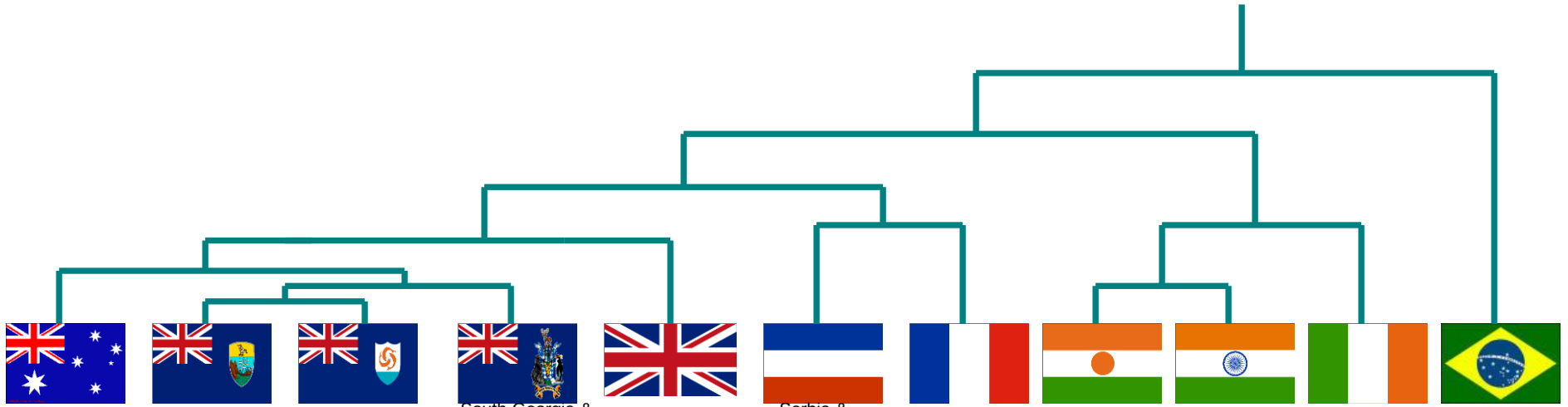


Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

Hierarchal clustering can sometimes show patterns that are meaningless or spurious

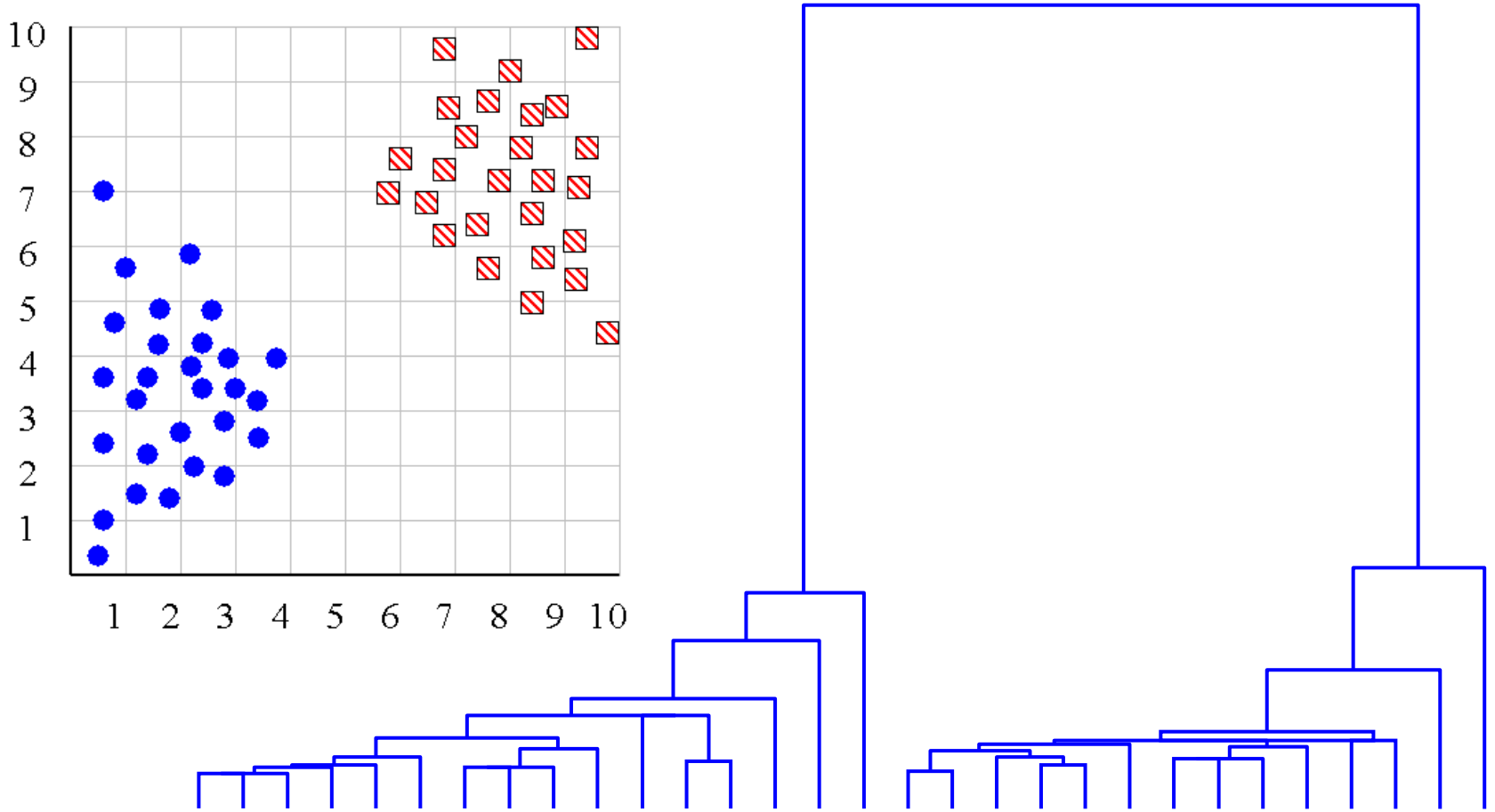
The tight grouping of Australia, Anguilla, St. Helena etc is meaningful; all these countries are former UK colonies

However the tight grouping of Niger and India is completely spurious; there is no connection between the two.



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

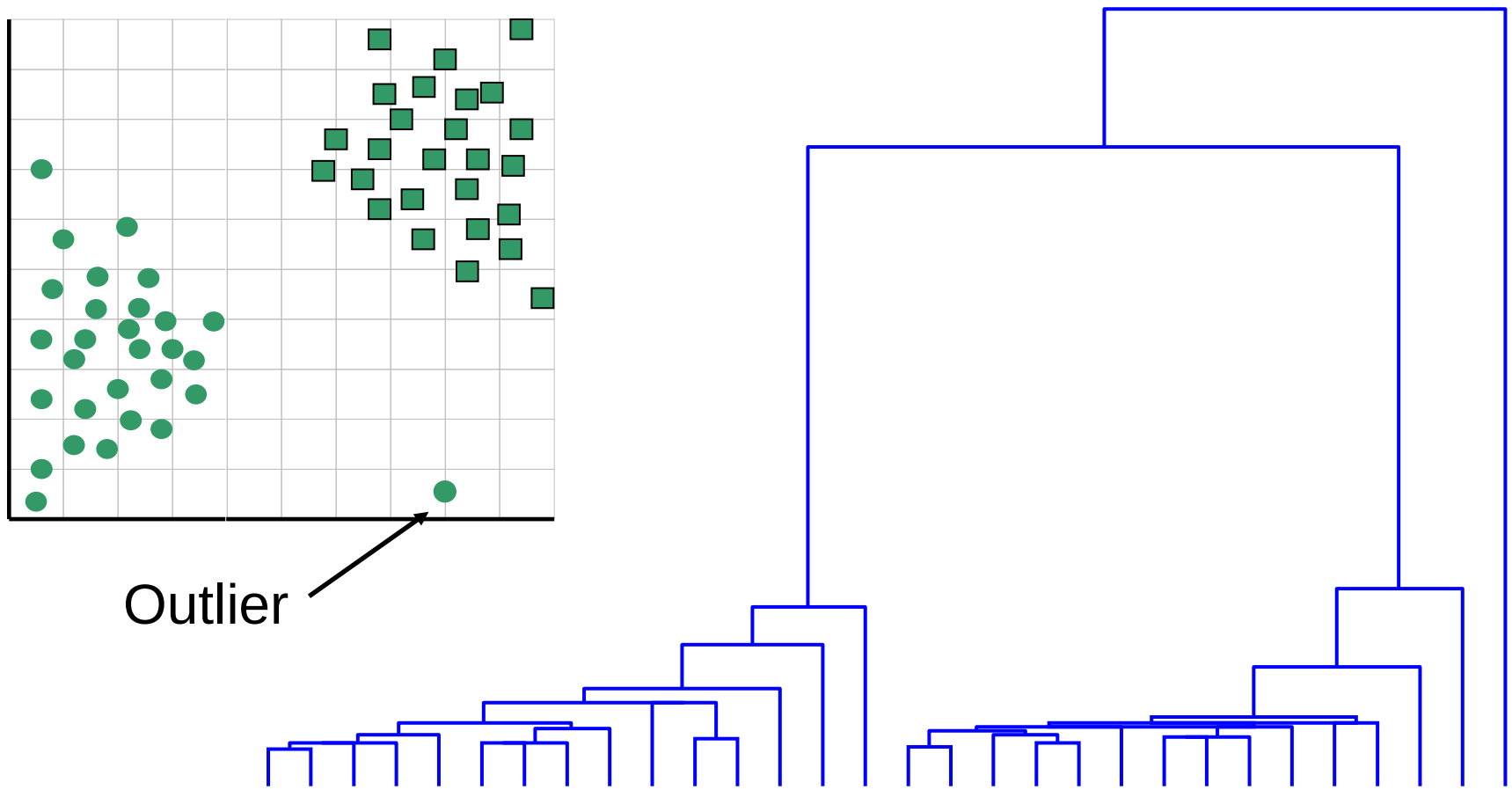
We can look at the dendrogram to determine the “correct” number of clusters.



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

One potential use of a dendrogram: detecting outliers

The single isolated branch is suggestive of a data point that is very different to all others

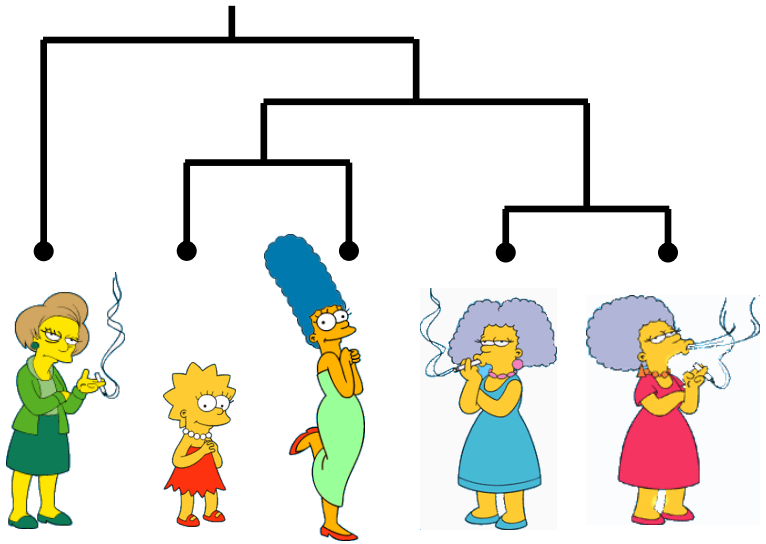


Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

Hierarchical Clustering

The number of dendrograms with n leafs $= (2n - 3)!/[(2^{(n-2)})(n-2)!]$

Number of Leafs	Number of Possible Dendrograms
2	1
3	3
4	15
5	105
...	…
10	34,459,425



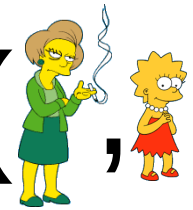
Since we cannot test all possible trees we will have to heuristic search of all possible trees. We could do this..

Bottom-Up (agglomerative): Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

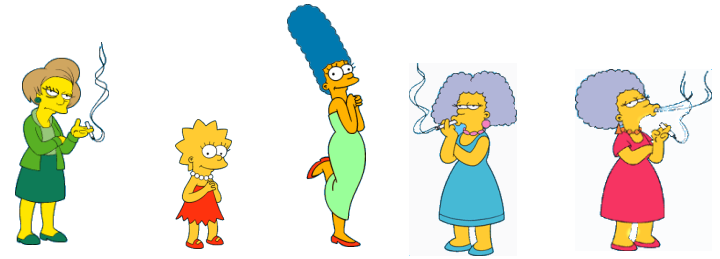
Top-Down (divisive): Starting with all the data in a single cluster, consider every possible way to divide the cluster into two. Choose the best division and recursively operate on both sides.

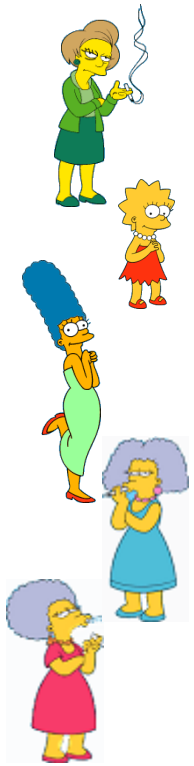
Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

We begin with a distance matrix which contains the distances between every pair of objects in our database.

D() = 8

D() = 1



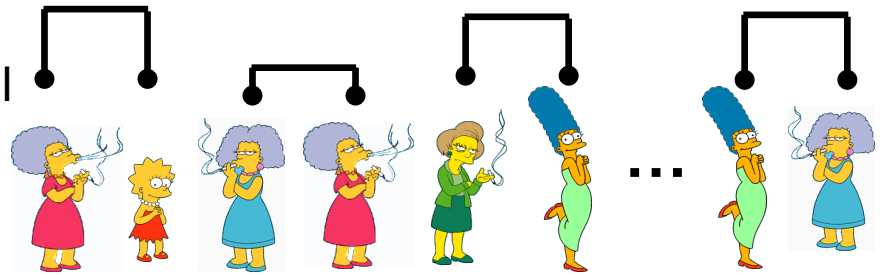


0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

Bottom-Up (agglomerative): Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

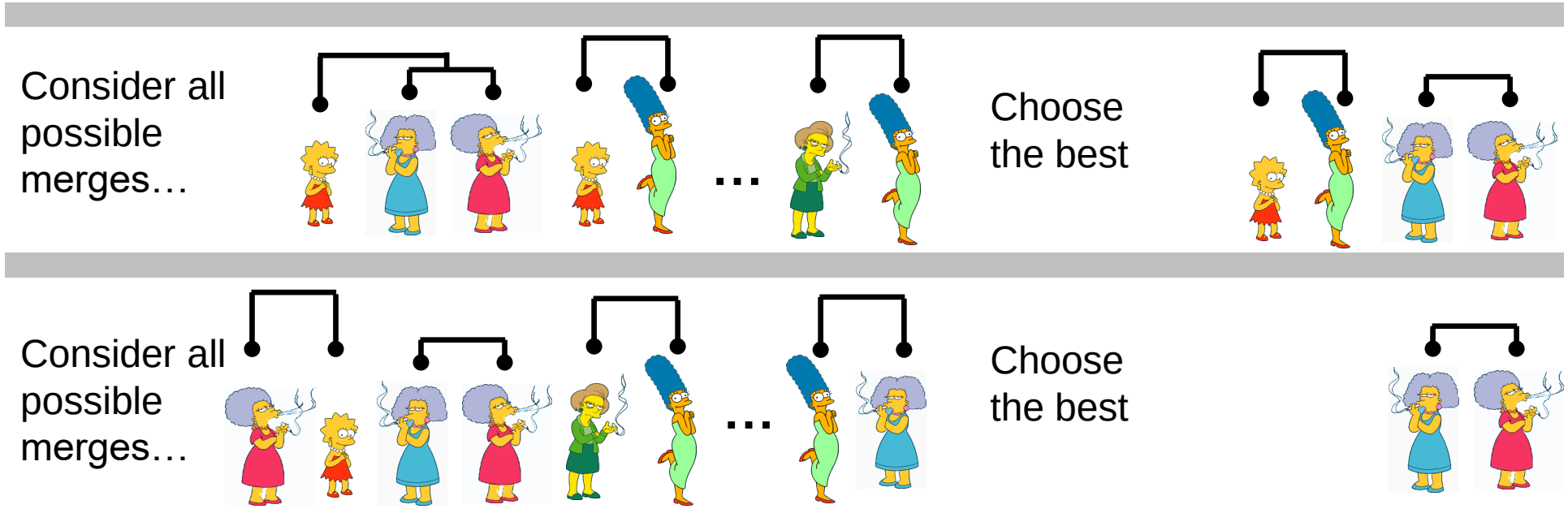
This slide and next 4 based on slides by Eamonn Keogh

Consider all possible merges…



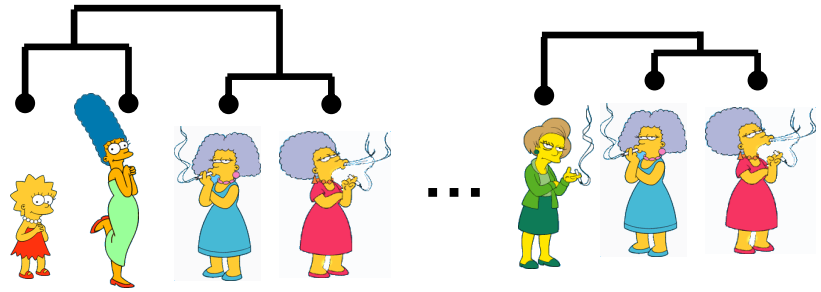
Choose the best

Bottom-Up (agglomerative): Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

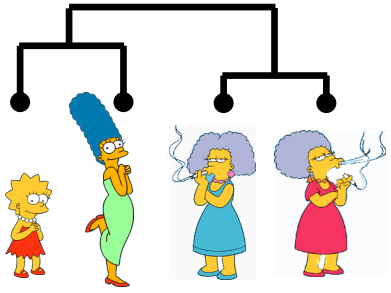


Bottom-Up (agglomerative): Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

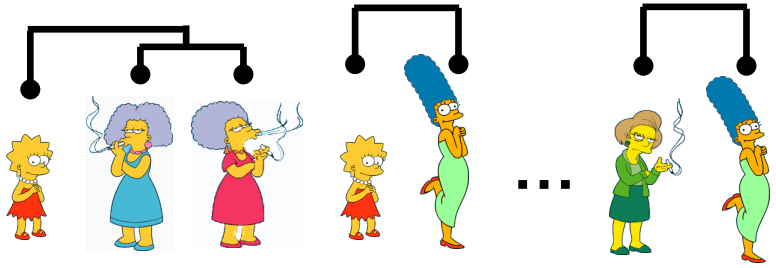
Consider all possible merges…



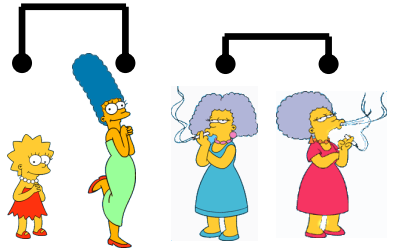
Choose the best



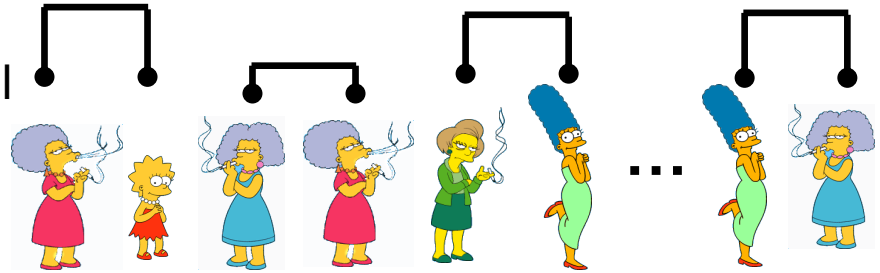
Consider all possible merges…



Choose the best

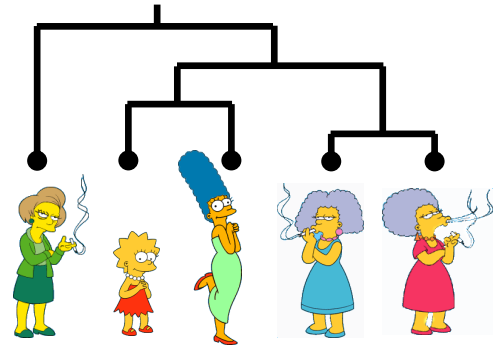


Consider all possible merges…

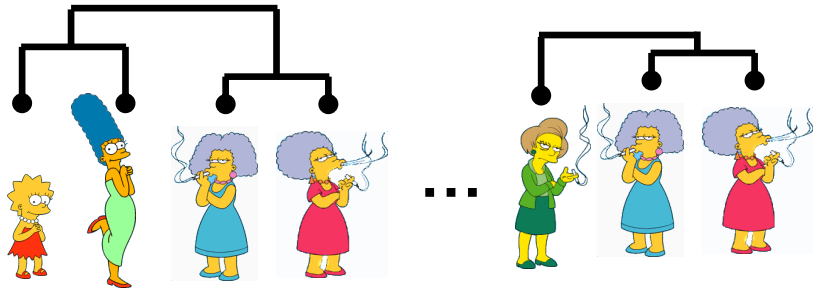


Choose the best

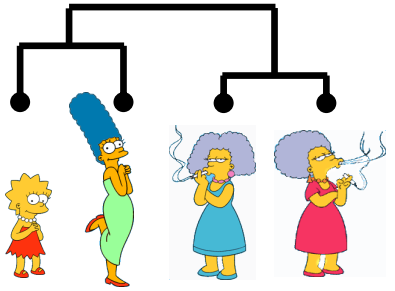
Bottom-Up (agglomerative): Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



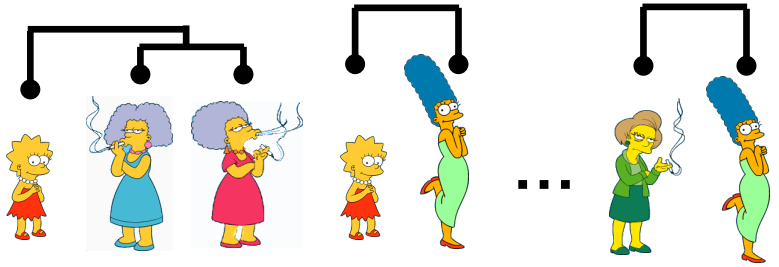
Consider all possible merges…



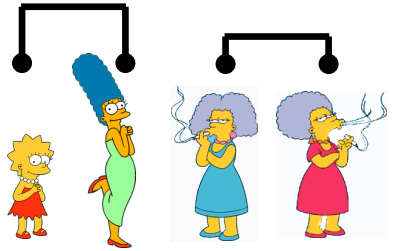
Choose the best



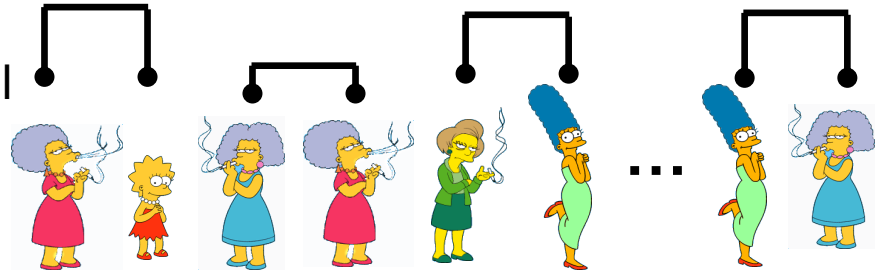
Consider all possible merges…



Choose the best



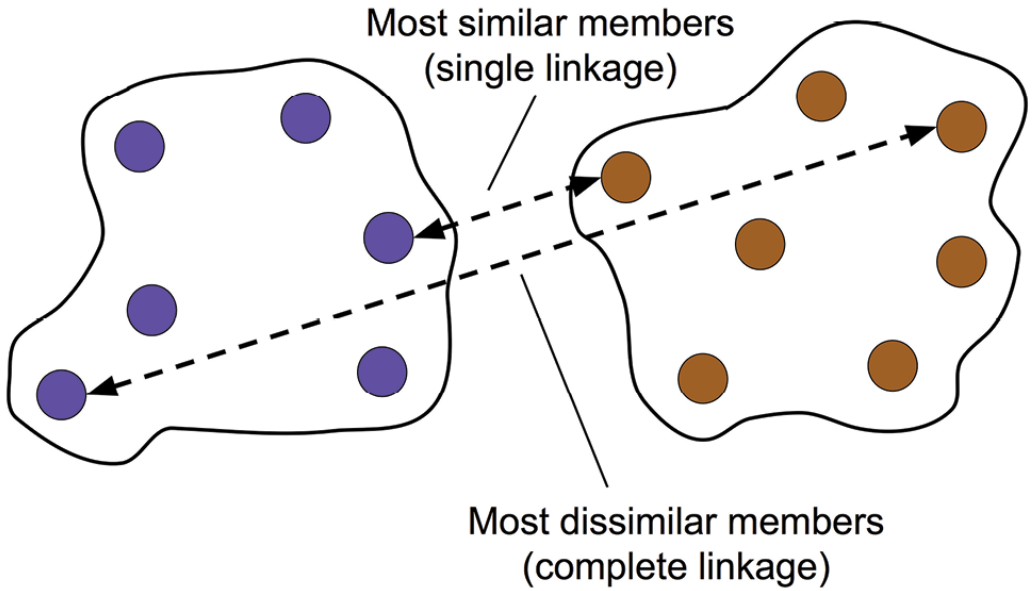
Consider all possible merges…



Choose the best

Distances

Figure by S. Raschka



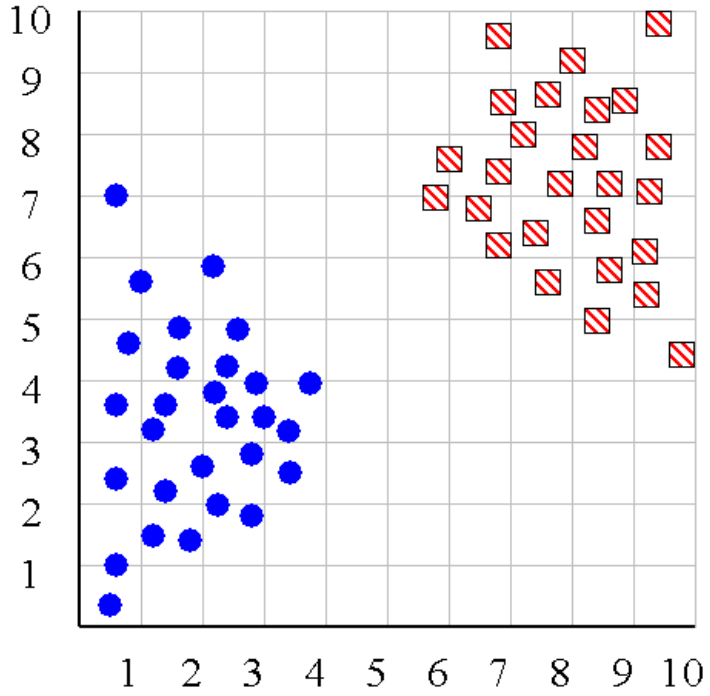
- Have a distance measure on pairs of objects, $d(x, y)$:
- Single linkage: $\text{dist}(\text{A}, \text{B}) = \min_{\text{x}\in A, x'\in B} \text{d}(\text{x}, \text{x}')$
- Complete linkage: $\text{dist}(\text{A}, \text{B}) = \max_{\text{x}\in A, x'\in B} \text{d}(\text{x}, \text{x}')$
- Average linkage: $\text{dist}(\text{A}, \text{B}) = \text{average}_{\text{x}\in A, x'\in B} \text{d}(\text{x}, \text{x}')$
- Ward's method $\text{dist}(\text{A}, \text{B}) = \frac{|A|\,|B|}{|A|+|B|} \|\text{mean}(A) - \text{mean}(B)\|^2$

Slide credit: Maria-Florina Balcan

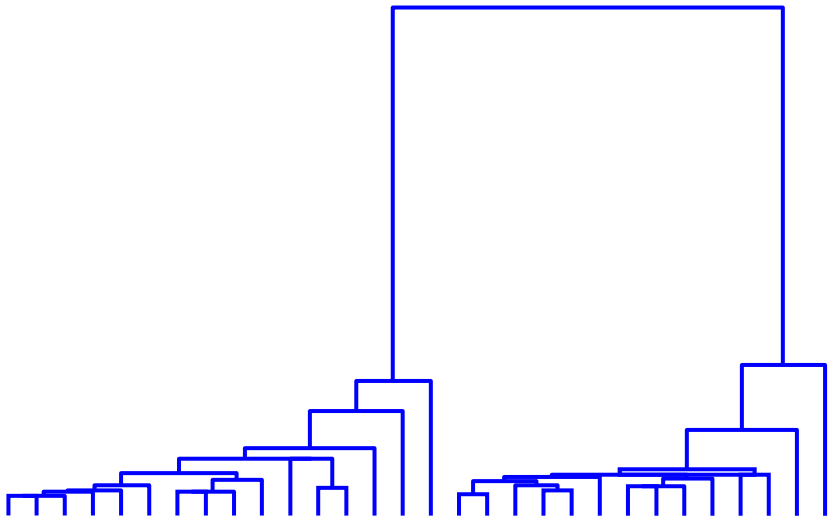
What difference does it make?

- **Single linkage**: $\text{dist}(A, B) = \min_{x \in A, x' \in B} d(x, x')$
  - At any time, distance between any two points in a connected components < r.
- **Complete linkage**: $\text{dist}(A, B) = \max_{x \in A, x' \in B} d(x, x')$
  - Keep max diameter as small as possible at any level
- **Ward's method** $\text{dist}(A, B) = \frac{|A|\,|B|}{|A|+|B|} \|\text{mean}(A) - \text{mean}(B)\|^2$
  - Merge the two clusters such that the increase in k-means cost is as small as possible.
  - Works well in practice

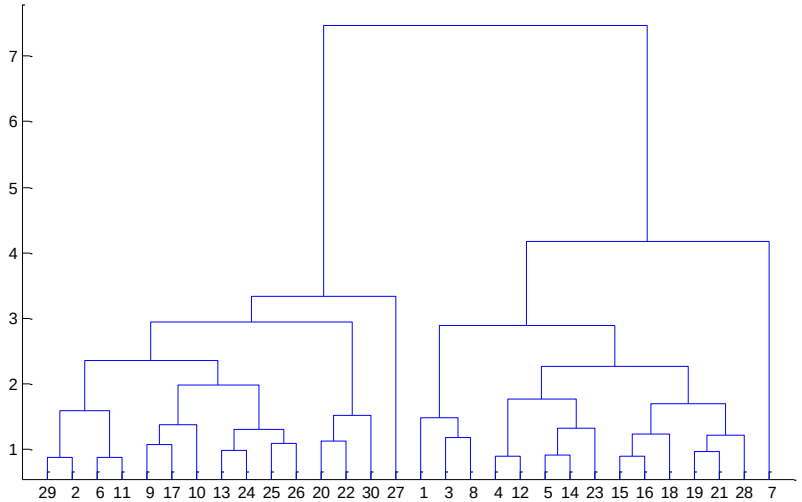
Slide credit: Maria-Florina Balcan



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

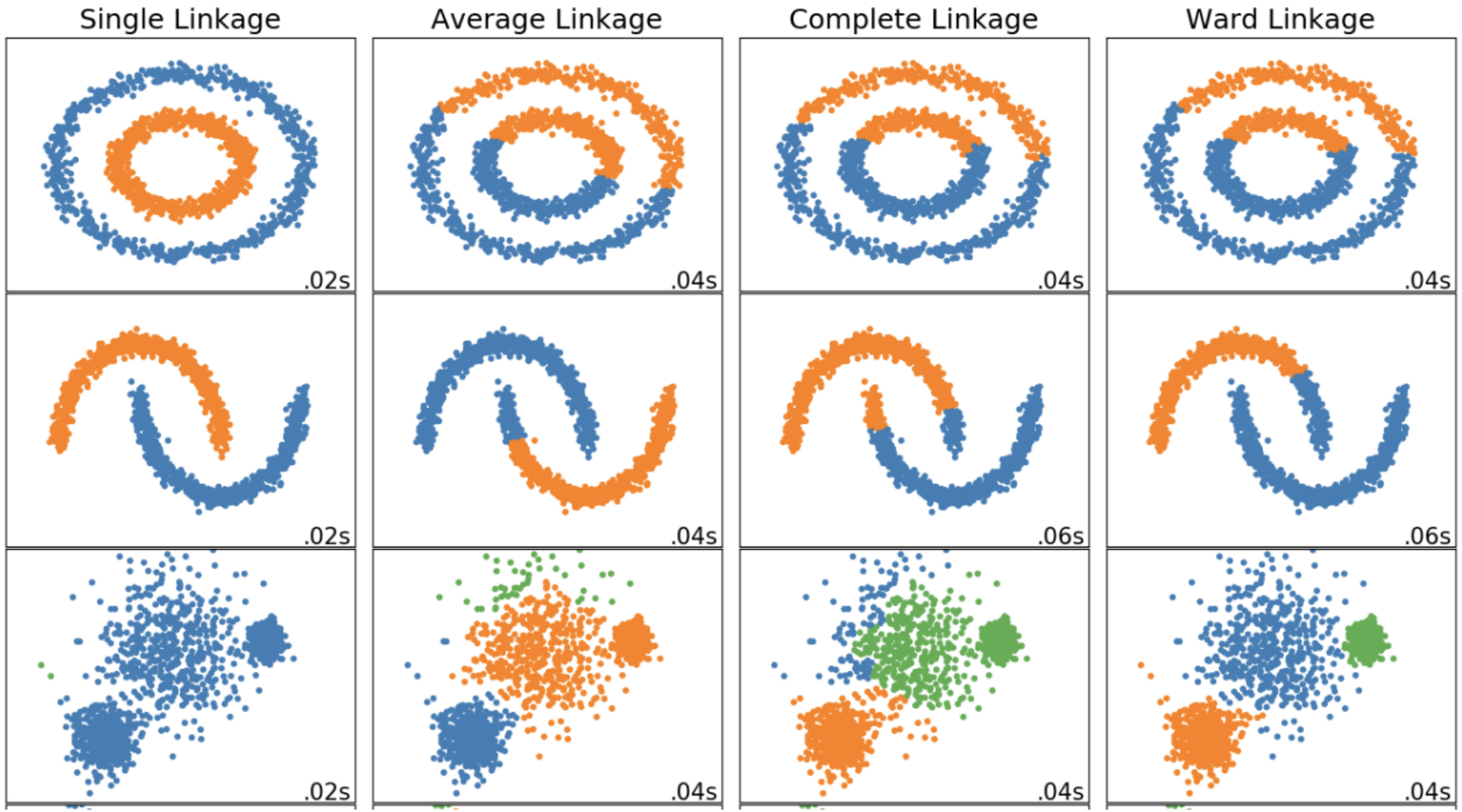


Single linkage



Average linkage

What difference does it make?



Ward linkage measures variance of clusters. The distance between two clusters, A and B, is how much the sum of squares would increase if we merged them.

Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

Hierarchical Clustering Methods Summary

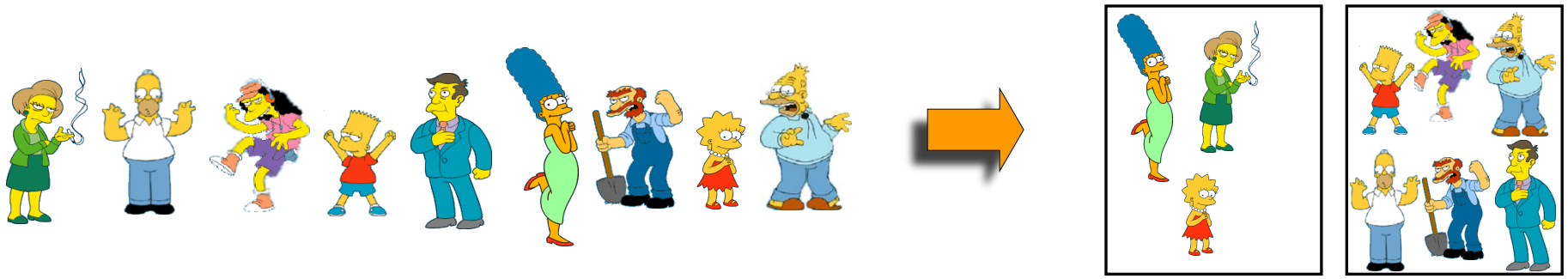
- No need to specify the number of clusters in advance
- Hierarchical nature maps nicely onto human intuition for some domains
- They do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
- Like any heuristic search algorithms, local optima are a problem
- Interpretation of results is (very) subjective

Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

Partitional Clustering

- Nonhierarchical, each instance is placed in exactly one of K non-overlapping clusters

- Since only one set of clusters is output, the user normally has to input the desired number of clusters K.

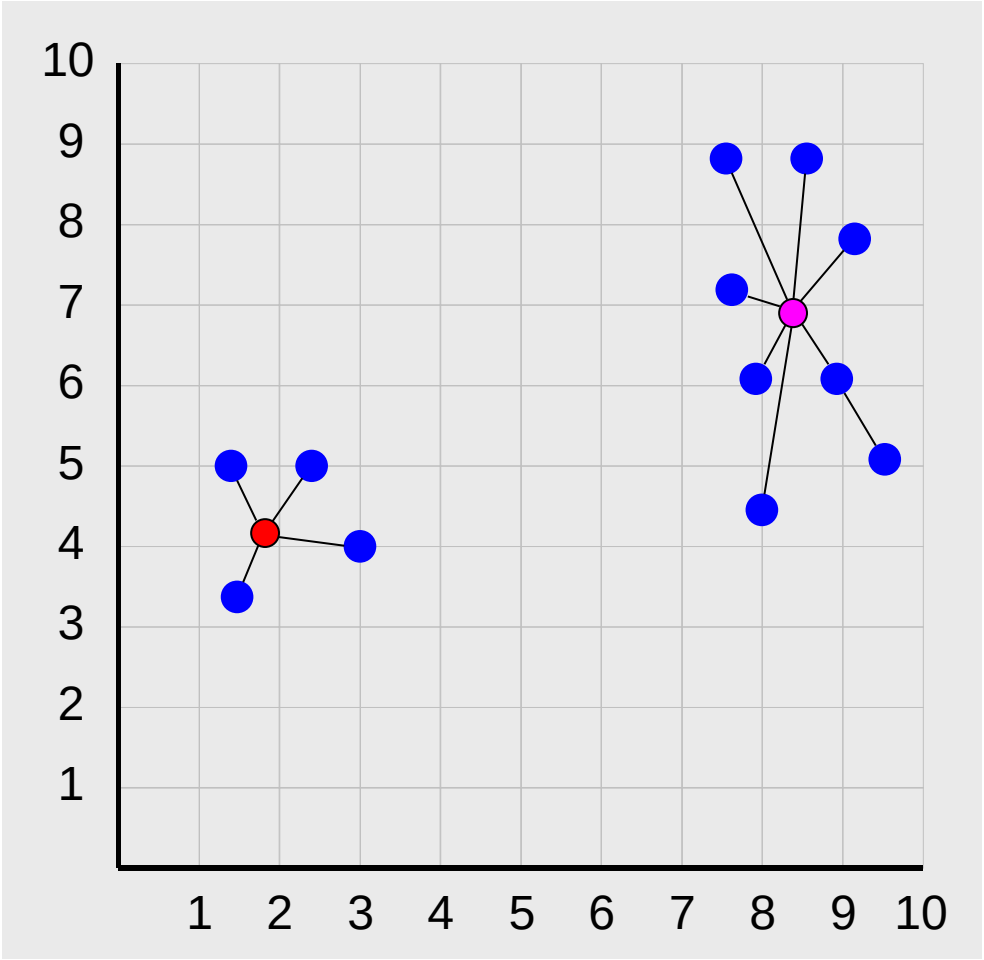


Squared Error

$$se_{K_i} = \sum_{j=1}^{m} \|t_{ij} - C_k\|^2$$

$$se_K = \sum_{j=1}^{k} se_{K_j}$$

Objective Function



Partition Algorithm 1: k-means

1. Decide on a value for *k*.

2. Initialize the *k* cluster centers (randomly, if necessary).

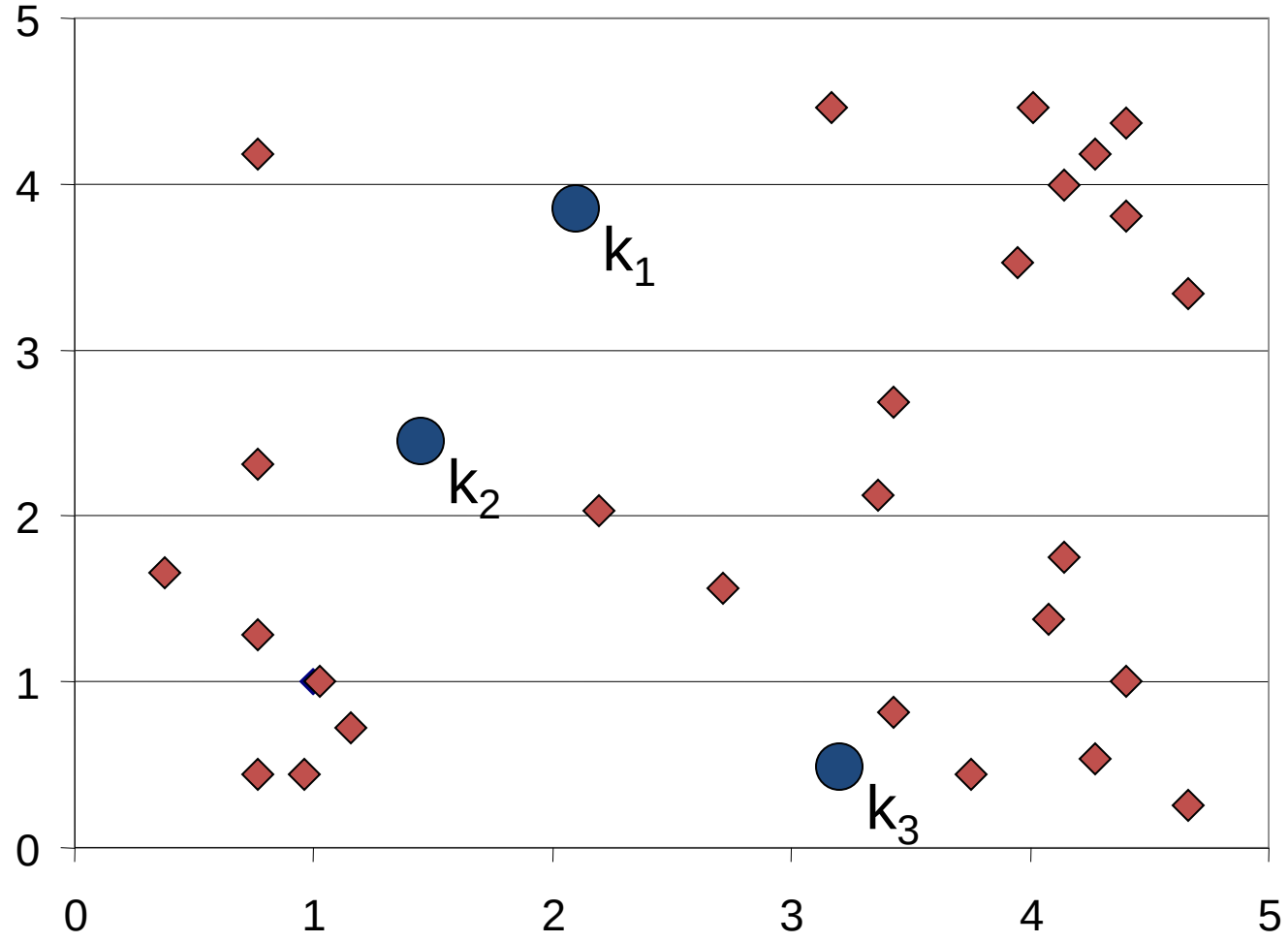
3. Decide the class memberships of the *N* objects by assigning them to the nearest cluster center.

4. Re-estimate the *k* cluster centers, by assuming the memberships found above are correct.

5. If none of the *N* objects changed membership in the last iteration, exit. Otherwise goto 3.

K-means Clustering: Step 1

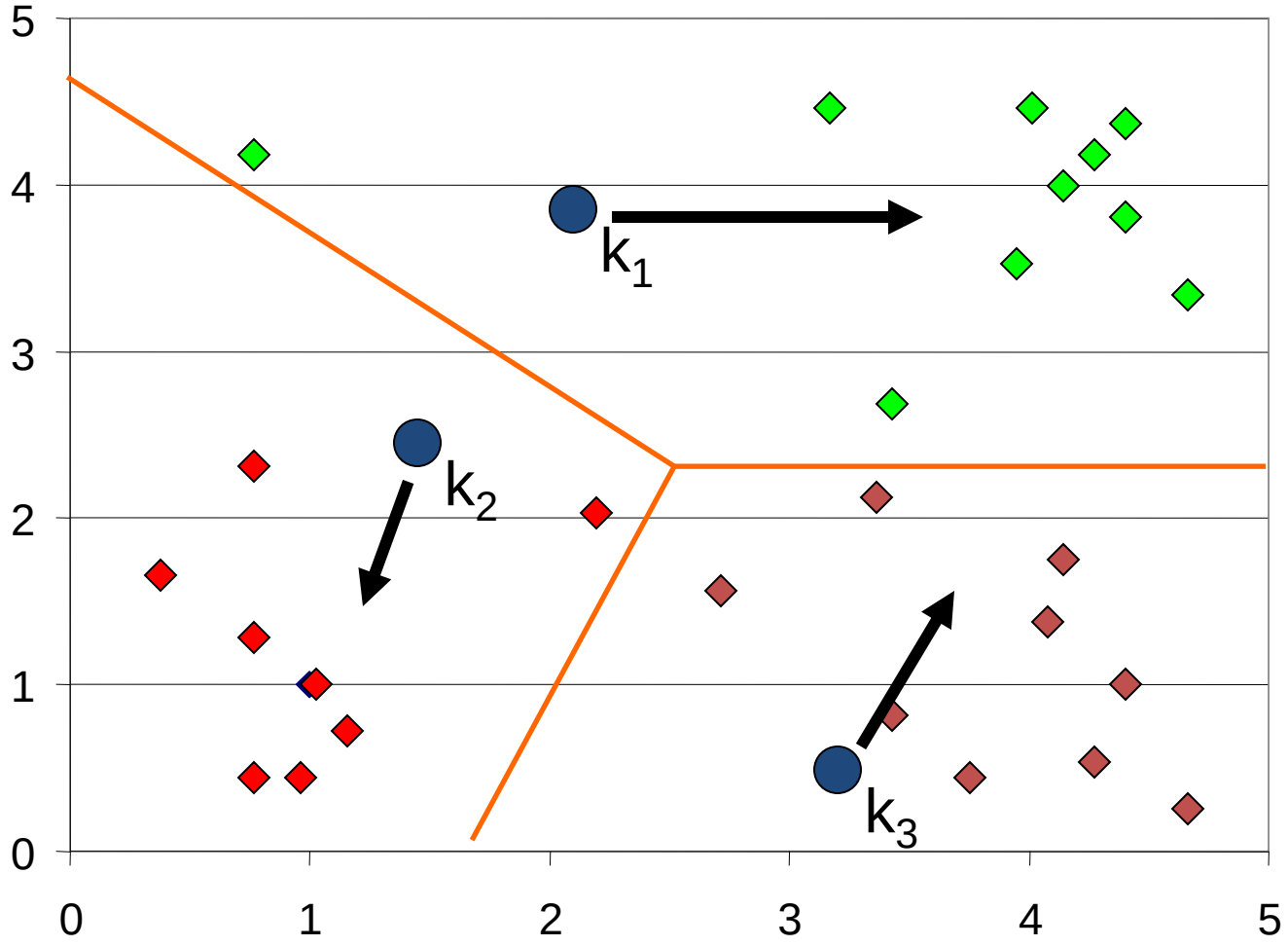
Algorithm: k-means, Distance Metric: Euclidean Distance



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

K-means Clustering: Step 2

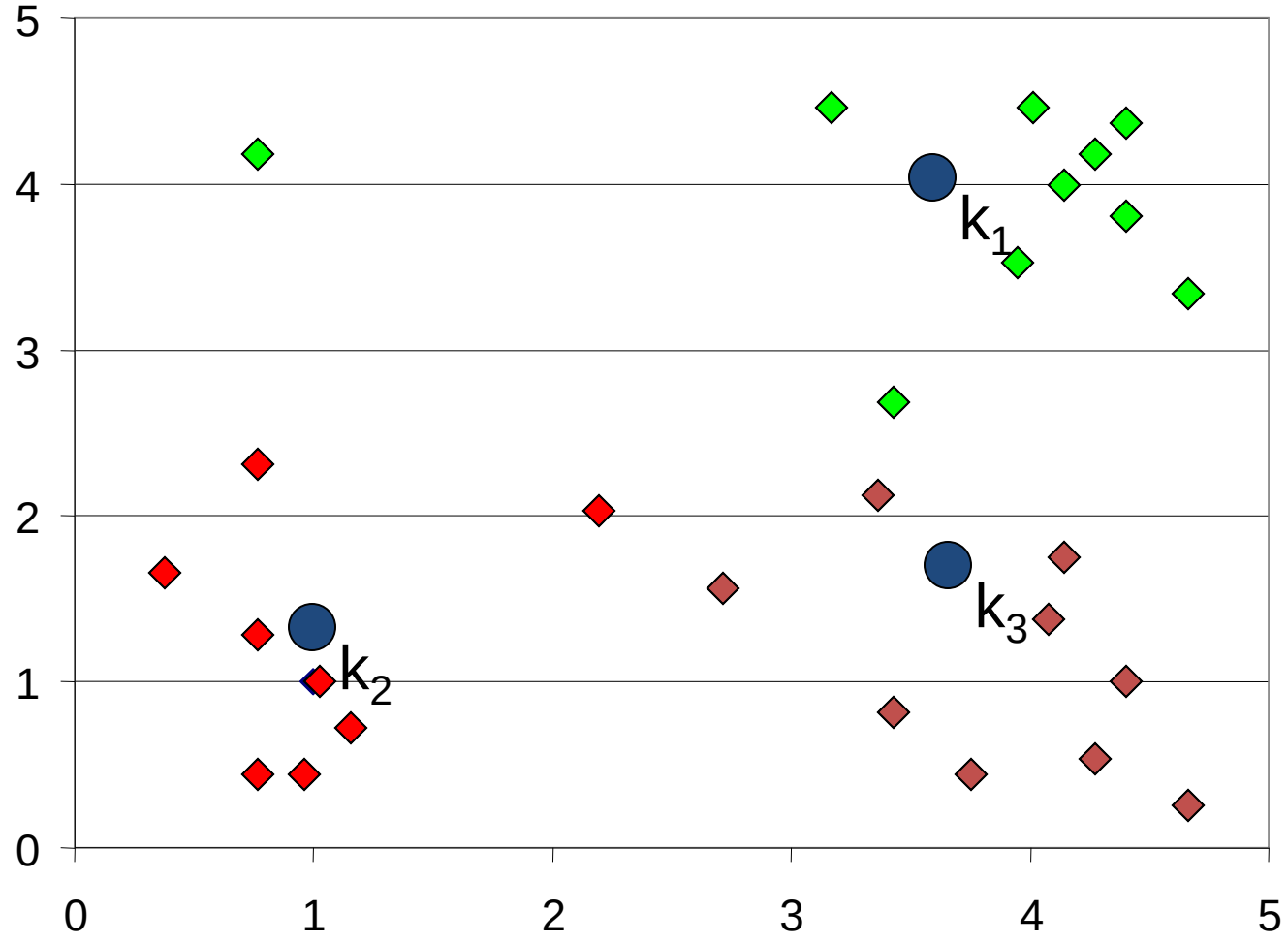
Algorithm: k-means, Distance Metric: Euclidean Distance



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

K-means Clustering: Step 3

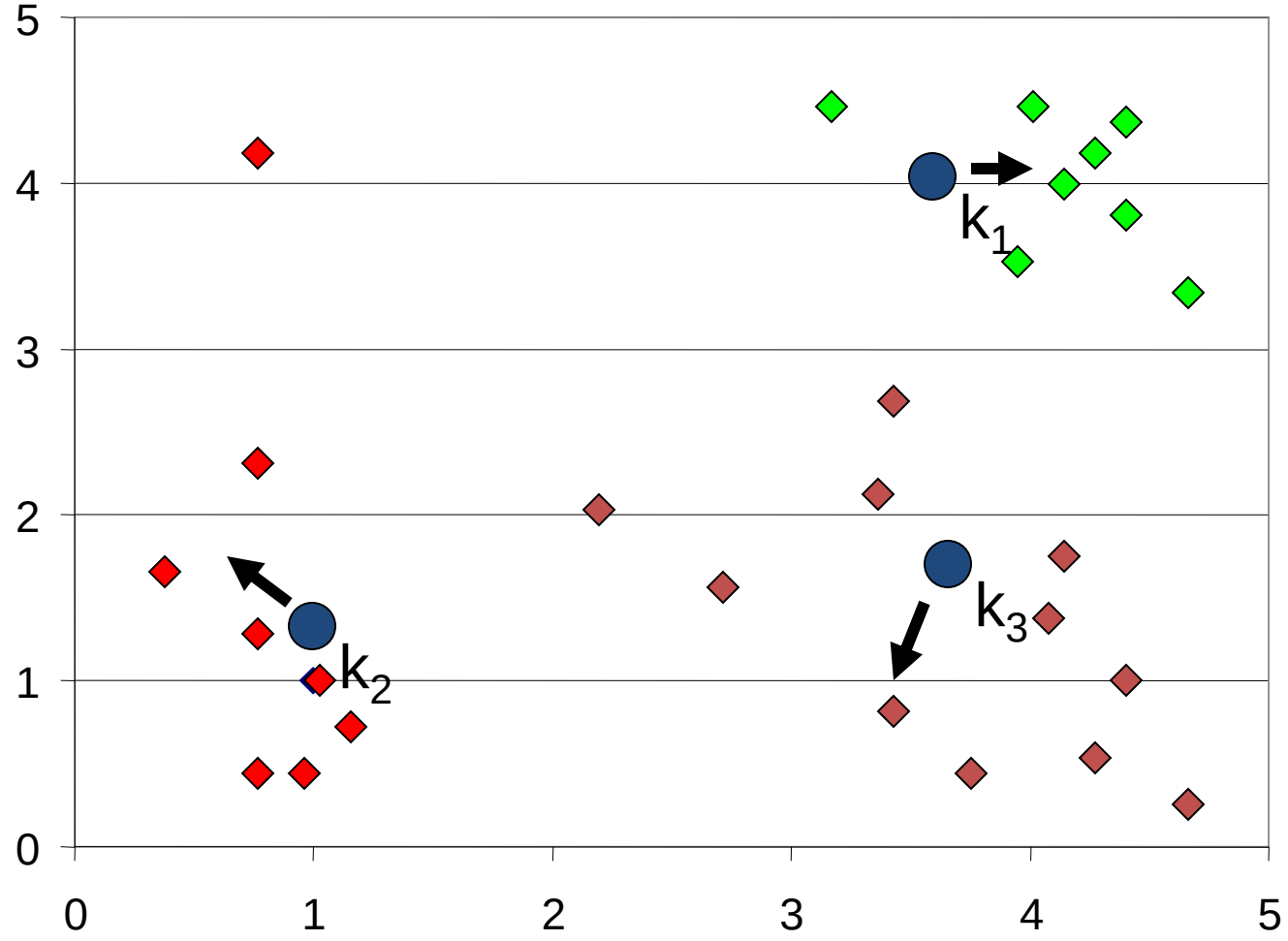
Algorithm: k-means, Distance Metric: Euclidean Distance



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

K-means Clustering: Step 4

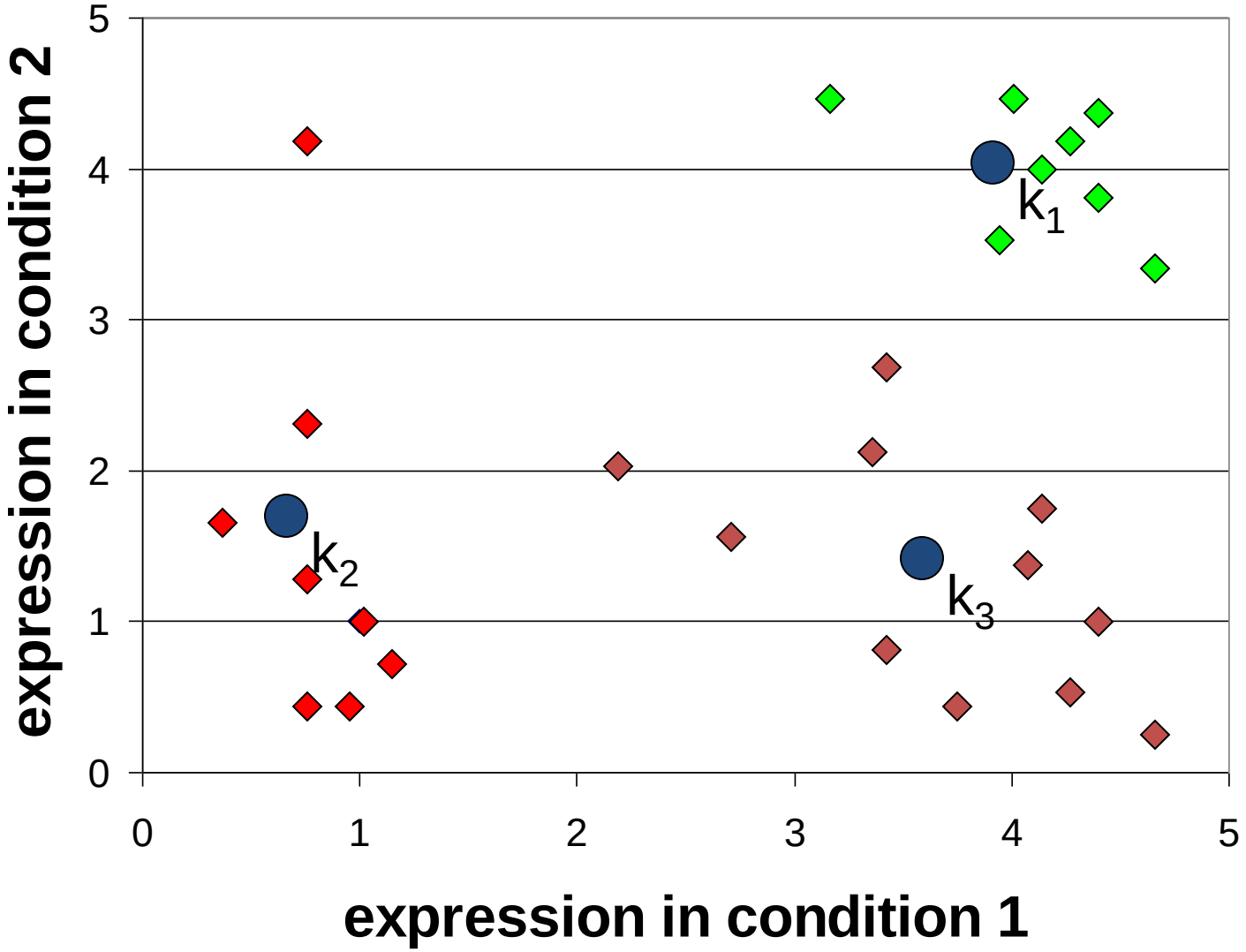
Algorithm: k-means, Distance Metric: Euclidean Distance



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

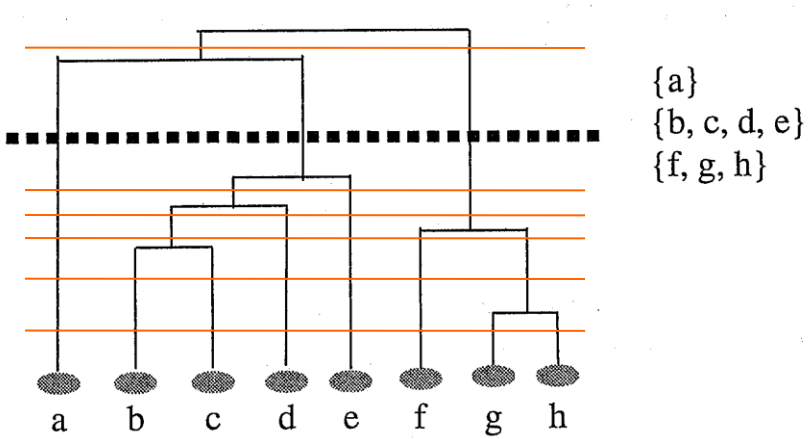
K-means Clustering: Step 5

Algorithm: k-means, Distance Metric: Euclidean Distance



Slide from Eamonn Keogh, from lecture by Carla Brodley, Tufts University

How many clusters to choose?



- Depends on goals
  - May know beforehand how many clusters you want - or at least a range (e.g. 2-10)
  - Could analyze the dendrogram and data after the full clustering to decide which subclustering level is most appropriate for the task at hand
  - Could use automated *cluster validity* metrics to help
- Could do stopping criteria during clustering

How many clusters to choose?

Compactness: members of a cluster are all similar and close together

- One measure of compactness of a cluster is the square distance of the cluster instances compared to the cluster centroid

$$Comp(C) = \sum_{i=1}^{|X_c|} (\mathbf{c} - \mathbf{x}_i)^2$$

- where **c** is the centroid of a cluster C, made up of instances X_c. Lower is better.
- The overall compactness of a particular clustering is just the sum of the compactness of the individual clusters
- Gives a numeric way to compare different clusterings by seeking clusterings which minimize the compactness metric

How many clusters to choose?

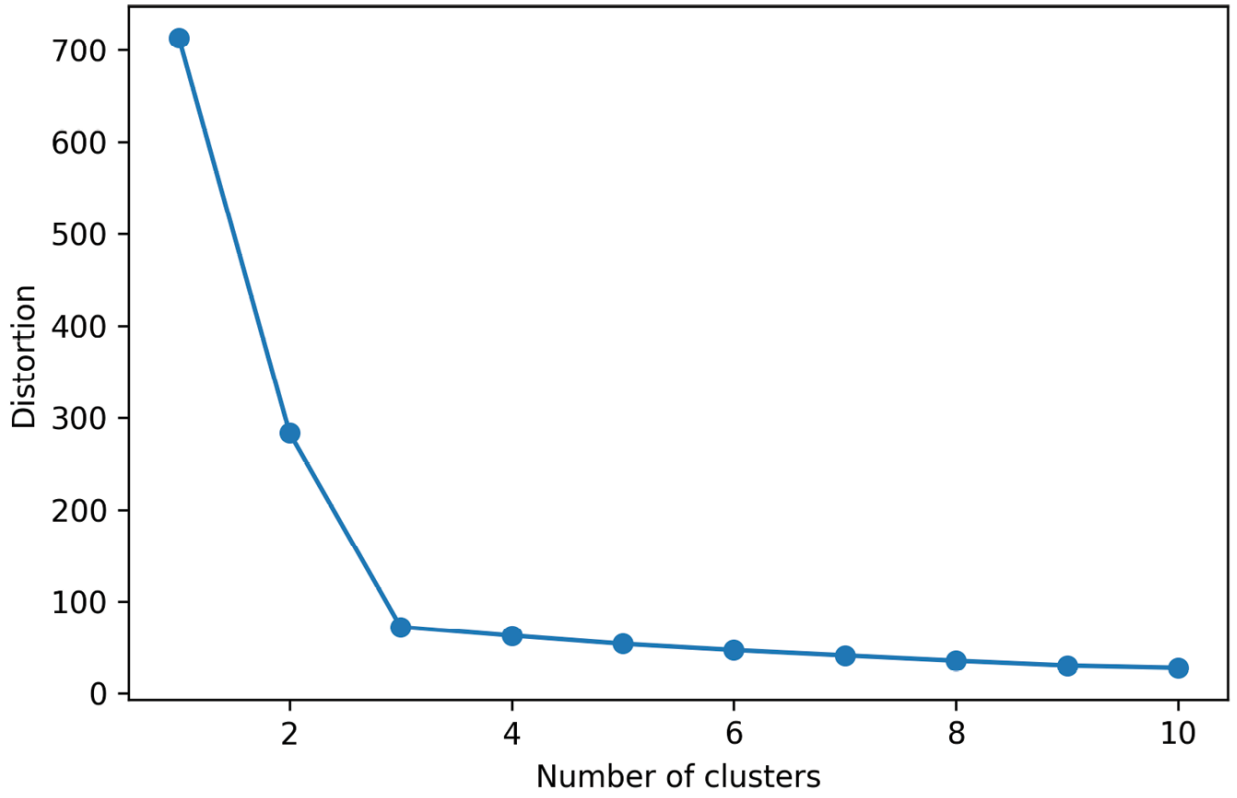
Separability: members of one cluster are sufficiently different from members of another cluster (cluster dissimilarity)

- One measure of the separability of two clusters is their squared distance. The bigger the distance the better.
- $dist_{ij} = (\mathbf{c}_i - \mathbf{c}_j)^2$ where $\mathbf{c}_i$ and $\mathbf{c}_j$ are two cluster centroids
- For a clustering which cluster distances should we compare?
- For each cluster we add in the distance to its closest neighbor cluster

$$Separability = \sum_{i=1}^{|C|} \min_j dist_{ij}(\mathbf{c}_i, \mathbf{c}_j)$$

- We would like to find clusterings where separability is maximized
- separability is usually maximized when there are are very few clusters

Elbow methods



Calculate cluster size as a function of number of clusters

Silhouette

We need techniques that find a balance between inter-cluster similarity and intra-cluster dissimilarity

Silhouette:

- Scores any clustering with an arbitrary number of unique clusters. Clustering can come from any clustering algorithm.
- $a(i)$ = average dissimilarity of instance i to all other instances in the cluster to which i is assigned – **Minimize**
  - Dissimilarity could be Euclidian distance, etc.
- $b(i)$ = the smallest average dissimilarity of instance i to all instances in the closest cluster to $b(i)$ – **Maximize**
- $b(i)$ is smallest for the best different cluster that i could be assigned to – the best cluster that you would move i to if needed

Silhouette

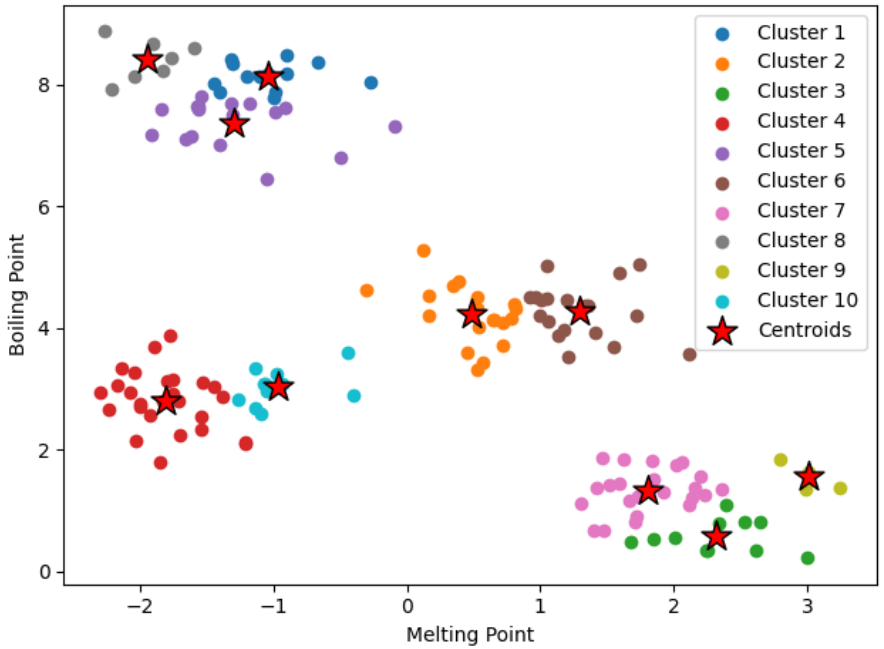
1. Calculate the **cluster cohesion**, $a^{(i)}$, as the average distance between an example, $\boldsymbol{x}^{(i)}$, and all other points in the same cluster.

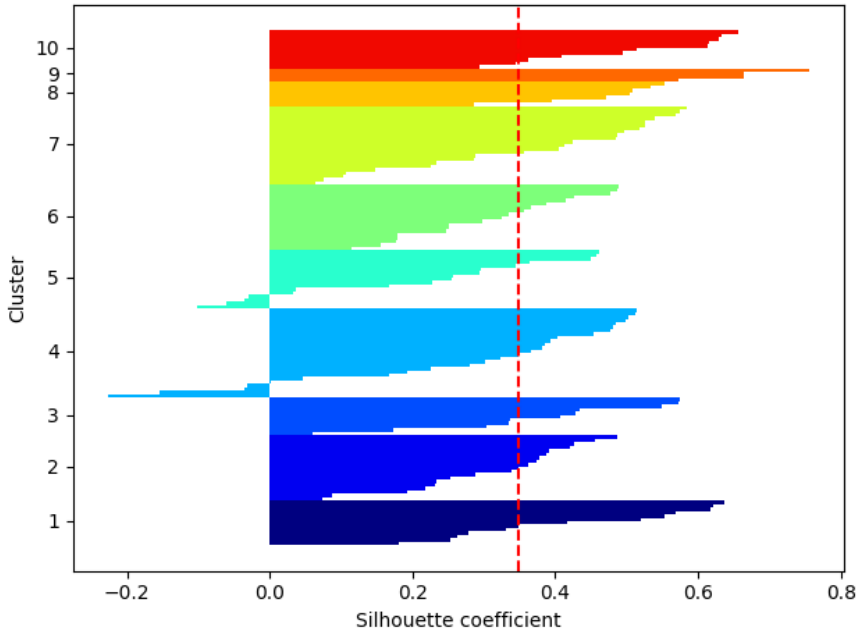
2. Calculate the **cluster separation**, $b^{(i)}$, from the next closest cluster as the average distance between the example, $\boldsymbol{x}^{(i)}$, and all examples in the nearest cluster.

3. Calculate the silhouette, $s^{(i)}$, as the difference between cluster cohesion and separation divided by the greater of the two, as shown here:

$$s^{(i)} = \frac{b^{(i)} - a^{(i)}}{max\{b^{(i)}, a^{(i)}\}}$$

Silhouette





- The quality of a single cluster can be measured by the average silhouette score of its members, (close to 1 is best)
- The quality of a total clustering can be measured by the average silhouette score of all the instances
- To find best clustering, compare total silhouette scores across clusterings with different *k* values and choose the highest

Summary of k-means clustering

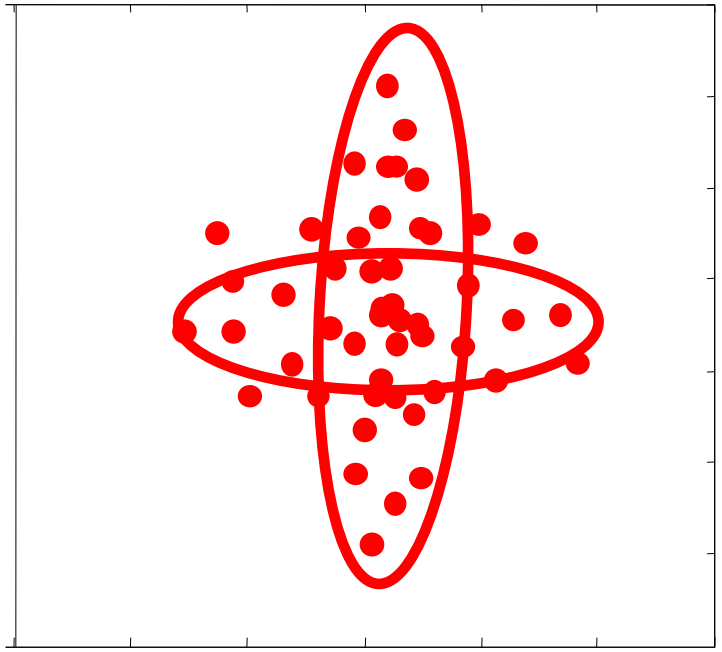
- **Strengths**
  - *Relatively efficient*: $O(tkn)$, where n is number of objects, k is number of clusters, and t is number of iterations. Normally, k, $t << n$.
  - Often terminates at a local optimum

- **Weakness**
  - Applicable only when mean is defined (what about categorical data)?
  - Need to specify k, the number of clusters, in advance
  - Unable to handle noisy data and outliers
  - Not suitable to discover clusters with non-convex shapes
  - Scales matter

Mixture of Gaussians

K-means algorithm

- Assigned each example to exactly one cluster
- What if clusters are overlapping?
  - Hard to tell which cluster is right
  - Maybe we should try to remain uncertain
- Used Euclidean distance
- What if cluster has a non-circular shape?



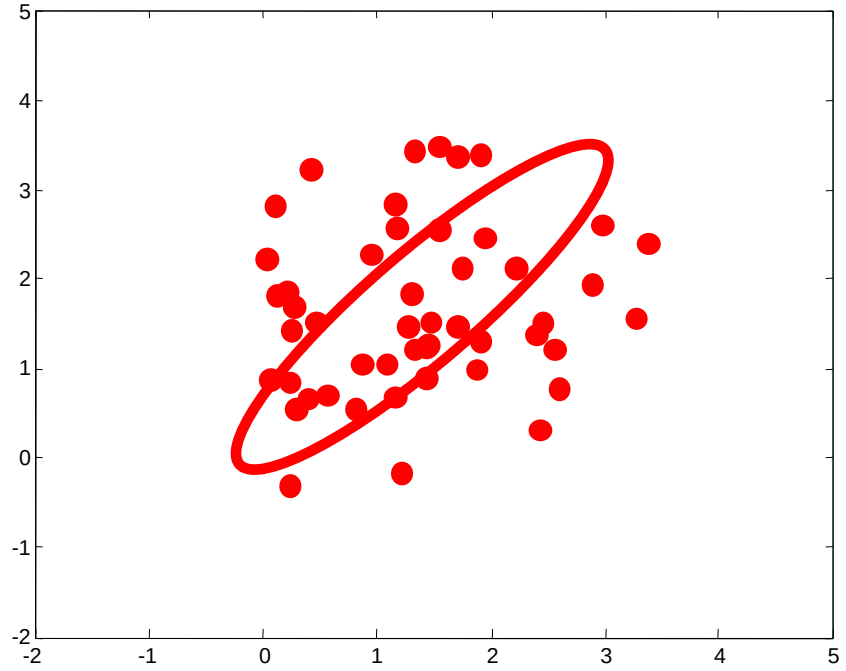
Gaussian mixture models

- Clusters modeled as Gaussian distributions
- EM algorithm: assign data to cluster with some *probability*

(adapted from) Prof. Alexander Ihler

Multivariate Gaussian Model

$$\mathcal{N}(\underline{x}\ ;\ \underline{\mu}, \Sigma) = \frac{1}{(2\pi)^{d/2}} |\Sigma|^{-1/2} \exp\left\{ -\frac{1}{2} (\underline{x} - \underline{\mu})^T \Sigma^{-1} (\underline{x} - \underline{\mu}) \right\}$$



Maximum Likelihood estimates

$$\hat{\mu} = \frac{1}{N} \sum_i x^{(i)}$$

$$\hat{\Sigma} = \frac{1}{N} \sum_i (x^{(i)} - \hat{\mu})^T (x^{(i)} - \hat{\mu})$$

We model each cluster using Gaussian distribution

(adapted from) Prof. Alexander Ihler

Expectation Maximization: E-Step

- Initialize parameters of each cluster: mean μ_c, Covariance Σ_c, size π_c

- **E-step ("Expectation")**
  - For each datum (example) x_i,
  - Compute r_{ic}, the probability that it belongs to cluster c
    - Compute its probability under model c
    - Normalize to sum to one (over clusters c)

$$r_{ic} = \frac{\pi_c \mathcal{N}(x_i \; ; \; \mu_c, \Sigma_c)}{\sum_{c'} \pi_{c'} \mathcal{N}(x_i \; ; \; \mu_{c'}, \Sigma_{c'})}$$

  - If x_i is very likely under the c^{th} Gaussian, it gets high weight
  - Denominator just makes probabilities to sum to one

(adapted from) Prof. Alexander Ihler

Expectation Maximization: M-Step

- Start with assignment probabilities r_{ic}
- Update parameters: mean μ_c, Covariance Σ_c, "size" π_c

- M-step ("Maximization")
  - For each Gaussian cluster x_c,
  - Update its parameters using the (weighted) data points

$$N_c = \sum_i r_{ic}$$

Total responsibility allocated to cluster c

$$\pi_c = \frac{N_c}{N}$$

Fraction of total assigned to cluster c

$$\mu_c = \frac{1}{N_c} \sum_i r_{ic} x_i$$

Weighted mean of assigned data

$$\Sigma_c = \frac{1}{N_c} \sum_i r_{ic} (x_i - \mu_c)^T (x_i - \mu_c)$$

Weighted covariance of assigned data (use new weighted means here)

(adapted from) Prof. Alexander Ihler

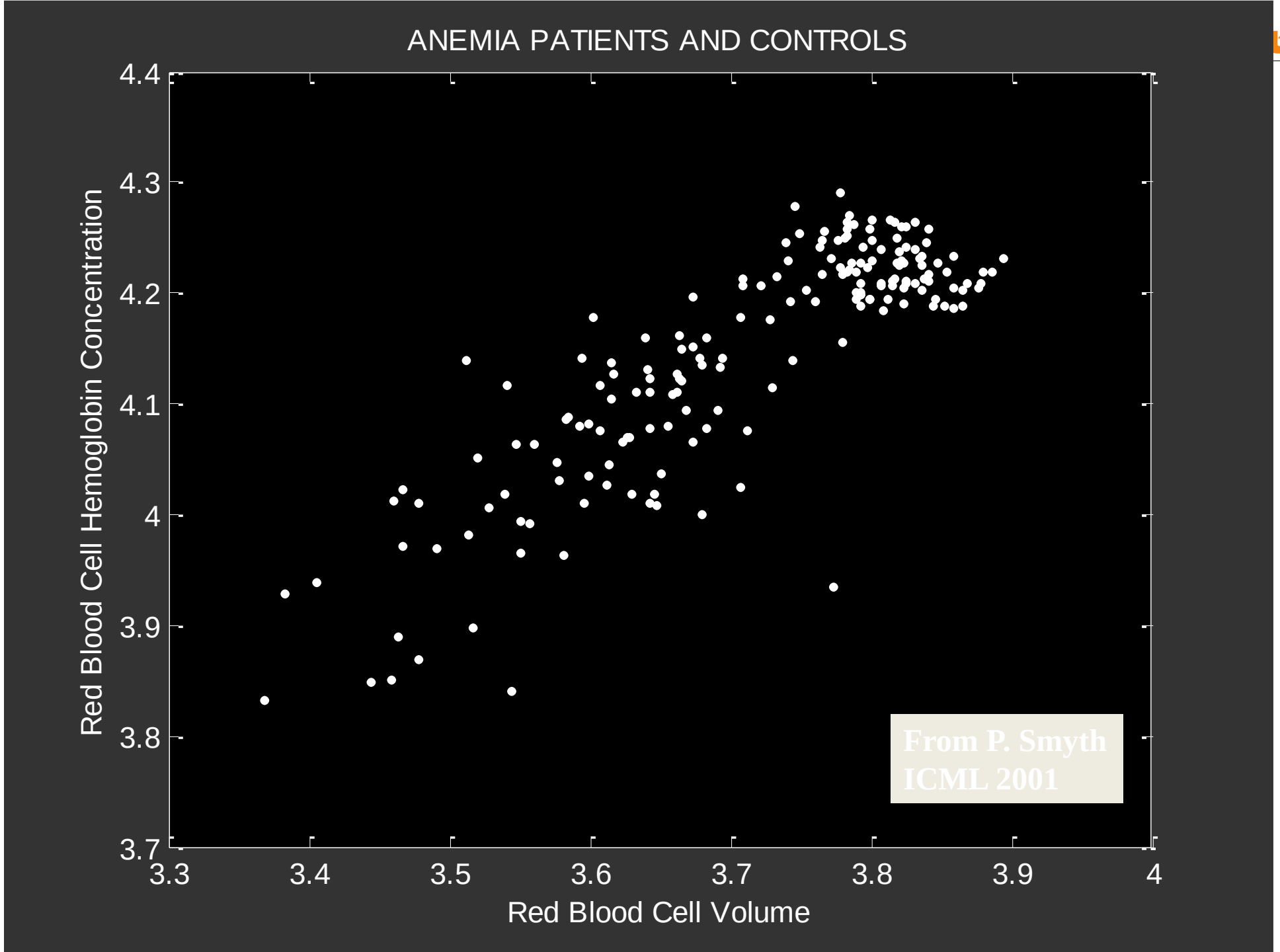
Expectation Maximization

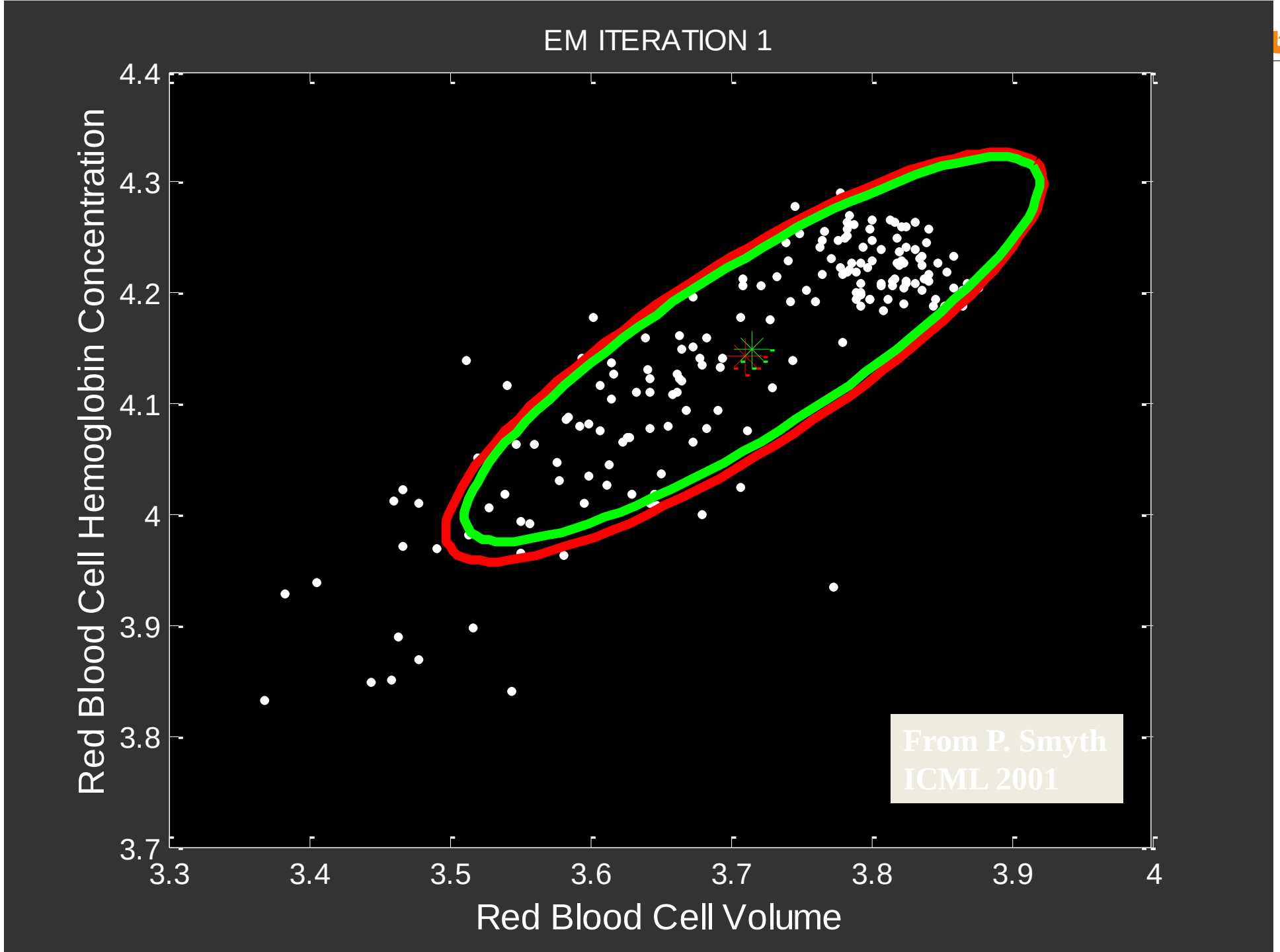
- Each step increases the log-likelihood of our model

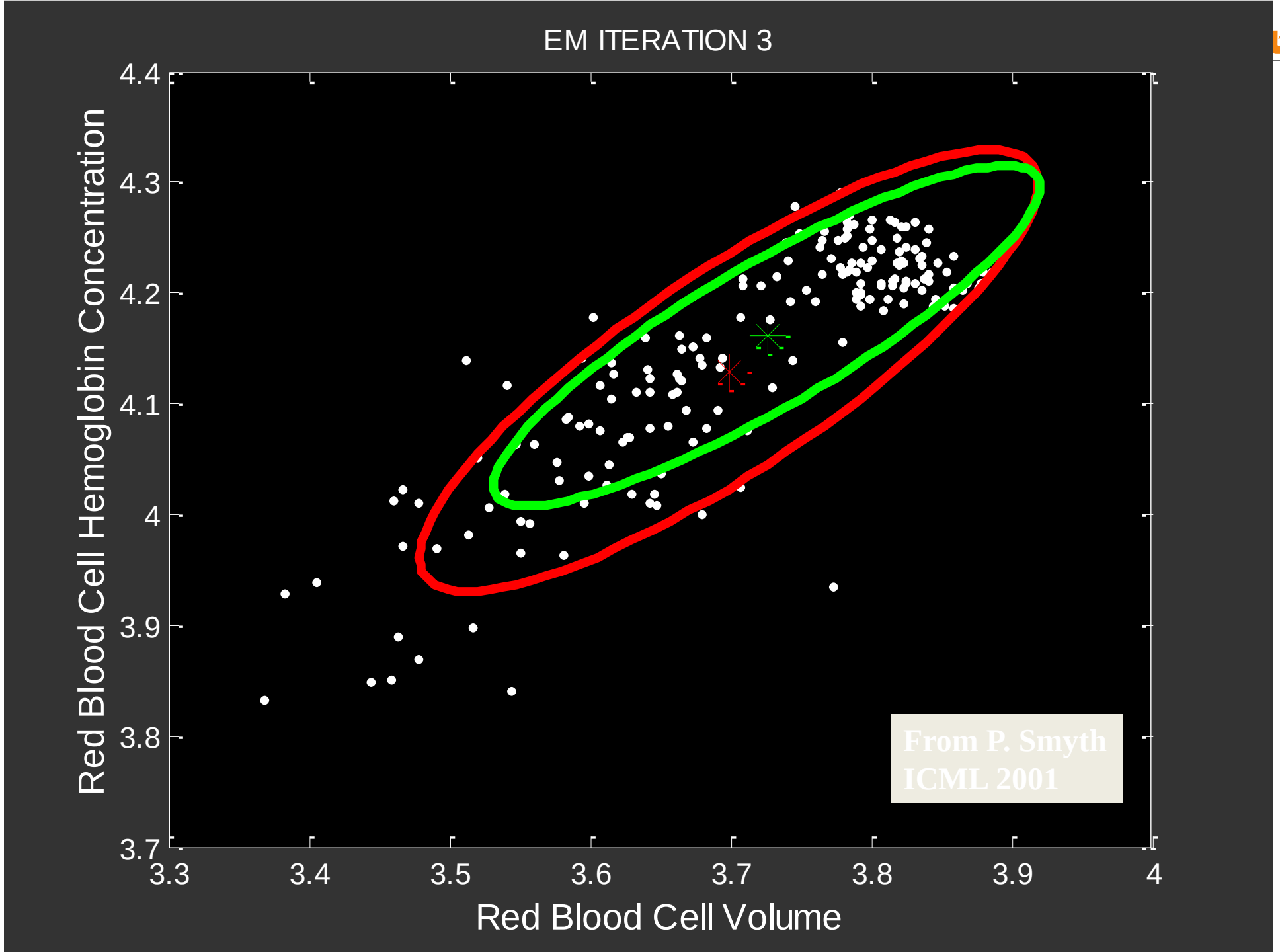
$$\log p(\underline{X}) = \sum_i \log \left[\sum_c \pi_c \, \mathcal{N}(x_i \; ; \; \mu_c, \Sigma_c) \right]$$

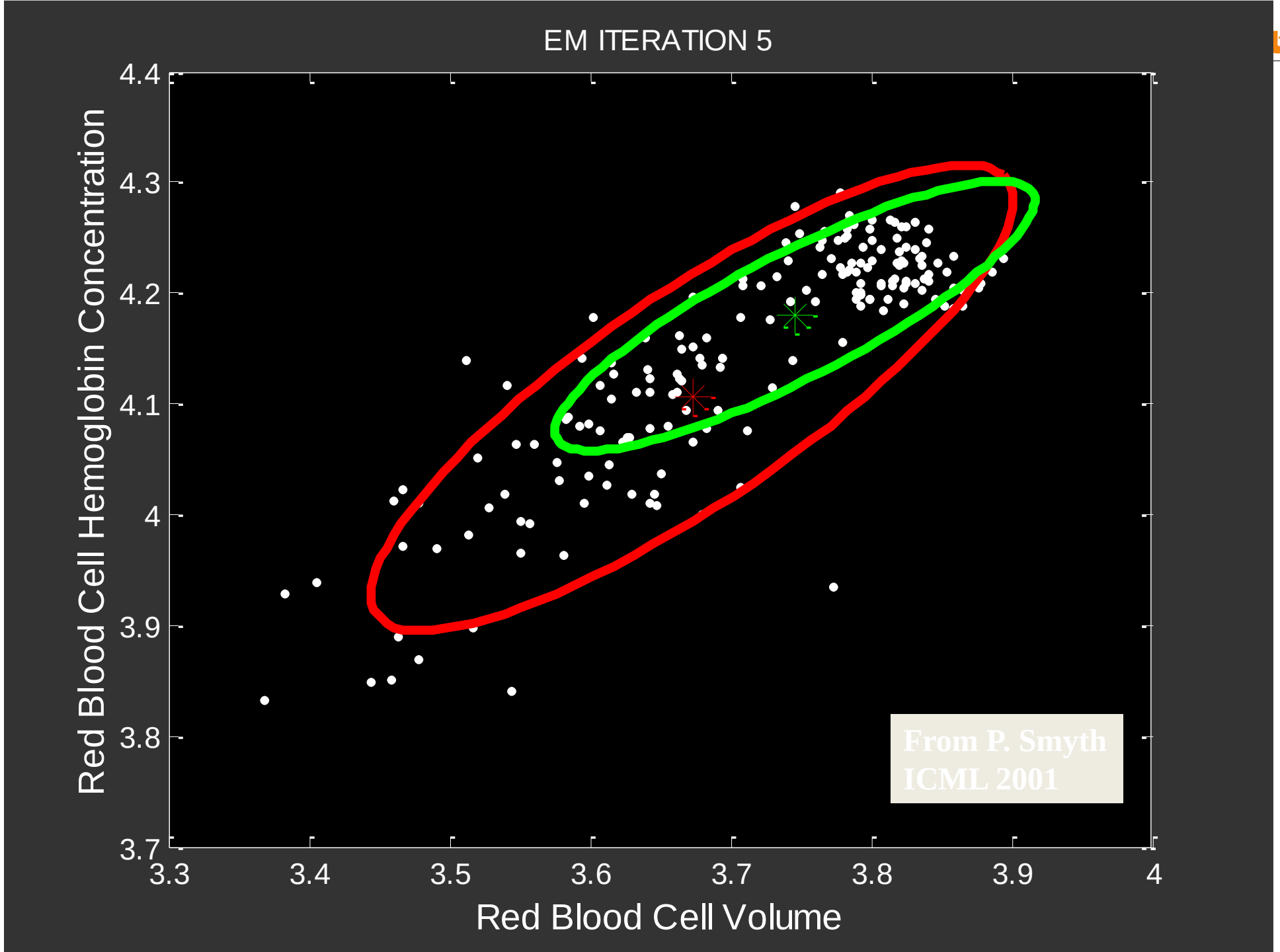
- Iterate until convergence
  - Convergence guaranteed – another ascent method

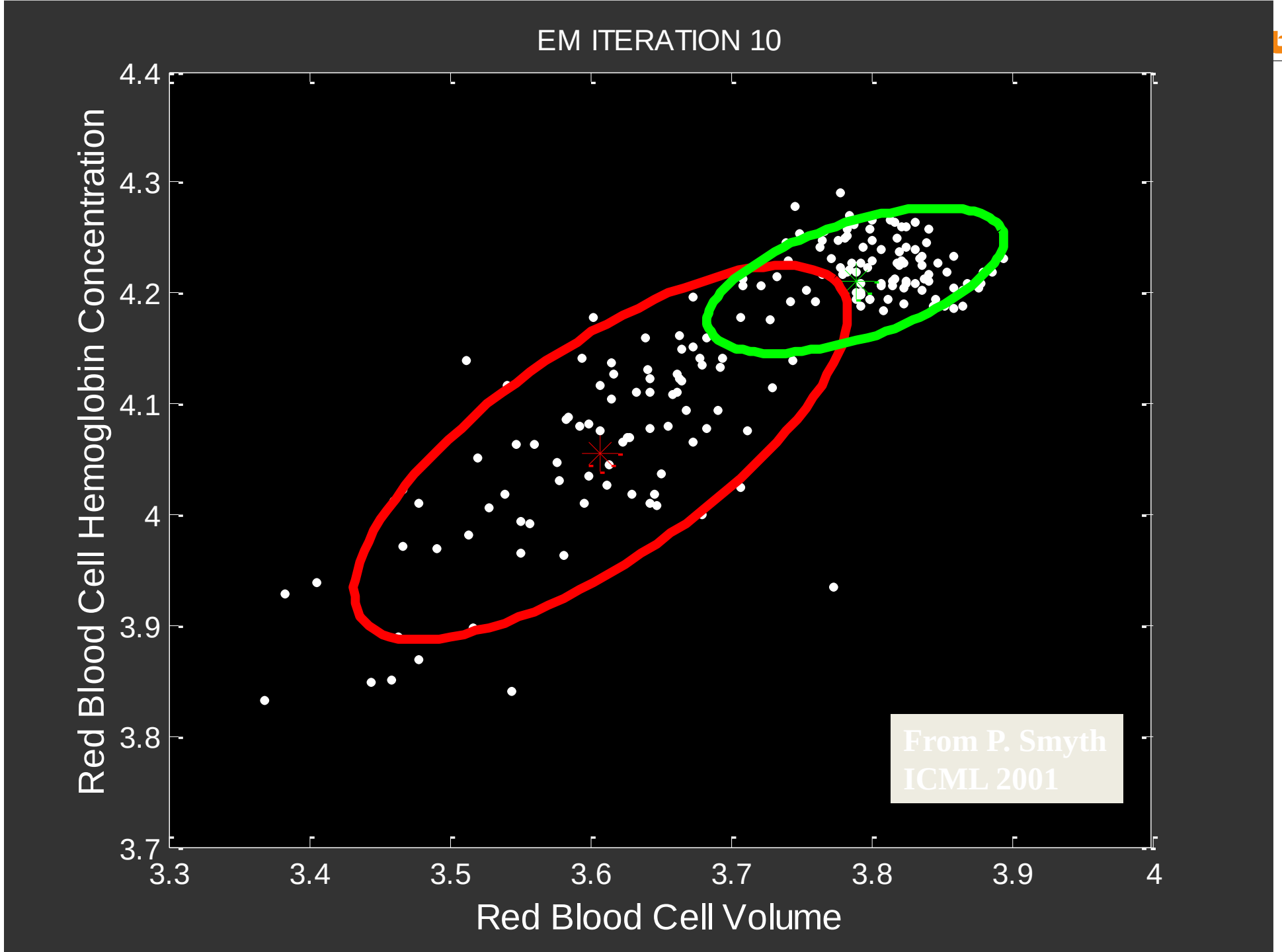
- What should we do
  - If we want to choose a single cluster for an “answer”?
  - With new data we didn’t see during training?

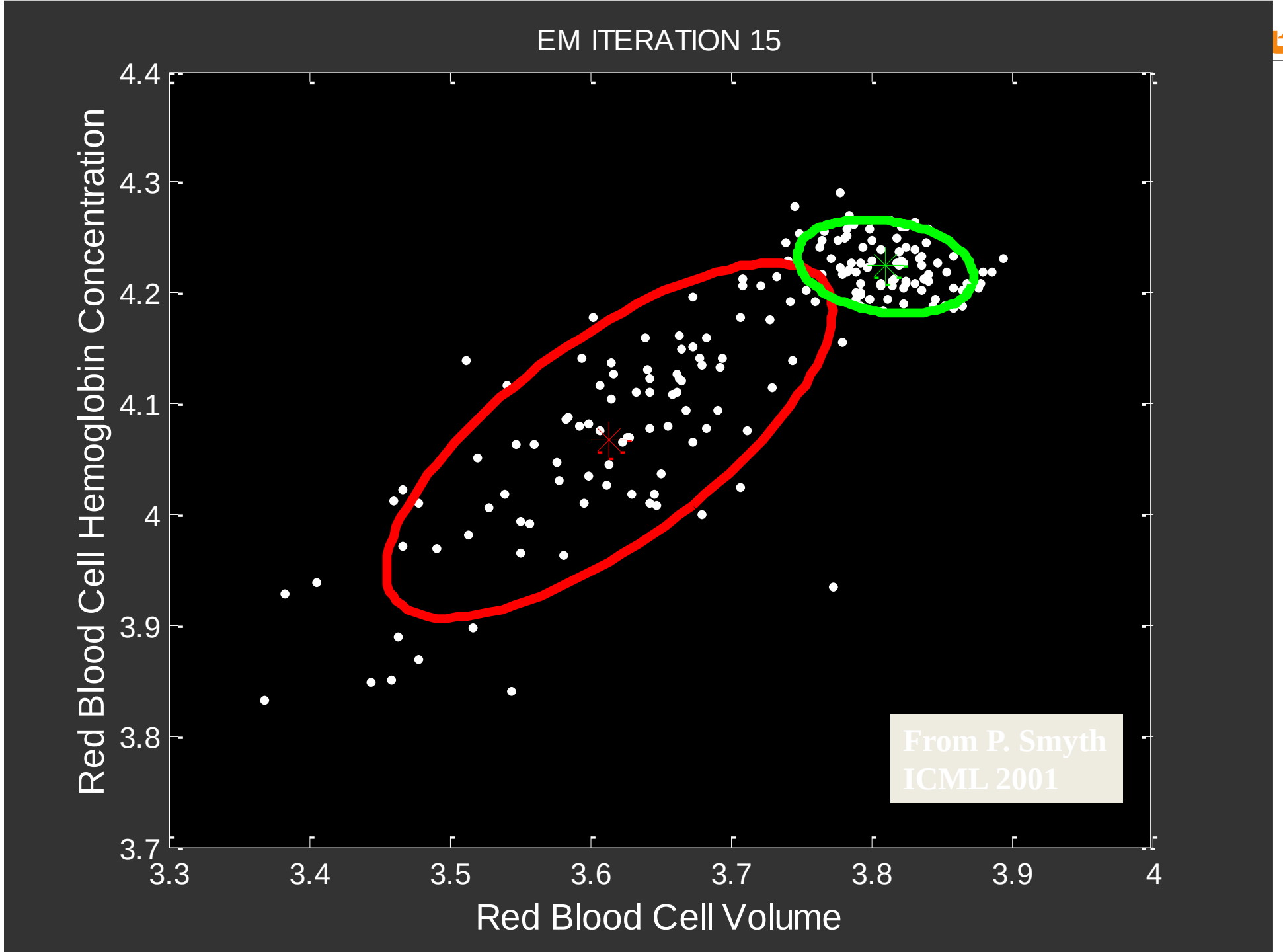

ANEMIA PATIENTS AND CONTROLS
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
From P. Smyth
ICML 2001

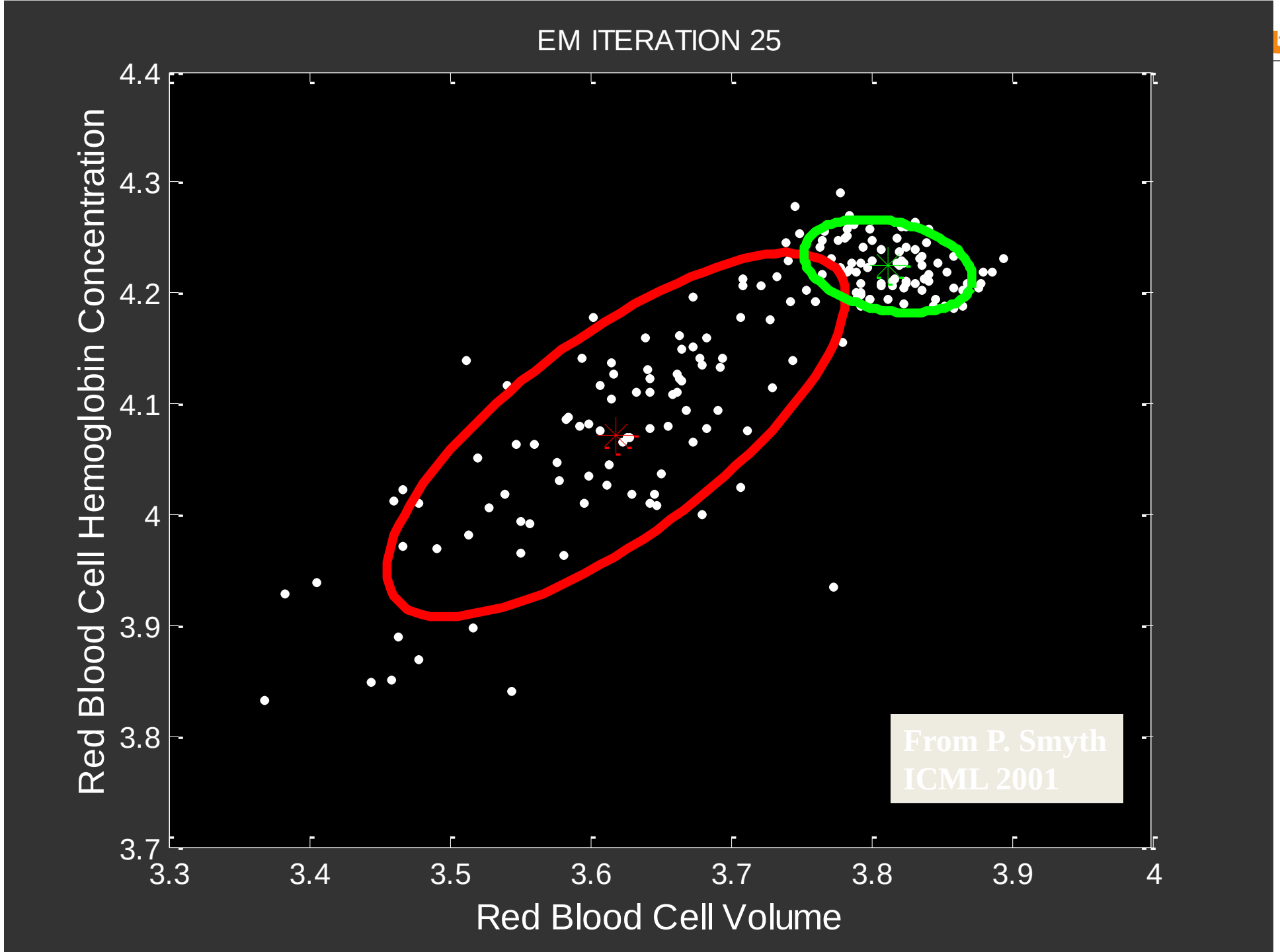

EM ITERATION 1
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
From P. Smyth
ICML 2001

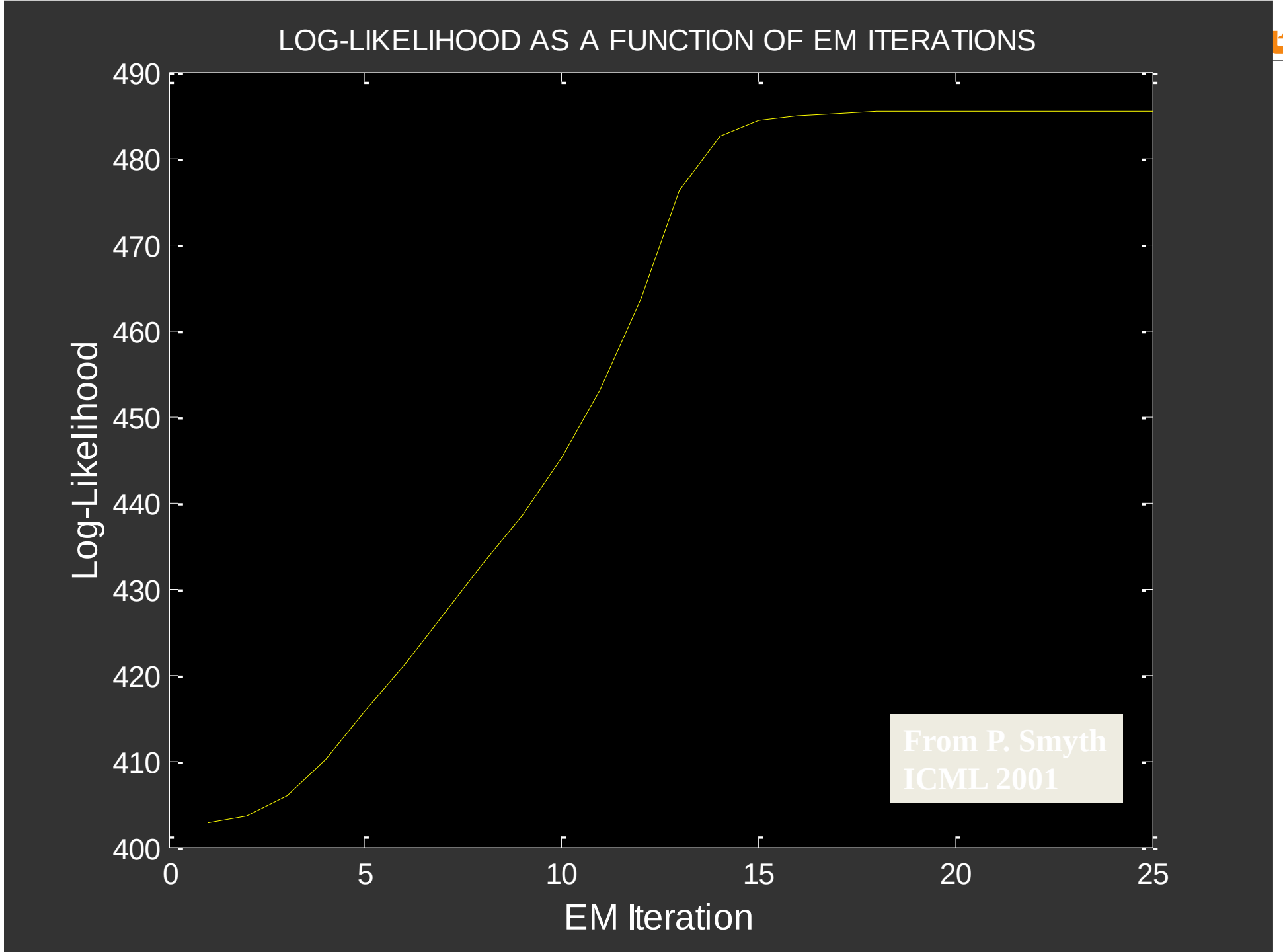

EM ITERATION 3
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
From P. Smyth
ICML 2001


EM ITERATION 5
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
From P. Smyth
ICML 2001

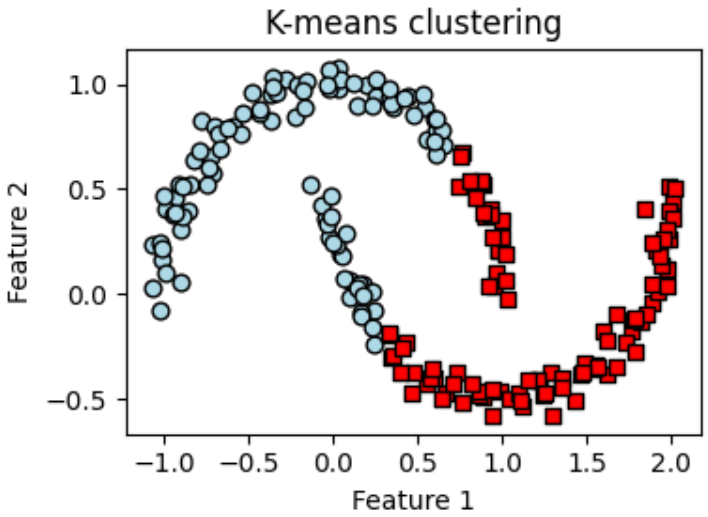

EM ITERATION 10
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
From P. Smyth
ICML 2001

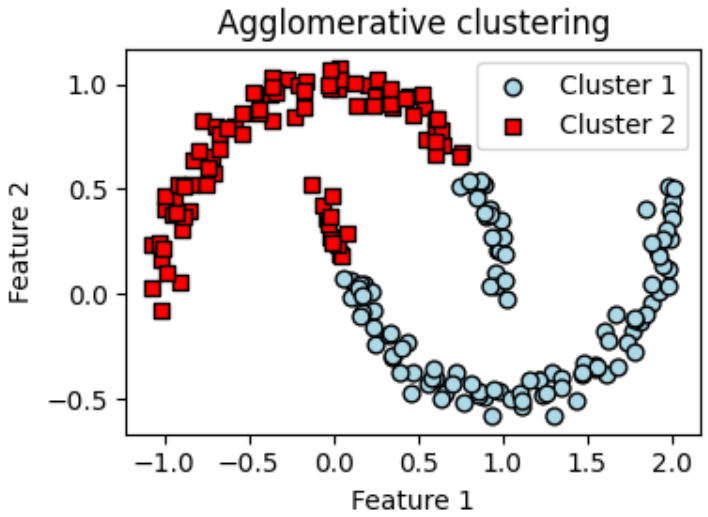

EM ITERATION 15
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
From P. Smyth
ICML 2001

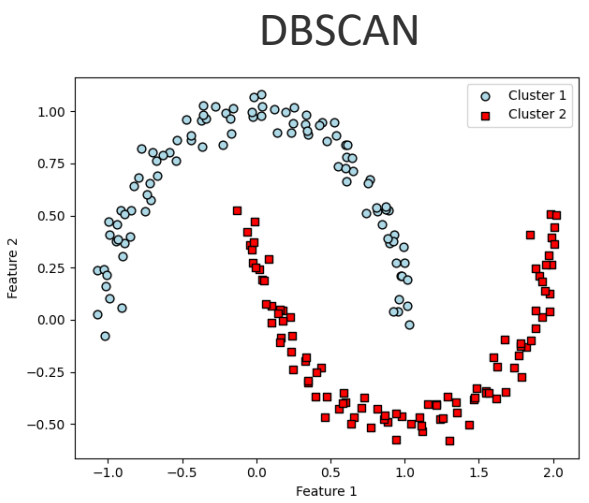

EM ITERATION 25
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
From P. Smyth
ICML 2001


LOG-LIKELIHOOD AS A FUNCTION OF EM ITERATIONS
Log-Likelihood
EM Iteration
490
480
470
460
450
440
430
420
410
400
0
5
10
15
20
25
From P. Smyth
ICML 2001

Density Based Clustering





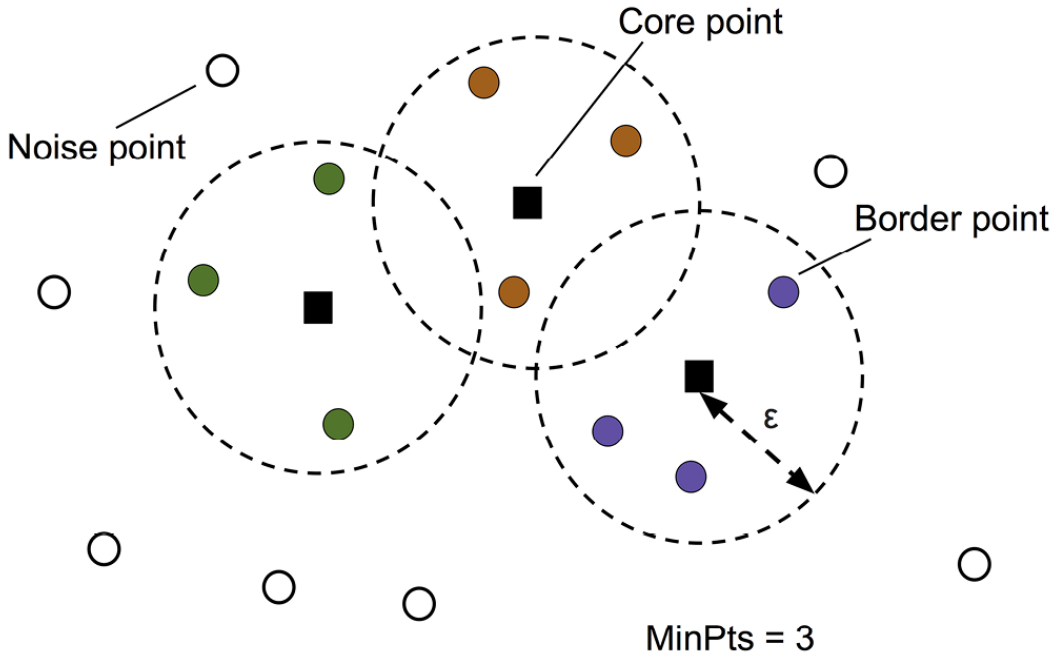


- Relies on a *density-based* notion of cluster: A *cluster* is defined as a maximal set of density-connected points
- Discovers clusters of arbitrary shape in spatial databases with noise

Density Based Clustering

- Two parameters**:**
  - ***Eps***: Maximum radius of the neighbourhood
  - ***MinPts***: Minimum number of points in an Eps-neighbourhood of that point
- $N_{Eps}(p)$: ***{q belongs to D | dist(p,q) <= Eps}***
- Directly density-reachable**:** A point ***p*** is directly density-reachable from a point ***q*** wrt. ***Eps***, ***MinPts*** if
  - ***p*** belongs to $N_{Eps}(q)$
  - core point condition: $|N_{Eps}(q)| >= MinPts$

Density Based Clustering



- Arbitrary select a point ***p***
- Retrieve all points density-reachable from ***p*** wrt ***Eps*** and ***MinPts***.
- If ***p*** is a core point, a cluster is formed.
- If ***p*** is a border point, no points are density-reachable from ***p*** and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed.

Summary

- In clustering, clusters are inferred from the data without human input (unsupervised learning)

- However, in practice, it is very domain specific:
  - Definition of distance in data space
  - Representation of data
  - Defining distance between clusters
  - Number of clusters
  - And so on.

- Practice, practice, practice!

Spectroscopic Imaging

Advancements in imaging led to a broad spectrum of the spectroscopic imaging techniques, in which response spectra are measured in each spatial location giving rise to 3- and higher dimensional data.

Scanning probe microscopy:

- Force-distance curve measurements
- Current-voltage measurements

Electron microscopy:

- Electron Energy Loss Spectroscopy

Optical microscopy:

- Hyperspectral imaging
- Time resolved measurements

Mass-spectrometry:

- Secondary ion MS imaging

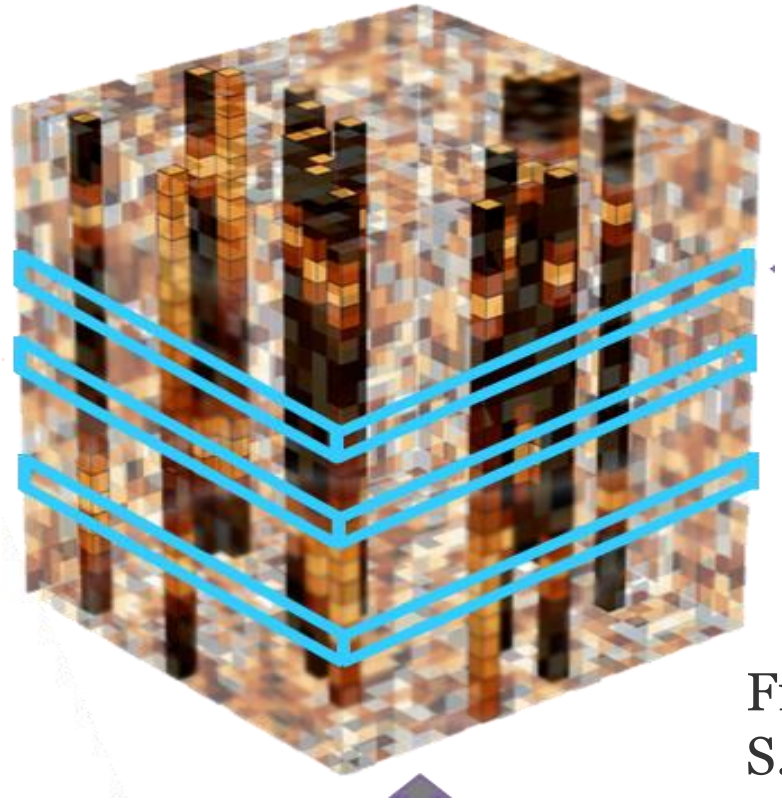
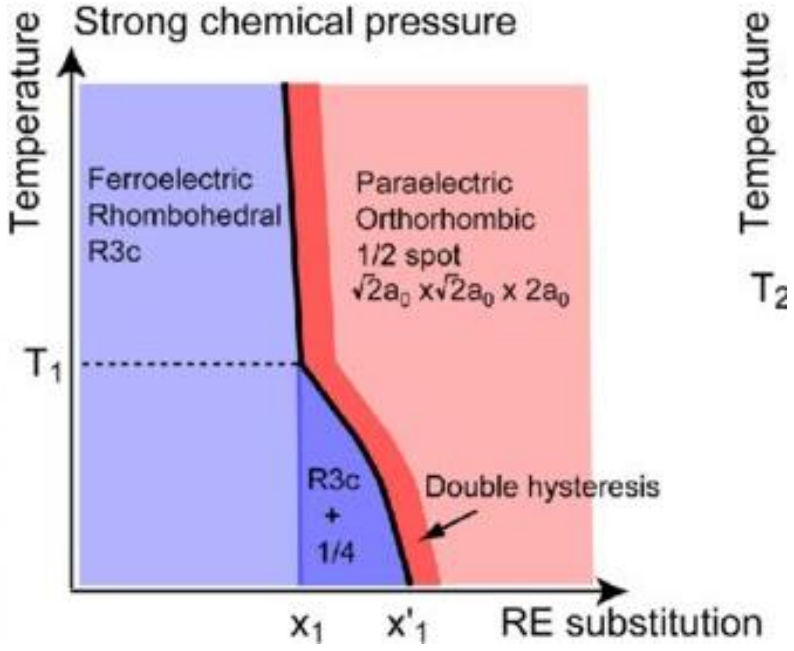
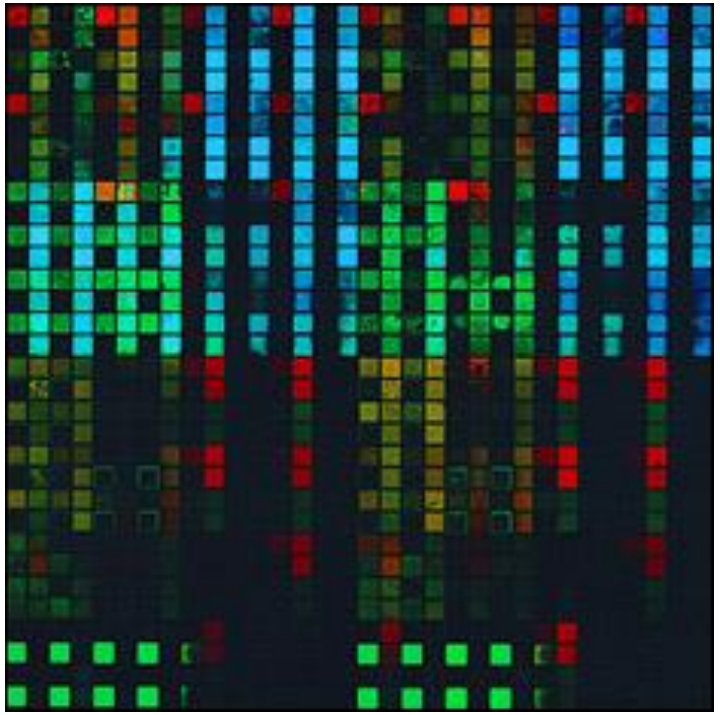


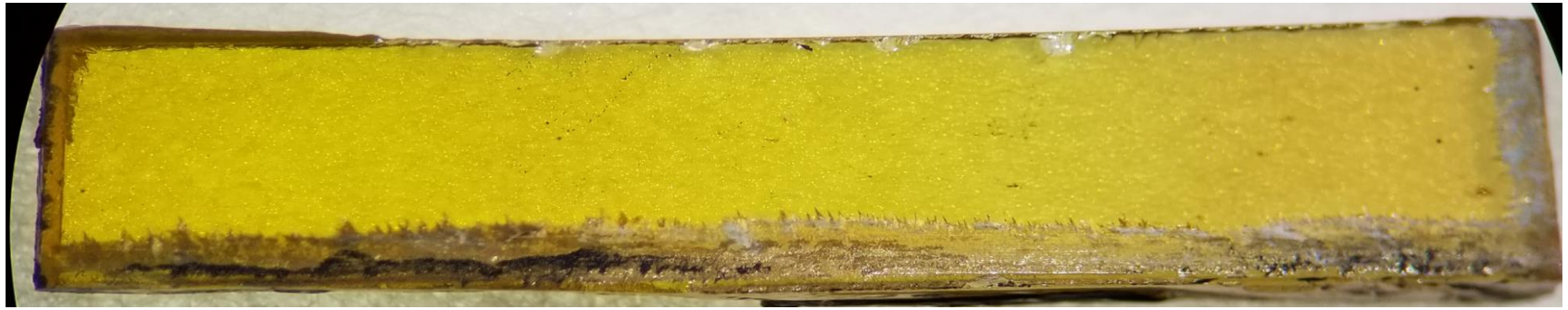
Figure by S. Jesse

Combinatorial Libraries



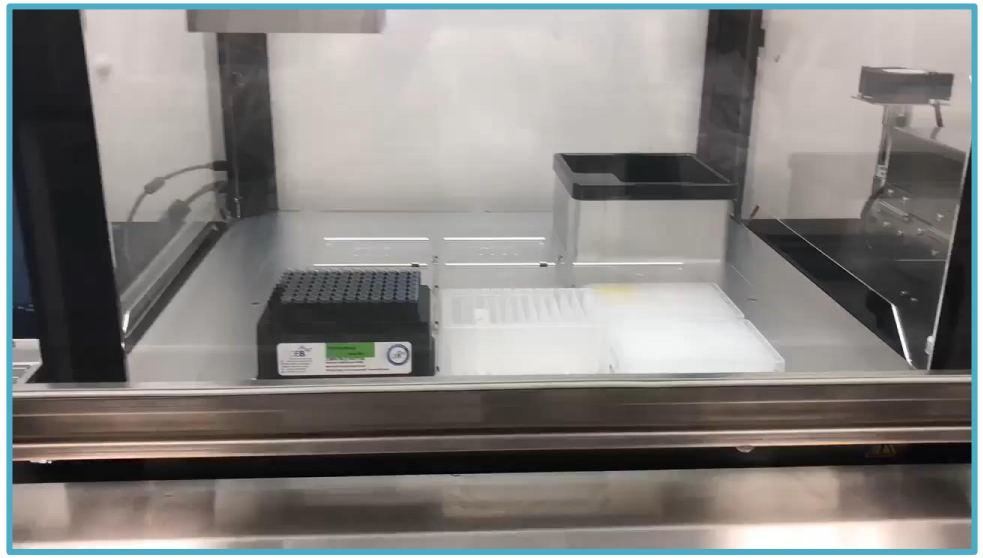


https://mse.umd.edu/research/spotlight/combinatorial

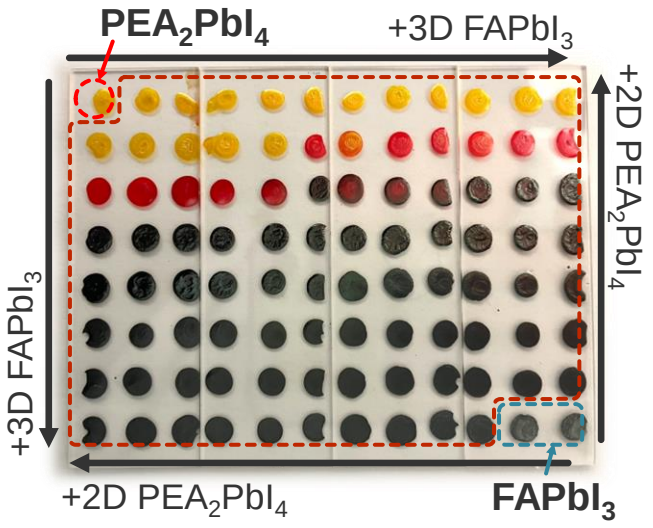


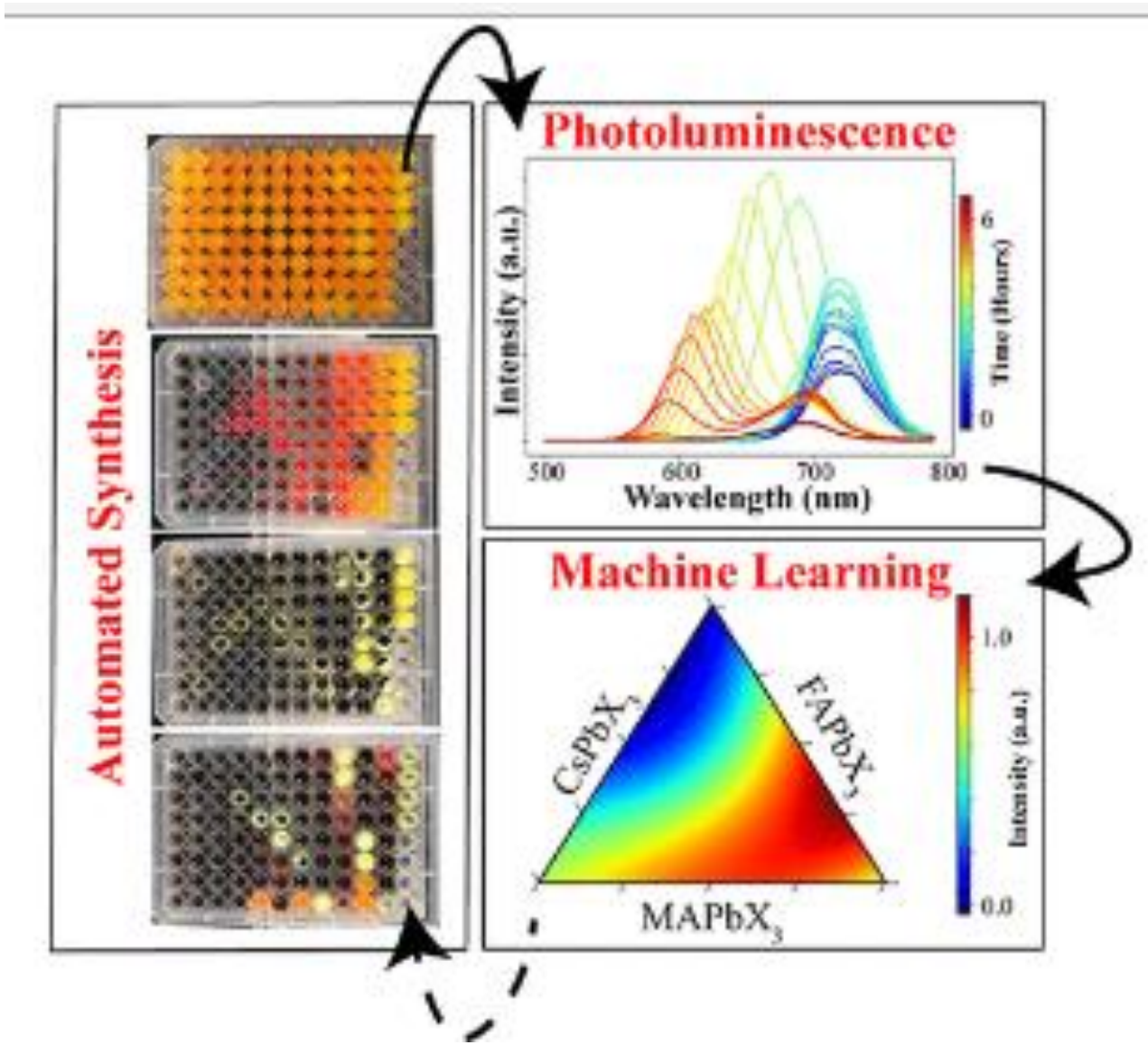
$BiFeO_3$ Linear est. 7%Sm $BiFeO_3$ 20%Sm $BiFeO_3$

Combinatorial Libraries

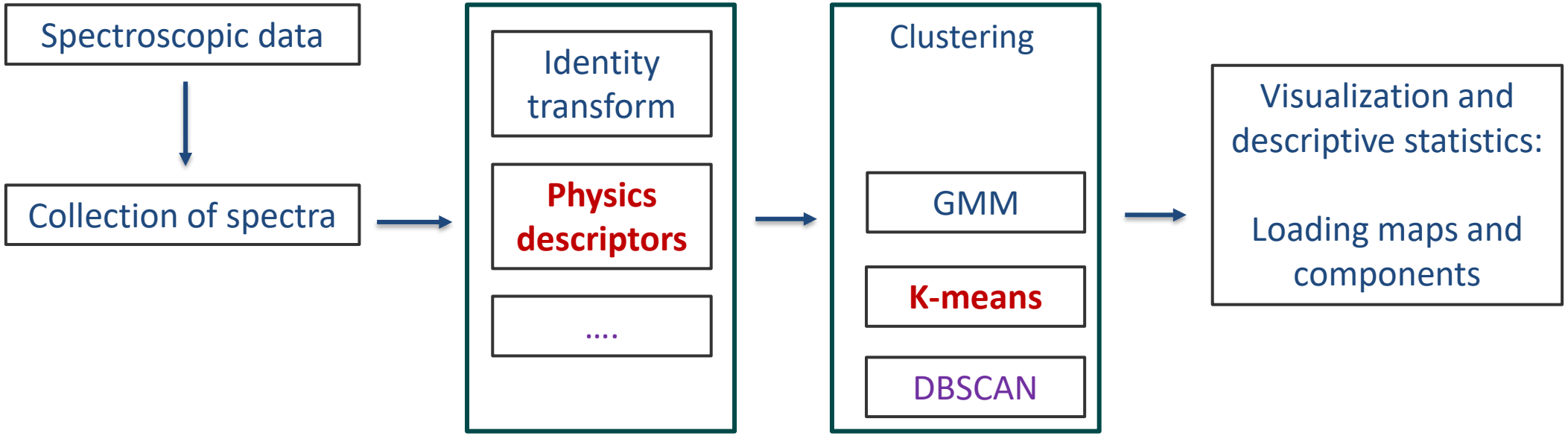


Quasi-2D MHP Film Array





Analysis pipeline

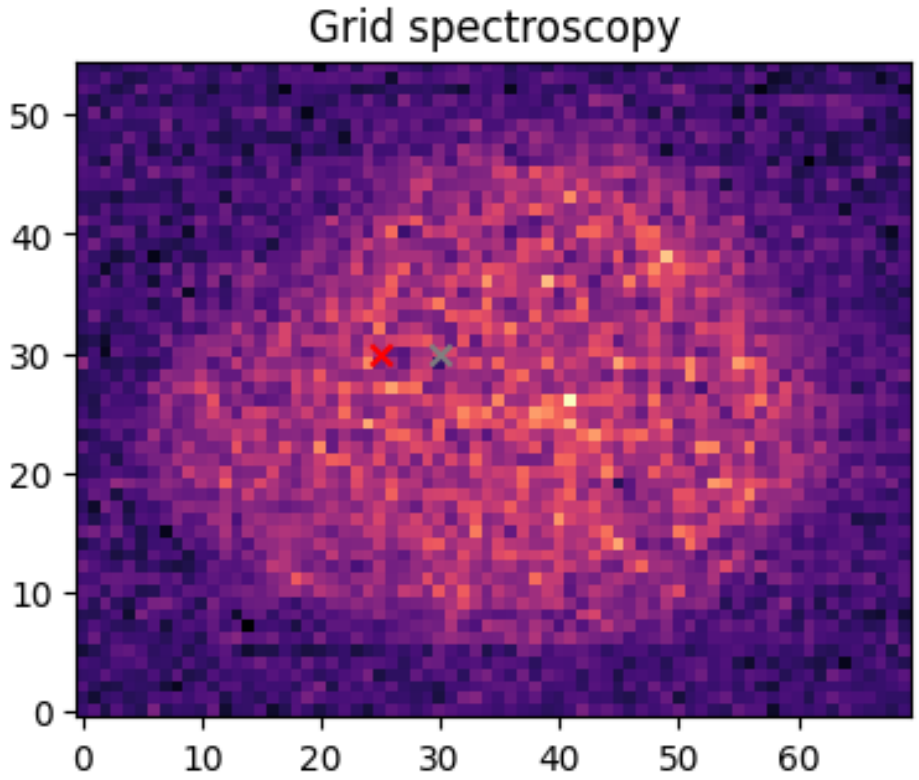


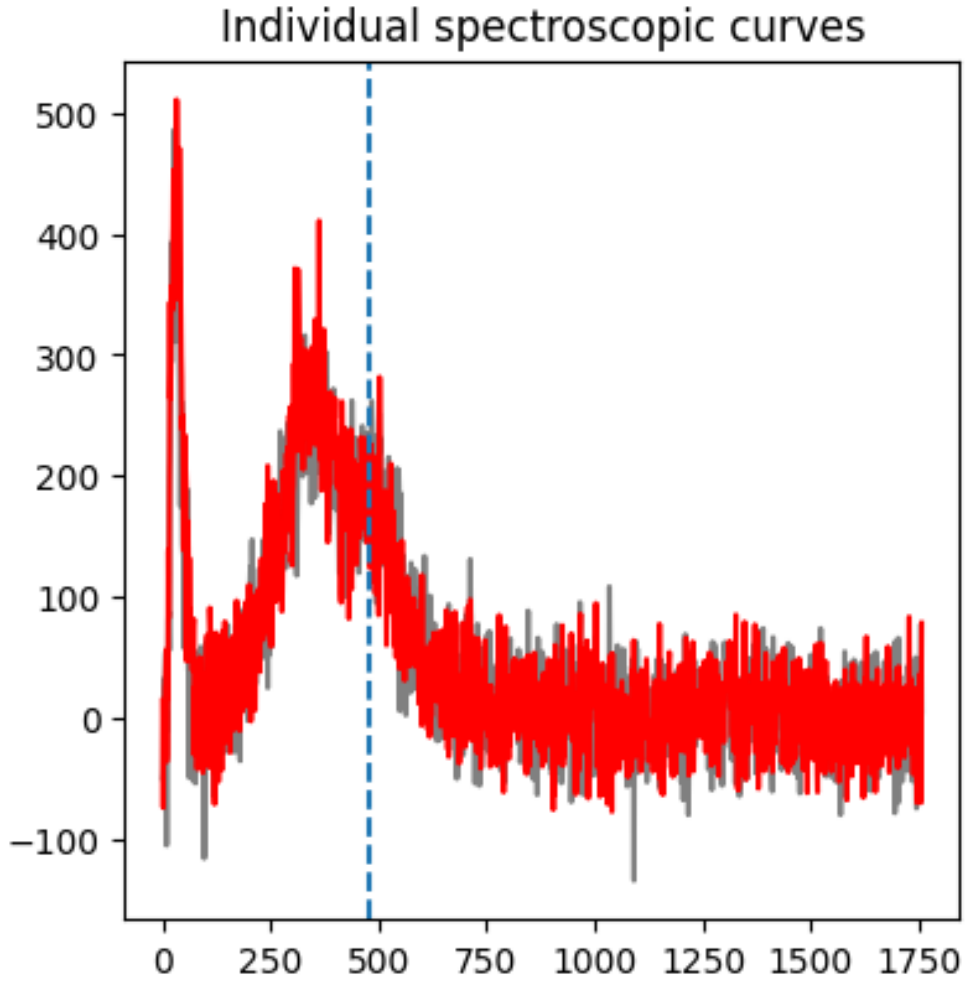
Pipelines are defined to

- Make analysis traceable, repeatable, explainable, and transferable
- Allow for hyperparameter tuning and optimization
- Efficiently use the memory

EELS Colab

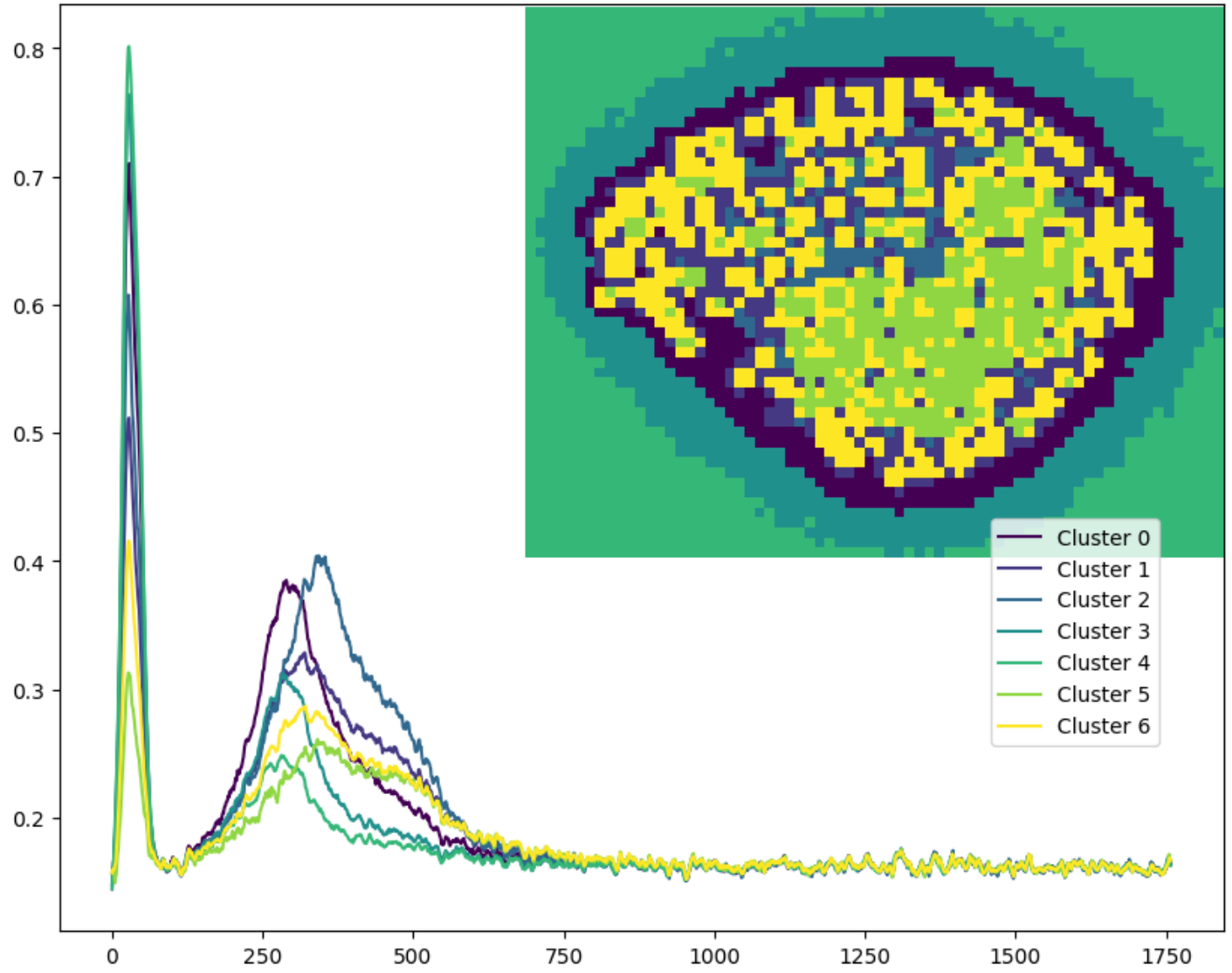
Clustering of spectral data





- The hyperspectral data set contains spectrum at each spatial position on the dense rectangular grid
- We use clustering to establish internal structure of this dataset

Clustering of spectral data

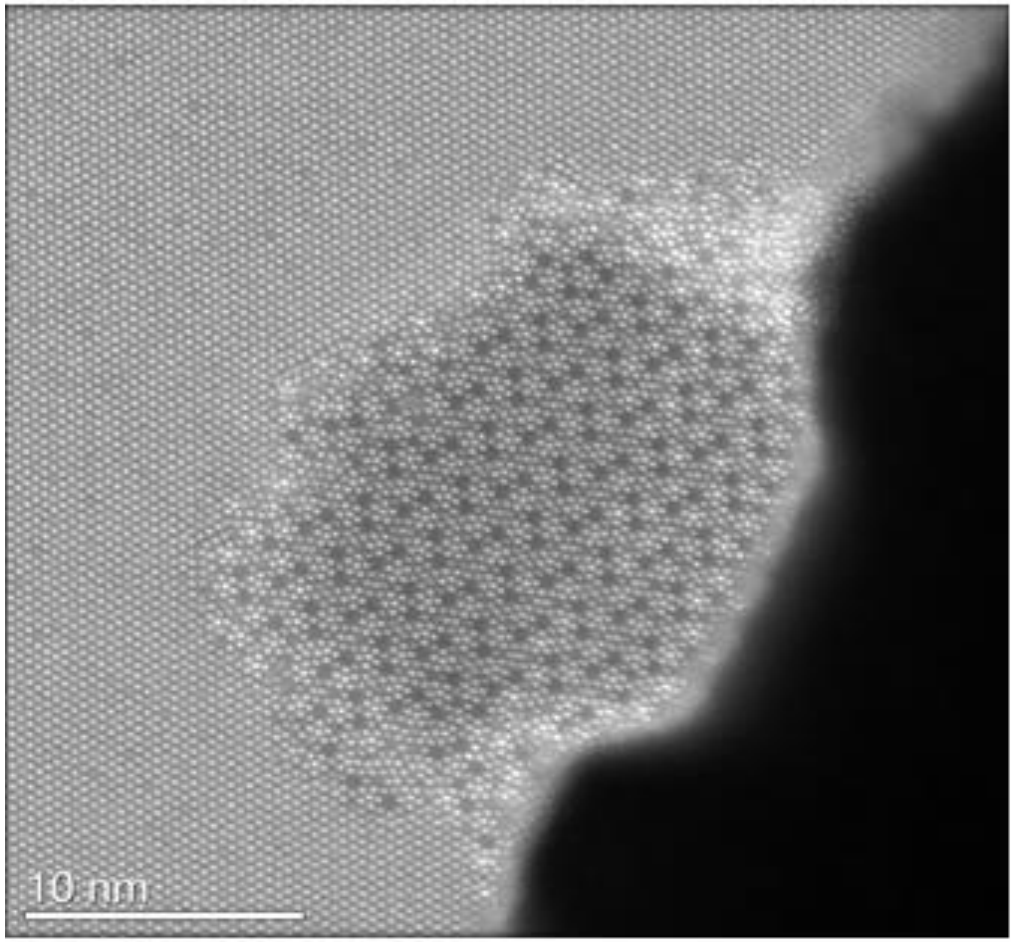


- Experiment with number of clusters
- Based on domain experience, explore the behavior of the components and images of class labels
- This is already “real” research

But what about images?

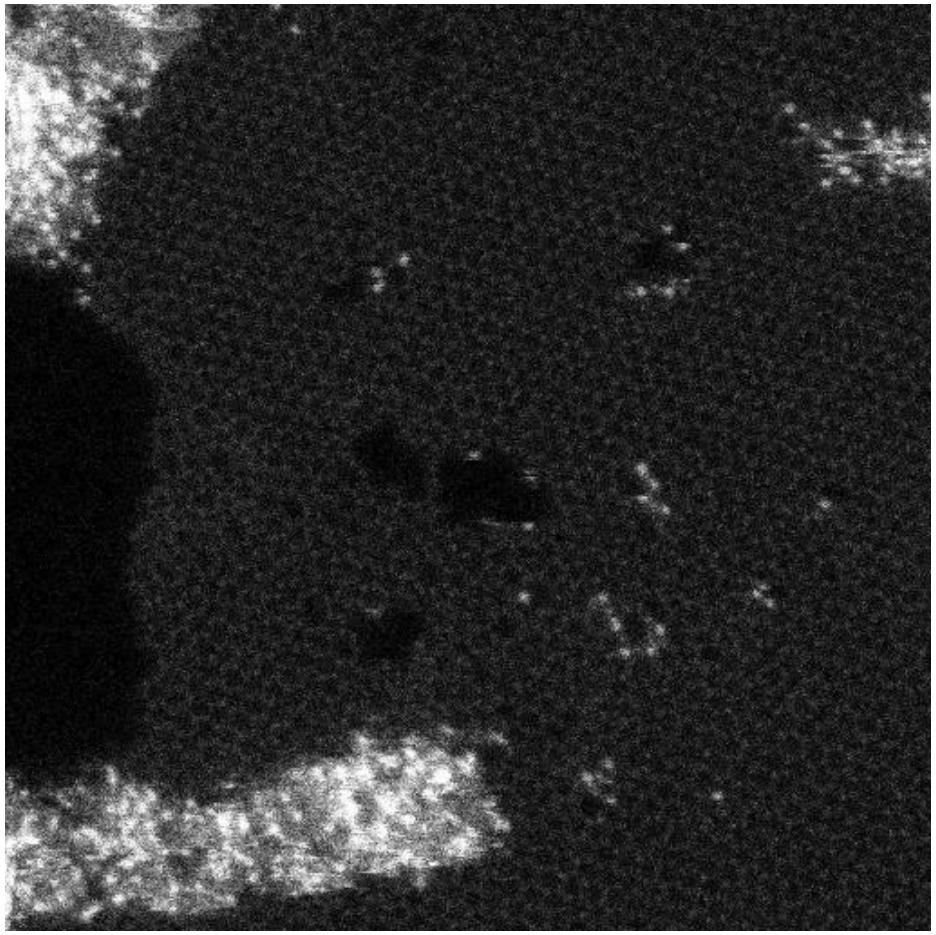
Chemically disordered systems

Mo-V-Ta complex oxide



Q. He et al, ACS Nano 9, 3470-3478

Si in graphene

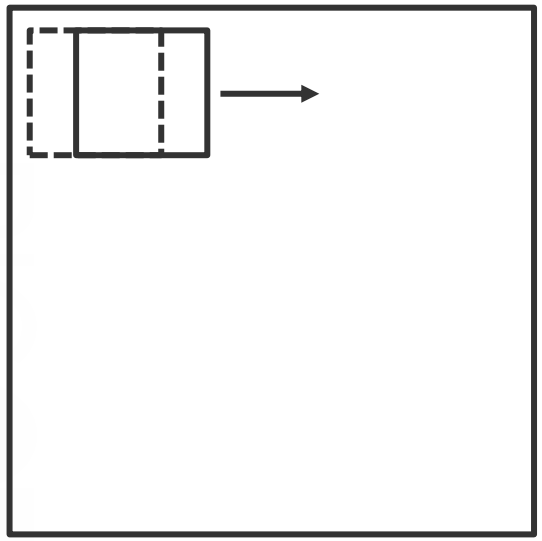


Data collected by O. Dyck (ORNL)

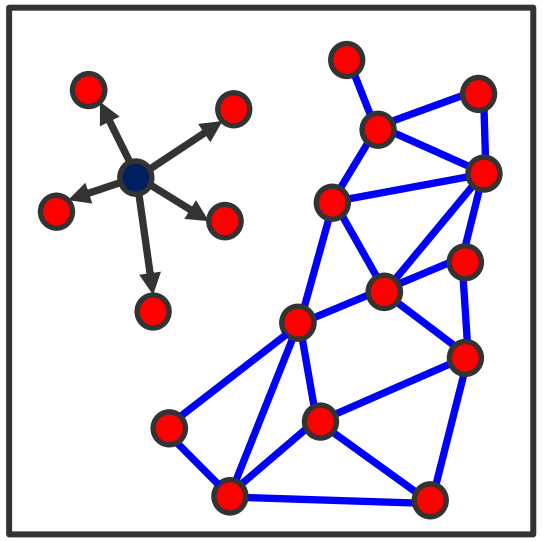
- What is the nature of the building blocks and relevant atomic configurations?
- Can we define single-phase regions and phase boundaries?

Constructing the descriptors

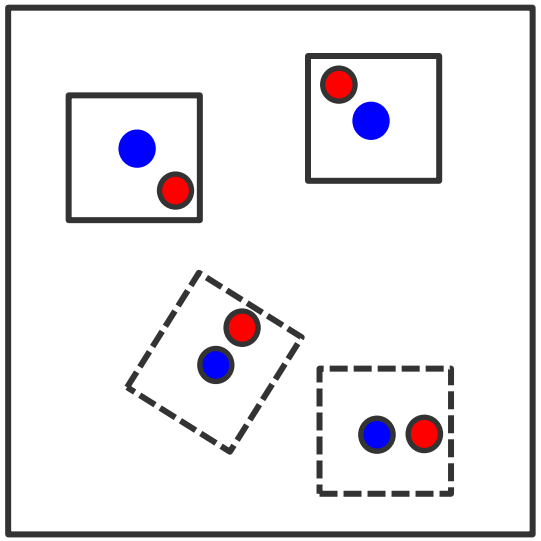
Continuous translational symmetry



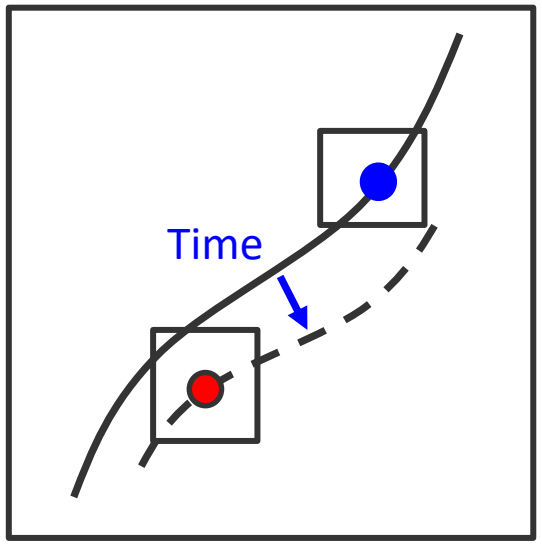
Atom based descriptions



Localized sub-images



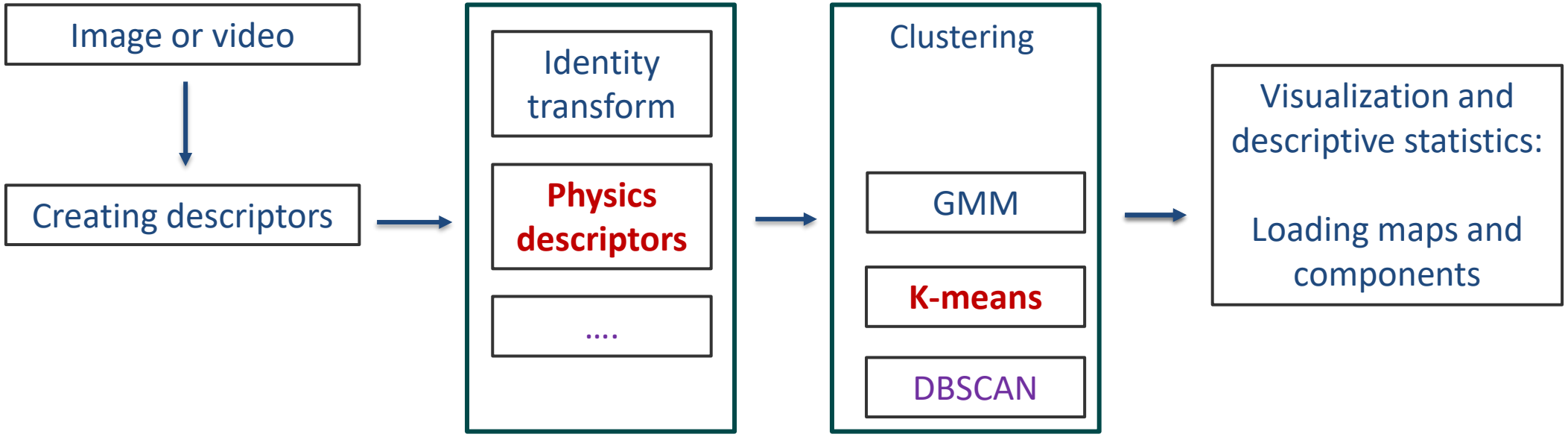
Time-delayed descriptors



The choice of the descriptor:

- Defines physical inferential biases and allows to introduce prior knowledge
- Determines the physical meaning of the analysis
- Establishes the analysis pipeline

Example of analysis pipeline

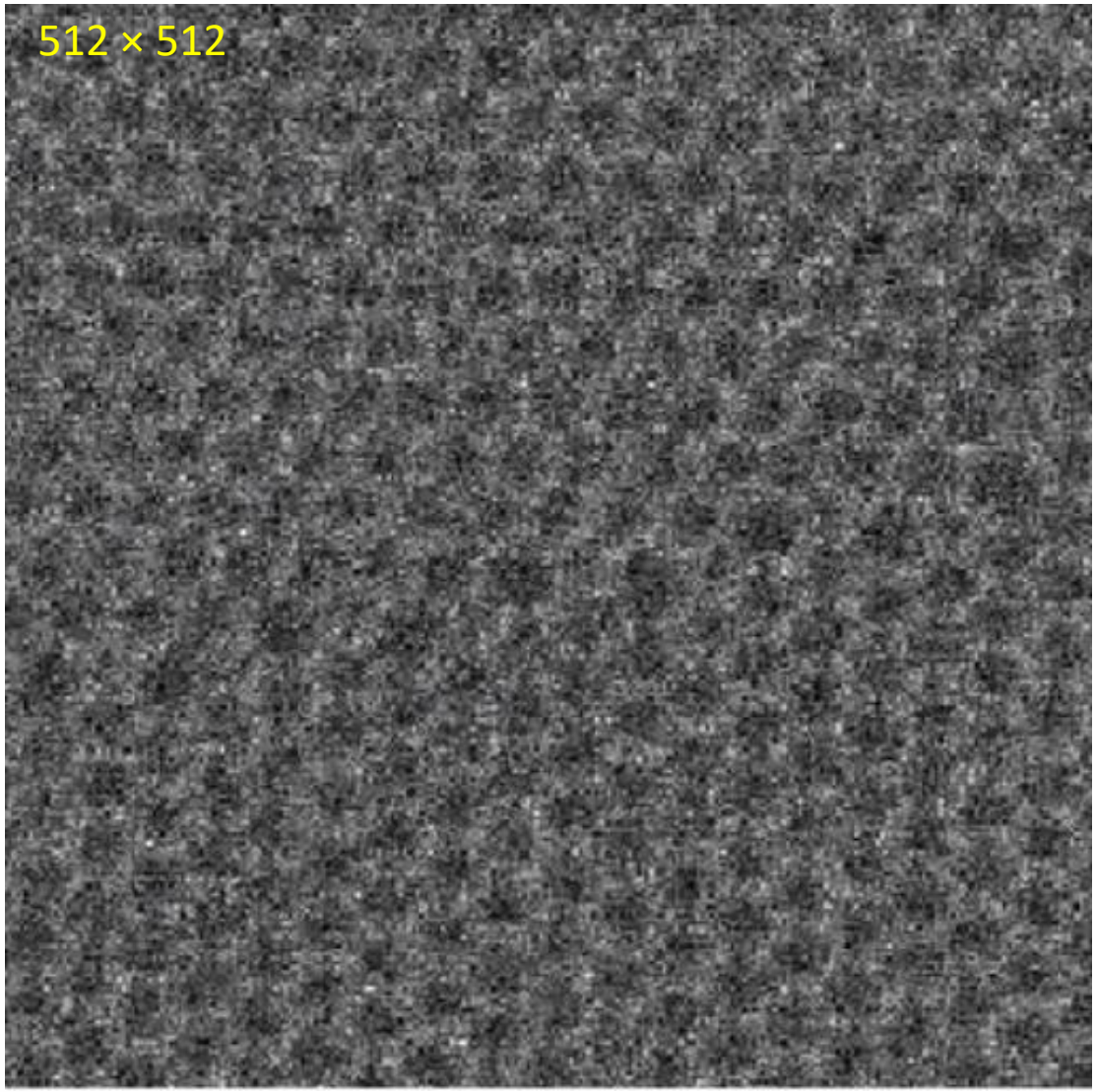


Pipelines are defined to

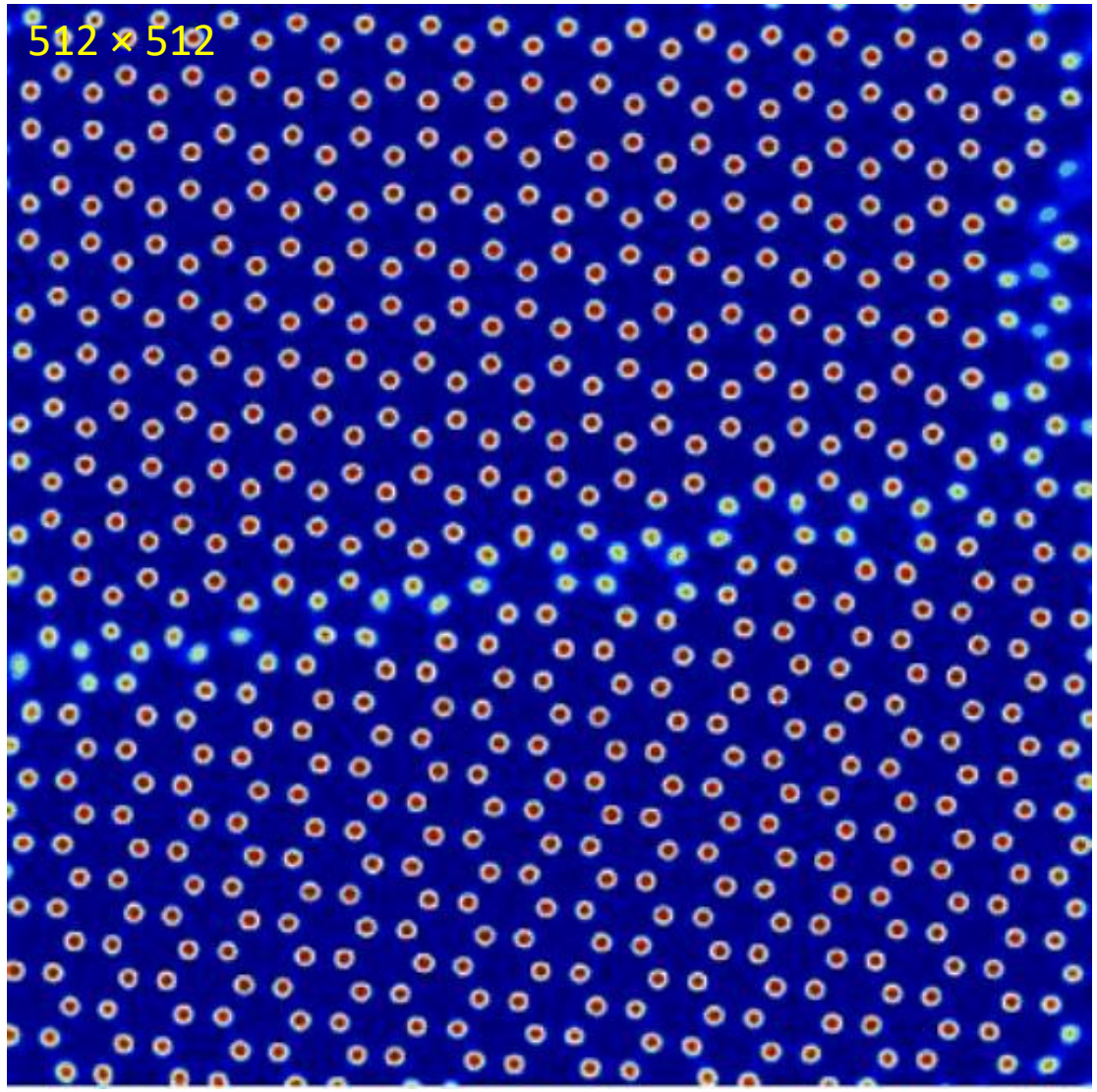
- Make analysis traceable, repeatable, explainable, and transferable
- Allow for hyperparameter tuning and optimization
- Efficiently use the memory

Application to electron microscopy data

Experiment



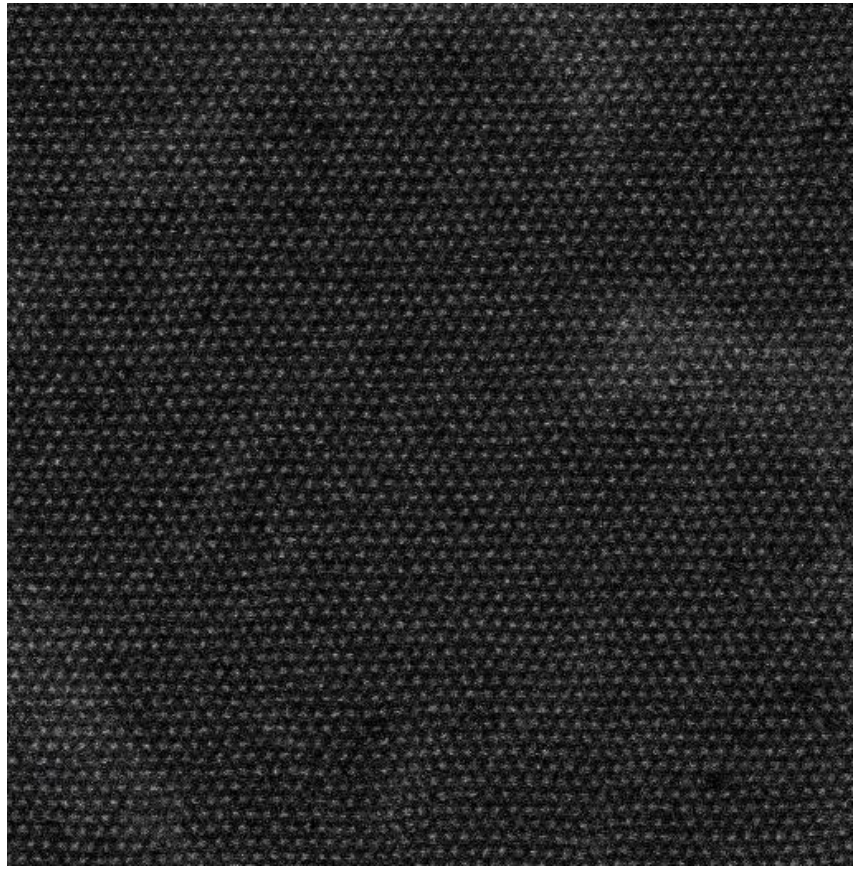
Neural Network Output



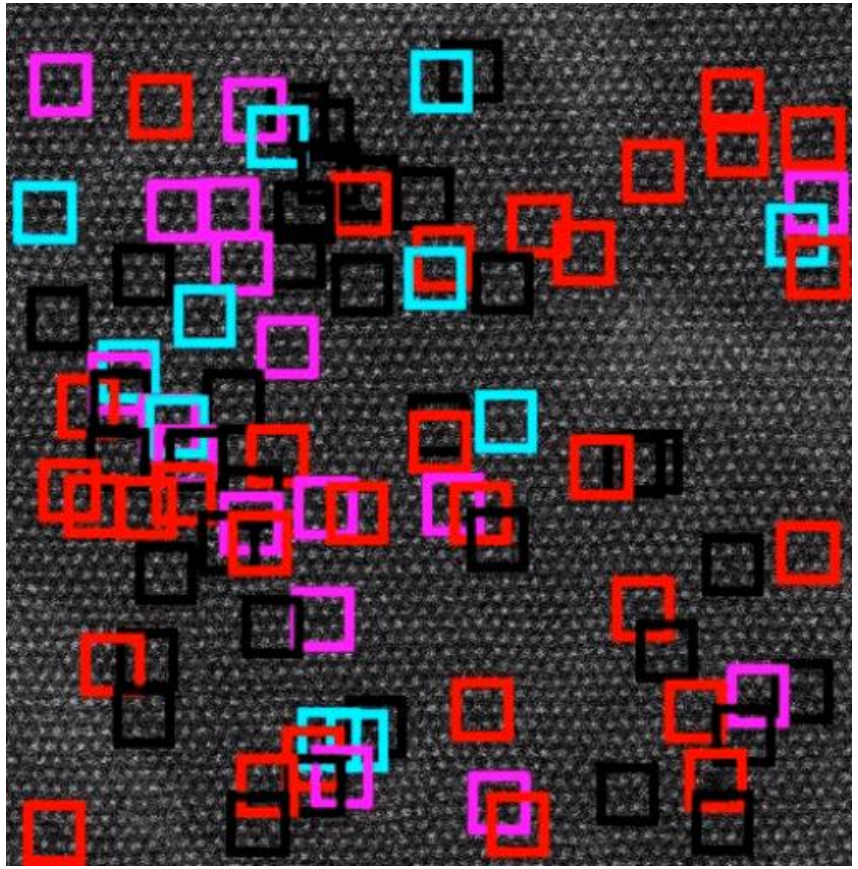
Slide by M. Ziatdinov

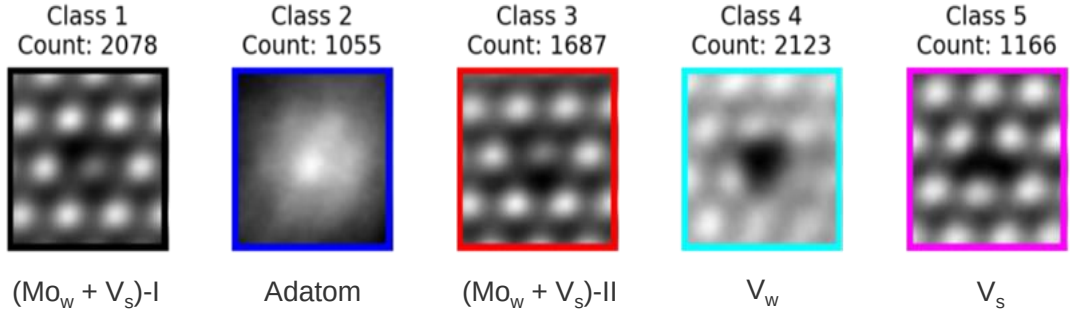
Exploring defect dynamics

Experimental



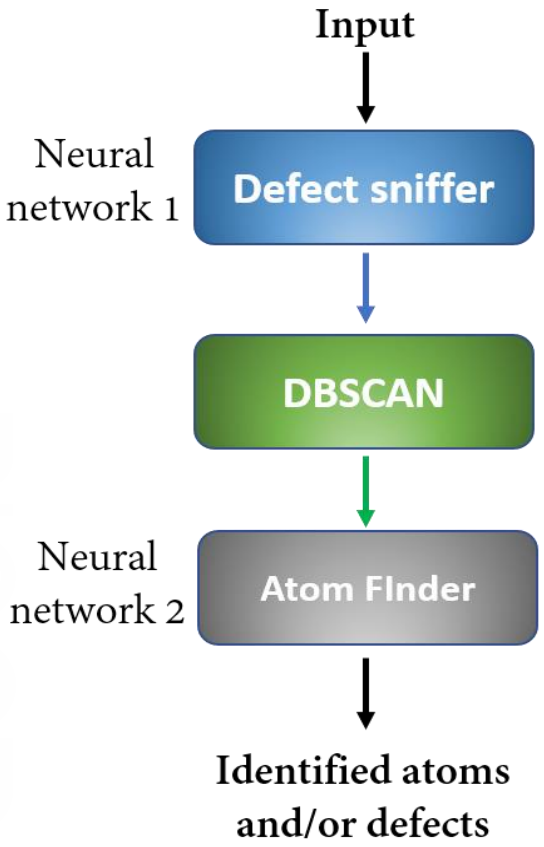
Decoded





Sample: WS2
E-beam energy: 60 kV
Data by Ondrej Dyck (CNMS/ORNL)

Practically: pipelines of simpler NNs



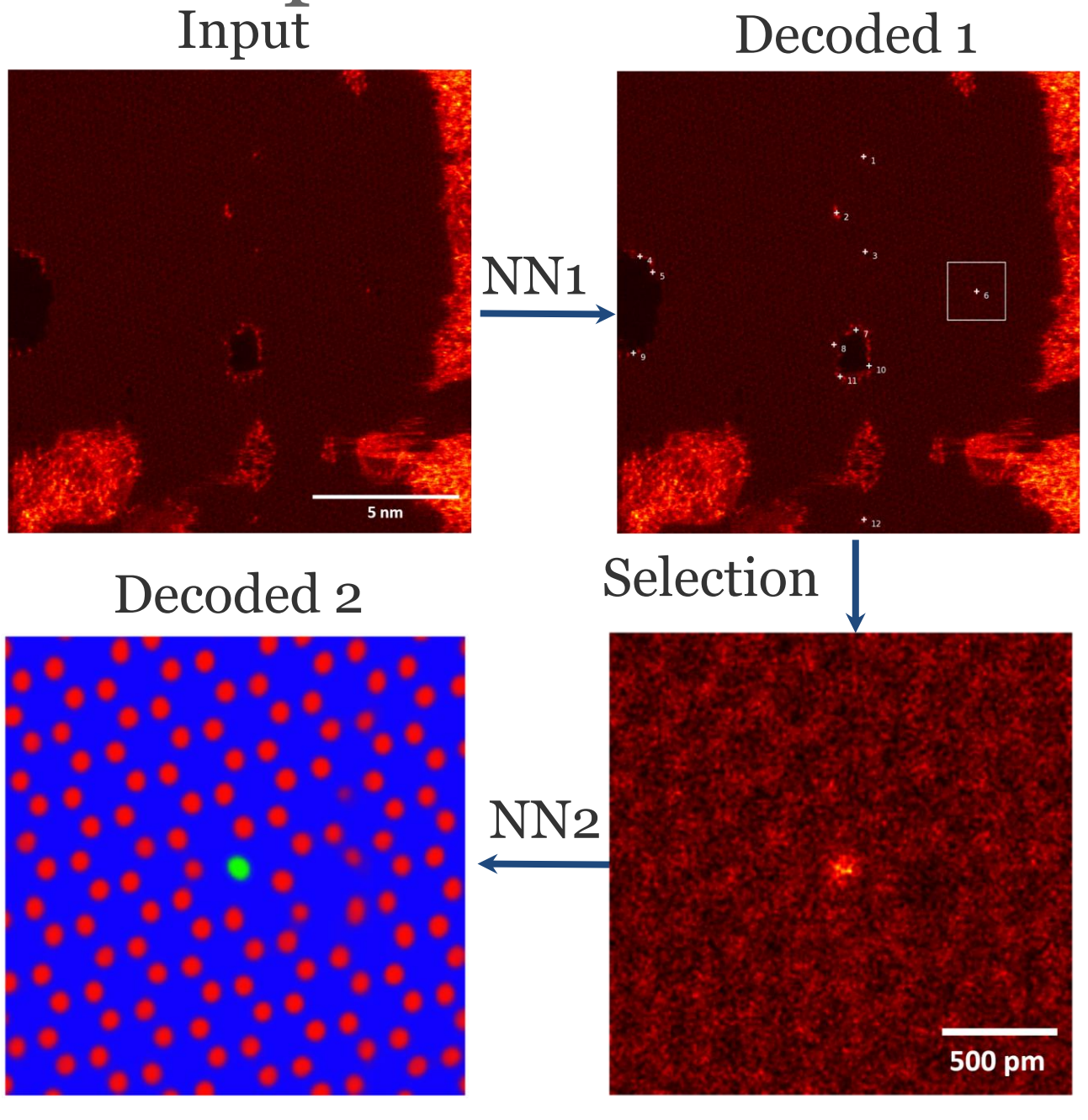
Finding needle in a haystack

We replace categorical cross entropy (CE) with focal loss (FL) function

$$CE(p_t) = -\log(p_t)$$

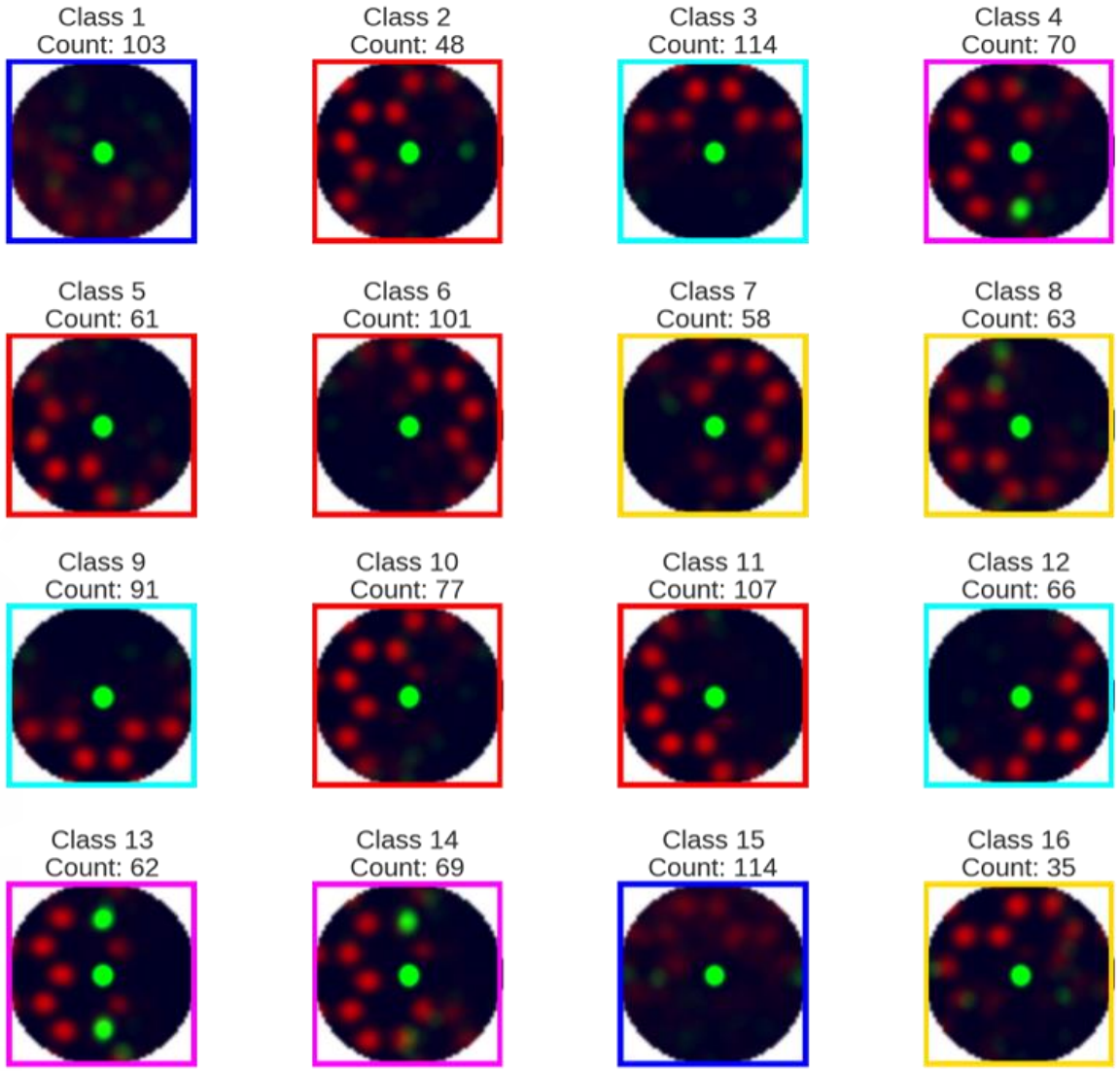
$$FL(p_t) = -(1-p_t)^{\gamma}\log(p_t)$$

T.-Y. Lin et al., arXiv 1708.02002 (2018)

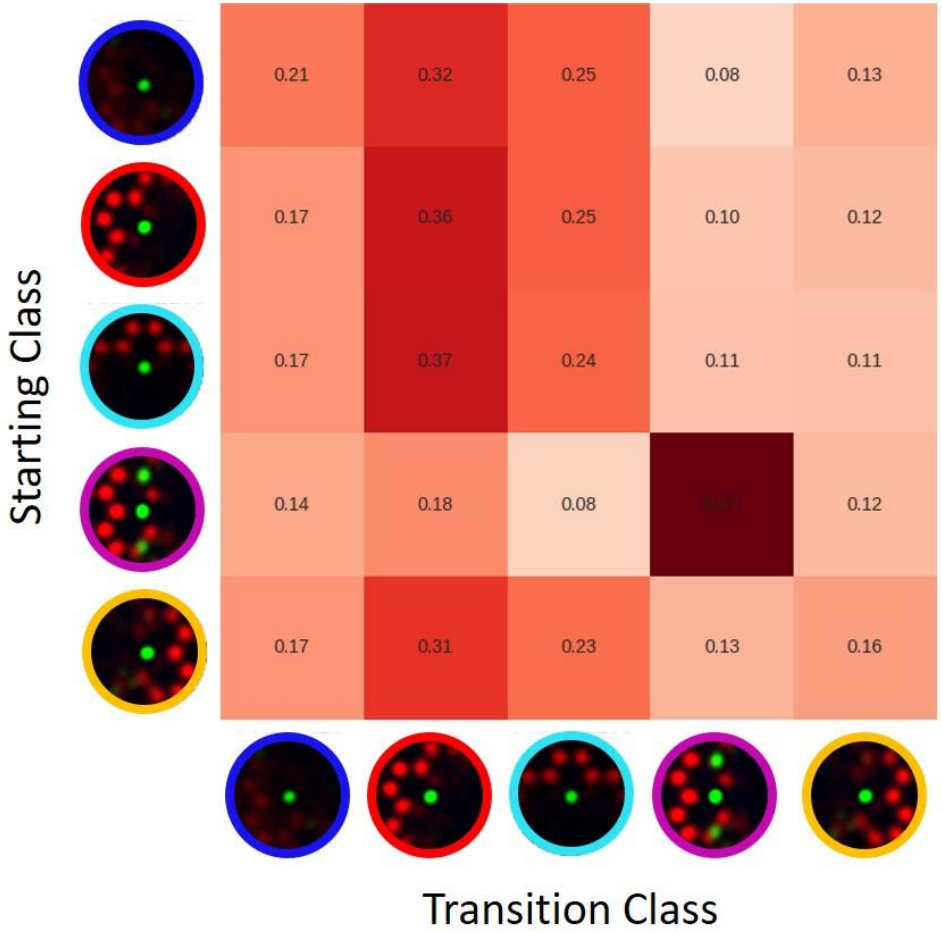


Classification of edge states

Derived classes of Si-C edge configurations



Transition probabilities matrix



- Gaussian mixture model
- Discrete rotation symmetry
- Markov state analysis

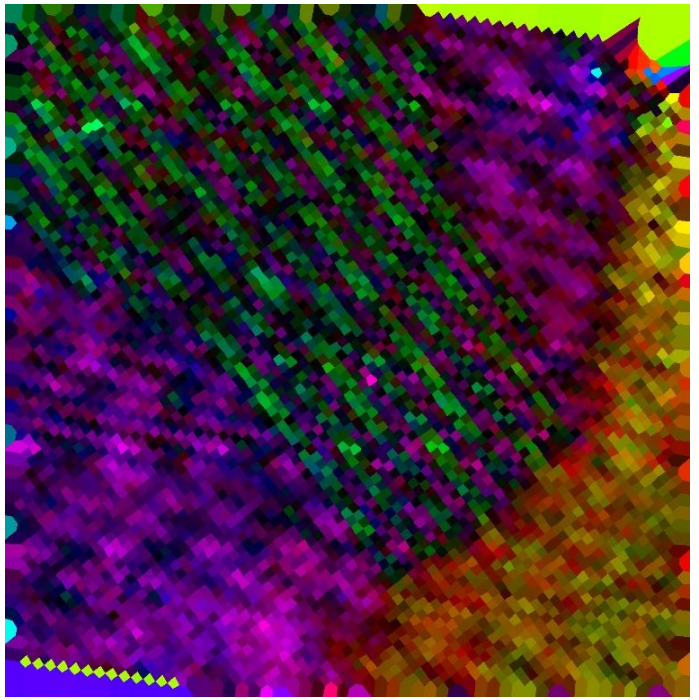
Ziatdinov *et al.*, ArXiv:1901.09322 (2019)

Crystalline materials

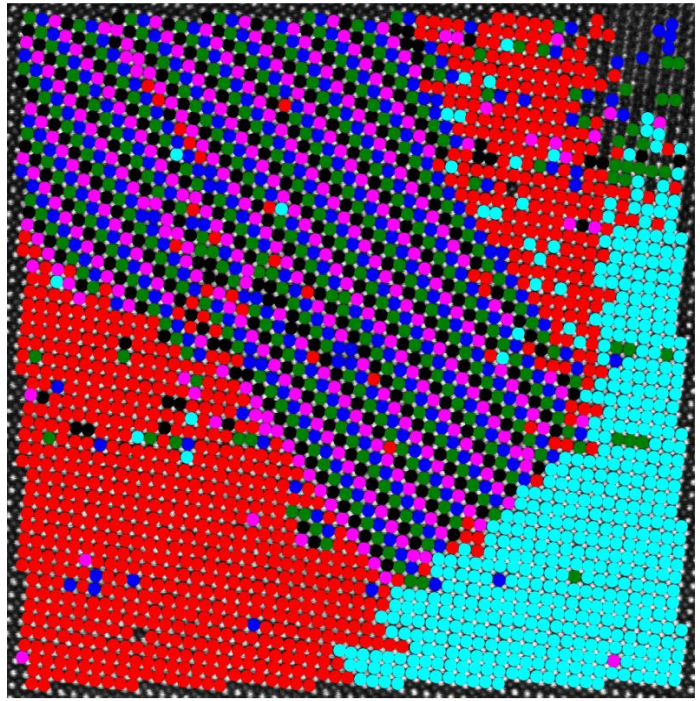
Experimental data (Ferroelectric LBFO)

Data by C. Nelson (MSTD ORNL)

Domain expert analysis

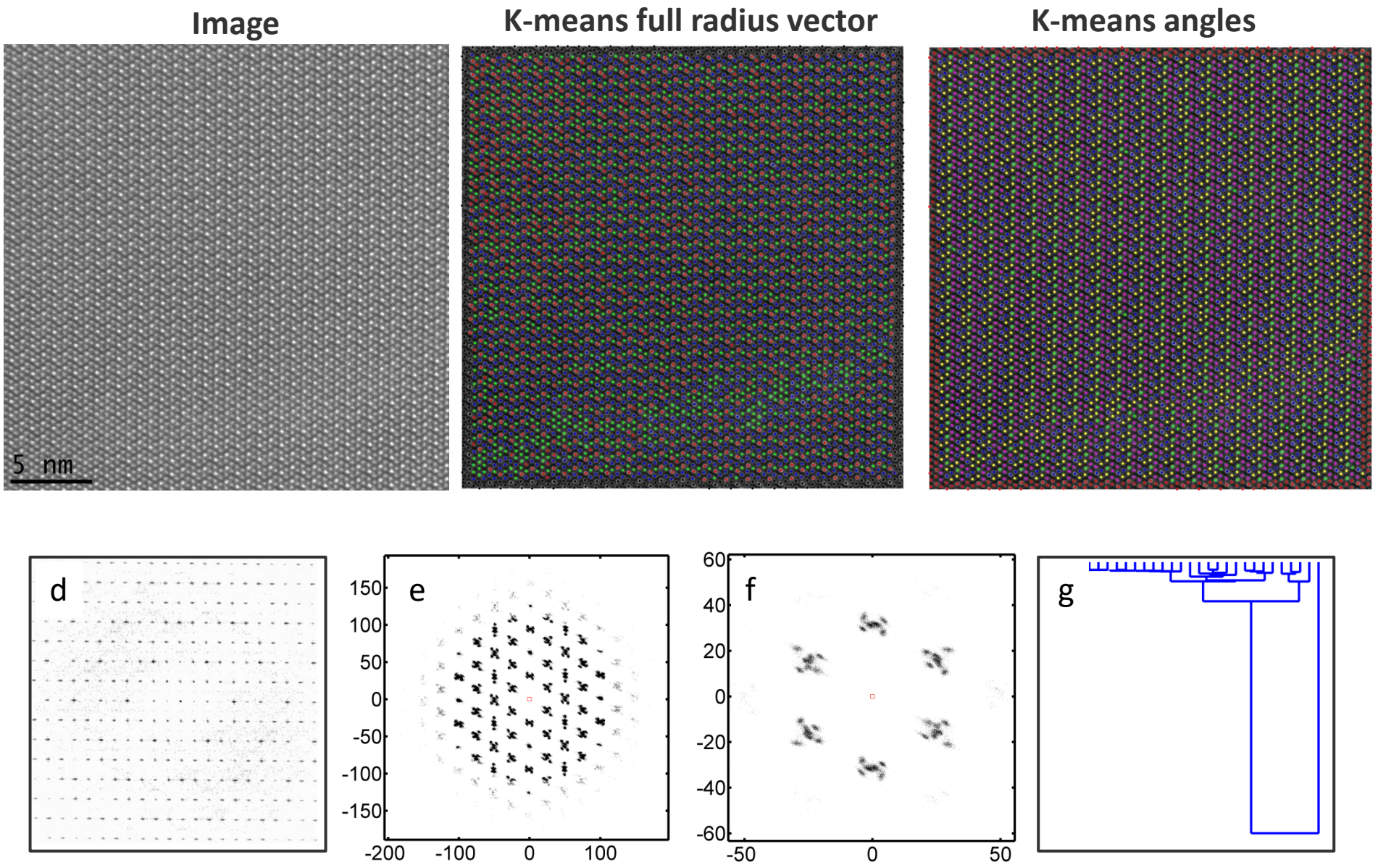


DCNN + k-means (~ 1-3 minutes)



CNN models work with patch by patch and sliding window approaches to image analysis. This allowed us to analyze high resolution images. CNN models were trained using Multislice simulations *and* simulations when atoms are modelled simply as 2D gaussians.

Local crystallography



High Dimensional Data – what should we do?

Examples of high D data:

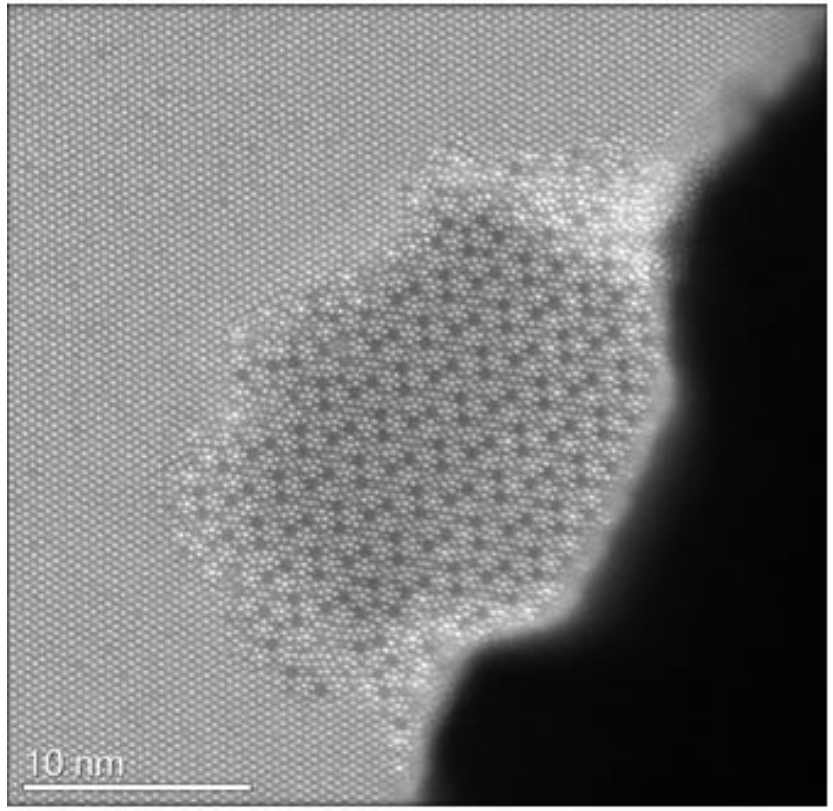
- Face recognition
- Image compression
- Gene expression analysis
- Spectroscopy
- 4D STEM
- X-Ray scattering

What do we want to accomplish?

- Reduce number of dimensions in data
- Find patterns in high-dimensional data
- Visualize data of high dimensionality

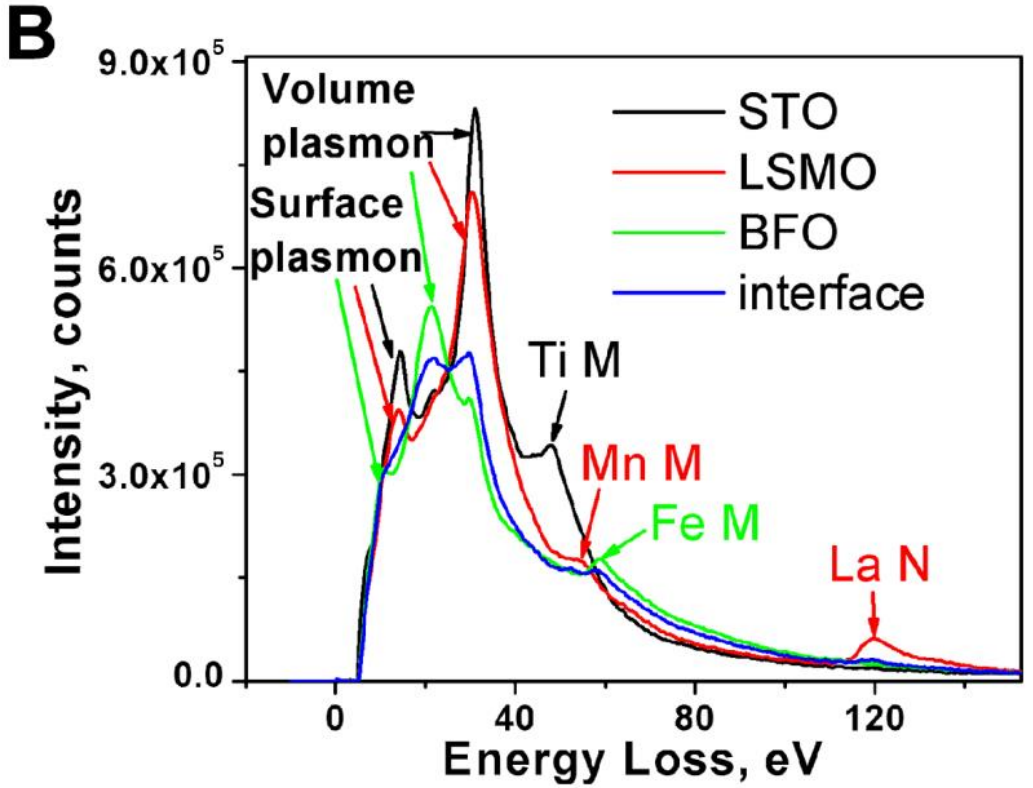
High D data in materials science?

Mo-V-Ta complex oxide



Q. He et al, ACS Nano 9, 3470-3478

Low-loss EELS spectra



A.Y. Borisevich et al., Phys. Rev. Lett. **105**, 087204 (2010).

- How many dimensions are in this data?
- Are all these dimensions necessary?
- For given acquisition time, how would the noise and signal balance?
- How do we extract “useful” information?

High Dimensional Data is often redundant

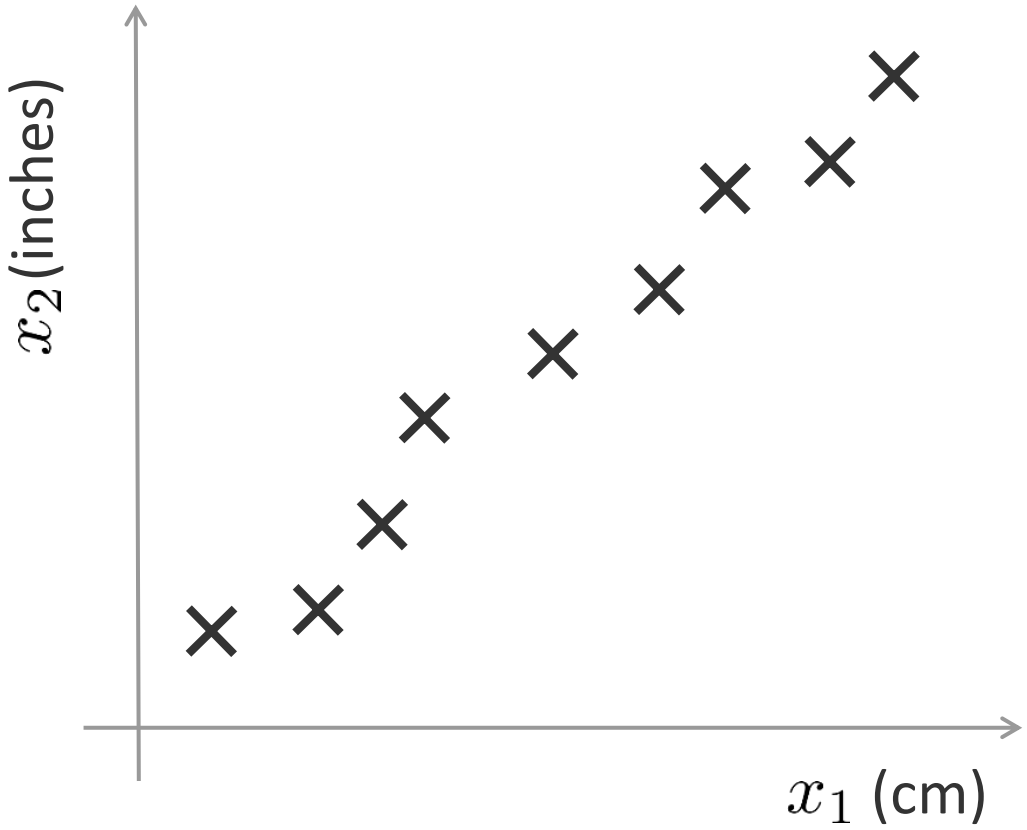
- We often need a method to simplify data on a large number of variables, and believe that there is some redundancy in those variables.
- Redundancy means that some of the variables are correlated with one another, possibly because they are measuring the same object or phenomenon.
- Because of redundancy, we believe that it should be possible to reduce the observed variables into a smaller number of artificial variables that will account for most of the variance in the observed variables.

Dimensionality Reduction Methods

- PCA (Principal Component Analysis):
  - Find projection that maximize the variance
- ICA (Independent Component Analysis):
  - Very similar to PCA except that it assumes non-Guassian features
- Multidimensional Scaling:
  - Find projection that best preserves inter-point distances
- LDA(Linear Discriminant Analysis):
  - Maximizing the component axes for class-separation
- Bayesian Linear Unmixing
  - Linear unmixing, non-negative, sum to one
- … constrained linear unmixing methods

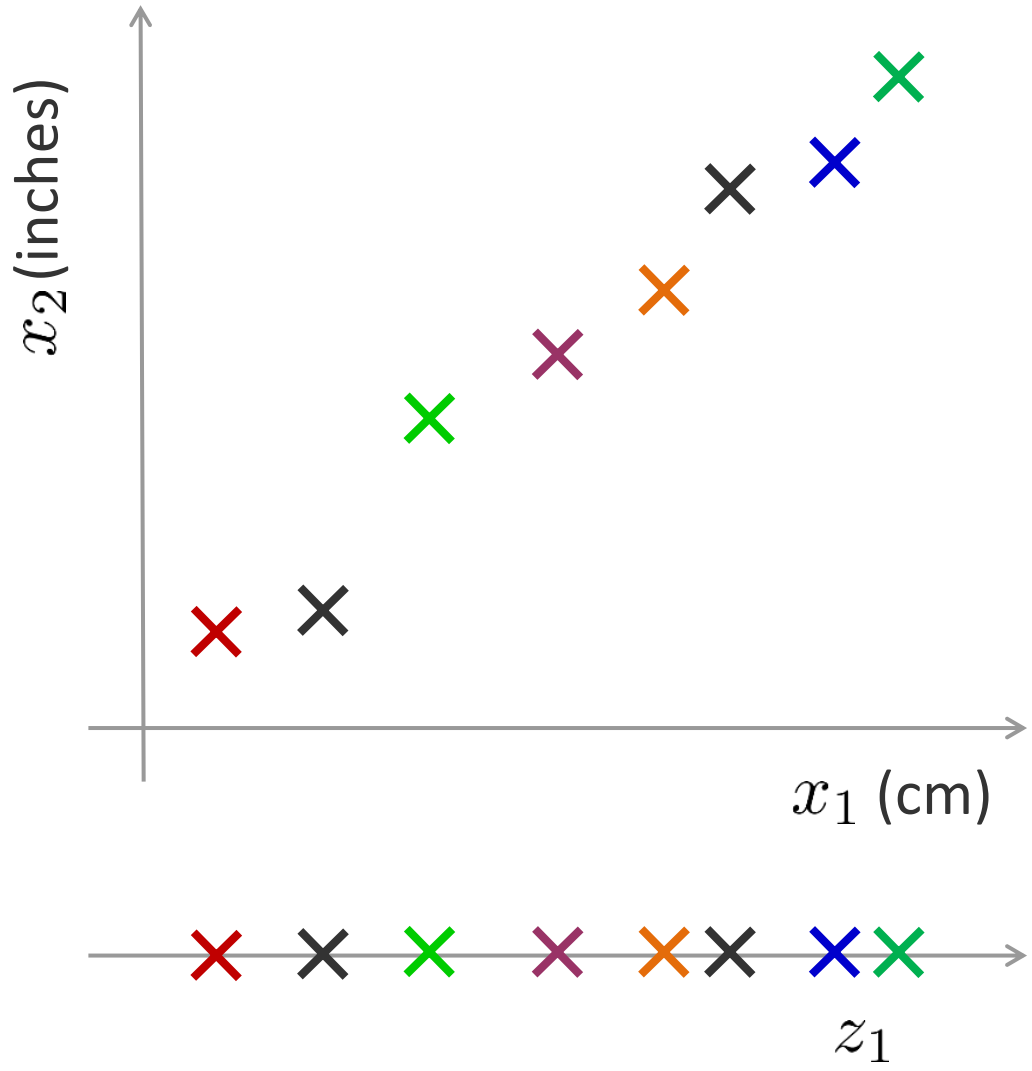
Manifold Hypothesis!

Simple Example



Reduce data from 2D to 1D

Simple Example



Reduce data from 2D to 1D

$$x^{(1)} \rightarrow z^{(1)}$$

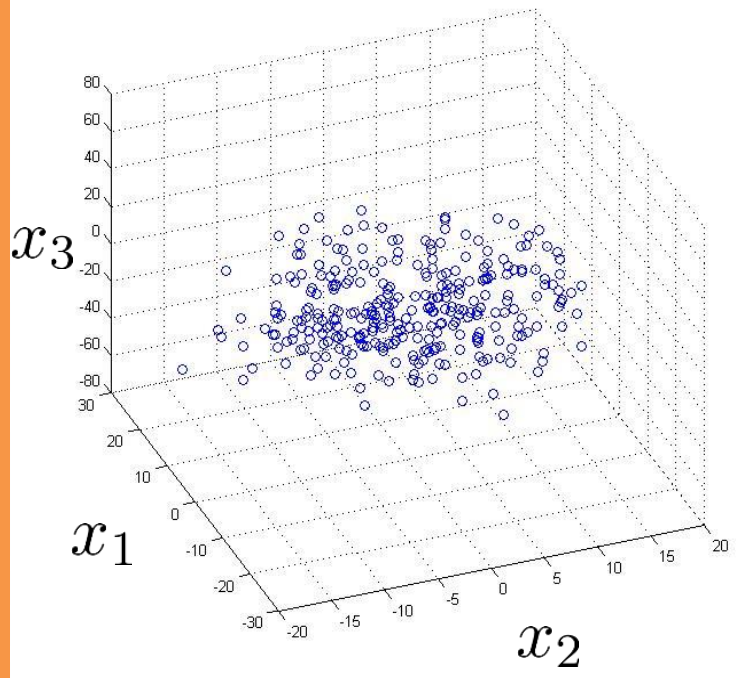
$$x^{(2)} \rightarrow z^{(2)}$$

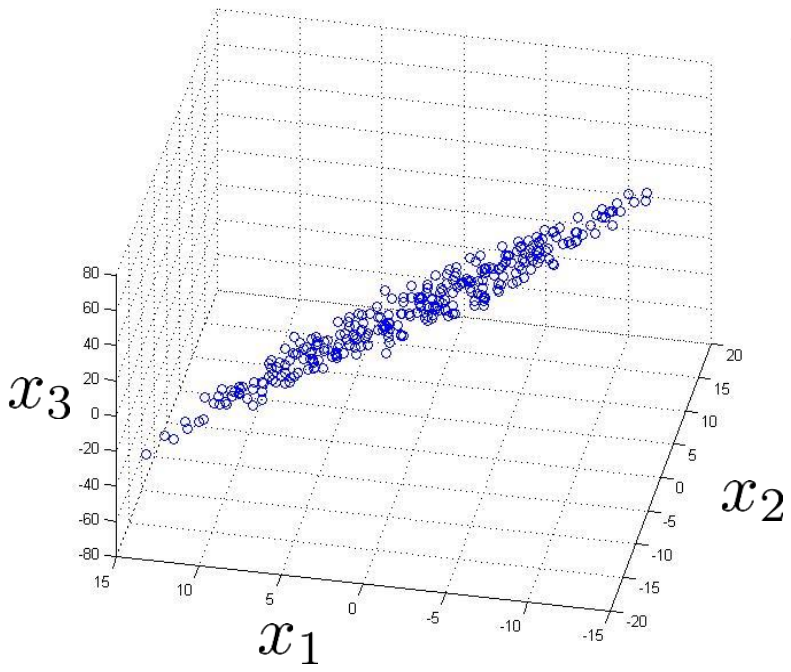
$$\vdots$$

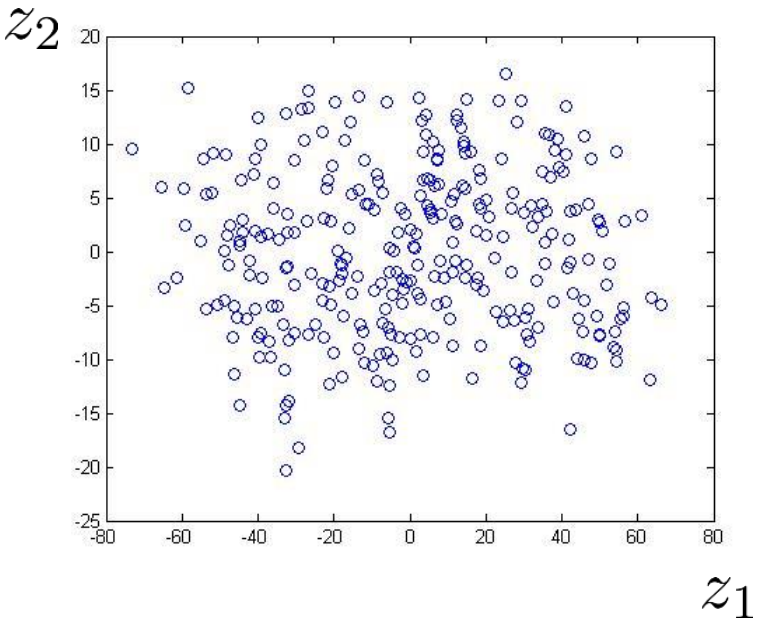
$$x^{(m)} \rightarrow z^{(m)}$$

Another Simple Example

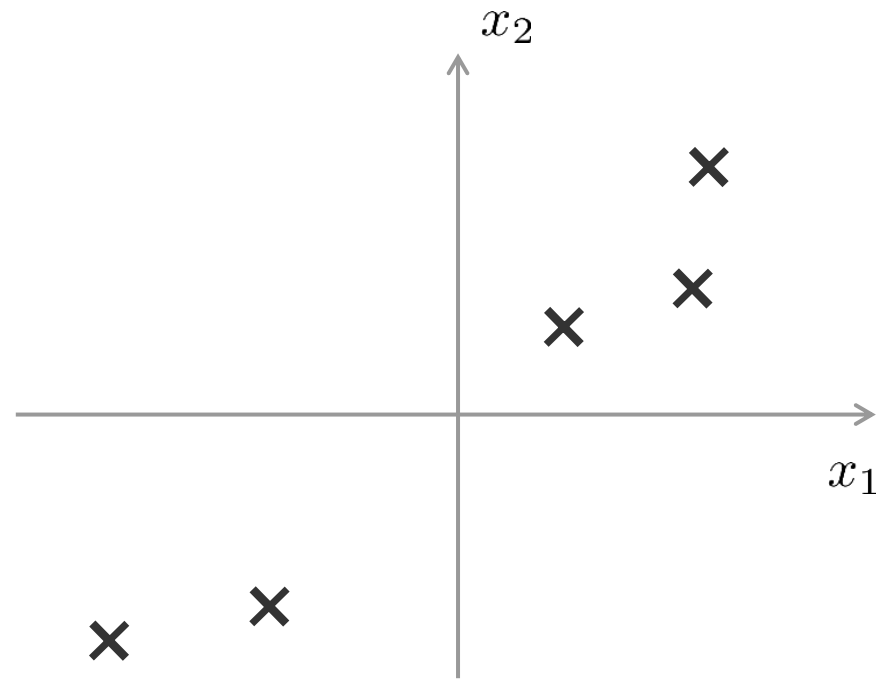
Reduce data from 3D to 2D



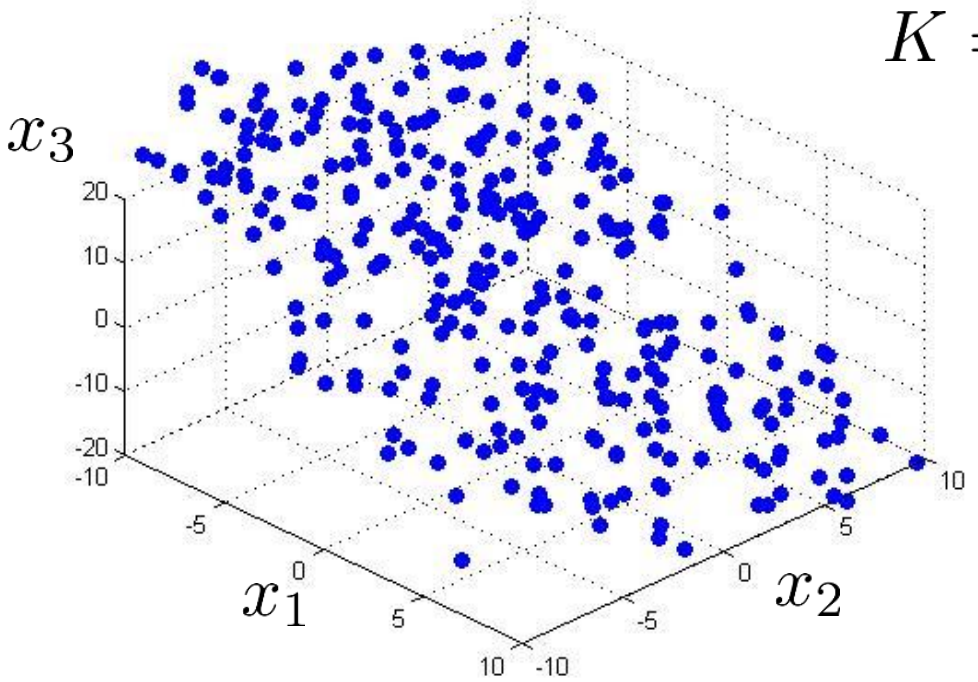




Generalize the problem



$3D \rightarrow 2D$
$K = 2$



Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)} \in \mathbb{R}^n$) onto which to project the data so as to minimize the projection error.

Reduce from n-dimension to k-dimension: Find k vectors $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ onto which to project the data, so as to minimize the projection error.

Variance and covariance

1D: Variance=(Standard deviation)^2

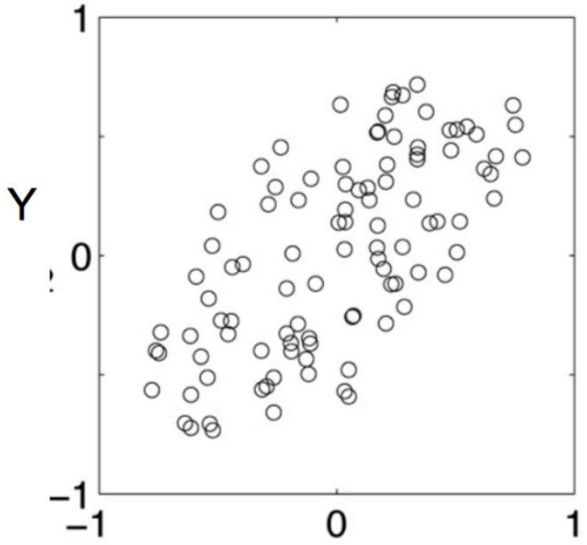
$$s^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)}$$

2D: Covariance: measures the correlation between X and Y

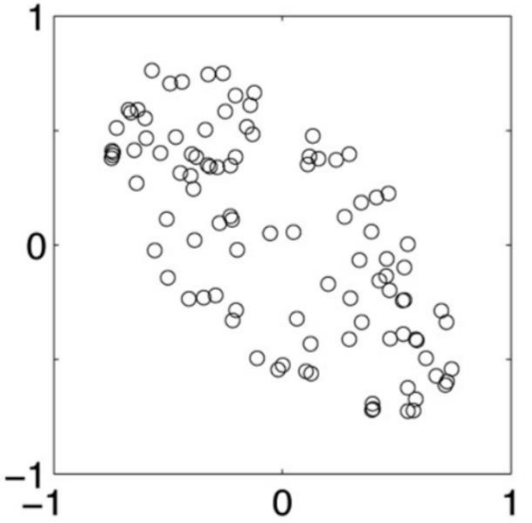
- Cov(X,Y)=0: independent
- Cov(X,Y)>0: move in the same direction
- Cov(X,Y)<0: move in opposite direction

$$\mathrm{cov}(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$

positive covariance



negative covariance



Multidimensional data: covariance matrix

- Contains covariance values between all possible dimensions (=attributes):

$$C^{nxn} = (c_{ij} \mid c_{ij} = \mathrm{cov}(Dim_i, Dim_j))$$

- Example for three attributes (x,y,z):

$$C = \begin{pmatrix} \mathrm{cov}(x,x) & \mathrm{cov}(x,y) & \mathrm{cov}(x,z) \\ \mathrm{cov}(y,x) & \mathrm{cov}(y,y) & \mathrm{cov}(y,z) \\ \mathrm{cov}(z,x) & \mathrm{cov}(z,y) & \mathrm{cov}(z,z) \end{pmatrix}$$

- Eigenvalues of covariance matrix contain information on the independent factors of variability
- Eigenvectors of covariance matrix provide the information on direcitons

Principal Component Analysis

- Center the data (subtract the mean $\boldsymbol{\mu} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n$ from each data point)
- Compute the $D \times D$ covariance matrix $\mathbf{S}$ using the centered data matrix $\mathbf{X}$ as

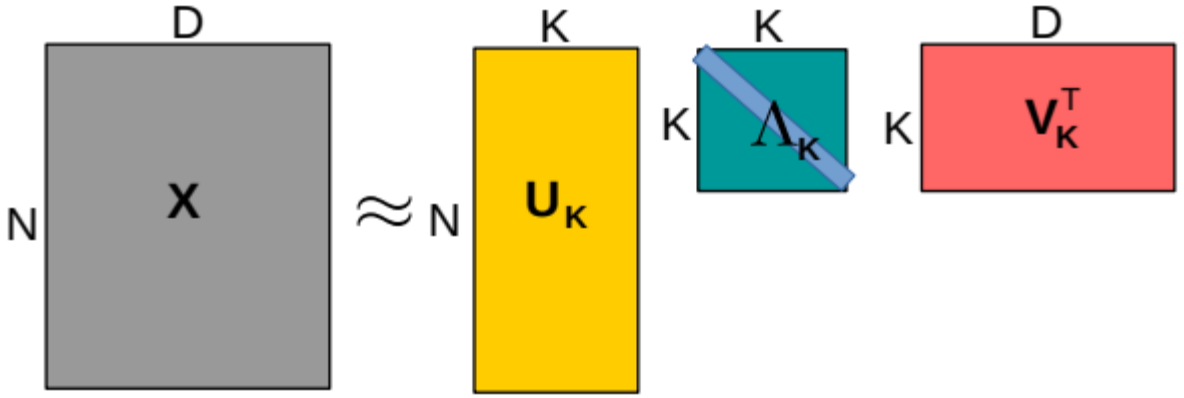
$$\mathbf{S} = \frac{1}{N}\mathbf{X}^\top\mathbf{X}$$ (Assuming $\mathbf{X}$ is arranged as $N \times D$)

- Do an eigen decomposition of the covariance matrix $\mathbf{S}$ (many methods exist)
- Take top $K < D$ leading eigvectors $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_K\}$ with eigen values $\{\lambda_1, \lambda_2, \ldots, \lambda_K\}$
- The K-dimensional projection/embedding of each input is $\boldsymbol{z}_n \approx \mathbf{W}_K^\top \boldsymbol{x}_n$
- Where $\mathbf{W}_\mathrm{K} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_K]$ is projection matrix of size $D \times K$

Adapted from Introduction to Machine Learning by Nisheeth

Singular Value Decomposition

- If we just use the top $K < \min\{N, D\}$ singular values, we get a rank-K SVD



$$\mathbf{X} \approx \hat{\mathbf{X}} = \sum_{k=1}^{K} \lambda_k \boldsymbol{u}_k \boldsymbol{v}_k^\top = \mathbf{U}_K \Lambda_K \mathbf{V}_K^\top$$

reconstruction error $\|\boldsymbol{X} - \hat{\boldsymbol{X}}\|$'

- Fact: SVD gives the best rank-K approximation of a matrix

- PCA is done by doing SVD on the covariance matrix **S** (left and right singular vectors are the sar and become eigenvectors, singular values become eigenvalues)

Adapted from Introduction to Machine Learning by Nisheeth

Principal Component Analysis

PCA: orthogonal transformation converting possibly correlated variables into linearly uncorrelated *principal components*

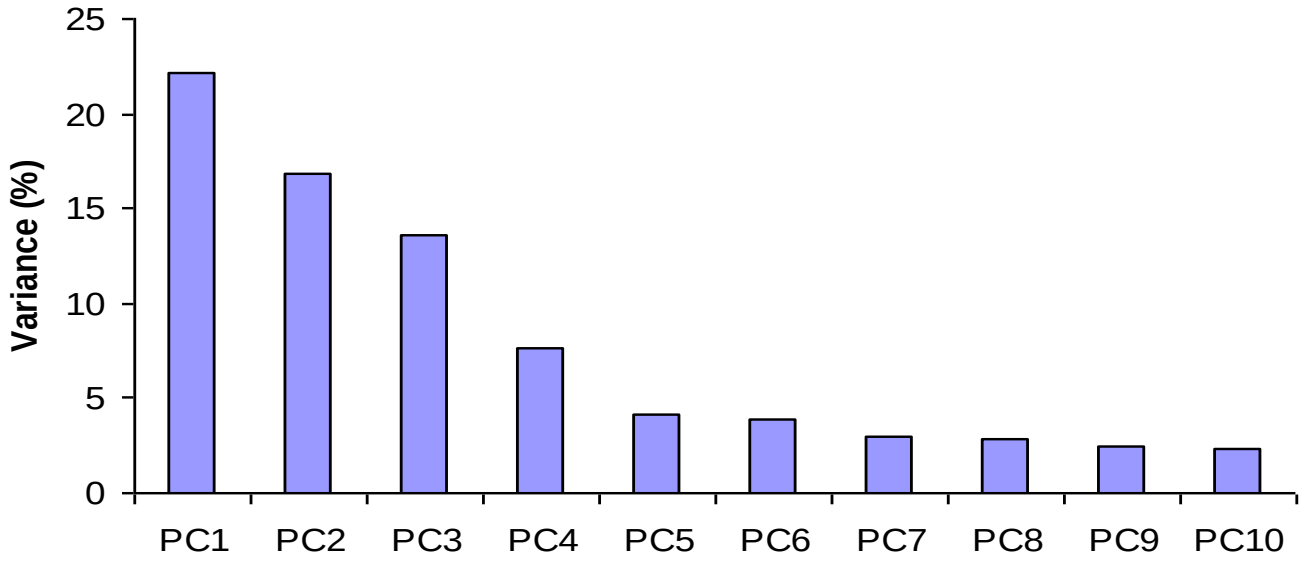
- PCA was invented by Karl Pearson in 1901, however the Singular Value Decomposition was independently derived some half a century earlier in Italy, Germany and France
- PCA transforms the data such that the greatest variance by any projection lies on the first coordinate
- Reveals internal structure of the data that best explains variance in the data set
- Since data often moves in clusters, PCA reveals those variables that drive the variance

Pearson, K. (1901). "On Lines and Planes of Closest Fit to Systems of Points is Space". *Philosophical Magazine Series 6* **2** (11): 559–572

PCA: Eigenvalues

Eigenvalues λ_j are used for calculation of [% of total variance] (V_j) for each component j:

$$V_j = 100 \cdot \frac{\lambda_j}{\sum_{x=1}^{n} \lambda_x} \qquad \sum_{x=1}^{n} \lambda_x = n$$



PCA: Components

- The first PC retains the greatest amount of variation in the sample
- The k^{th} PC retains the k^{th} greatest fraction of the variation in the sample
- The k^{th} largest eigenvalue of the correlation matrix C is the variance in the sample along the k^{th} PC
- The least-squares view: PCs are a series of linear least squares fits to a sample, each orthogonal to all previous ones

PCA Components and Loadings

- Technique useful for compression and classification of data
- Find new descriptors smaller than original variables
- Retain most of sample's information - correlation between original variables
- New descriptors are principal components (PCs)
- Loadings represent the “fraction” of PCs in initial data
- Uncorrelated, and ordered by fraction of total information retained in each PC

PCA in scikit-learn

sklearn.decomposition.PCA

class sklearn.decomposition.**PCA**(*n_components=None, *, copy=True, whiten=False, svd_solver='auto', tol=0.0, iterated_power='auto', n_oversamples=10, power_iteration_normalizer='auto', random_state=None*) [source]

Methods

fit(X[, y])	Fit the model with X.
fit_transform(X[, y])	Fit the model with X and apply the dimensionality reduction on X.
get_covariance()	Compute data covariance with the generative model.
get_feature_names_out([input_features])	Get output feature names for transformation.
get_metadata_routing()	Get metadata routing of this object.
get_params([deep])	Get parameters for this estimator.
get_precision()	Compute data precision matrix with the generative model.
inverse_transform(X)	Transform data back to its original space.
score(X[, y])	Return the average log-likelihood of all samples.
score_samples(X)	Return the log-likelihood of each sample.
set_output(*[, transform])	Set output container.
set_params(**params)	Set the parameters of this estimator.
transform(X)	Apply dimensionality reduction to X.

Examples: Eigenfaces

- When viewed as vectors of pixel values, face images are extremely high dimensional. Image of 100x100 pixels has 10,000 dimensions.
- However, very few of 100x100 vectors are valid face images
- We want to effectively represent the subspace of face images

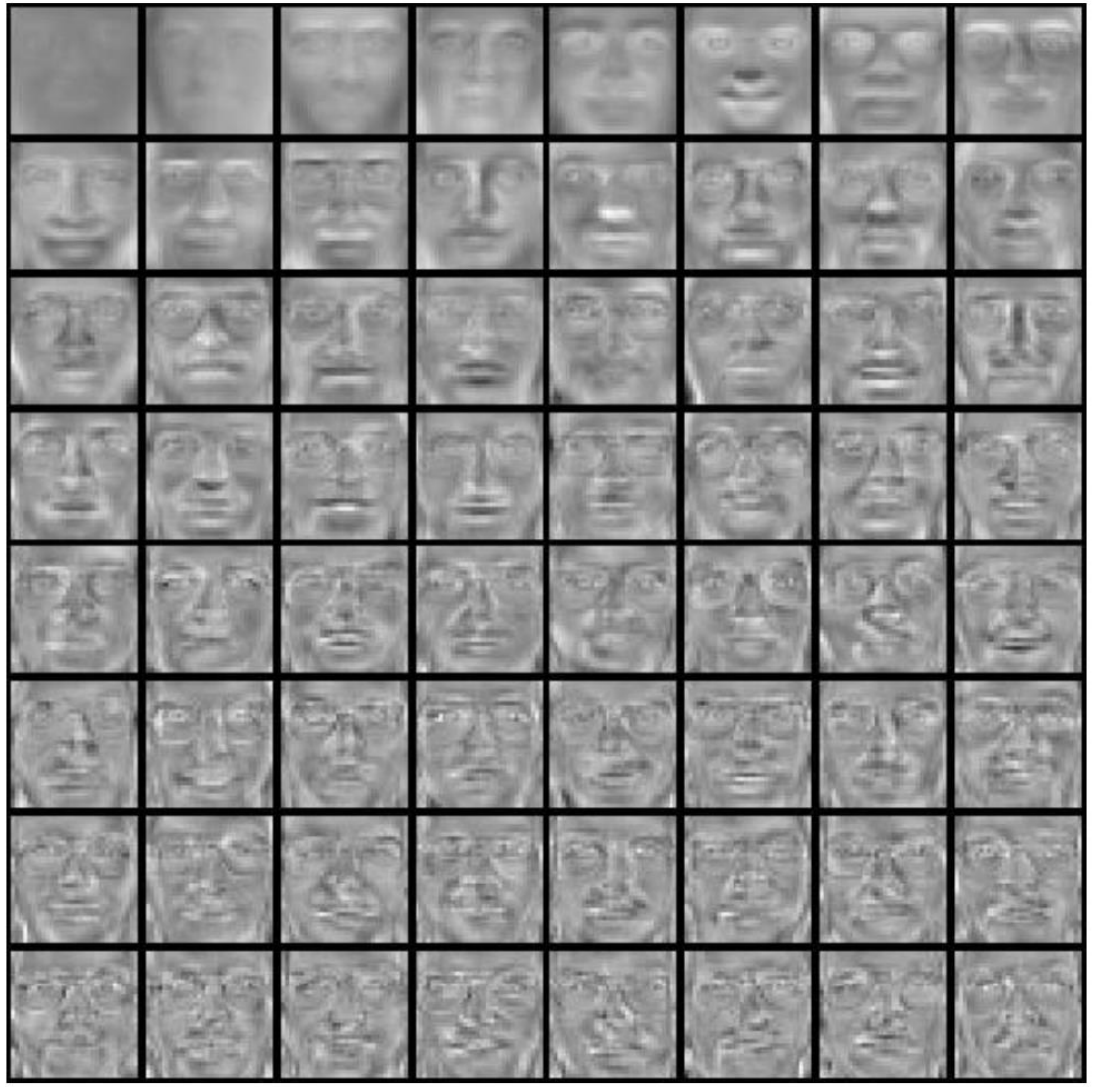
Adapted form Fereshteh Sadeghi

slide by Derek Hoiem

Examples: Eigenfaces

Mean: μ

Top eigenvectors: $u_1,\ldots u_k$



Adapted from Fereshteh Sadeghi slide by Derek Hoiem

Representation and reconstruction

- Face **x** in "face space" coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)]$$

$$= \quad w_1, \ldots, w_k$$

- Reconstruction:

$$\hat{x} = \mu + w_1 u_1 + w_2 u_2 + w_3 u_3 + w_4 u_4 + \ldots$$

slide by Derek Hoiem

Adapted from Fereshteh Sadeghi

Reconstruction

P = 4

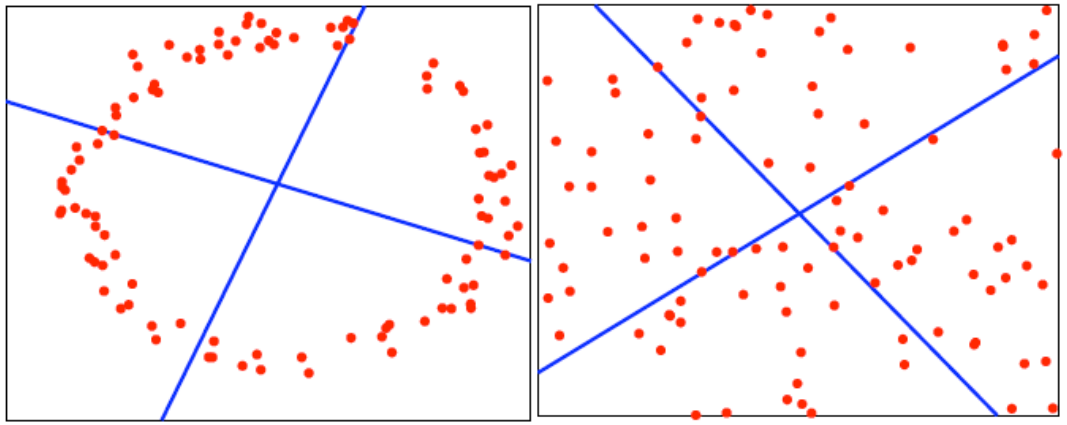
P = 200

P = 400

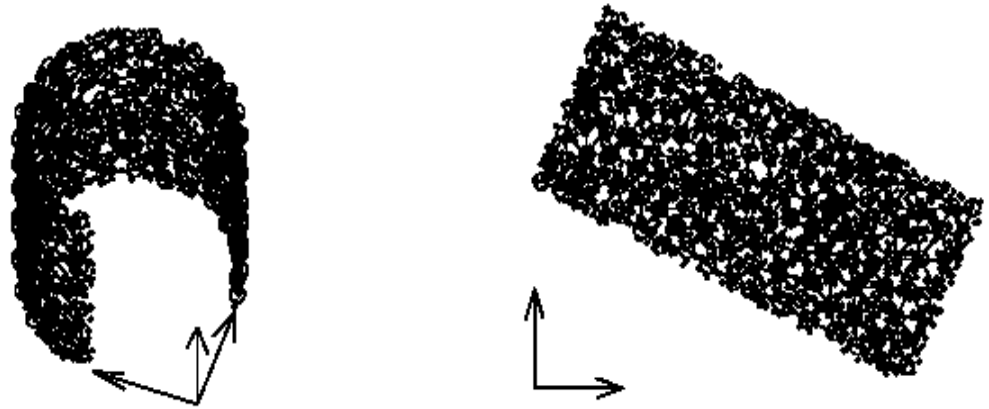
We can represent faces well by 400 components – rather then 10,000!

Adapted from Fereshteh Sadeghi

Limitations of PCA



PCA will make no difference between these examples



Non linear projection of a horseshoe

Non-linear PCA

- Suppose that instead of using the points $\mathbf{x}_i$ as is, we wanted to go to some different *feature space* $\phi(\mathbf{x}_i) \in \Re^N$
- E.g. using polar coordinates instead of cartesian coordinates would help us deal with the circle
- In the higher dimensional space, we can then do PCA
- The result will be non-linear in the original data space!
- Similar idea to support vector machines

Kernel PCA

Kernel PCA is an unsupervised manifold learning technique that maps data points to a generally lower-dimensional space. It generalizes the Principal Components Analysis approach to non-linear transformations using the kernel trick (Schölkopf, Smola and Müller, 1996; Schölkopf, Smola and Müller, 1998; Schölkopf, Burges and Smola, 1999). The algorithm implicitly finds the leading eigenvectors and eigenvalues of the covariance of the projection $\phi(x)$ of the data in "feature space", where $\phi(x)$ is such that the kernel $K_n(x, y) = \phi(x) \cdot \phi(y)$ (i.e. K_n must not have negative eigenvalues). If the data is

(http://research.microsoft.com/users/Cambridge/nicolasl/papers/eigen_dimred.pdf)

PCA in scikit-learn

sklearn.decomposition.PCA

class sklearn.decomposition.**PCA**(*n_components=None, *, copy=True, whiten=False, svd_solver='auto', tol=0.0, iterated_power='auto', n_oversamples=10, power_iteration_normalizer='auto', random_state=None*) [source]

Methods

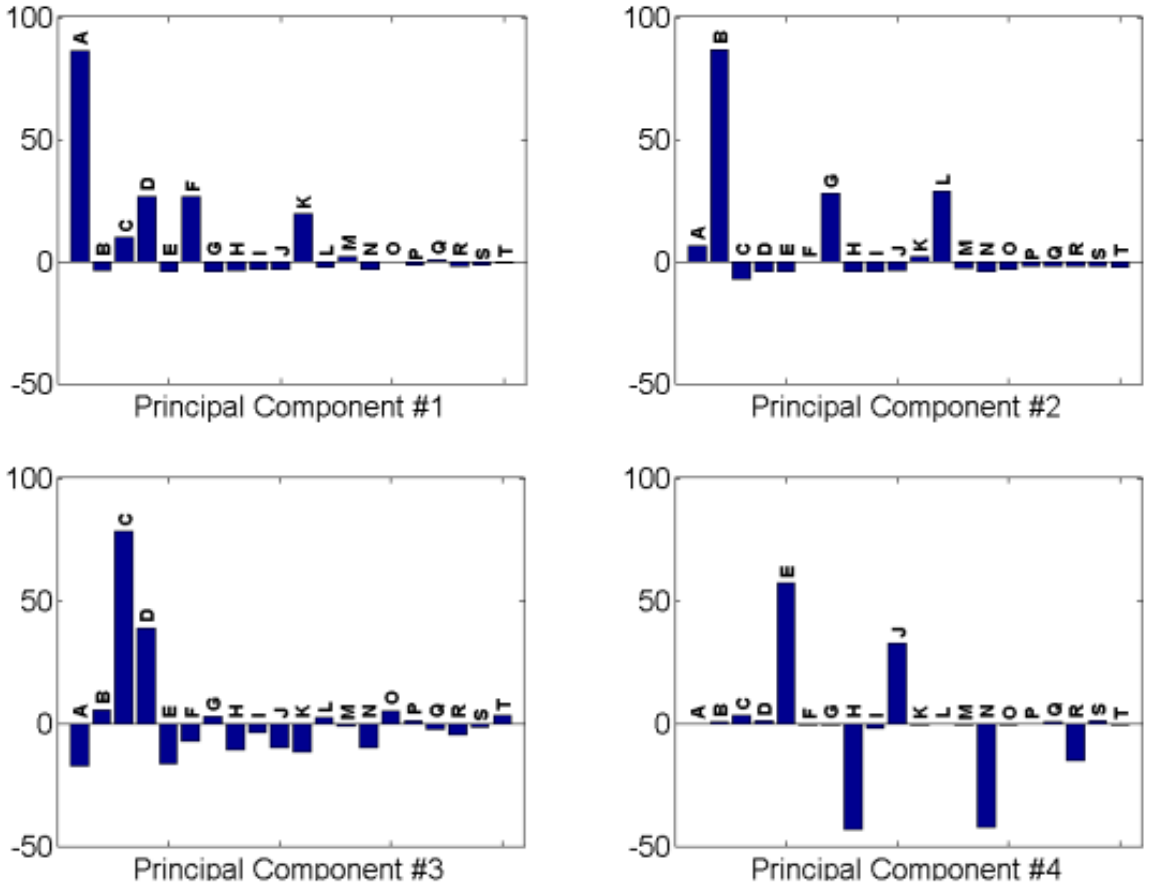
fit(X[, y])	Fit the model with X.
fit_transform(X[, y])	Fit the model with X and apply the dimensionality reduction on X.
get_covariance()	Compute data covariance with the generative model.
get_feature_names_out([input_features])	Get output feature names for transformation.
get_metadata_routing()	Get metadata routing of this object.
get_params([deep])	Get parameters for this estimator.
get_precision()	Compute data precision matrix with the generative model.
inverse_transform(X)	Transform data back to its original space.
score(X[, y])	Return the average log-likelihood of all samples.
score_samples(X)	Return the log-likelihood of each sample.
set_output(*[, transform])	Set output container.
set_params(**params)	Set the parameters of this estimator.
transform(X)	Apply dimensionality reduction to X.

- In scikit-learn, PCA is implemented as **probabilistic model**: can estimate the likelihood of data based on the amount of variance it explains.
- **Incremental PCA:** add partial_fit method
- **SparsePCA:** introduce sparsity in decomposition

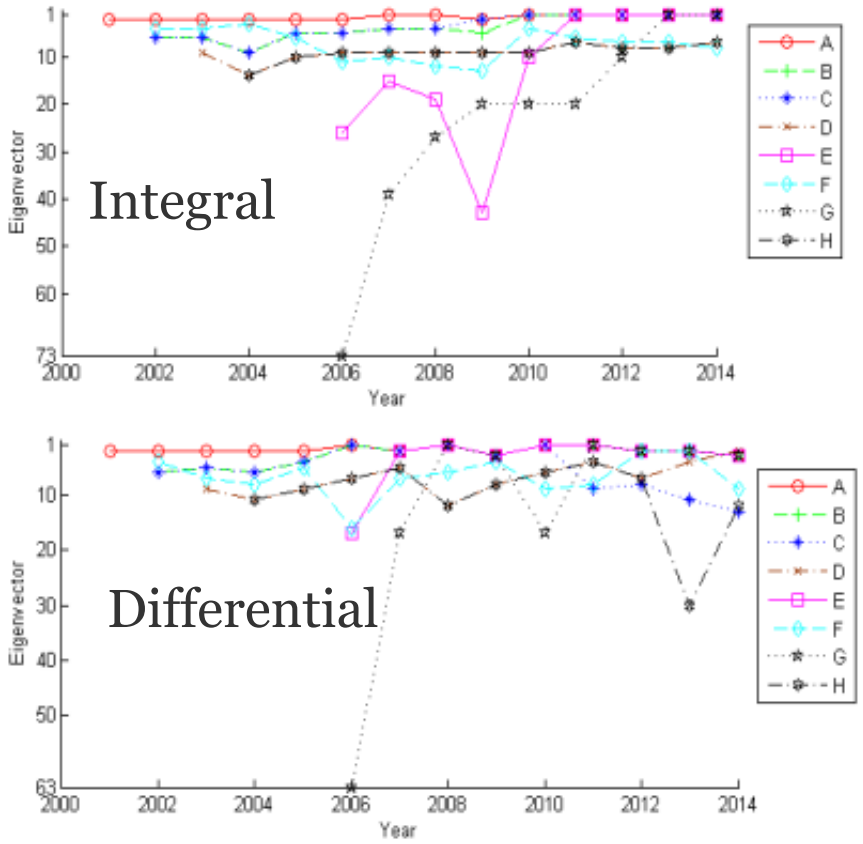
$$(U^*, V^*) = \underset{U,V}{\arg\min} \frac{1}{2}||X - UV||_{\text{Fro}}^2 + \alpha||V||_{1,1}$$
$$\text{subject to } ||U_k||_2 <= 1 \text{ for all } 0 \leq k < n_{components}$$

- … and there is always more (dictionary methods, etc)

PCA can be applied to everything



Publication activity



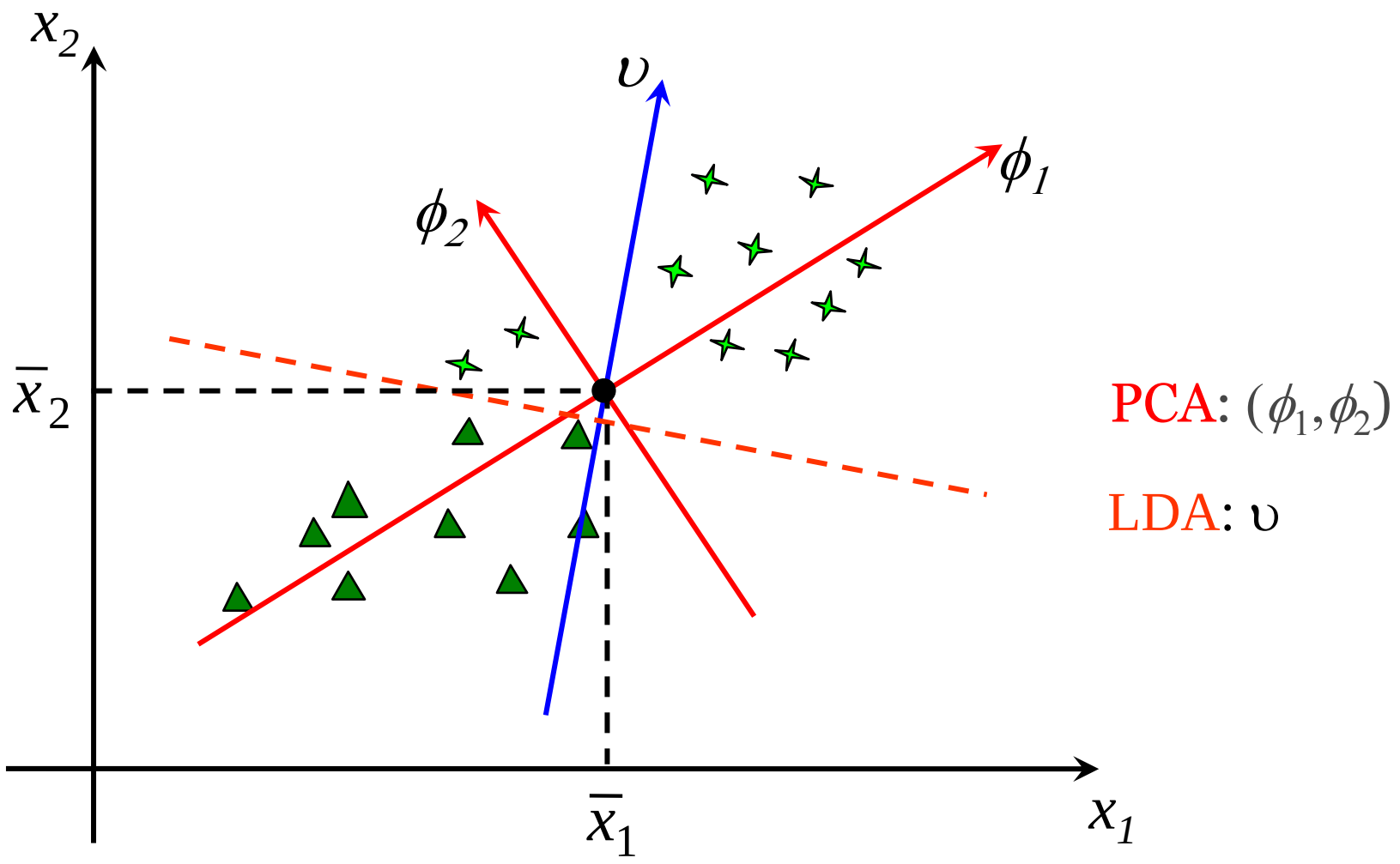
Publication analysis: The dimensionality of space, N, is defined by the total number of scientists. Each publication in this case then defines a single point in this space. For example, for the 28-dimensional space of authors of {A,B,C, … , Z} the publication authored by A and C will be represented as {1,0,1, …, 0}, and by A,B, and Z as {1,1,0, …, 0,1}, where all missing elements are zeroes.

https://arxiv.org/abs/1502.03439

Linear Discriminant Analysis

- Linear Discriminant Analysis, or simply LDA, is a feature extraction technique that has been used successfully in many statistical pattern recognition problems.
- LDA is often called Fisher Discriminant Analysis (FDA).
- The primary purpose of LDA is to separate samples of distinct groups by transforming then to a space which maximises their between-class separability while minimising their within-class variability.
- It assumes implicitly that the true covariance matrices of each class are equal because the same within-class scatter matrix is used for all the classes considered.
- If we remove this assumption, we get Quadratic Discriminant Analysis

Geometric Idea of LDA



From Intelligent Data Analysis and Probabilistic Inference by Longin Jan Latecki Temple University

LDA Steps

1. Compute the d-dimensional mean vectors.
2. Compute the scatter matrices
3. Compute the eigenvectors and corresponding eigenvalues for the scatter matrices.
4. Sort the eigenvalues and choose those with the largest eigenvalues to form a $d \times k$ dimensional matrix
5. Transform the samples onto the new subspace.

LDA Method

- Let the between-class scatter matrix S_b be defined as

$$S_b = \sum_{i=1}^{g} N_i (\bar{x}_i - \bar{x})(\bar{x}_i - \bar{x})^T$$

- and the within-class scatter matrix S_w be defined as

$$S_w = \sum_{i=1}^{g} (N_i - 1) S_i = \sum_{i=1}^{g} \sum_{j=1}^{N_i} (x_{i,j} - \bar{x}_i)(x_{i,j} - \bar{x}_i)^T$$

- where $x_{i,j}$ is an n-dimensional data point j from class p_i, N_i is the number of training examples from class p_i, and g is the total number of classes or groups.

From Intelligent Data Analysis and Probabilistic Inference by Longin Jan Latecki Temple University

LDA Method

- The sample mean, sample covariance, and grand mean vector are given respectively by:

$$\bar{x}_i = \frac{1}{N_i}\sum_{j=1}^{N_i} x_{i,j}$$

$$S_i = \frac{1}{(N_i - 1)}\sum_{j=1}^{N_i}(x_{i,j} - \bar{x}_i)(x_{i,j} - \bar{x}_i)^T$$

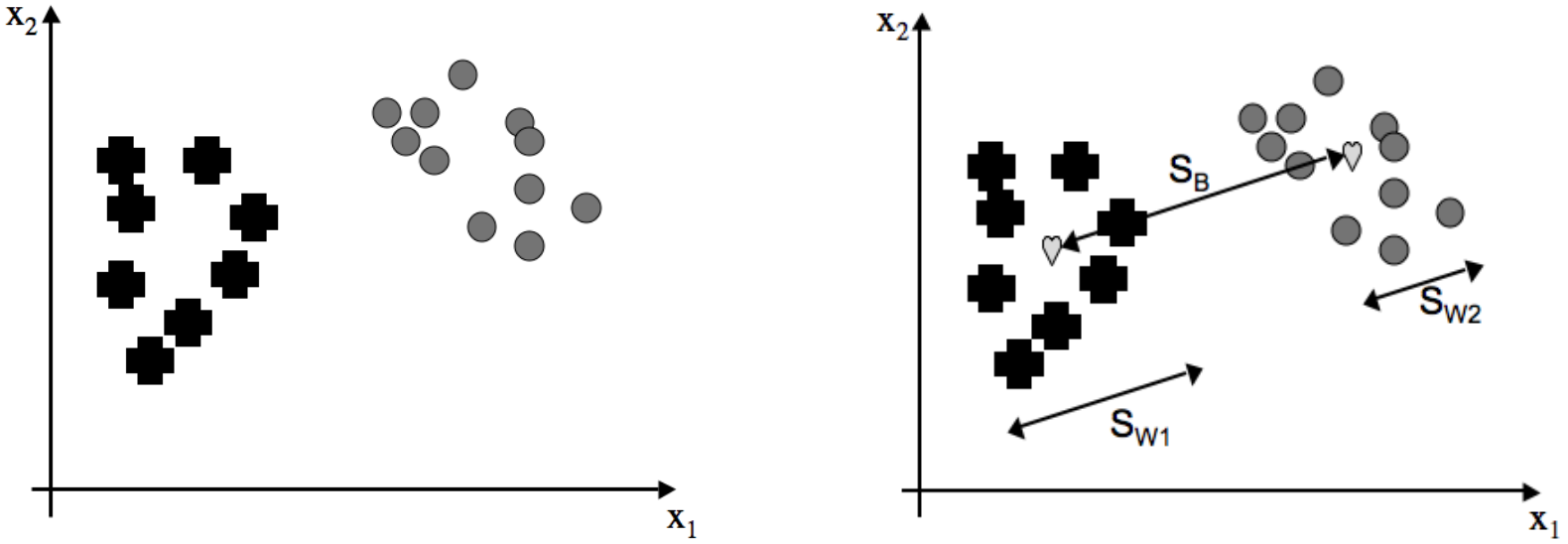
$$\bar{x} = \frac{1}{N}\sum_{i=1}^{g} N_i \bar{x}_i = \frac{1}{N}\sum_{i=1}^{g}\sum_{j=1}^{N_i} x_{i,j} \qquad N = N_1 + N_2 + \ldots + N_g$$

From Intelligent Data Analysis and Probabilistic Inference by Longin Jan Latecki Temple University

LDA Method

The objective of LDA is to find a projection matrix P_{lda} that maximises the ratio of the determinant of S_b to the determinant of S_w (Fisher's criterion) as:

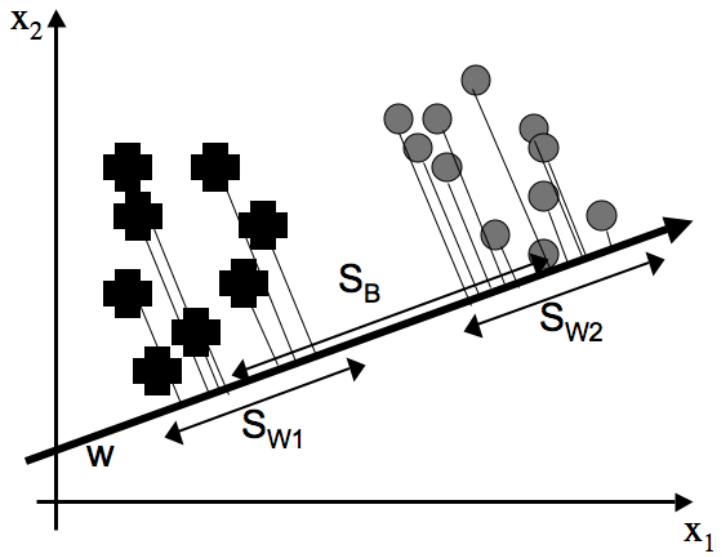
$$P_{lda} = \arg\max_{P} \frac{\left| P^T S_b P \right|}{\left| P^T S_w P \right|}$$



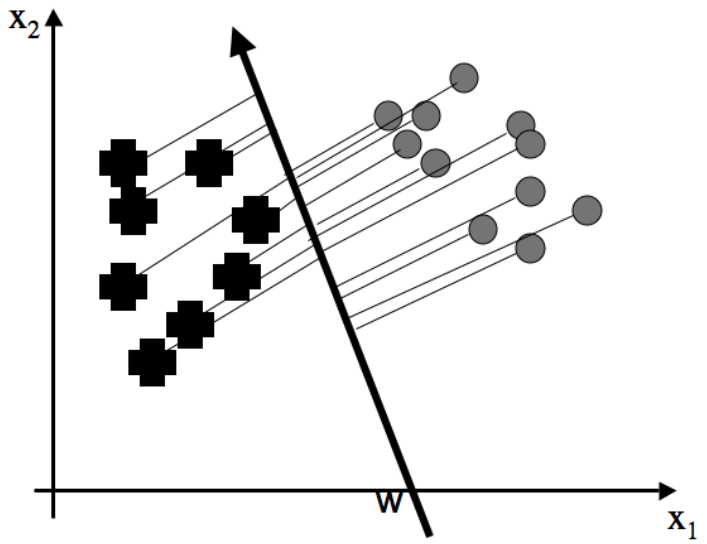
From Intelligent Data Analysis and Probabilistic Inference by Longin Jan Latecki Temple University

LDA Method

- So Fisher's criterion tries to find the projection that:
  - Maximises the variance of the class means
  - Minimises the variance of the individual classes



Good projection



Bad projection

From Intelligent Data Analysis and Probabilistic Inference by Longin Jan Latecki Temple University

Classification using LDA

- The LDA is an axis projection.
- Once the projection is found all the data points can be transformed to the new axis system along with the class means and covariances.
- Allocation of a new point to a class can be done using a distance measure such as the Mahalanobis distance.

Given a vector $\mathbf{x}$, a mean vector μ, and a covariance matrix $\mathbf{\Sigma}$, the Mahalanobis Distance D_M is defined as:

$$D_M(\mathbf{x}, \mu) = \sqrt{(\mathbf{x} - \mu)^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \mu)}$$

where:

- $\mathbf{x}$ is the vector for which the Mahalanobis Distance is being computed.
- μ is the mean vector of the distribution.
- $\mathbf{\Sigma}$ is the covariance matrix of the distribution.
- ^{T} denotes a matrix transpose.
- $\mathbf{\Sigma}^{-1}$ is the inverse of the covariance matrix.

From Intelligent Data Analysis and Probabilistic Inference by Longin Jan Latecki Temple University

LDA vs. PCA

- LDA is supervised ML method, PCA is unsupervised method
- LDA seeks directions that are efficient for *discriminating* data whereas PCA seeks directions that are efficient for *representing* data.
- The directions that are discarded by PCA might be exactly the directions that are necessary for distinguishing between groups.

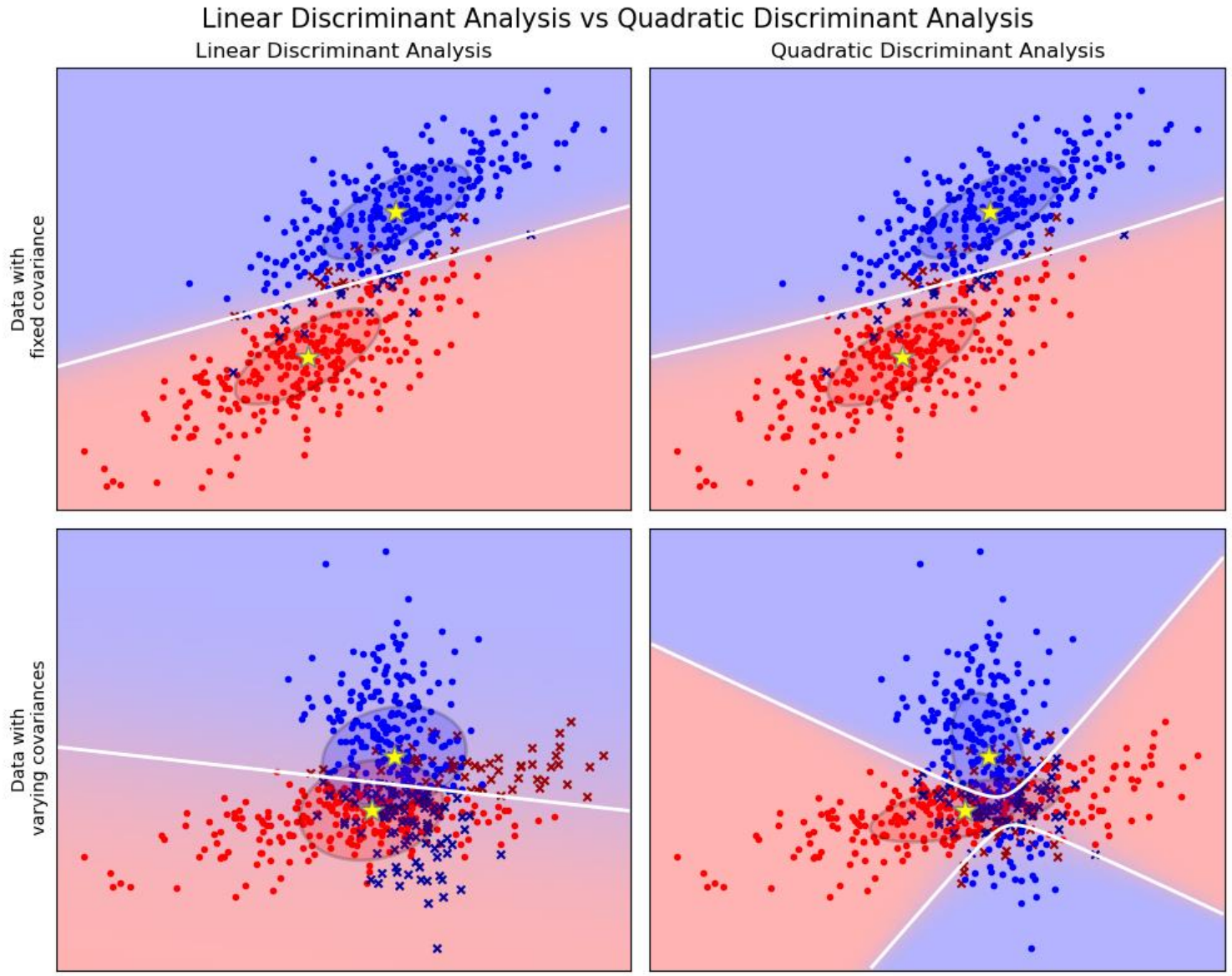
From Intelligent Data Analysis and Probabilistic Inference by Longin Jan Latecki Temple University

sklearn.discriminant_analysis.LinearDiscriminantAnalysis

class sklearn.discriminant_analysis.**LinearDiscriminantAnalysis**(*solver='svd', shrinkage=None, priors=None, n_components=None, store_covariance=False, tol=0.0001, covariance_estimator=None*) [source]

decision_function(X)	Apply decision function to an array of samples.
fit(X, y)	Fit the Linear Discriminant Analysis model.
fit_transform(X[, y])	Fit to data, then transform it.
get_feature_names_out([input_features])	Get output feature names for transformation.
get_metadata_routing()	Get metadata routing of this object.
get_params([deep])	Get parameters for this estimator.
predict(X)	Predict class labels for samples in X.
predict_log_proba(X)	Estimate log probability.
predict_proba(X)	Estimate probability.
score(X, y[, sample_weight])	Return the mean accuracy on the given test data and labels.
set_output(*[, transform])	Set output container.
set_params(**params)	Set the parameters of this estimator.
set_score_request(*[, sample_weight])	Request metadata passed to the `score` method.
transform(X)	Project data to maximize class separation.

Quadratic Discriminant Analysis



https://scikit-learn.org/stable/modules/lda_qda.html#lda-qda

LDA vs. QDA

Bayesian classification framework:

$$P(y=k|x) = \frac{P(x|y=k)P(y=k)}{P(x)} = \frac{P(x|y=k)P(y=k)}{\sum_l P(x|y=l) \cdot P(y=l)}$$

QDA approximation:

$$P(x|y=k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_k)^t \Sigma_k^{-1}(x-\mu_k)\right)$$

LDA is a special case of QDA, where the Gaussians for each class are assumed to share the same covariance matrix: $\Sigma_k = \Sigma$. If so

$$\log P(y=k|x) = \omega_k^t x + \omega_{k0} + Cst.$$

where $\omega_k = \Sigma^{-1}\mu_k$ and $\omega_{k0} = -\frac{1}{2}\mu_k^t \Sigma^{-1}\mu_k + \log P(y=k)$. These quantities correspond to the `coef_` and `intercept_` attributes, respectively.

Note: Relation with Gaussian Naive Bayes

If in the QDA model one assumes that the covariance matrices are diagonal, then the inputs are assumed to be conditionally independent in each class, and the resulting classifier is equivalent to the Gaussian Naive Bayes classifier `naive_bayes.GaussianNB`.

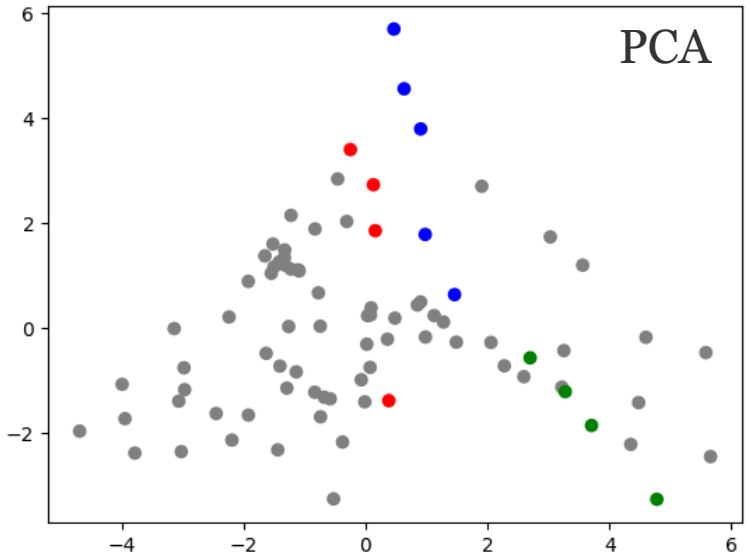
https://scikit-learn.org/stable/modules/lda_qda.html#lda-qda

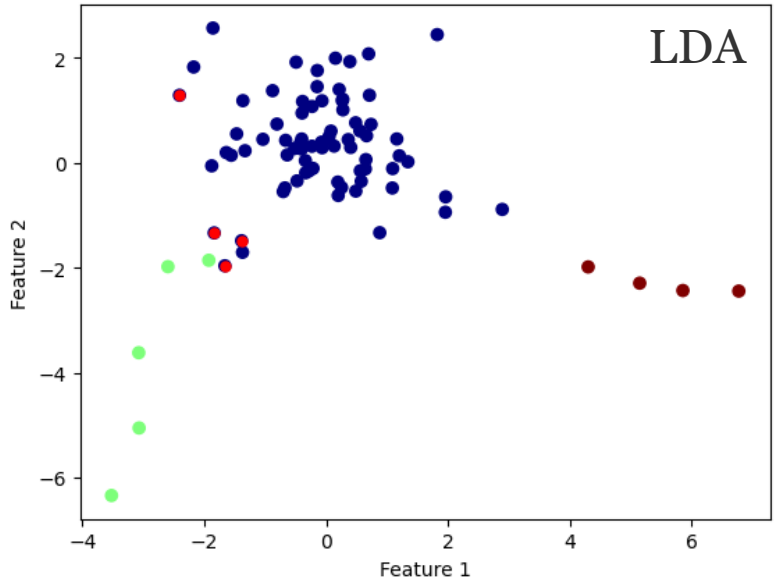
PCA vs. LDA for elements

Alkali

Alkali earth

Halogens

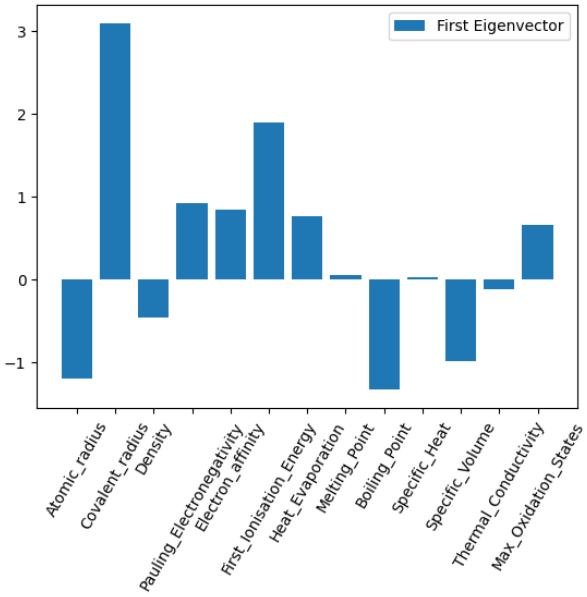




Alkali

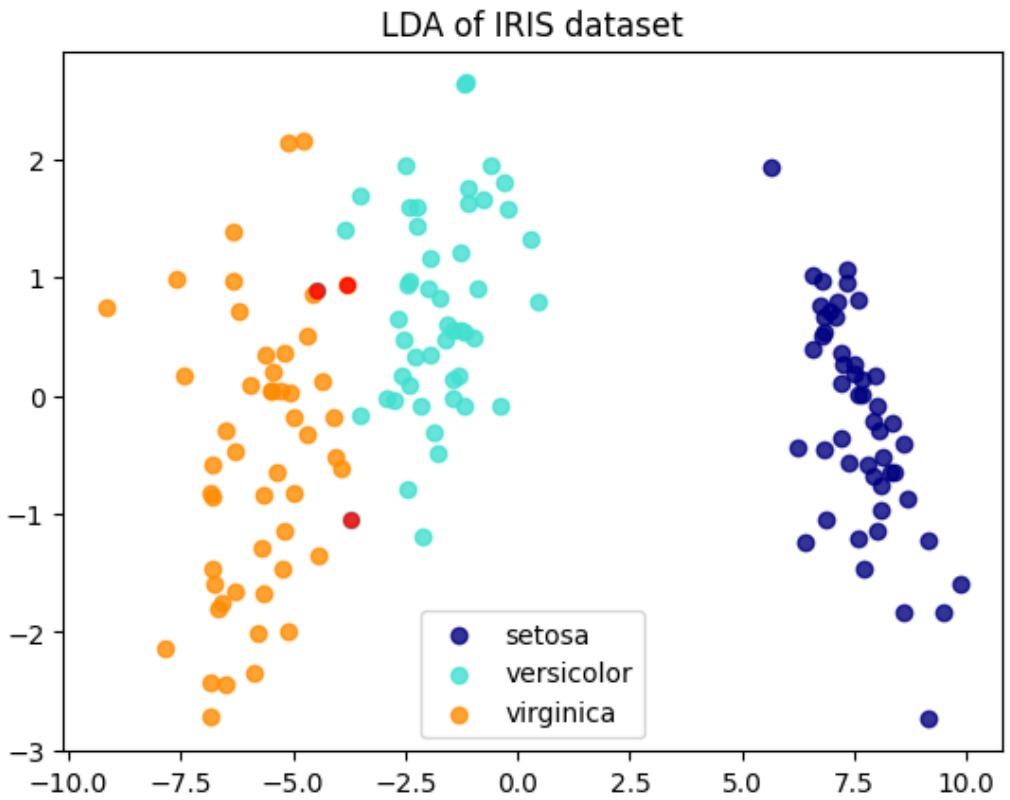
Alkali earth

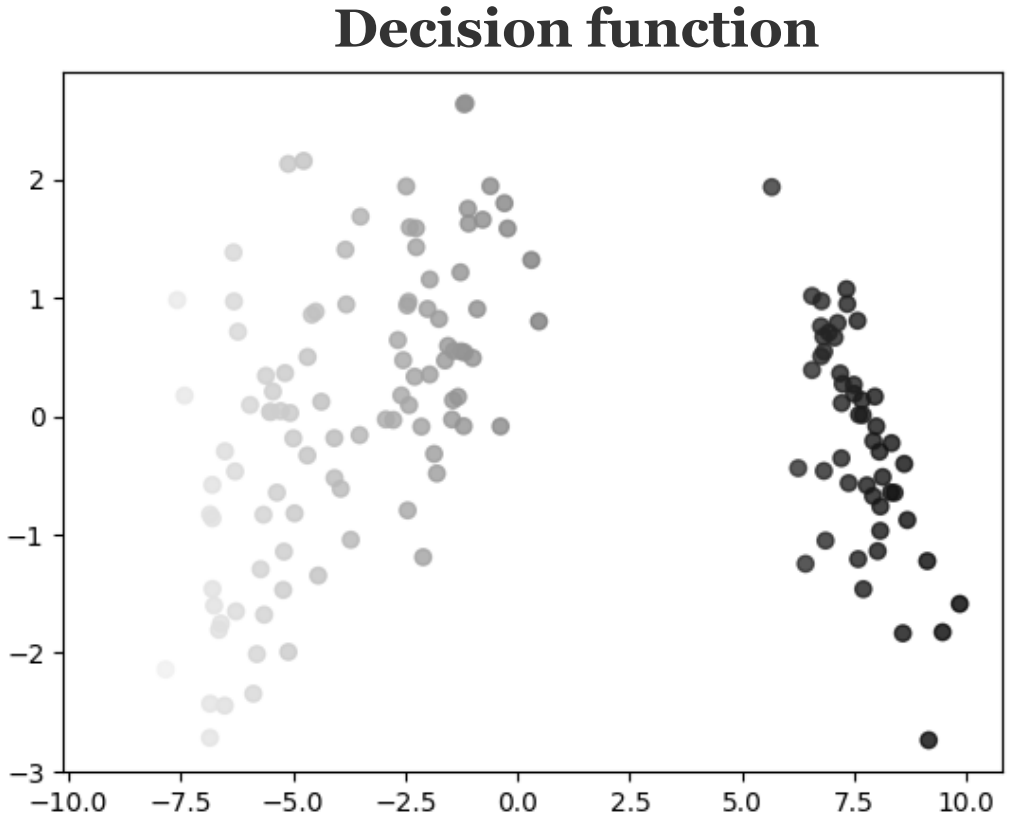
Halogens



- Our element space is described by 13 descriptors
- In PCA, we found 2 linear combinations of these descriptors that describe this data set best.
- Alkali, alkali-earth, and halogens are close to each other in PCA space
- LDA finds best representation to separate alkali and halogens from everything else
- In LDA representation, alkali earth are close to alkali

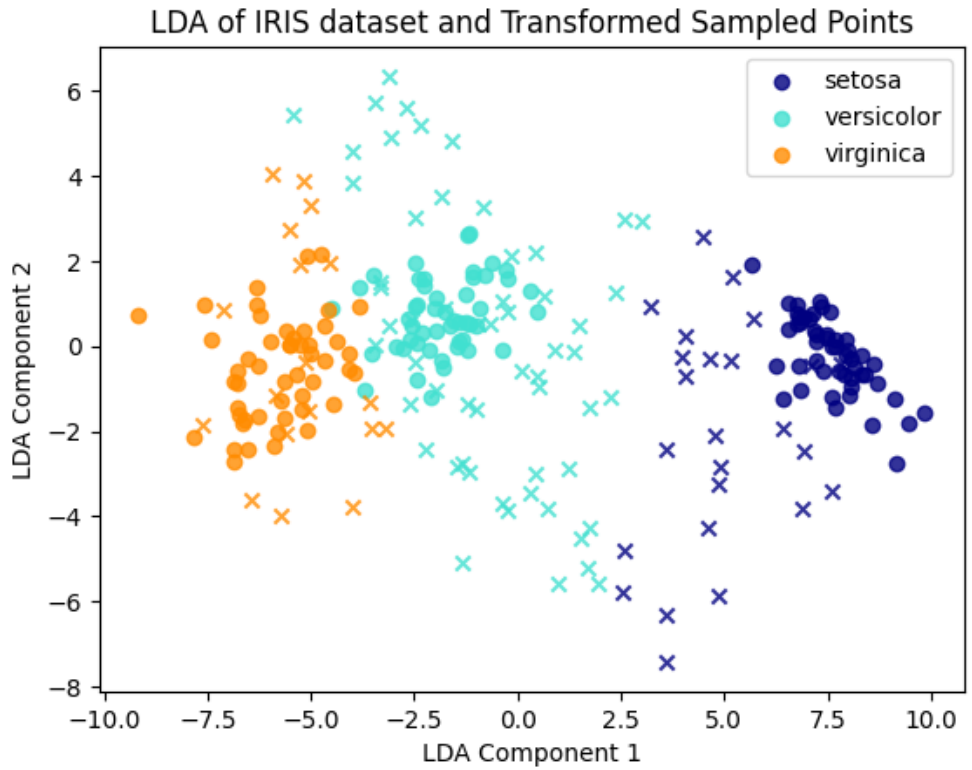
What else can LDA give us?

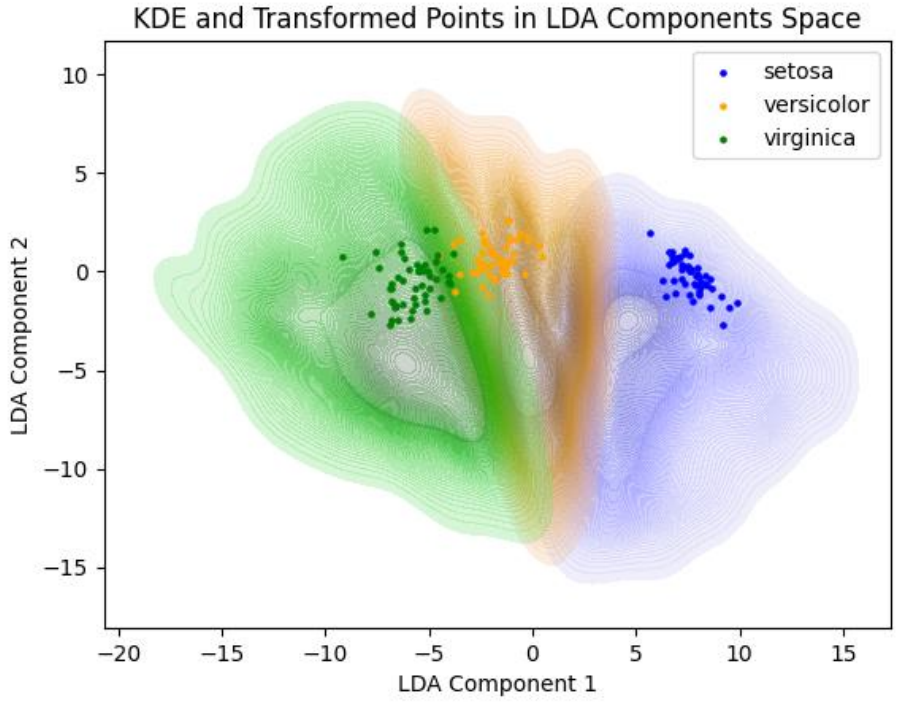




- Decision function allows us to quantify how likely is the feature to belong to certain class

Visualizing the decision surfaces





- Generate multiple points uniformly distributed in the original high-dimensional space
- Perform the LDA transform
- Calculate the KDE

Why these methods are useful?

Pro:

1.Computational Efficiency
2. Interpretability
3. Less Data Requirement
4. Applicability to Linear Problems
5. Ease of Implementation and Use
6. Feature Reduction and Visualization
7. Foundation for More Complex Models

Con:

1. Assumption of Linearity
2. Assumption of Gaussian distributions
3. Limited Complexity
4. Independence Assumption in PCA and CCA
5. Vulnerability to Outliers
6. Feature Importance Ambiguity

Linear methods in Deep Learning Era

1. Preprocessing: To reduce the dimensionality of data or extract more informative features before feeding them into a neural network.

2. Visualization: To visualize high-dimensional embeddings learned by neural networks.

3. Post-Processing of Embeddings: To further process or analyze embeddings learned by a neural network for various tasks. Specifically, **CCA** can be used to measure the similarity of learned representations from different neural networks or layers, often used in analyzing and comparing embeddings.

4. Improving Model Robustness and Generalization: To enhance the robustness and generalization capabilities of deep learning models: feature extraction or transformation, potentially enhancing the robustness and generalization of neural network models.

5. Facilitating Transfer Learning: To adapt learned representations for transfer learning applications, e.g. transform embeddings for alignment, compatibility, or adaptation

6. Interpreting Neural Network Decisions: To interpret and understand the decisions or representations learned by neural networks, e.g. analyze and interpret the feature representations or activations within a neural network, contributing to model interpretability and transparency.

Extra slides – examples of the dimensionality reduction and clustering for spectroscopic and imaging data

Spectroscopic Imaging

Scanning probe microscopy:

- Force-distance curve measurements
- Current-voltage measurements
- Piezoresponse force/electrochemical strain spectroscopy

Electron microscopy:

- Electron Energy Loss Spectroscopy

Optical microscopy:

- Hyperspectral imaging
- Time resolved measurements

Mass-spectrometry:

- Secondary ion MS imaging

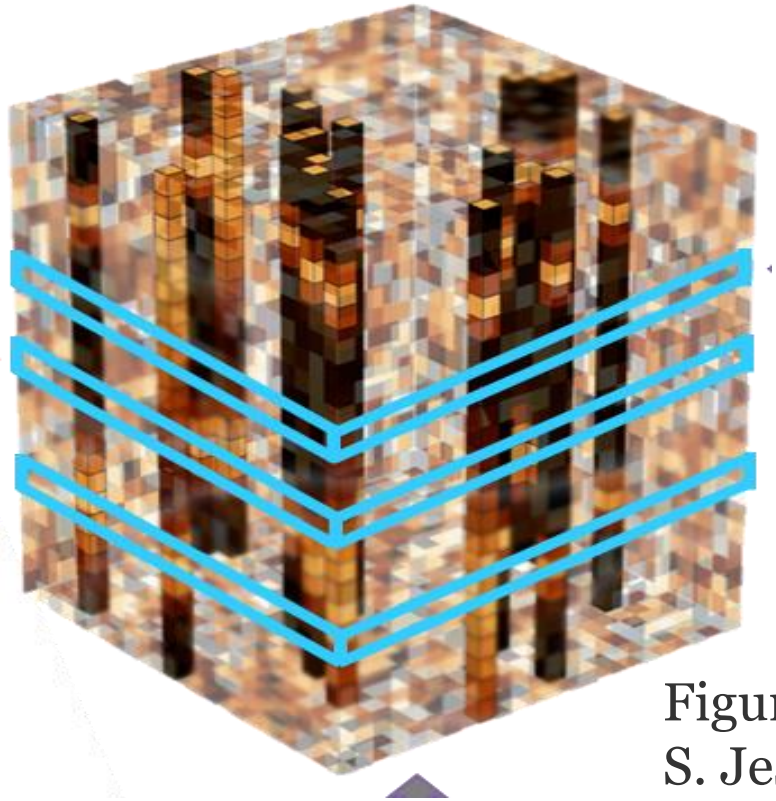
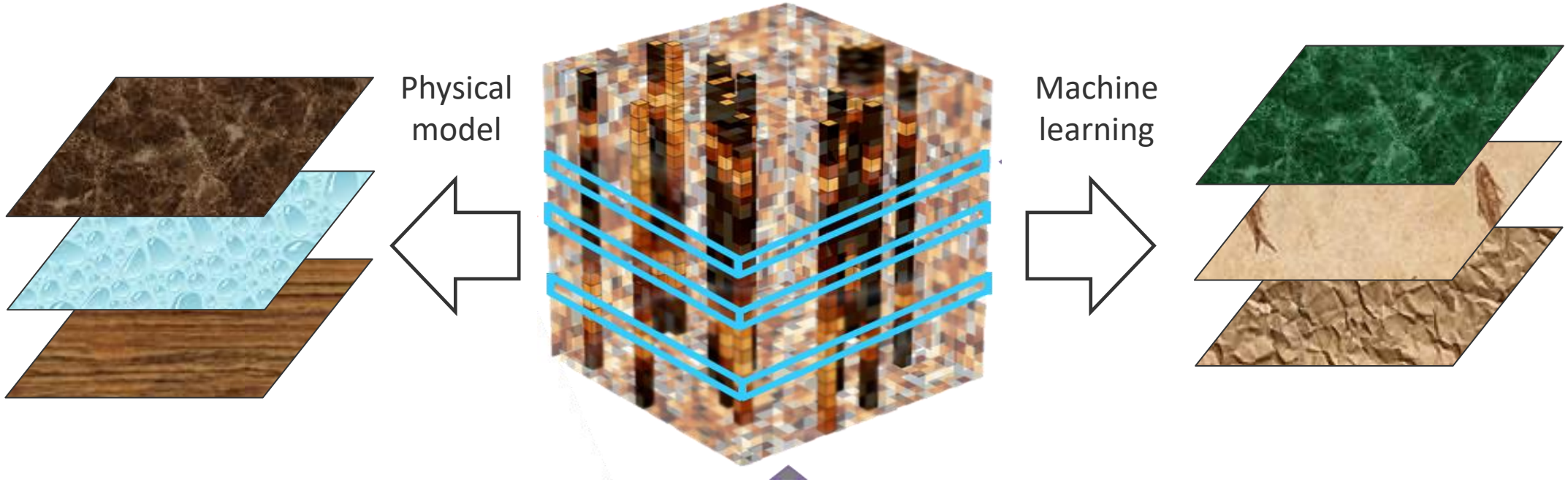


Figure by S. Jesse

In many cases, measured signal can be represented or approximated as a linear combination of signals. However, their functional forms are generally unknown

Very important: convolution with resolution function is also mixing

Physics vs. ML based analysis



- If we have physical model, we can extract relevant parameters from data
- Imperfect model: epistemic uncertainty
- Noisy data: aleatoric uncertainty
- Analysis results do not depend on sampling of data in x,y

- If we don't have physical model, we can learn intrinsic structure of data
- **Unsupervised learning:** based on data only
  - But not really (definition of distance)
  - Analysis can depend on sampling of data
- **Supervised learning:** based on prior examples
  - Out of distribution shifts

Physics-informed ML: Combines strengths (and limitations) of both

General linear unmixing

$$S(\mathbf{x}, \mathbf{R}) = \sum_i a_i(\mathbf{x})\, w_i(\mathbf{R}) + \mathrm{N}$$

We start with:

- **x** is the spatial variable, x = (x,y)
- R is the (vector) parameter variable

Overall, for MxM image and P point in spectra, we have M^2P data points

We aim to get:

- $a_i(\mathbf{x})$ are loading maps
- $w_i(\mathbf{R})$ are endmembers/eigenvectors
- N is noise

Overall, we can have (maximum) P loading maps of M^2 size. However, not all of them will have useful information

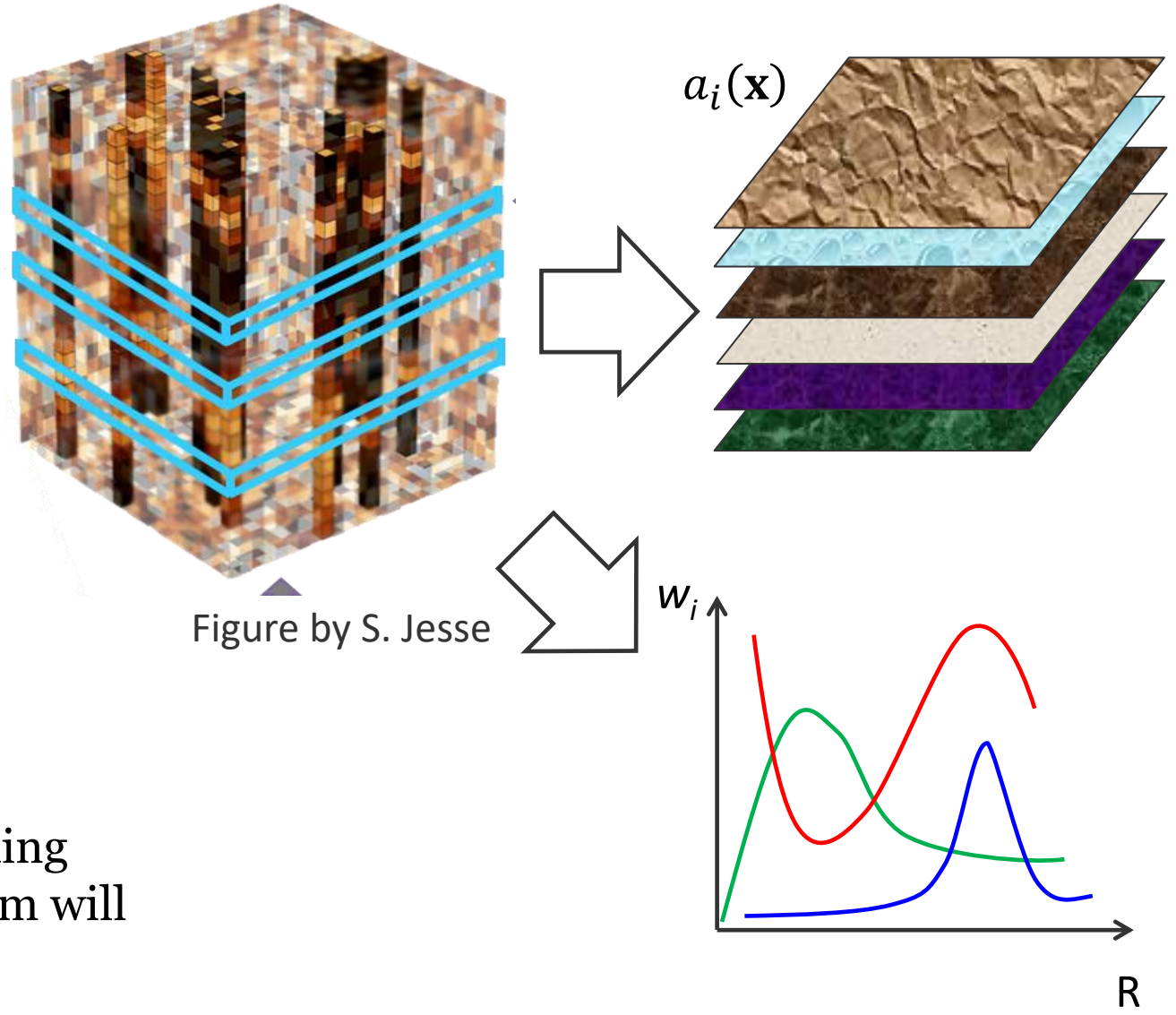
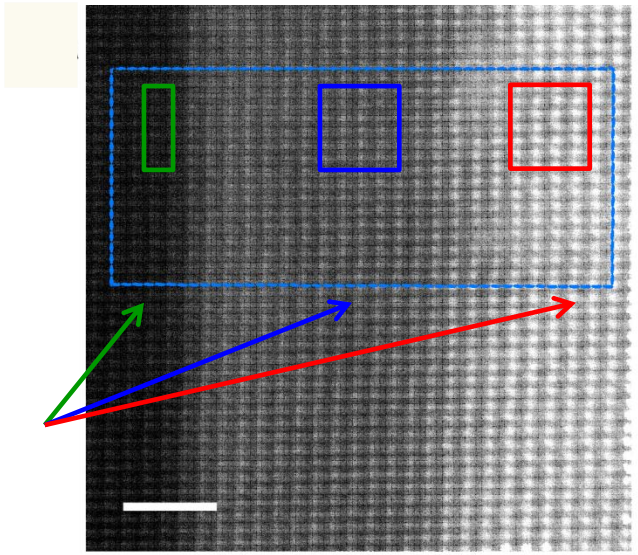


Figure by S. Jesse

Multiple Linear Regression

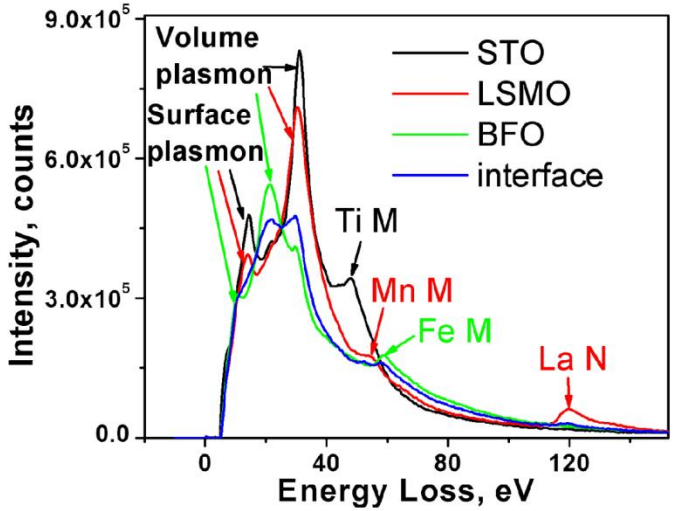
Linear mixing $S(\mathbf{x}, \mathbf{R}) = \sum_i a_i(\mathbf{x}) w_i(\mathbf{R}) + N$ but $w_i(\mathbf{R})$ are **known**

STEM of STO/LSMO/BFO interface



Standard spectra

Low-loss EELS spectra of three components

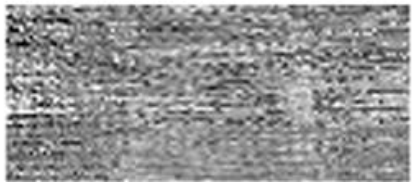


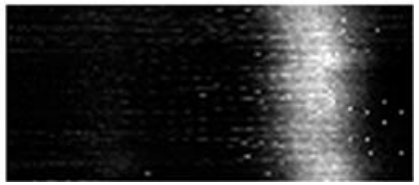
A.Y. Borisevich et al, *Suppression of Octahedral Tilts and Associated Changes in Electronic Properties at Epitaxial Oxide Heterostructure Interfaces*, Phys. Rev. Lett. **105**, 087204 (2010).

Fit coefficient map **residuals map** **χ2 map**

"Chemistry": 35 to 125 eV

"Plasmons" 5 to 35 eV





Principal Component Analysis

$$S(\mathbf{x}, \mathbf{R}) = \sum_i a_i(\mathbf{x})\, w_i(\mathbf{R})$$

- In PCA, the eigenvectors $w_i(\mathbf{R})$ are orthonormal and are arranged such that corresponding eigenvalues are placed in descending order by variance
- Can be used to separate “real data” from “noise” – but needs cut-off/selection criteria
- PCA eigenvectors generally do not have defined physical meaning
- PCA is a starting point for many other unmixing methods

Ultramicroscopy 34 (1990) 165–178
North-Holland

EELS elemental mapping with unconventional methods I. Theoretical basis: image analysis with multivariate statistics and entropy concepts

Pierre Trebbia *
Laboratoire de Physique des Solides, Bâtiment 510, F-91405 Orsay Cedex, France

and

Noël Bonnet
Unité INSERM 314 et Université de Reims, 21 rue Clément Ader, F-51100 Reims, France

Received 7 June 1990

Electron energy loss filtered images recorded within a transmission analytical electron microscope are now widely used for the mapping of the elemental distribution of a given atomic species in a specimen prepared as a thin film. Such an image processing may produce both valuable results and artifacts if a careful inspection of all the hypotheses needed by the calculation is not carried out. This paper presents some general statistical methods for a contrast information analysis of a noisy image data set. After a brief introduction of different concepts such as contrast, variance, information and entropy, two unconventional approaches for image analysis are explained: the relative entropy computed with respect to a pure random and signal-free image and the factorial analysis of correspondence (a branch of multivariate statistics). In the companion article (part II), these concepts are applied to real experiments and the results compared with those obtained with a conventional method. Although electron energy loss spectroscopy is the only technique considered here, these methods for image analysis can be applied to a wide variety of noisy data sets (spectra, images, ...) recorded from various sources (electrons, photons, ...).

Why historical papers matter:

1. Often contain elementary introductions
2. Deep insights into principles
3. Surprisingly prescient predictions
4. Comparison with the present: see the big picture

““Those who cannot remember the past are condemned to repeat it.”

George Santayana,
The Life of Reason, 1905

Available online at www.sciencedirect.com
ScienceDirect
Ultramicroscopy 106 (2006) 1024–1032
ultramicroscopy
www.elsevier.com/locate/ultramic

Mapping chemical and bonding information using multivariate analysis of electron energy-loss spectrum images

M. Bosman[a,*], M. Watanabe[b], D.T.L. Alexander[c], V.J. Keast[a]

[a]*Australian Key Centre for Microscopy and Microanalysis, University of Sydney, Sydney, NSW 2006, Australia*
[b]*Department of Materials Science and Engineering, Lehigh University, Bethlehem, PA 18015, USA*
[c]*Department of Materials Science and Metallurgy, University of Cambridge, Pembroke St. CB2 3QZ, UK*

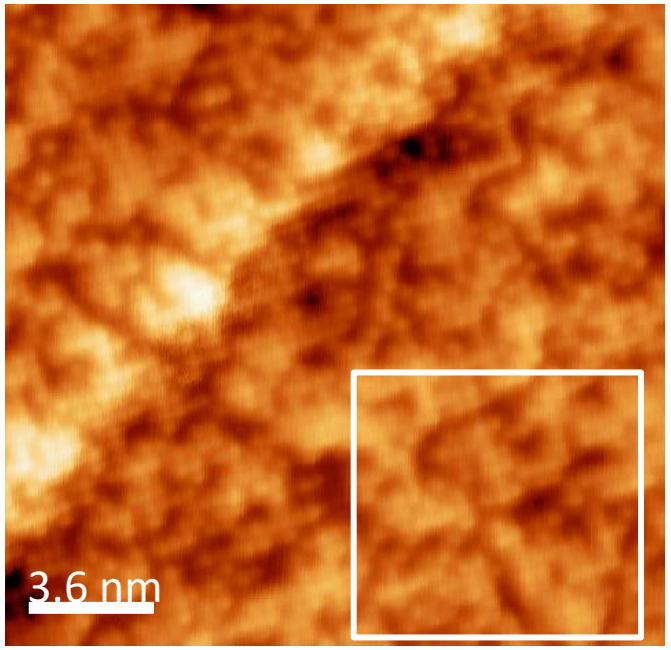
Received 23 June 2005; received in revised form 26 October 2005; accepted 18 April 2006

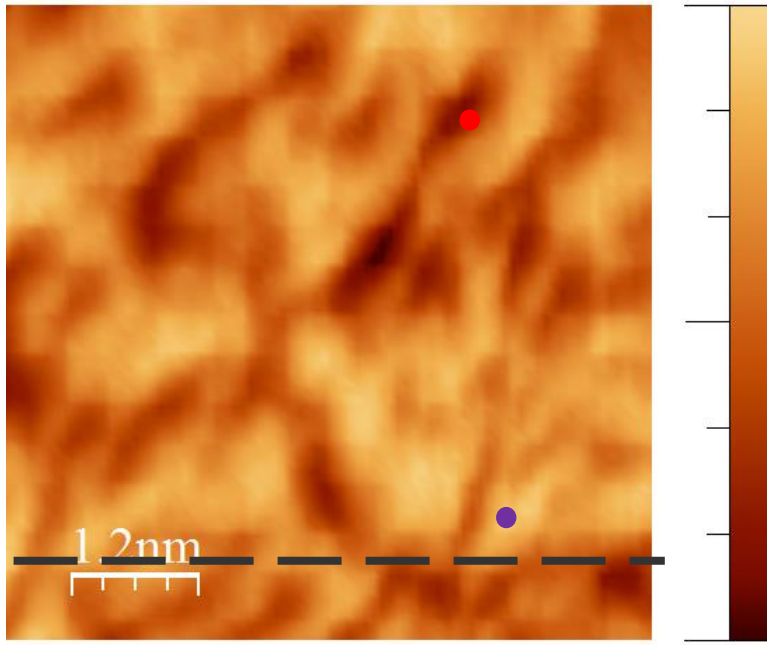
Abstract

Electron energy-loss spectroscopy (EELS) in the transmission electron microscope (TEM) is used to obtain high-resolution information on the composition and the type of chemical bonding of materials. Spectrum imaging, where a full EEL spectrum is acquired and stored at each pixel in the image, gives an exact correlation of spatial and spectral features. However, determining and extracting the important spectral components from the large amount of information contained in a spectrum image (SI) can be difficult. This paper demonstrates that principal component analysis of EEL SIs can be used to extract chemically relevant components. With weighted or two-way scaled principal component analysis, both compositional and bonding information can be extracted. Mapping of the chemical variations in a partially reduced titanium dioxide sample and the orientation-dependent bonding in boron nitride and carbon nanotubes are given as examples.

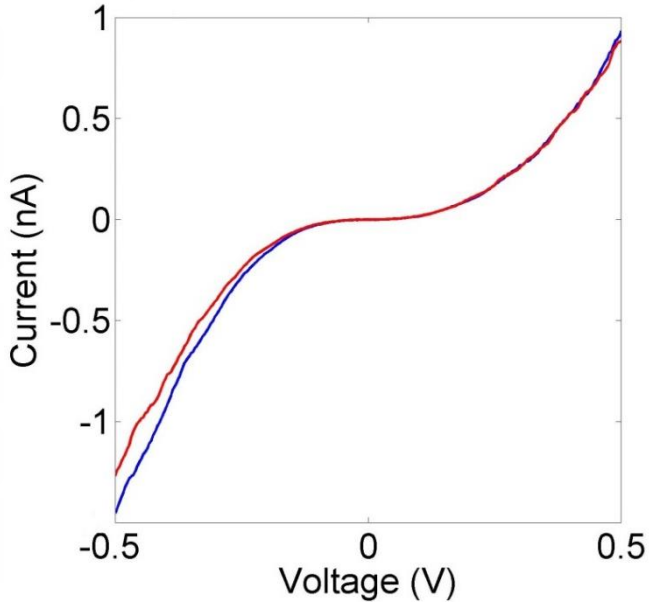
Why did the PCA on EELS started to grow in 2005 – 2010?

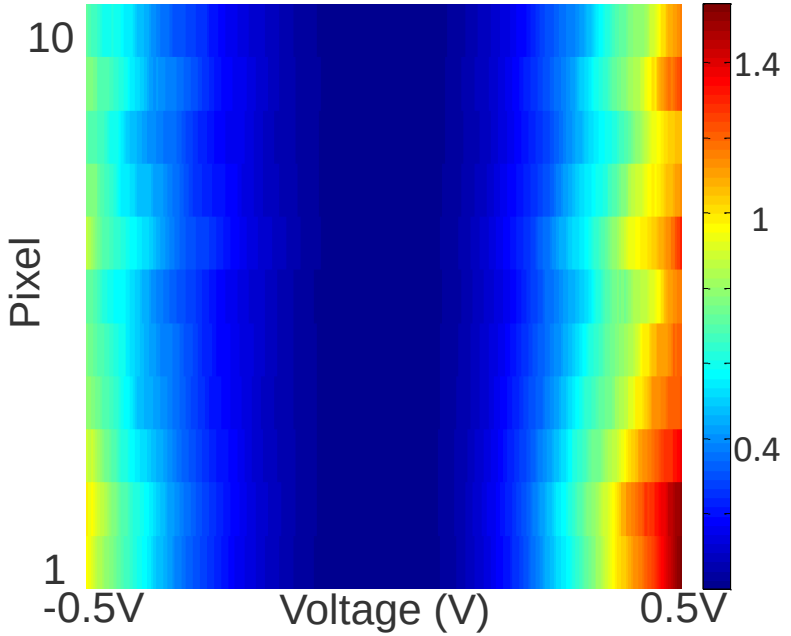
Grain boundary by STM





Topographical STM of FeSeTe, T = 82K 15x15nm^2, 50mV, 100pA, white rectangle represents area where CITS was performed; (b) CITS 80x80 pixel graphical average of the spectrographic data.





M. Ziatdinov, A. Maksov, L. Li, A. Sefat, P. Maksymovych, and S.V. Kalinin, *Deep data mining in a real space: Separation of intertwined electronic responses in a lightly-doped* $BaFe_2As_2$, Nanotechnology **27**, 475706 (2016).

Eigenvectors and loadings

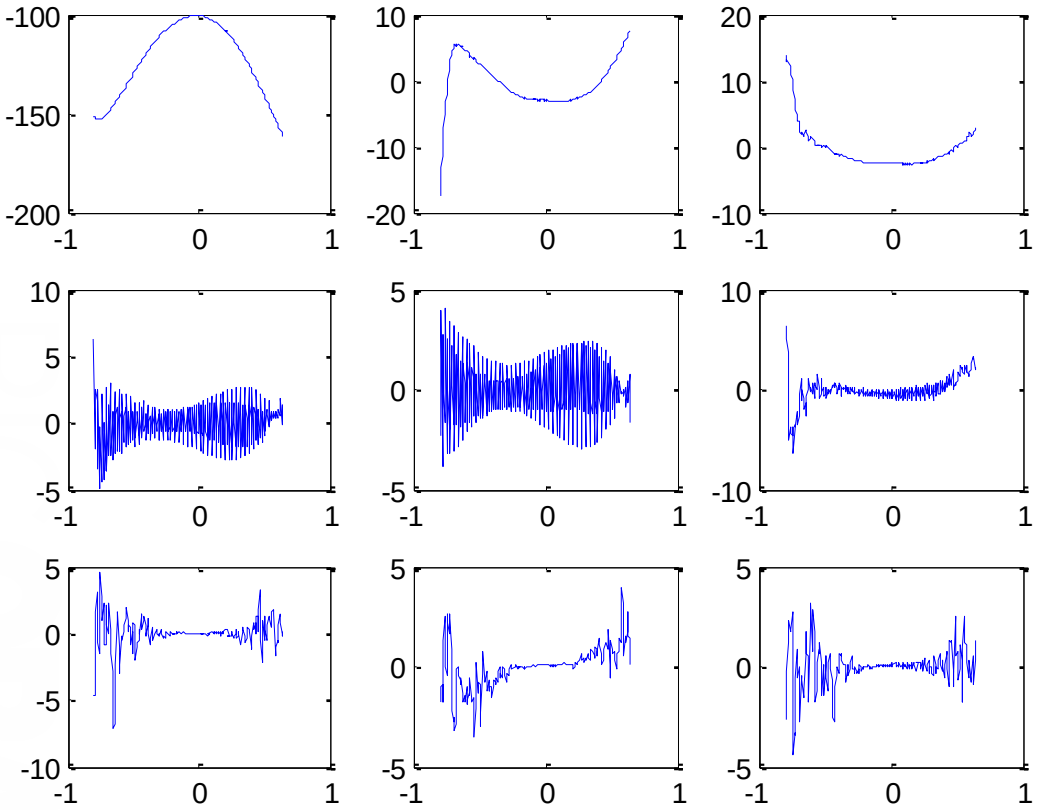


Figure by A. Belianinov

Scree plot and correlations

- Semi log plot indicating the “weight” of each component as a function of all components
- Only the first few components contain useful info, while others are dominated by noise

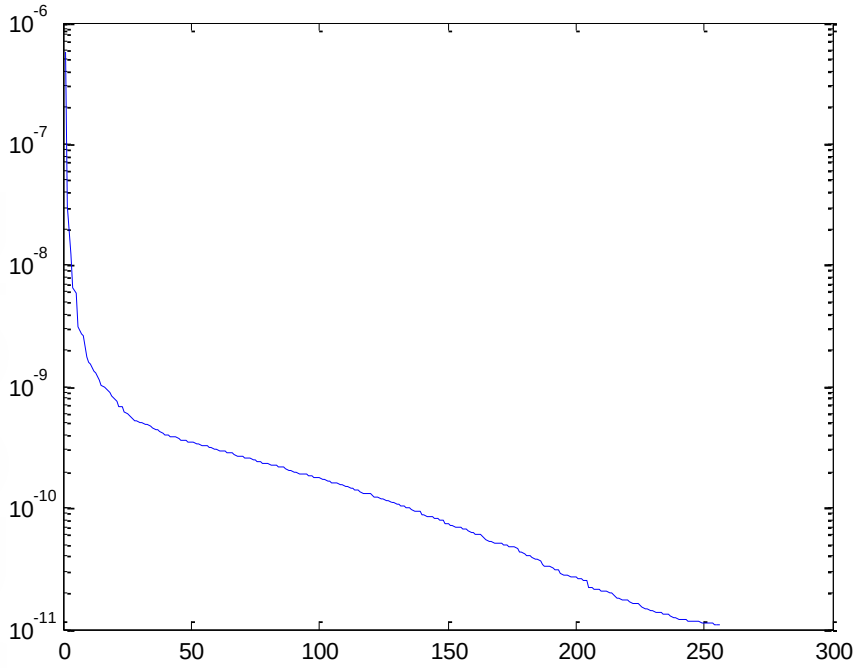
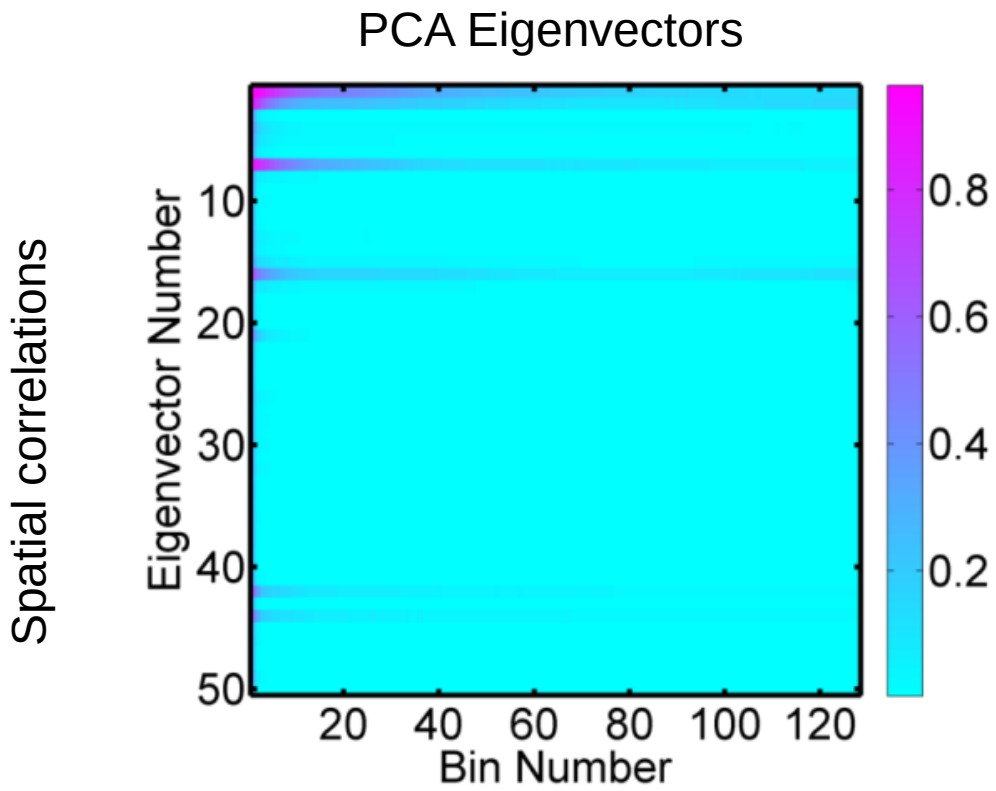


Figure by A. Belianinov

- We can also analyze correlations in images

For AFM data

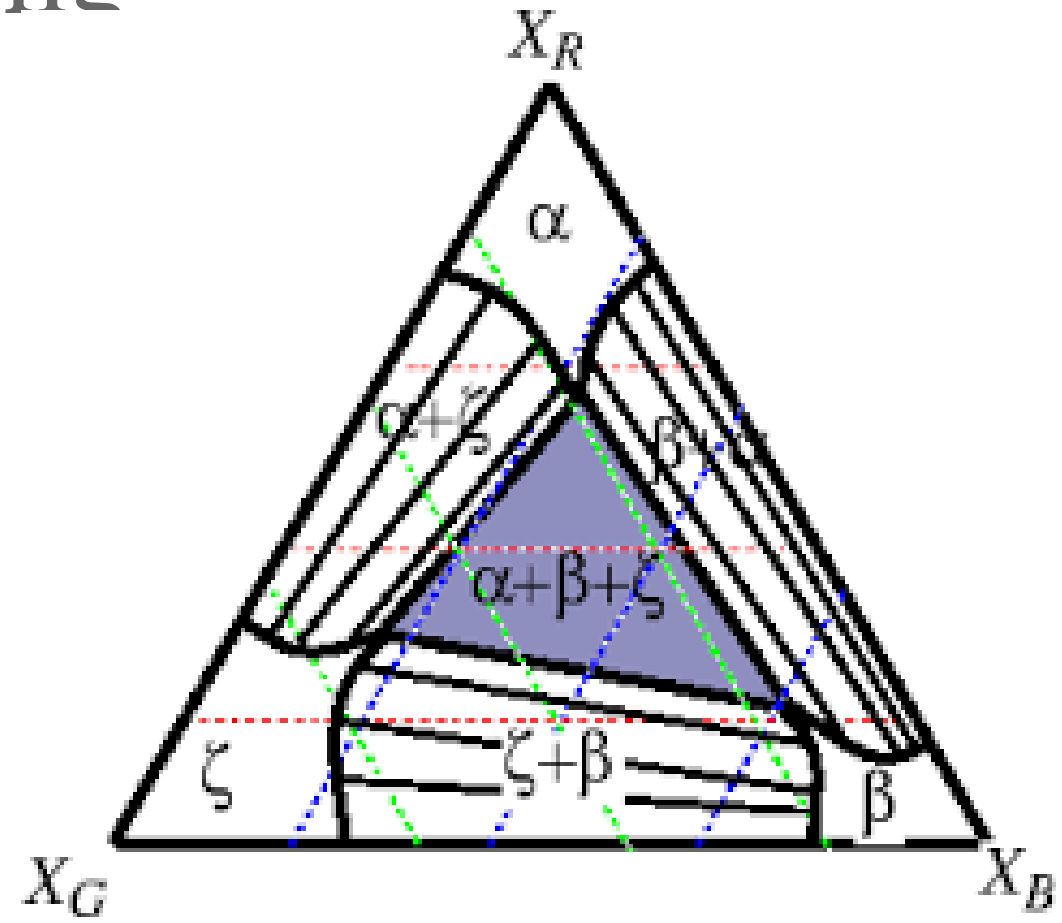


Bayesian Linear Unmixing

$$S(\mathbf{x},\mathbf{R}) = \sum_{i=1}^{K} a_i(\mathbf{x}) w_i(\mathbf{R}) + \mathrm{N}$$

$$\sum_{i=1}^{K} a_i(\mathbf{x}) = 1$$

- The eigenvectors $w_i(\mathbf{R})$ are non-negative, $w_i(\mathbf{R}) \geq 0$
- The loading coefficients sum to 1
- The number of eigenvectors, K, is a priori unknown



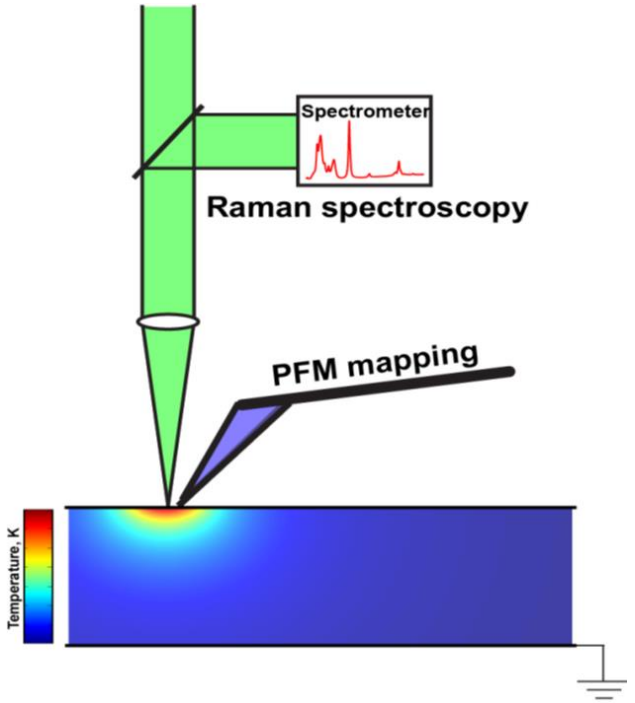
BLU is ideally suited for certain classes of problems, e.g. conduction through parallel channels, optical or electronic spectra of mixtures, etc

Laser heating induced phase transitions

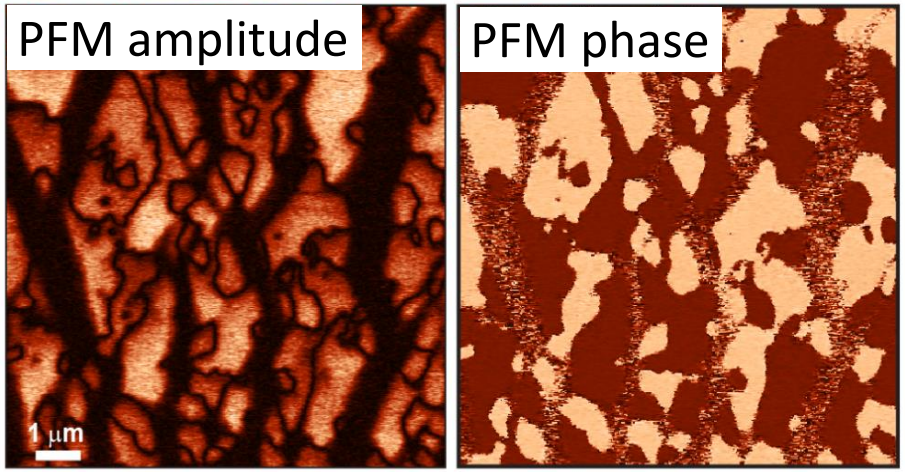
- Copper indium thiophosphate ($Cu_{0.77}In_{1.12}P_2S_6$) layered ferroelectric
  - Ferroelectric state at room temperature
  - Curie temperature T_c = 320 K
  - Non-polar $In_{4/3}P_2S_6$ inclusions

- Combined Atomic Force Microscopy (AFM) and confocal Raman spectroscopy investigative approach
  - AFM – topography measurements
  - Piezoresponse force microscopy (PFM) – static ferroelectric domain structure
  - Raman – crystallographic structure via Raman spectra

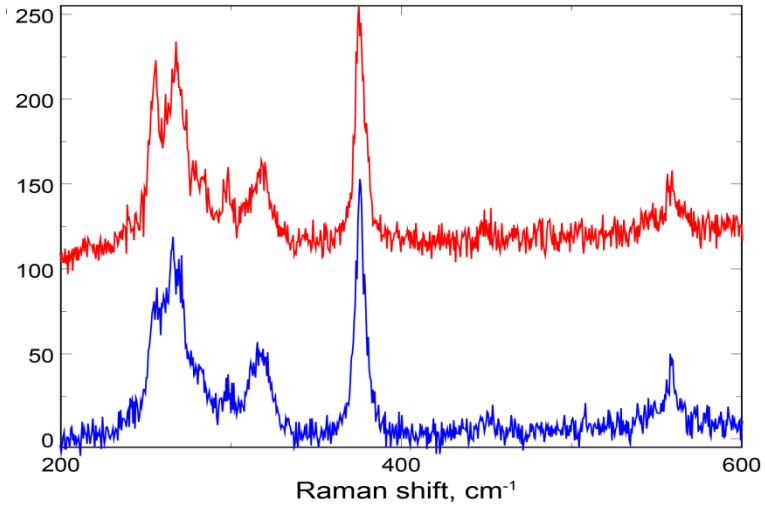
Experimental scheme



Ferroelectric domain structure



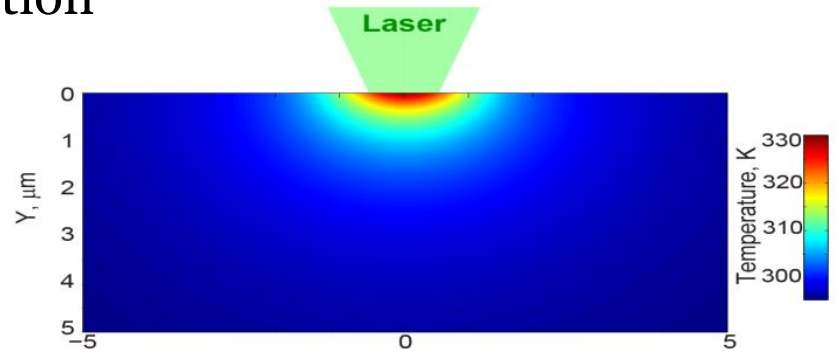
Single point Raman spectra



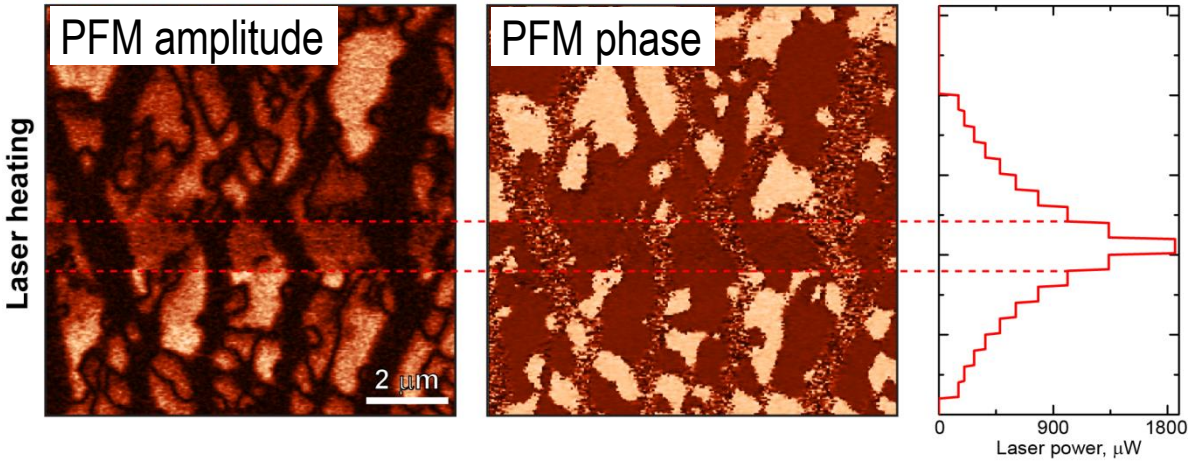
A. Ievlev, ACS Nano **9**, 12442 (2015).

Laser heating induced phase transition

Laser can be used for local heating to induce ferroelectric- paraelectric phase transition

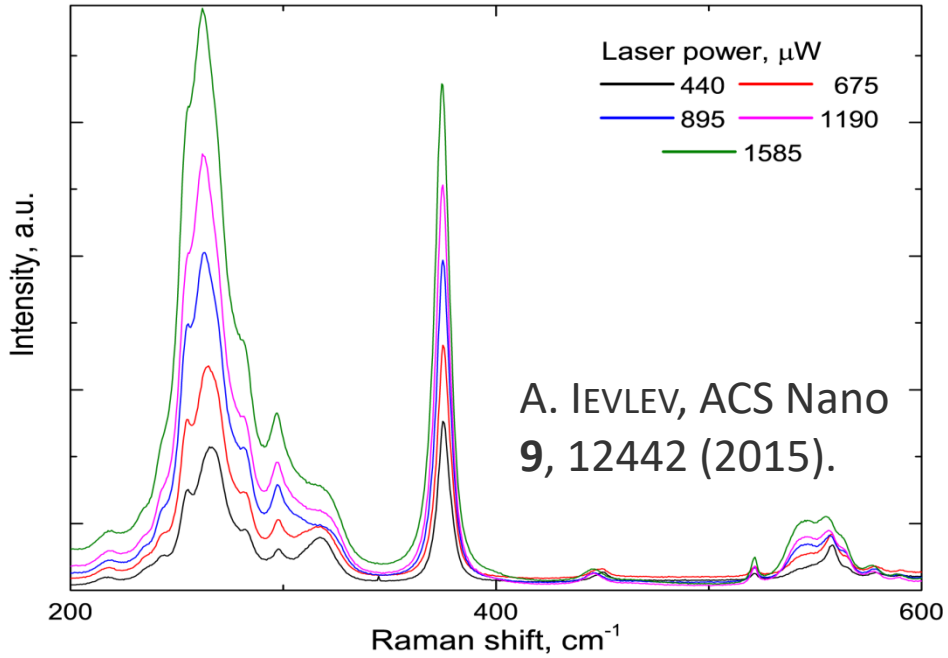


Domain structure evolution **Laser power**



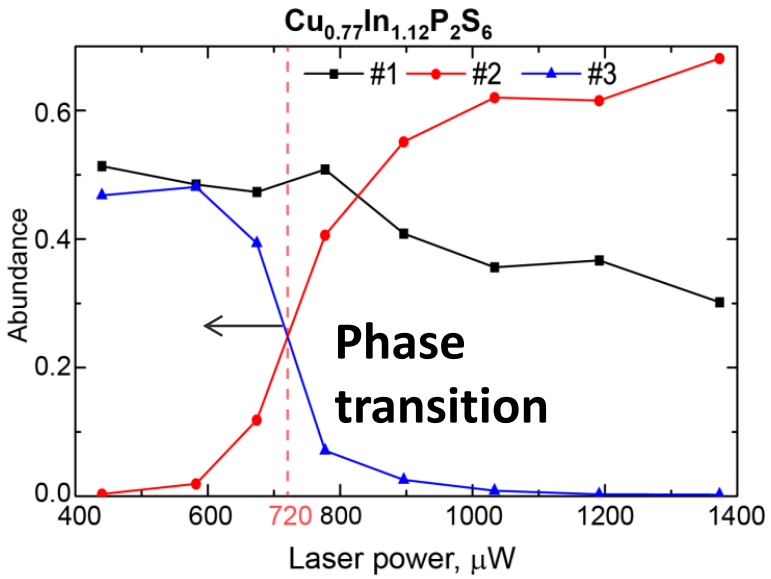
- Measurements with variation of the laser power
  - PFM – *in-situ* change in the domain structure above T_c
  - Raman – evolution of the Raman spectra through the phase transition
- Comprehensive analysis of Raman spectra is complicated by inhomogeneous chemical composition and high noise level
- Bayesian Linear Unmixing can be used for automated identification of spectra evolution

Raman spectra evolution (averaged)



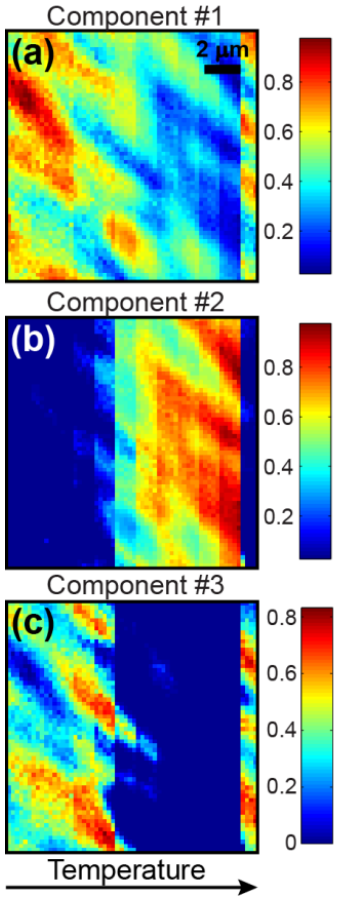
BLU separation of components

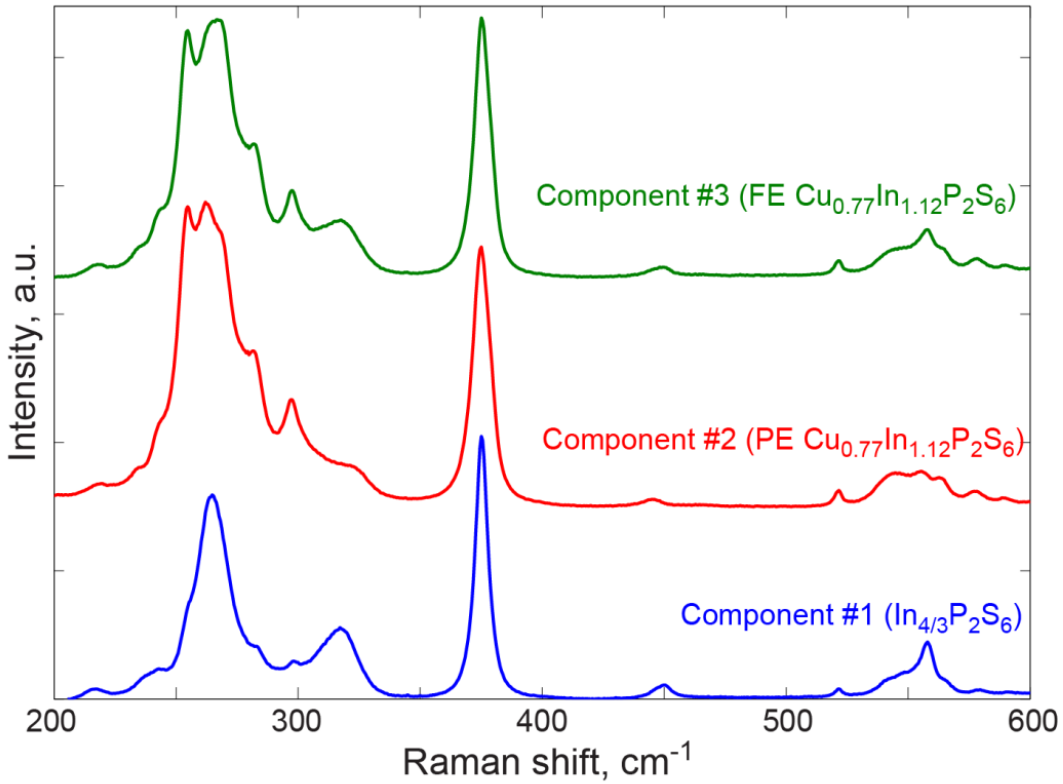
Spatial concentration of components



A. Ievlev, ACS Nano **9**, 12442 (2015).

Results of BLU: components and loading maps





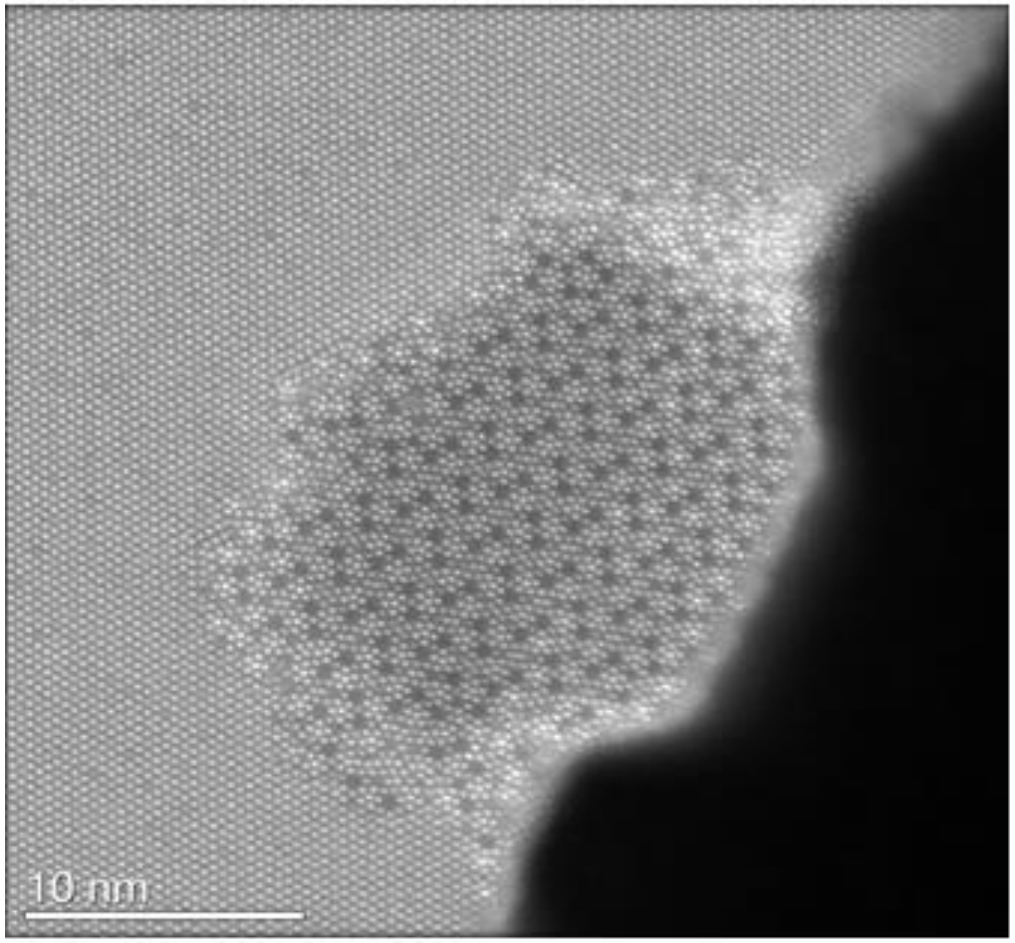
Unmixing showed presence of three independent components in Raman spectra:

1. Non-polar $In_{4/3}P_2S_6$ – weak changes in intensity with temperature
2. Paraelectric $CuInP_2S_6$ above T_c – appears at higher laser powers
3. Ferroelectric $CuInP_2S_6$ below T_c – disappears at higher temperatures

13_PCA_CL.ipynb

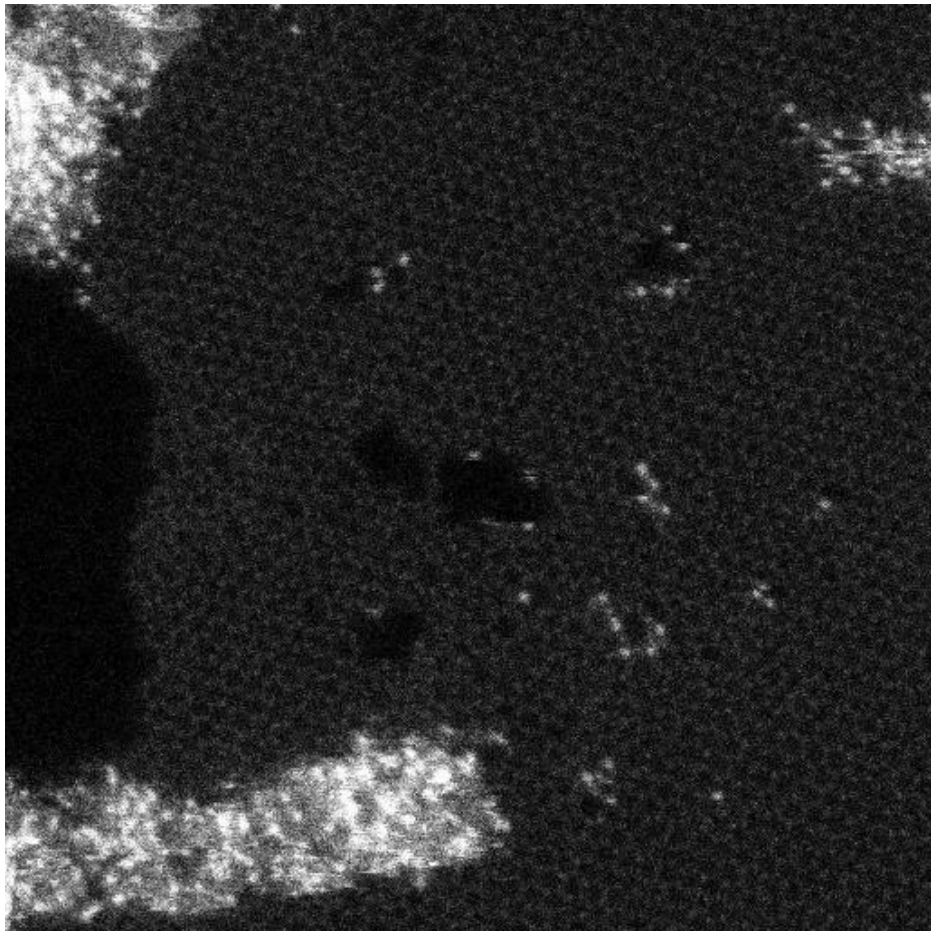
Chemically disordered systems

Mo-V-Ta complex oxide



Q. He et al, ACS Nano 9, 3470-3478

Si in graphene

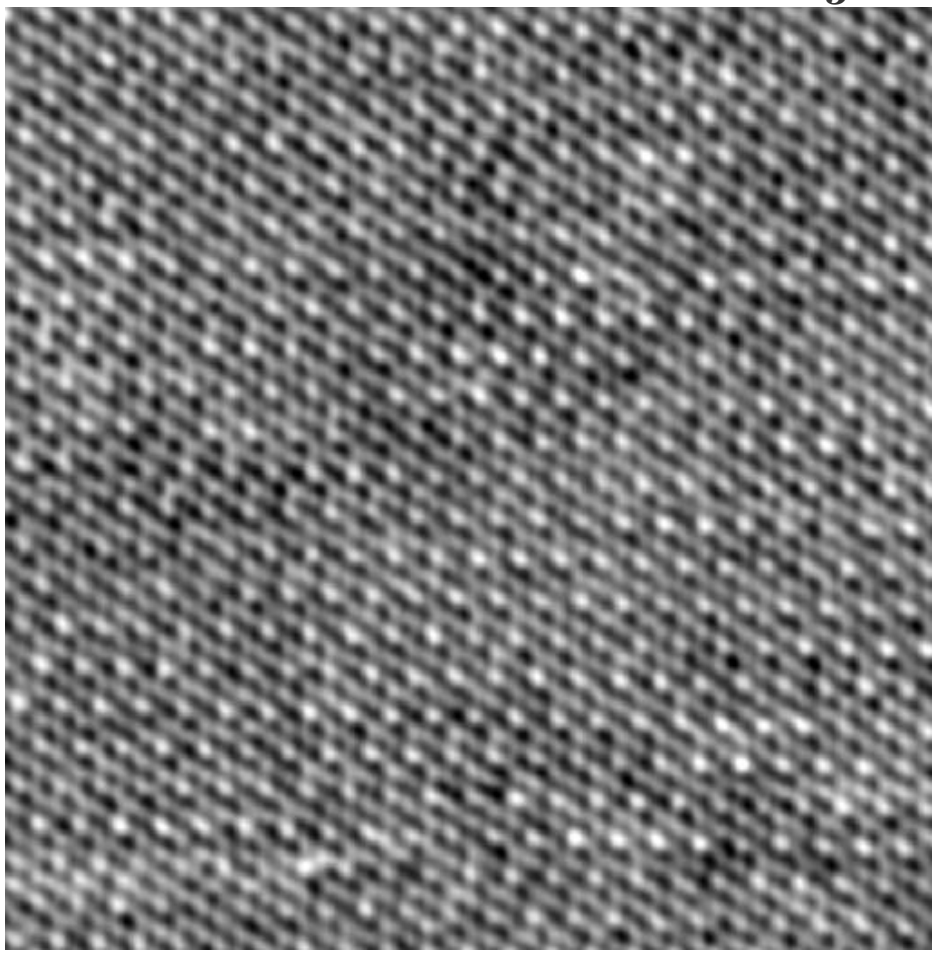


Data collected by O. Dyck (ORNL)

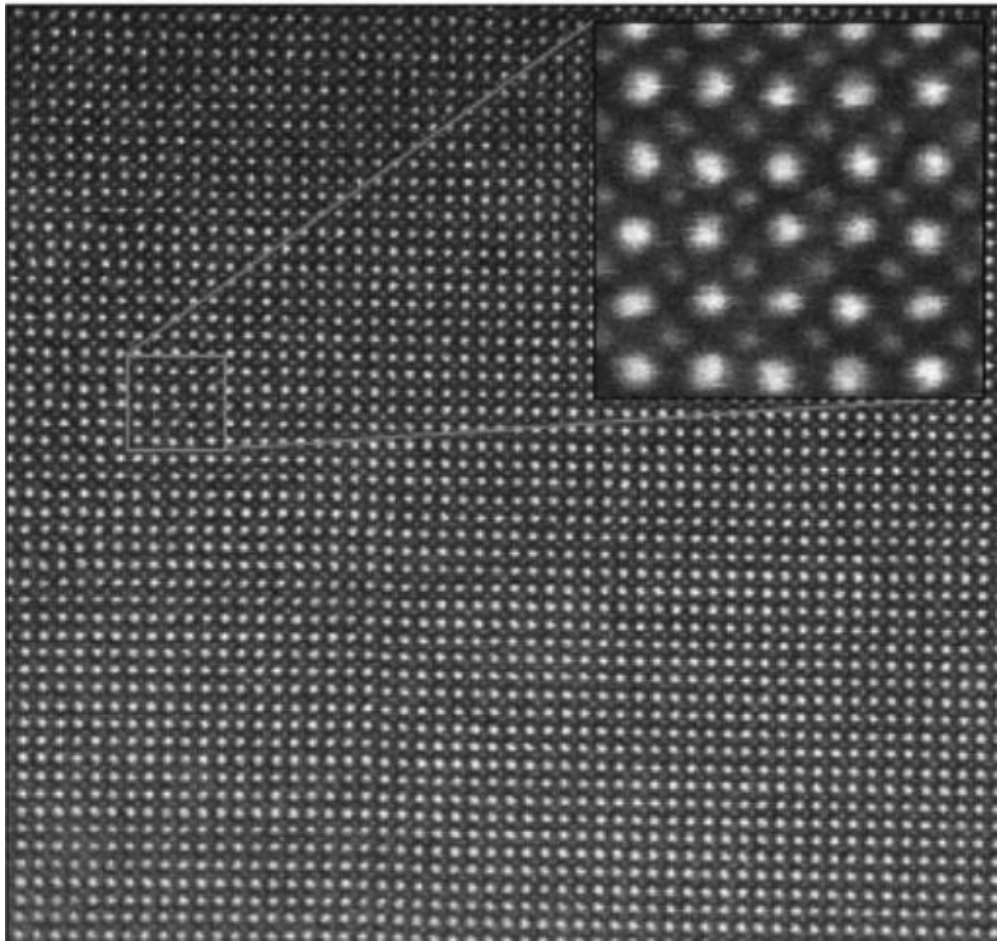
- What is the nature of the building blocks and relevant atomic configurations?
- Can we define single-phase regions and phase boundaries?

But what about subtle distortions?

Electronic structure in $RuCl_3$



$BiFeO_3$ on $SrRuO_3$



- Can we identify ferroelectric and ferroic variants and associated topological defects?
- What is the nature of the phases?

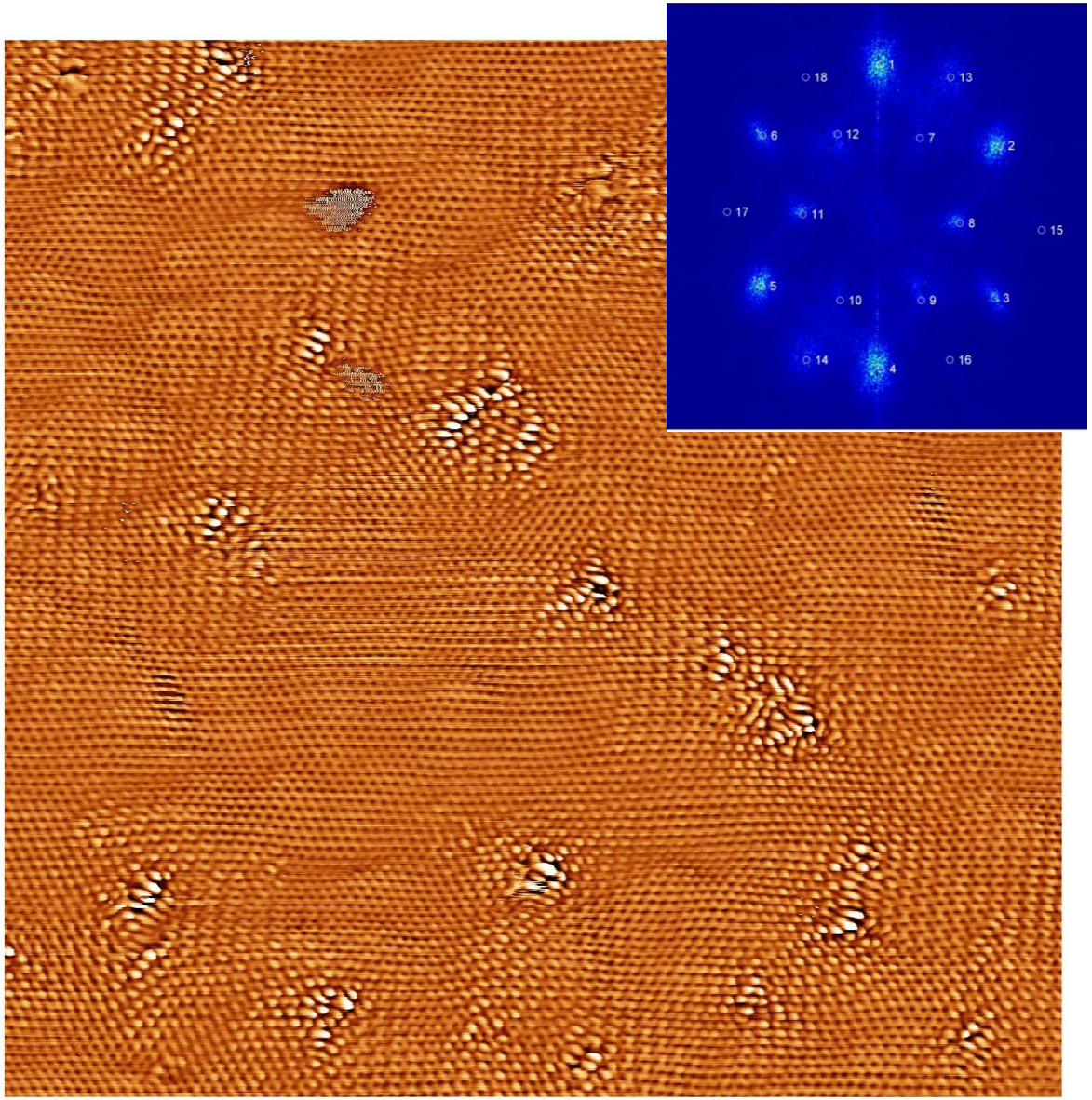
Can global FFT help?

- Global FFT – everything is averaged:
  - Drift
  - Extended defects
  - Multiple grains

We are averaging out all interesting phenomena except for small spatially uniform structural distortions

- Solution – sliding window approaches:
  - Fit FFT peaks: amplitudes, positions
  - Multivariate analysis

Note: window can be also tied to a specific feature, such a selected atom. Then we explore atomic neighborhood



General linear unmixing

$$S(\mathbf{x}, \mathbf{R}) = \sum_i a_i(\mathbf{x})\, w_i(\mathbf{R}) + \mathrm{N}$$

We start with:

- **x** is the spatial variable, x = (*x*,*y*)
- R is the (vector) parameter variable

We aim to get:

- $a_i(\mathbf{x})$ are loading maps
- $w_i(\mathbf{R})$ are endmembers/eigenvectors
- N is noise

The M pixel 2D image is transformed to M/N pixel image of more complex structure.

Our loading map is 2D image, and endmembers/eigenvectors are 2D images

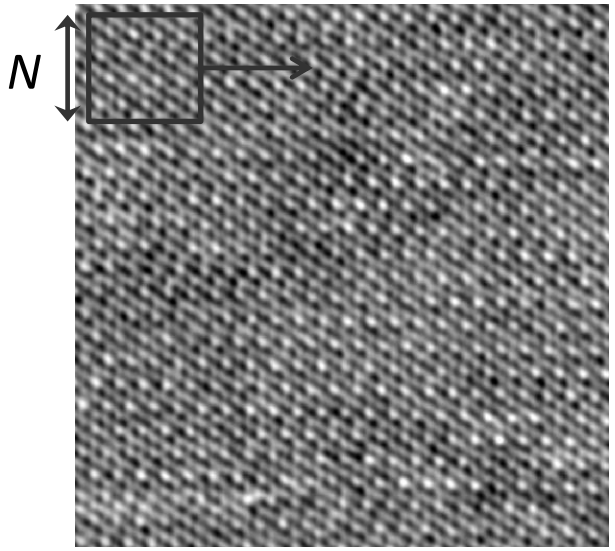
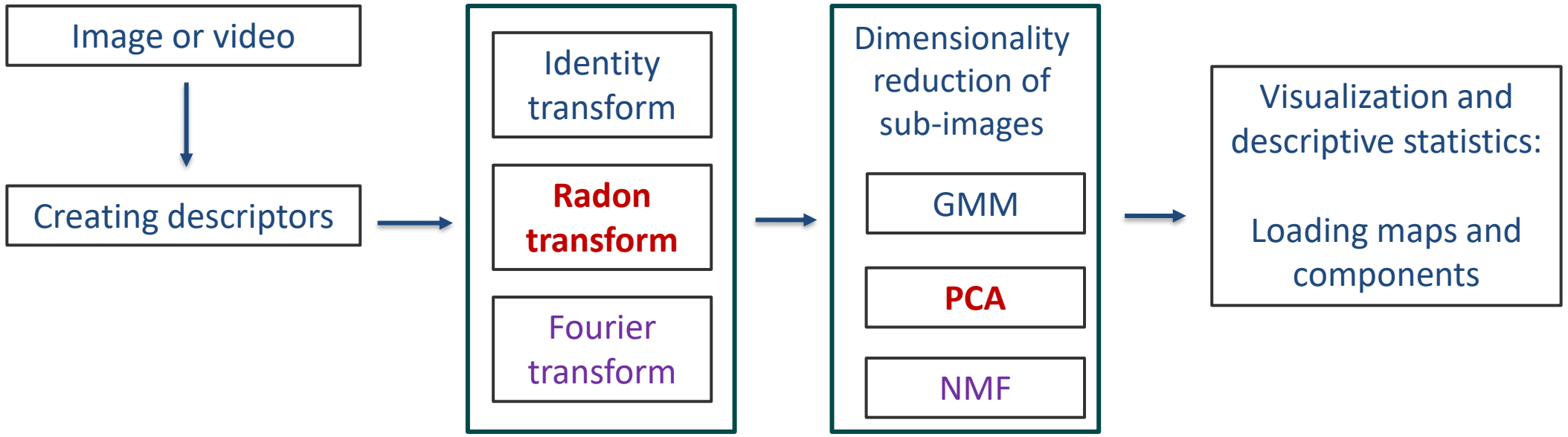


Figure by M. Ziatdinov

Sliding image transforms:

- Fast Fourier Transforms
- Correlation functions
- Intensity histograms
- Structural descriptors

Example of analysis pipeline



Pipelines are defined to

- Make analysis traceable, repeatable, explainable, and transferable
- Allow for hyperparameter tuning and optimization
- Efficiently use the memory

Sliding PCA-FFT

Can we use PCA of FFT transform in sliding windows to find periodicity?

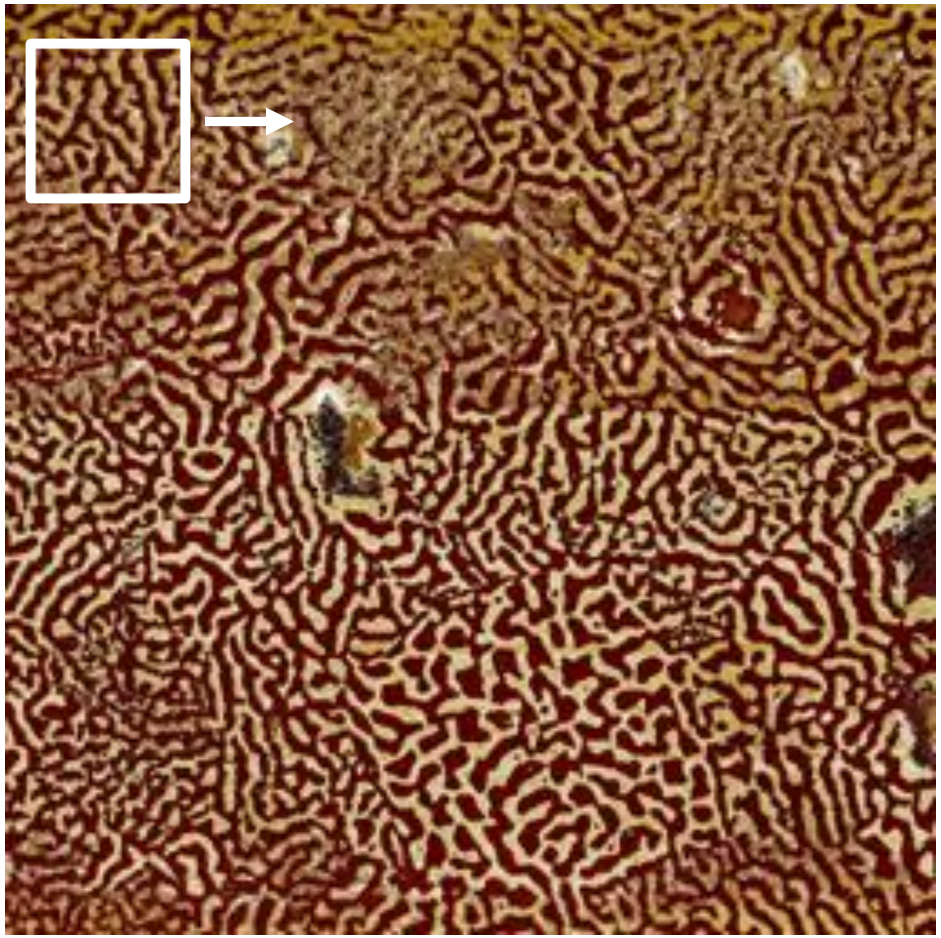
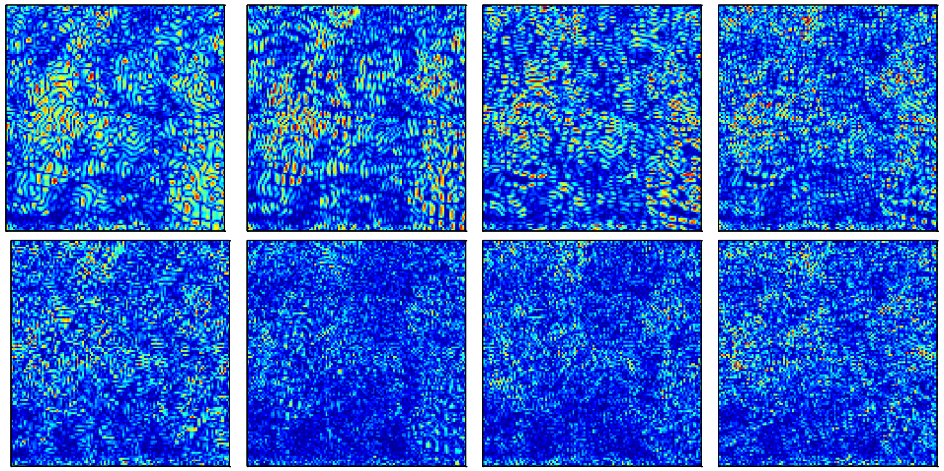
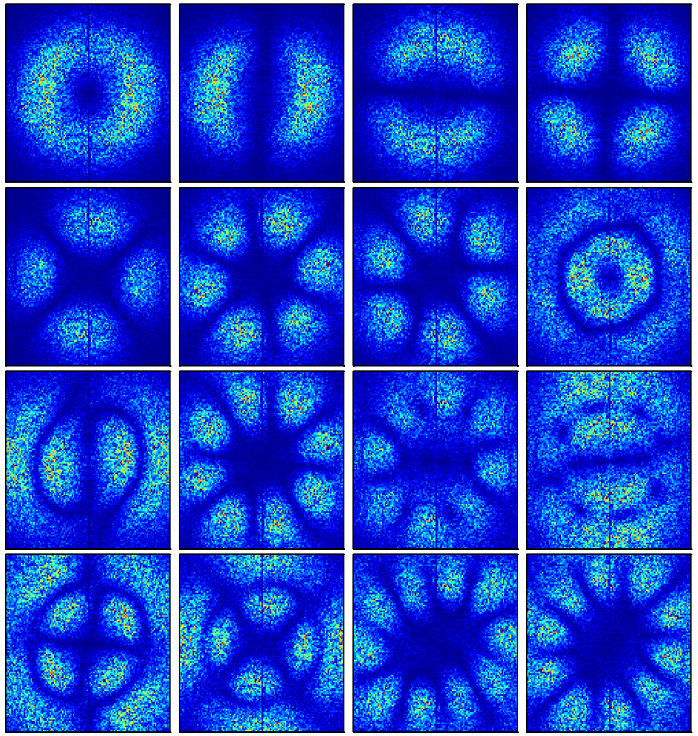


Figure by S. Jesse, data D. Gobelic

First 8 maps



First 16 eigenvectors



Spectral Unmixing: N-FINDR

Spectra for a given pixel is assumed to be a linear combination of the end-member spectra (+ Gaussian noise). The mixing proportions sum to 1

Physics constraint

$$p_{ij} = \sum_k e_{ik} c_{kj} + \varepsilon \qquad \sum_k c_{kj} = 1$$

- Let E be the matrix of end-members (here, 3).

$$\boldsymbol{E} = \left[\overrightarrow{e_1} \quad \overrightarrow{e_2} \quad \overrightarrow{e_3}\right] \qquad V\left(\frac{\mathbf{1}}{(\boldsymbol{l}-1)!}\right)|\det(E)|$$

- Iteratively select endmembers, accepting the new selection if the volume increases

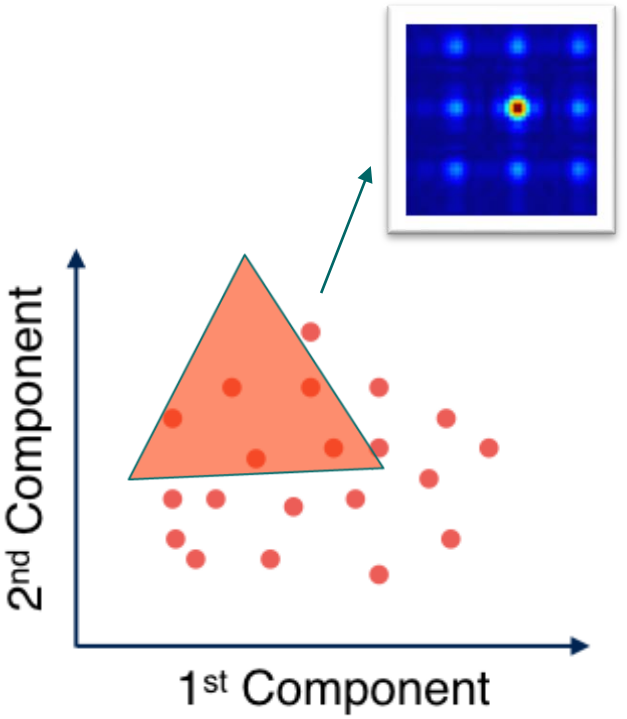


Figure by R. Vasudevan

Ideal test case

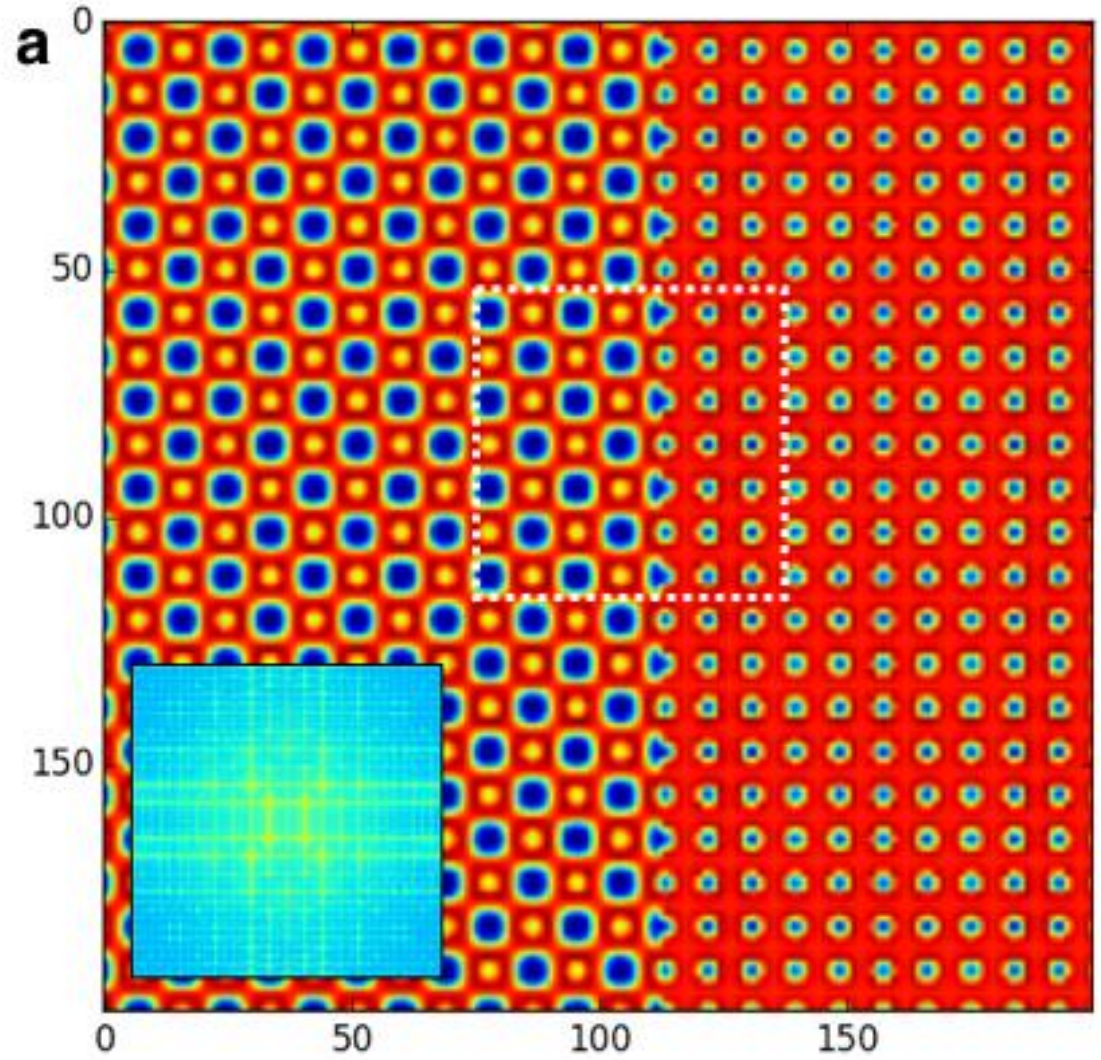
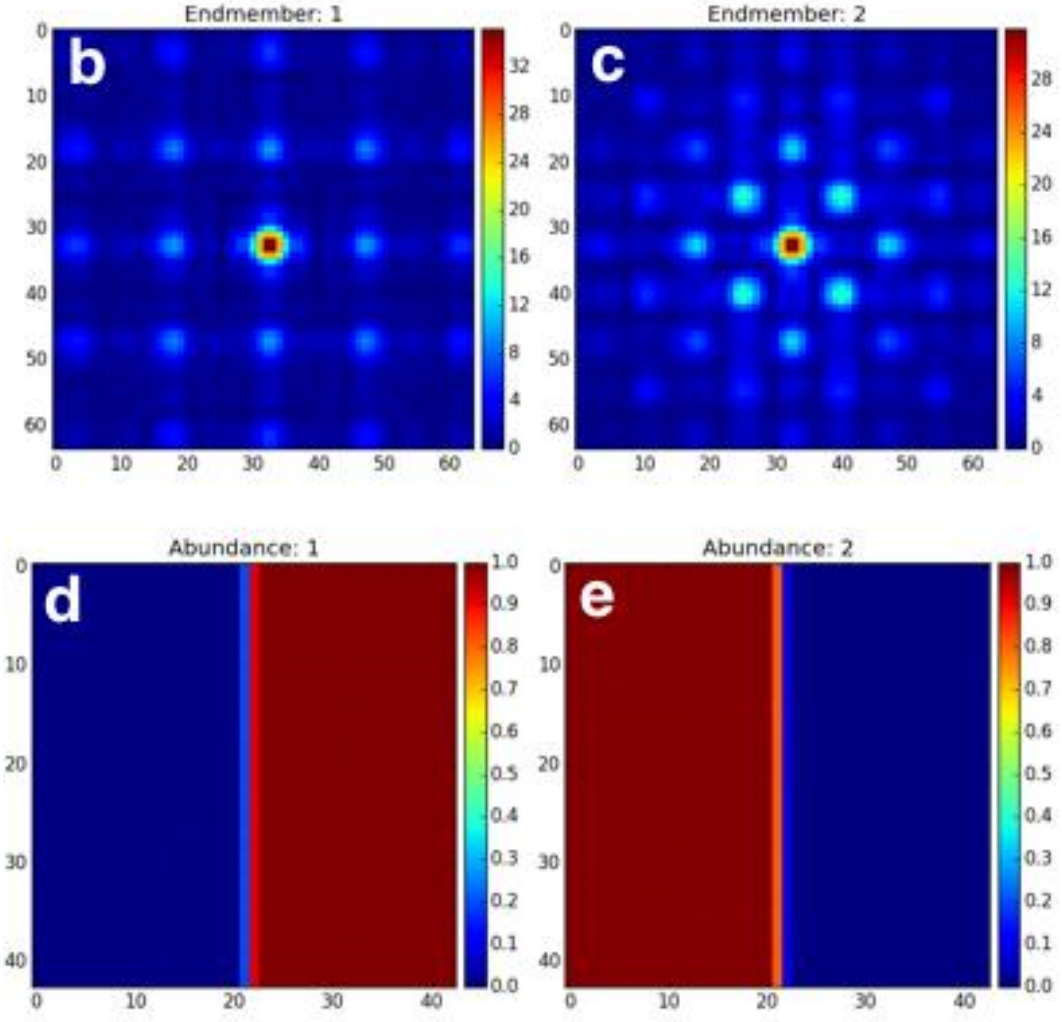


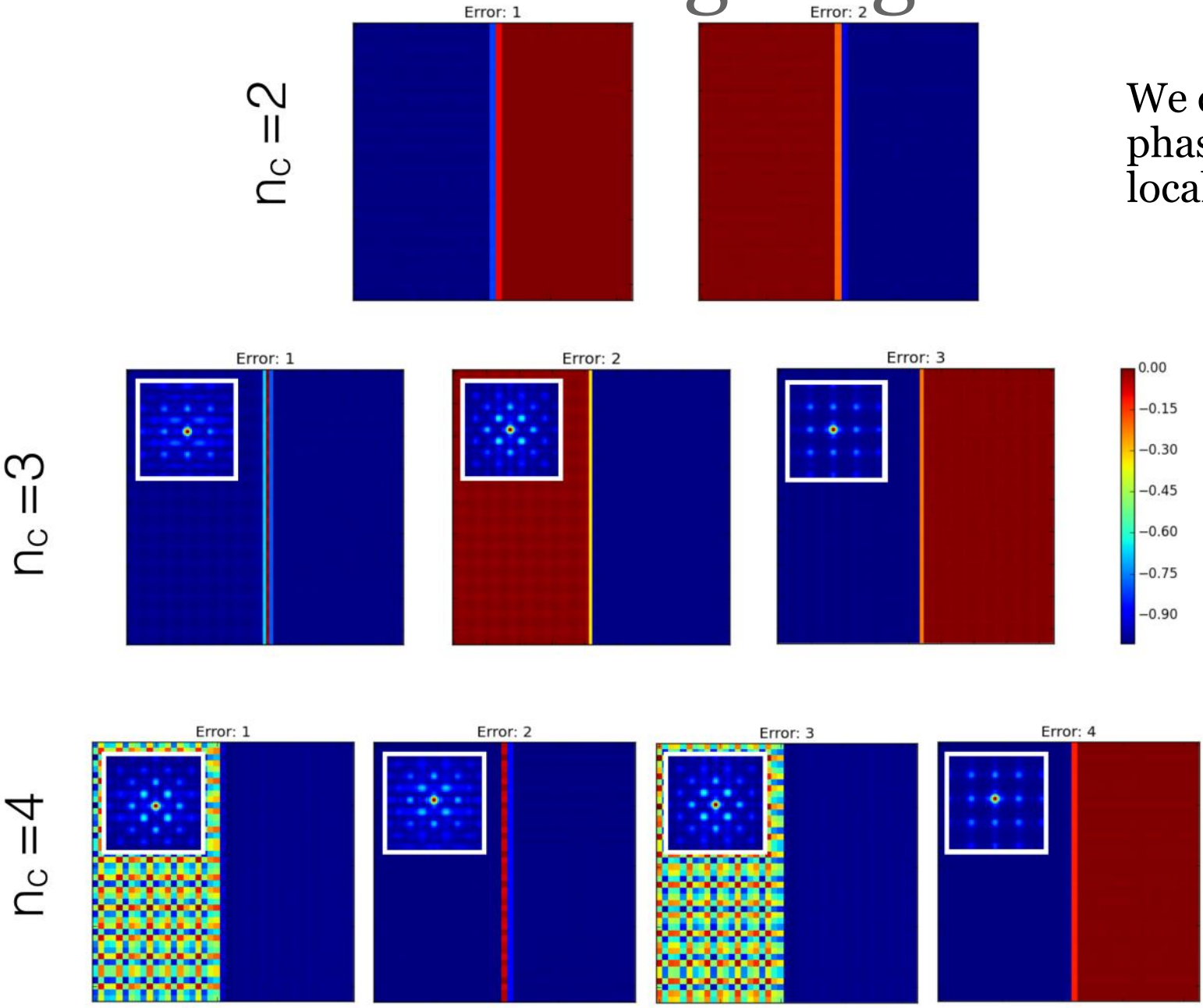
Figure by R. Vasudevan



Main idea:

- FFT amplitudes are non-negative;
- FFT removes translation

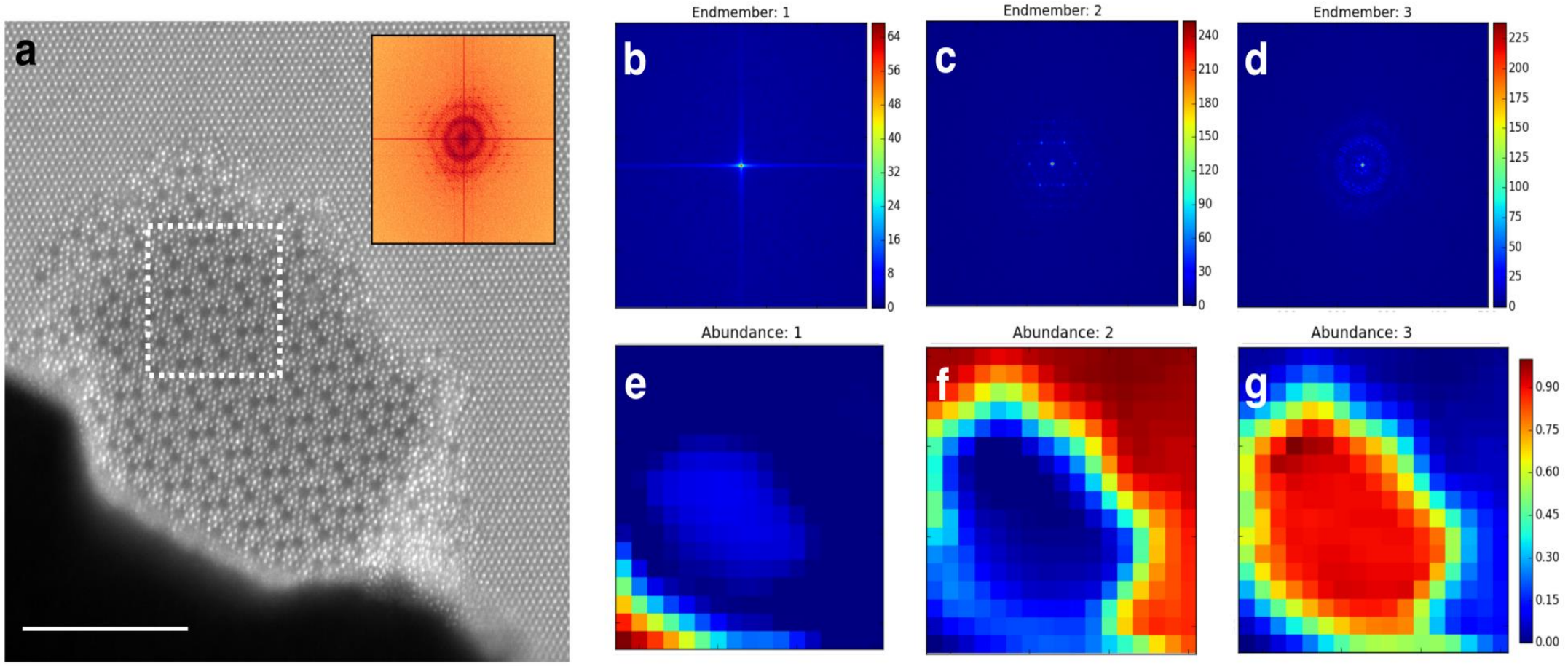
N-FINDR for image segmentation



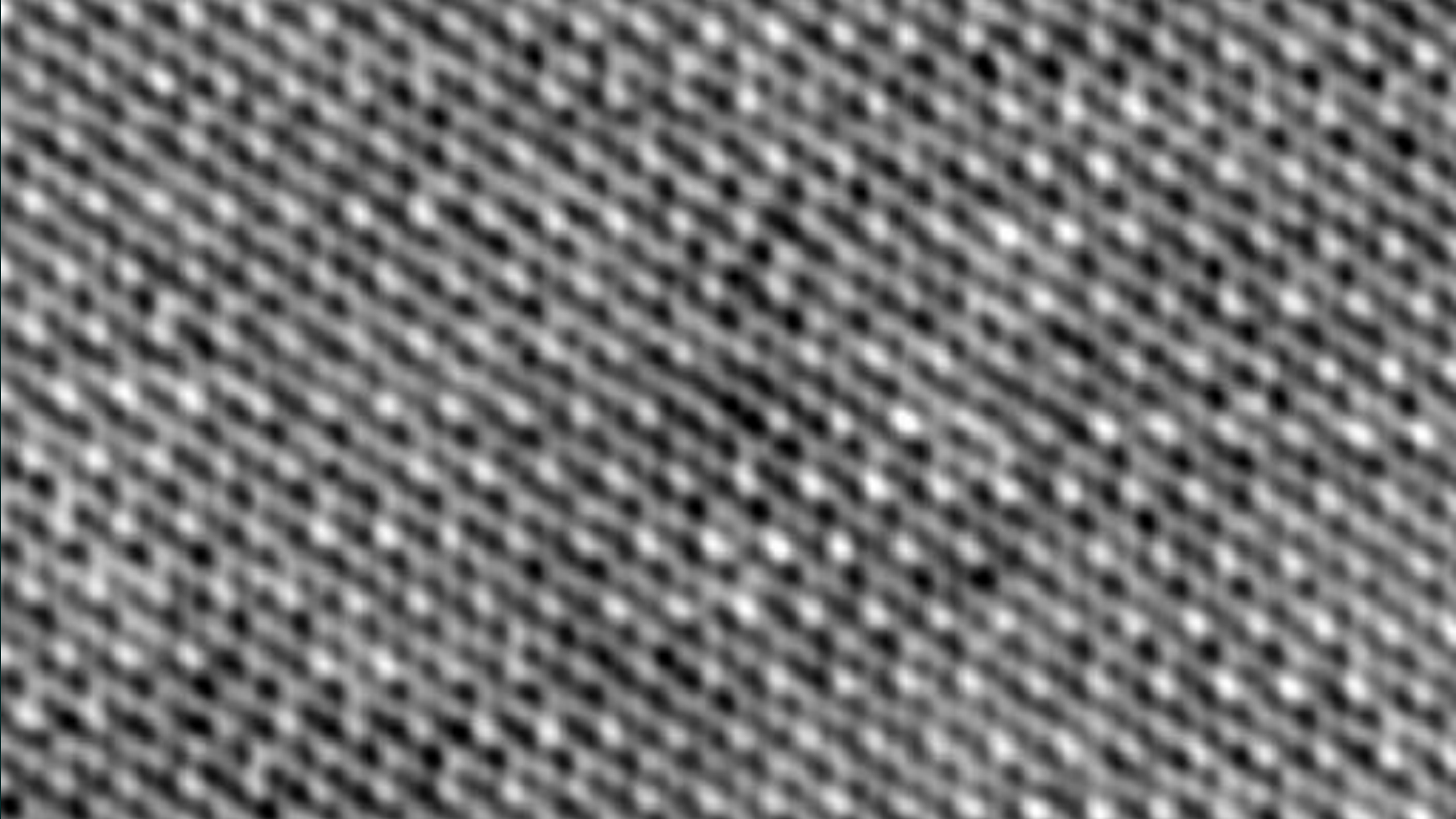
We can determine number of phases based on spatial localization and geometry

Figure by R. Vasudevan

N-FINDR for chemically separated images

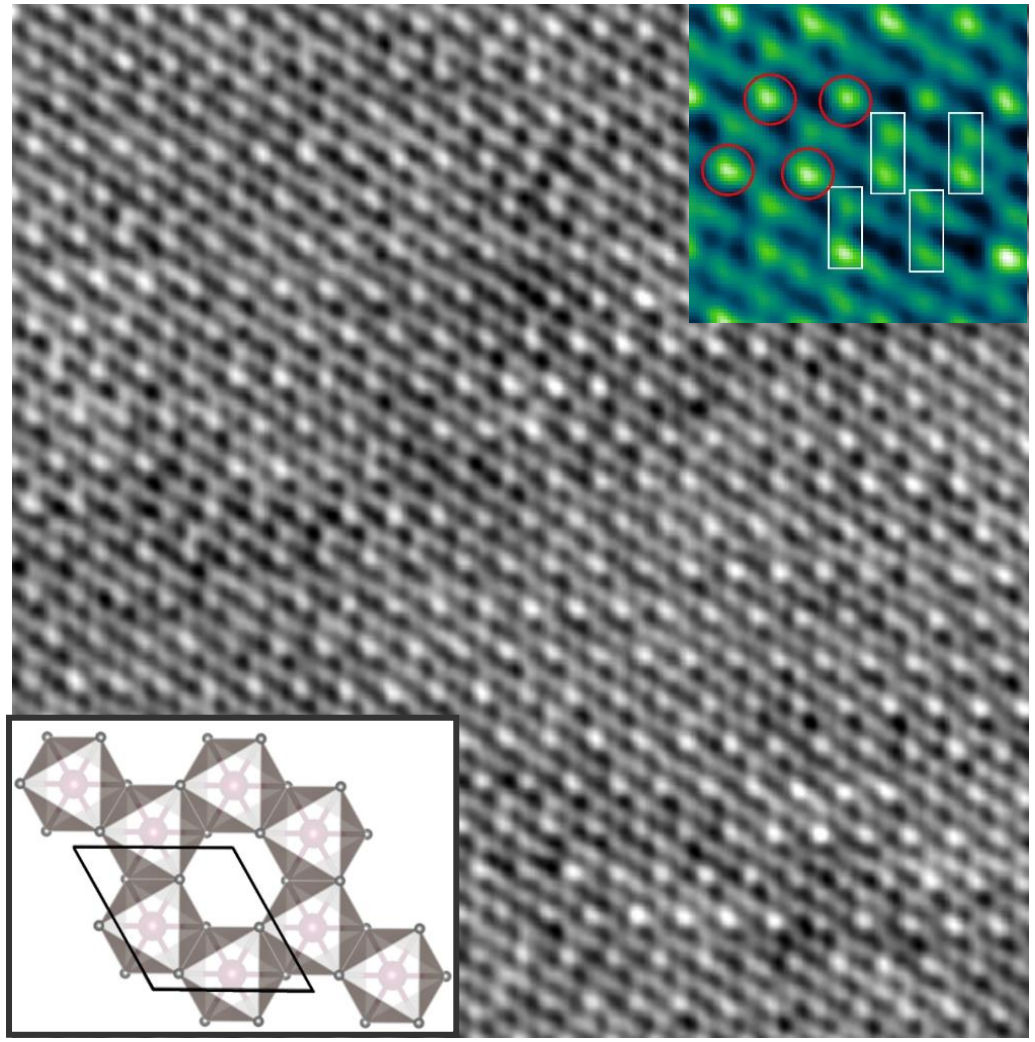


Q. He, J. Woo, A. Belianinov, V.V. Guliants, A.Y. Borisevich, *Better catalysts through microscopy: mesoscale M1/M2 intergrowth in Molybdenum–Vanadium based complex oxide catalysts for propane ammoxidation*, ACS Nano 9, 3470-3478

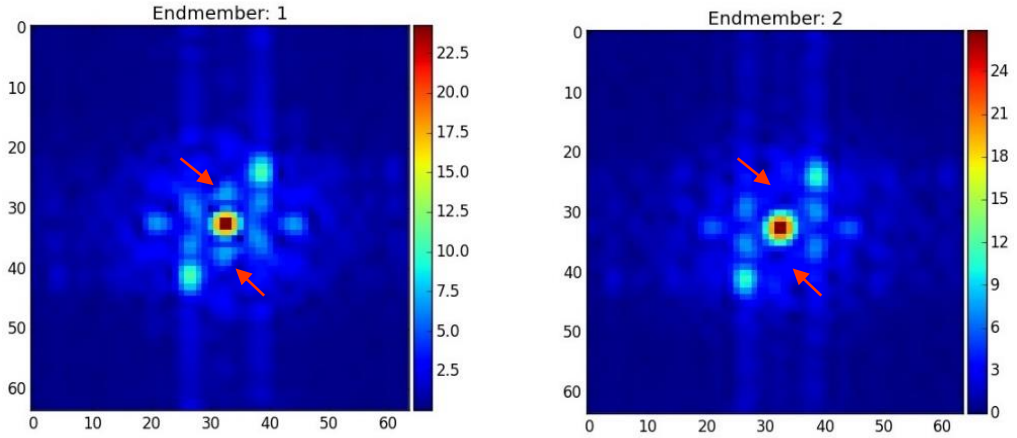


NFINDR for coexisting order parameters

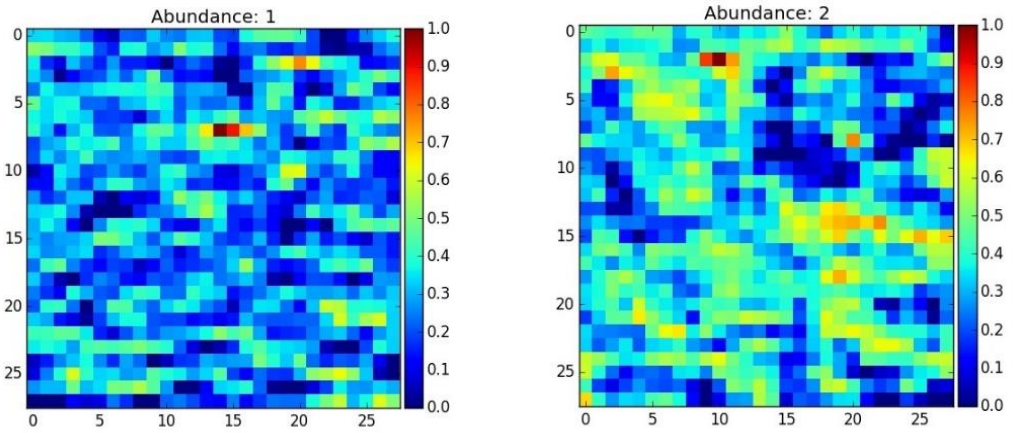
Input experimental image



FFT endmembers



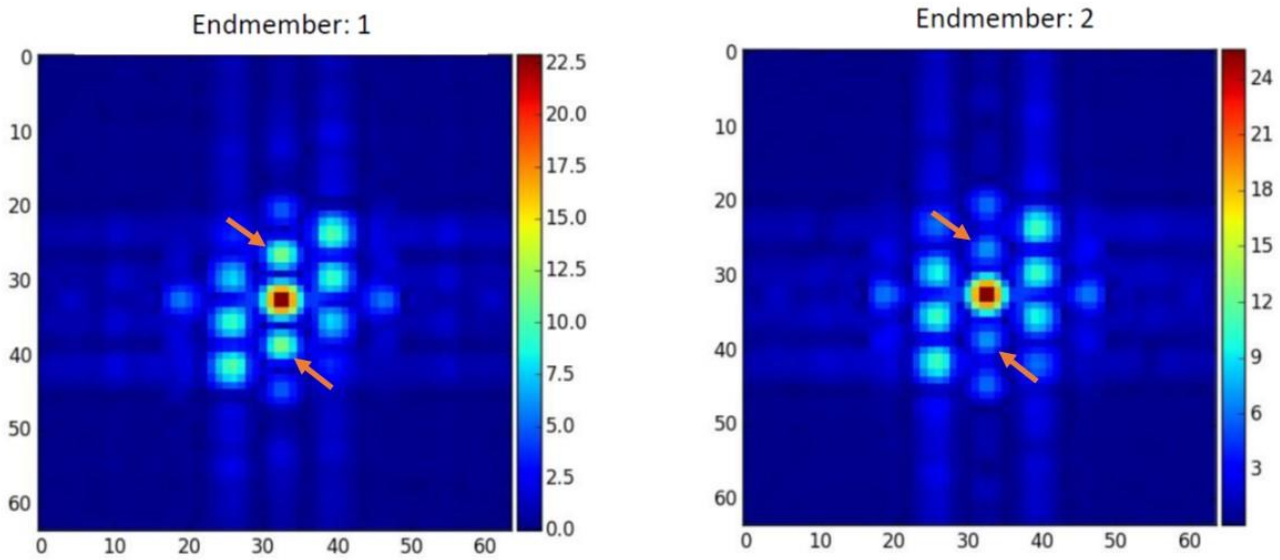
Real-space abundance maps



In a good agreement with test case, 2 spots in the “inner hexagon” are strongly suppressed in the 2nd component reflecting a fine structure of charge ordered pattern

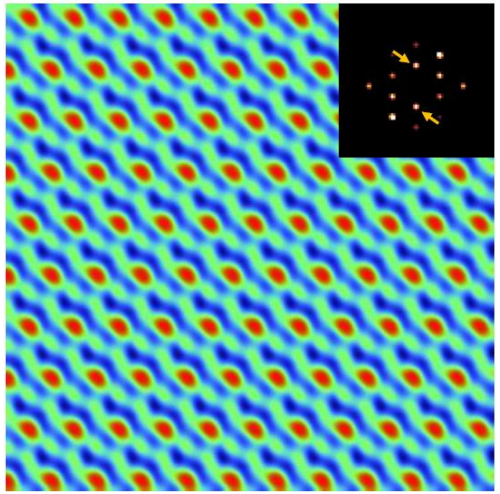
NFIND-R for coexisting order parameters

FFT endmembers

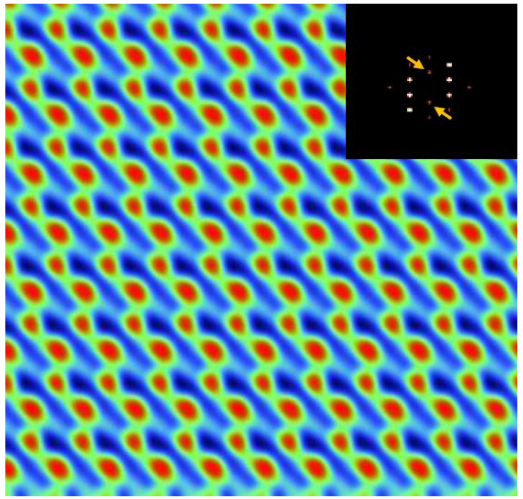


In the 2nd component, 2 spots in the "inner hexagon" are strongly suppressed reflecting a fine structure of charge ordered pattern

Real-space images of corresponding phases



Hexagonal superlattice



Dimer superlattice

M. Ziatdinov, A. Banerjee, A. Maksov, T. Berlijn, W. Zhou, H.B. Cao, J.Q. Yan, C.A. Bridges, D.G. Mandrus, S.E. Nagler, A.P. Baddorf, and S.V. Kalinin, *Atomic-scale observation of structural and electronic orders in the layered compound* α-$RuCl_3$, Nature Comm. 7, 13774 (2016).

Sliding FFT:

- We always have a problem of window size:
  - too large – loose spatial resolution,
  - too small – FFT behaves poorly due to edge effects
- Interpretation of FFT data is complicated (too much data if fit each peak, unclear meaning of the unmixing components)
- Natural descriptor for atomically resolved images – atomic coordinates!

Local crystallography

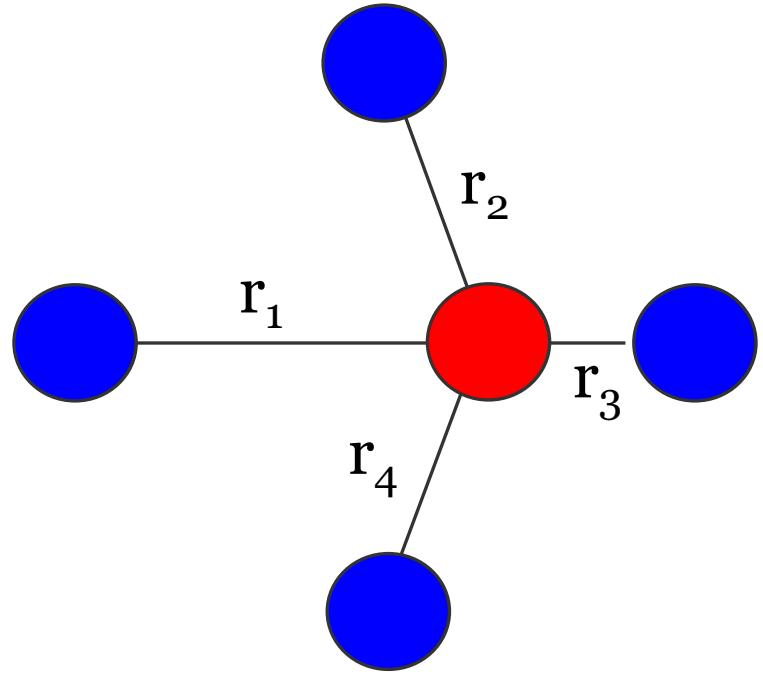
For each atom, define nearest neighbors and generate array of the corresponding radius-vectors of the form

$NA_{ij} = (rx_1, ry_1, rx_2, ry_2, rx_3, ry_3, rx_4, ry_4)_{ij}$

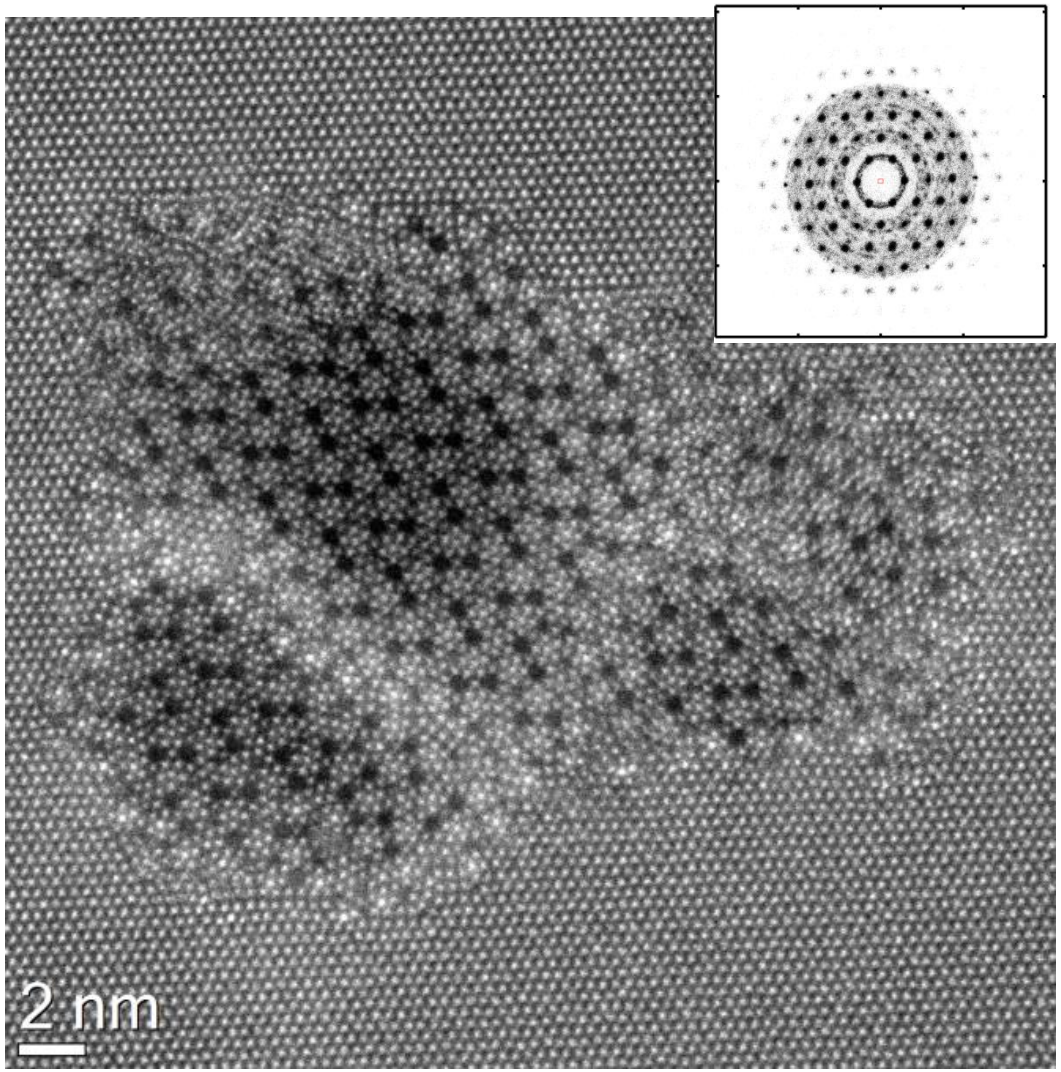
Indexes 1,2,3,4 are chosen in the same sense for all atoms (generalization for different lattice and/or next coordination sphere obvious)

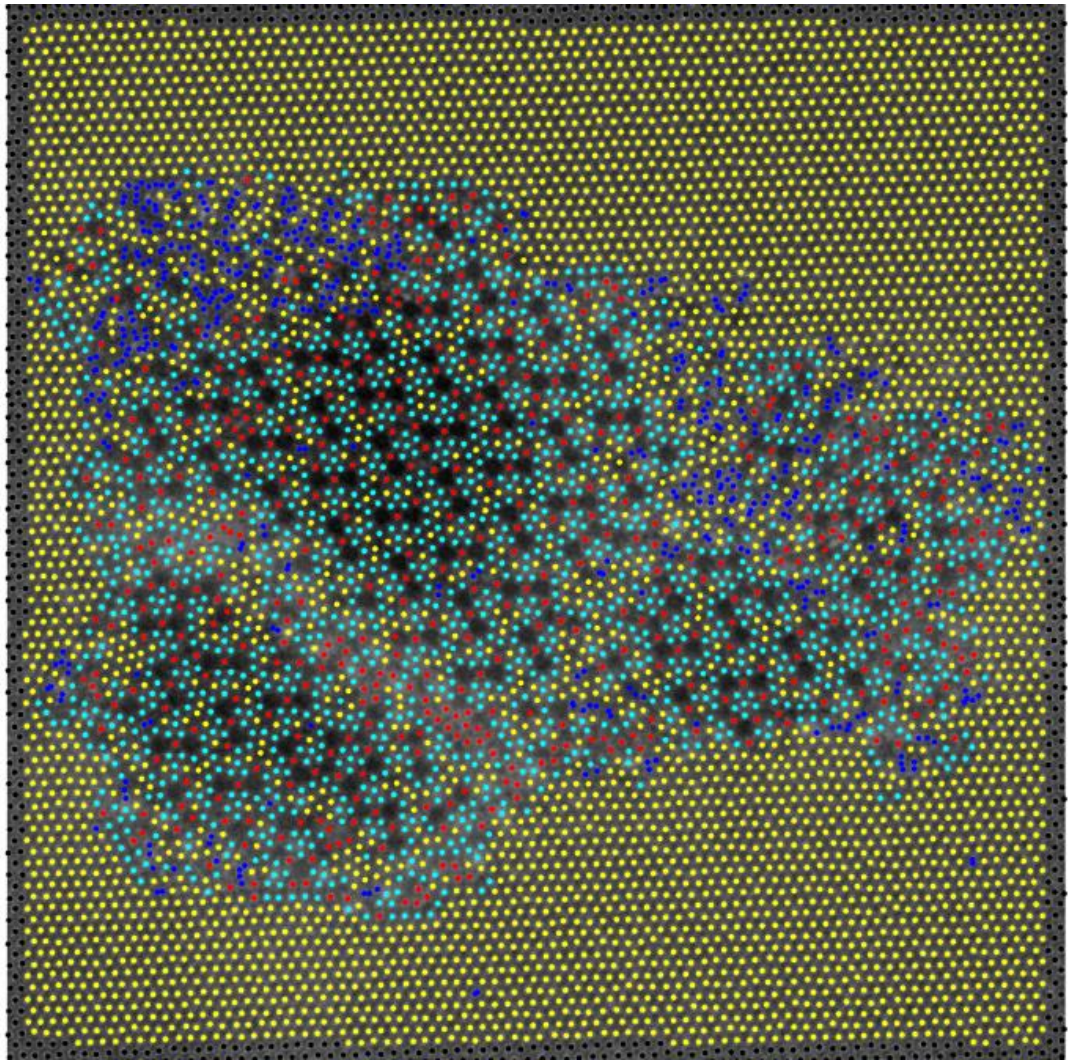
Then, phase/ferroic variant identification problem can be reduced to finding equivalent (in statistical sense) groups of nearest neighbors

We can also use group theory to make hypotheses, e.g. add translation symmetry operations, i.e. i -> i+1 and j -> j+1 for lattice doubling)

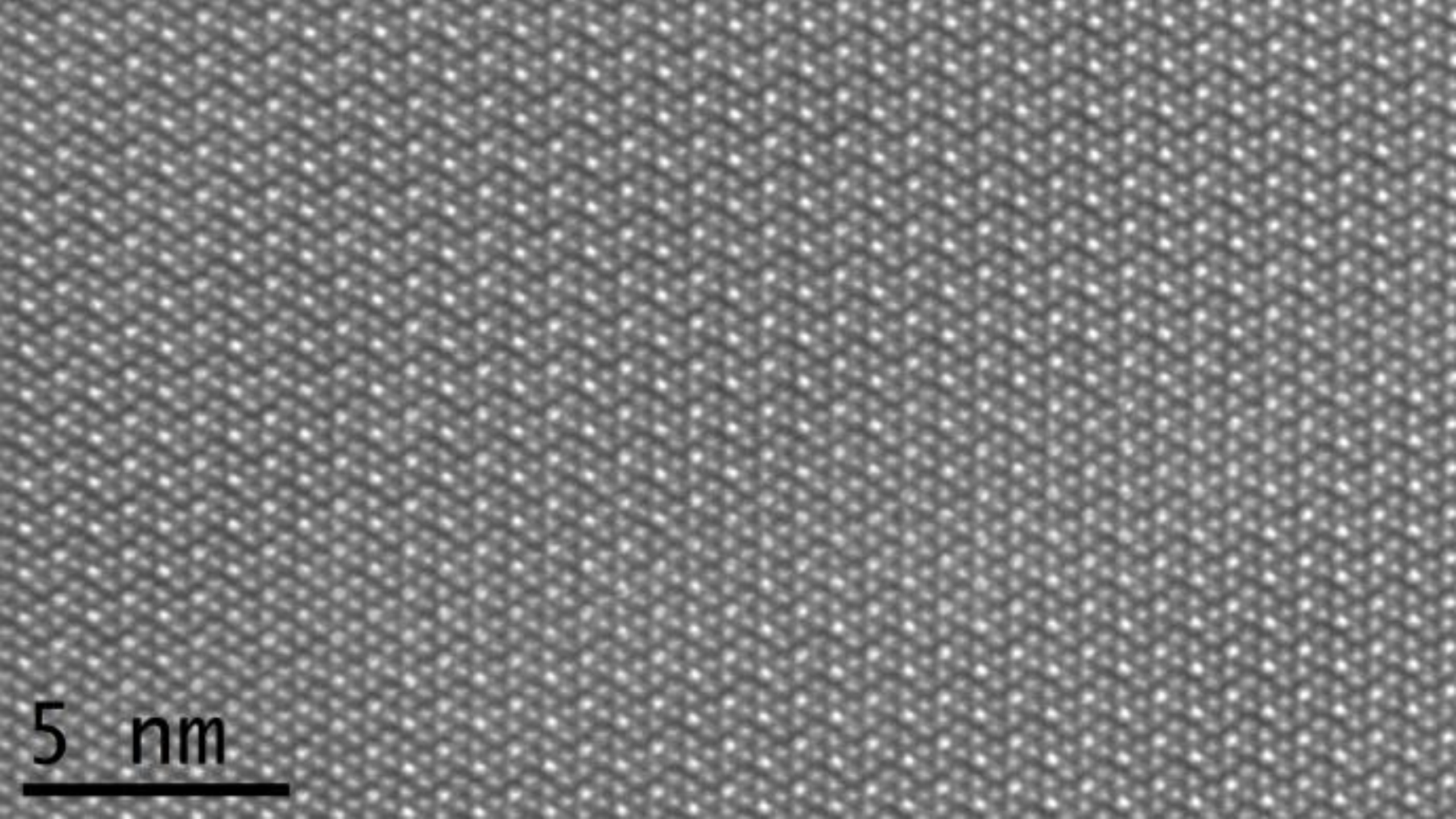


Local crystallography: k-means



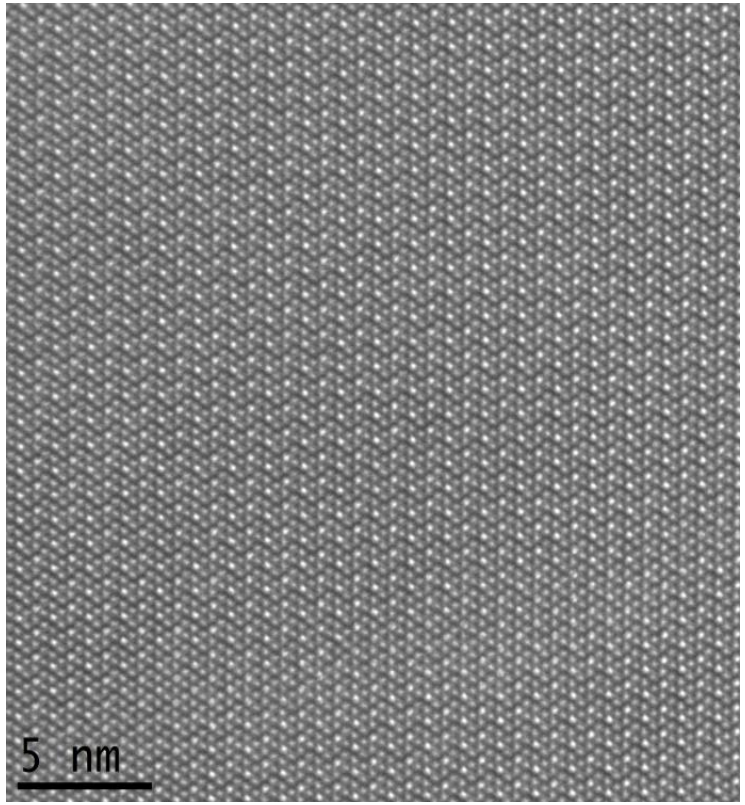


A. BELIANINOV, Q. HE, M. KRAVCHENKO, S. JESSE, A. BORISEVICH, and S.V. KALININ, *Identification of phases, symmetries, and defects through local crystallography*, Nat. Comm. **6**, 7801 (2015).

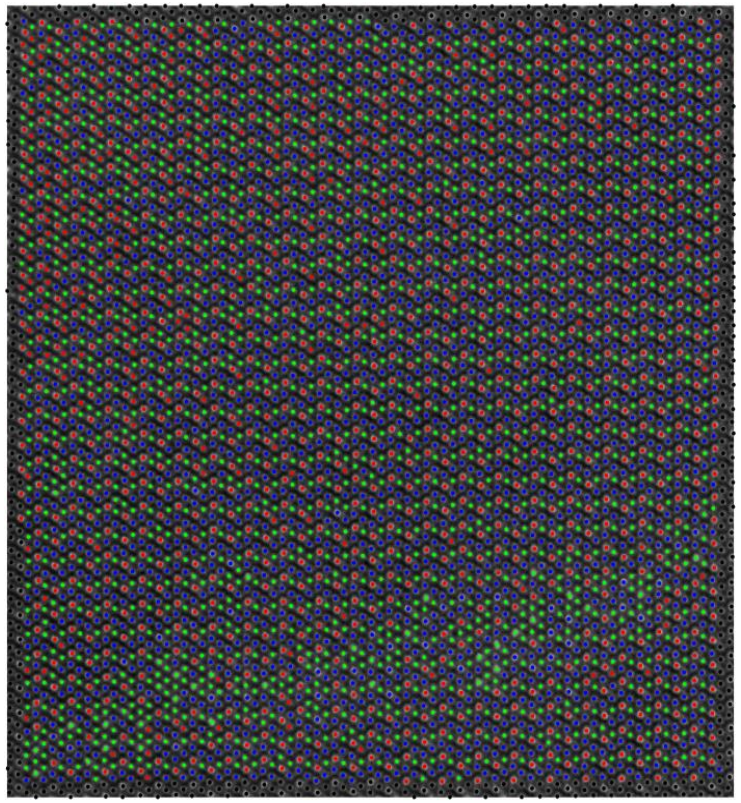

5 nm

Local crystallography

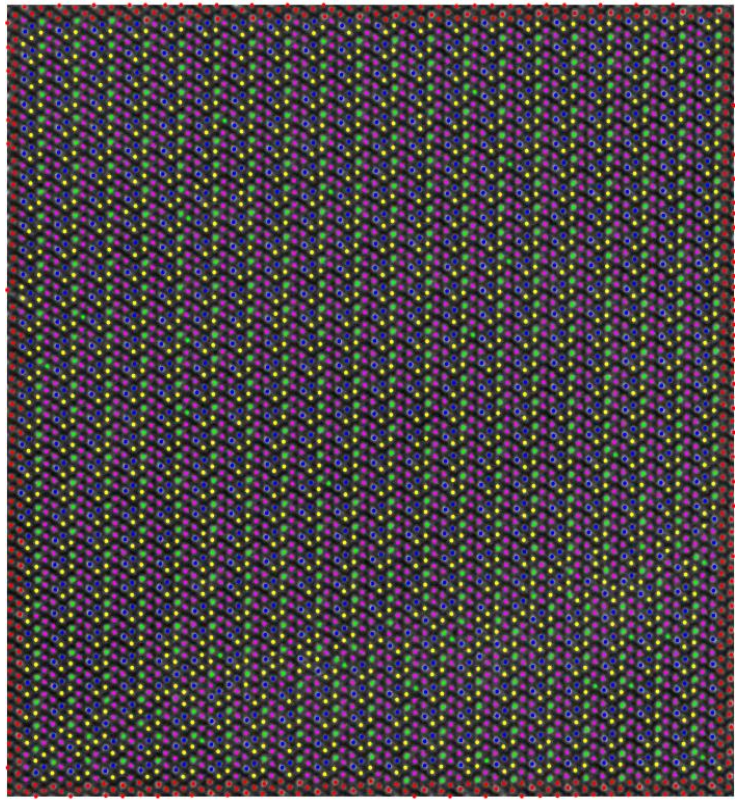
Image



K-means full vector



K-means angles



Normal modes: group theory

- **Group Representations** map group elements onto matrices, ensuring matrix multiplication aligns with the group operation.
- Molecules have **symmetries** defined by point groups linked to symmetry operations, such as rotations.
- Molecules have various **vibrational modes** with specific symmetries. Using the molecule's point group, one can deduce which modes are spectroscopically active.
- **Selection Rules:** Group representations dictate which vibrational modes appear in techniques like IR or Raman. Some modes may be IR-active but not Raman-active based on symmetry.
- **Vibrational frequencies** indicate energy differences between vibrational levels that are connected to group representations

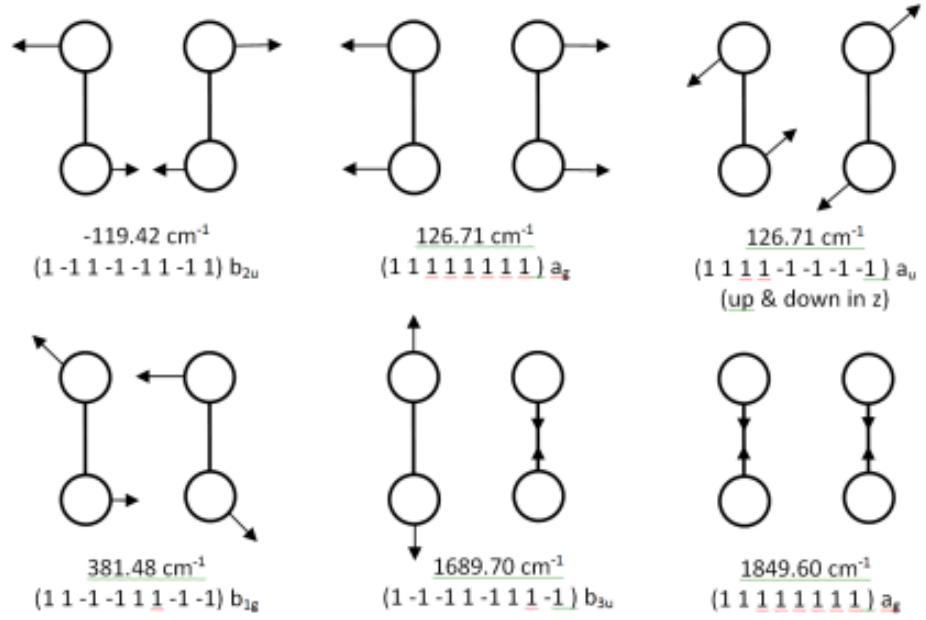


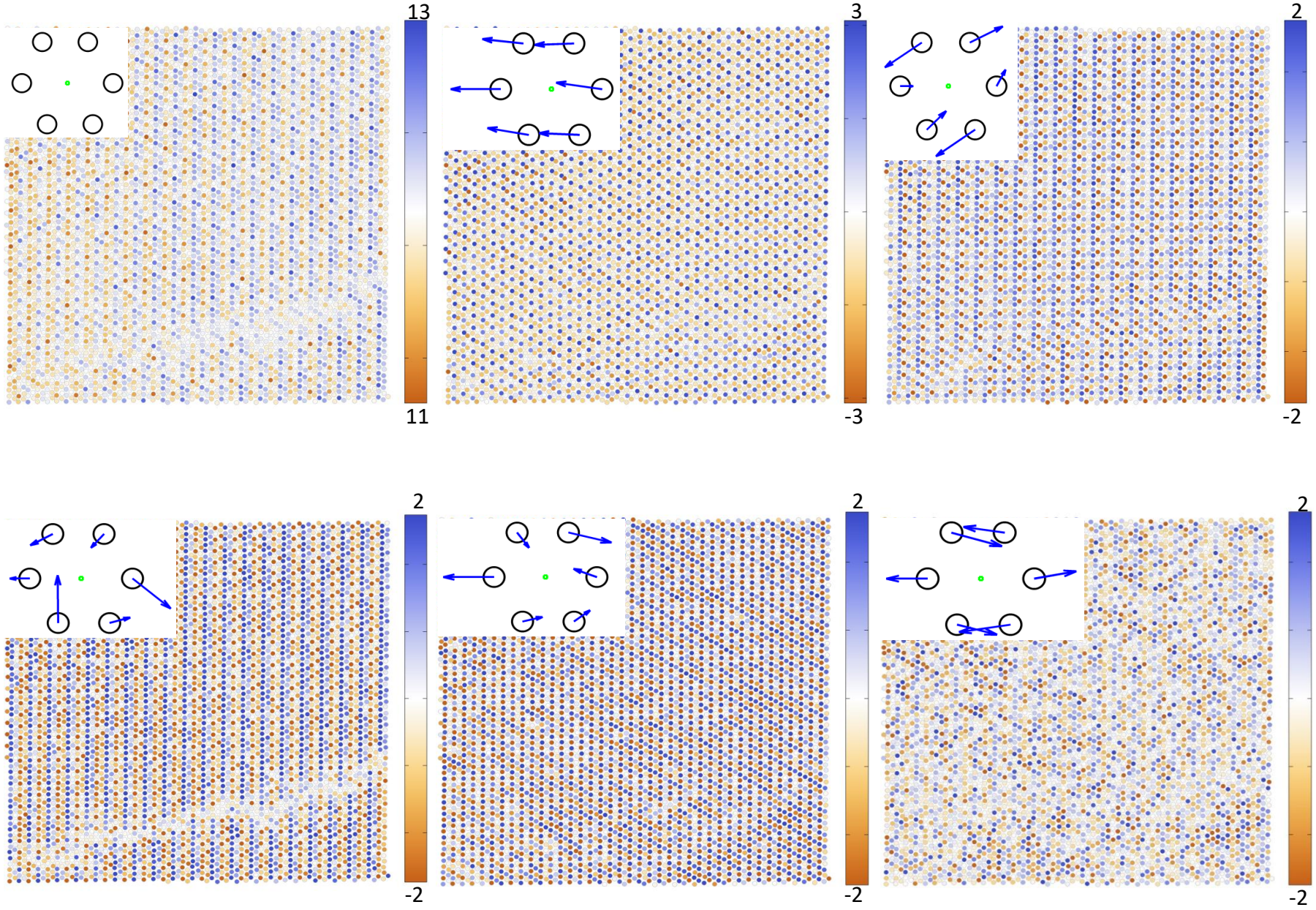
Figure 2: Sketches of Normal Modes of O_4^+

Table 2: Character Table for Point Group D_{2h}

D_{2h}	E	$C_2(z)$	$C_2(y)$	$C_2(x)$	i	$\sigma(xy)$	$\sigma(xz)$	$\sigma(yz)$		
A_g	1	1	1	1	1	1	1	1		x^2, y^2, z^2
B_{1g}	1	1	-1	-1	1	1	-1	-1	R_z	xy
B_{2g}	1	-1	1	-1	1	-1	1	-1	R_y	xz
B_{3g}	1	-1	-1	1	1	-1	-1	1	R_x	yz
A_u	1	1	1	1	-1	-1	-1	-1		
B_{1u}	1	1	-1	-1	-1	-1	1	1	z	
B_{2u}	1	-1	1	-1	-1	1	-1	1	y	
B_{3u}	1	-1	-1	1	-1	1	1	-1	x	

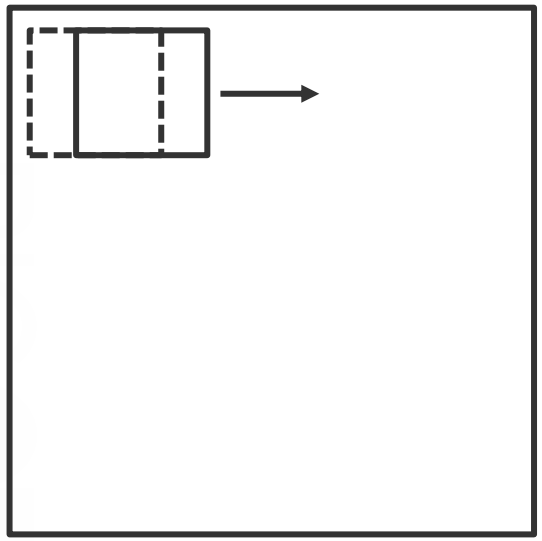
http://vergil.chemistry.gatech.edu/notes/grpthy-vib/grpthy-vib.pdf

Local crystallography: PCA

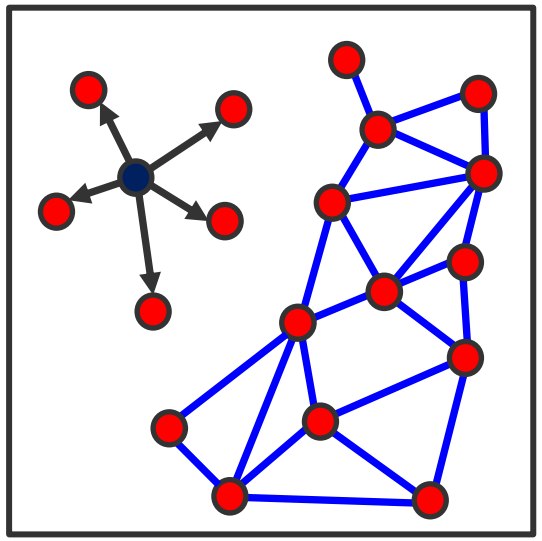


Constructing the descriptors

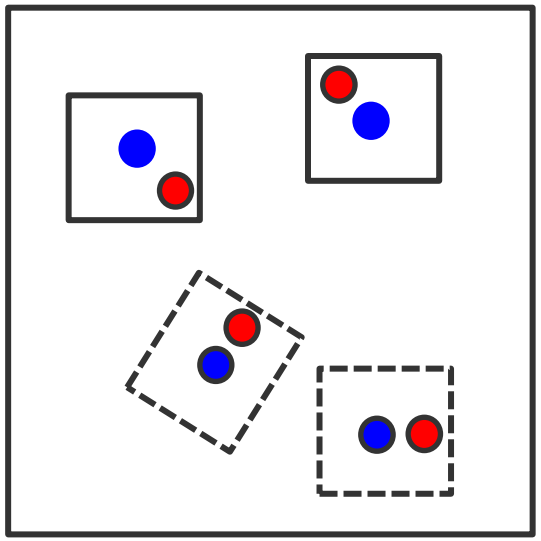
Continuous translational symmetry



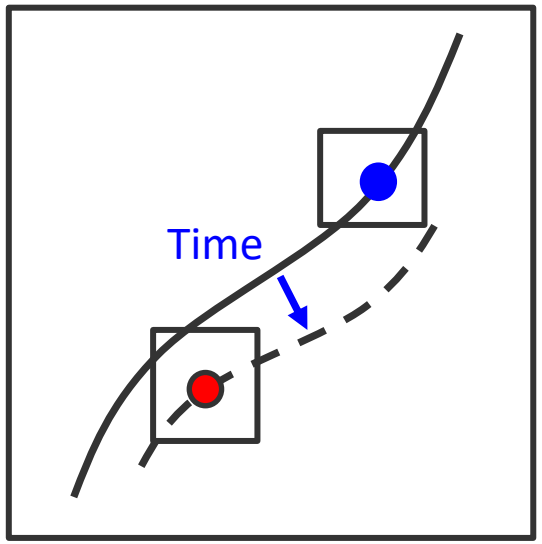
Atom based descriptions



Localized sub-images



Time-delayed descriptors

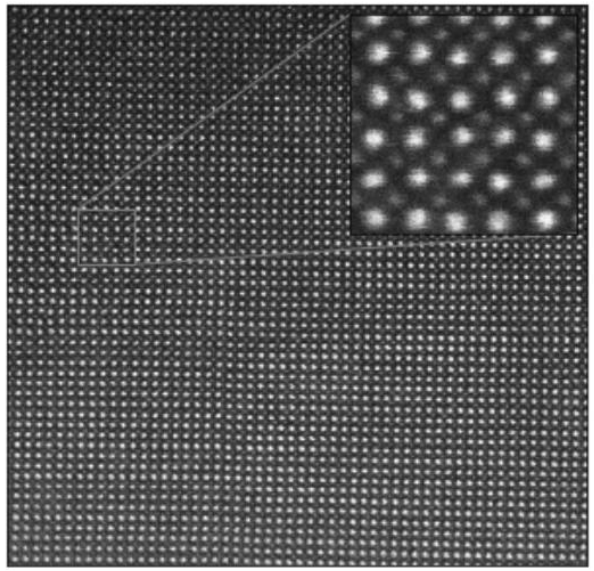


The choice of the descriptor:

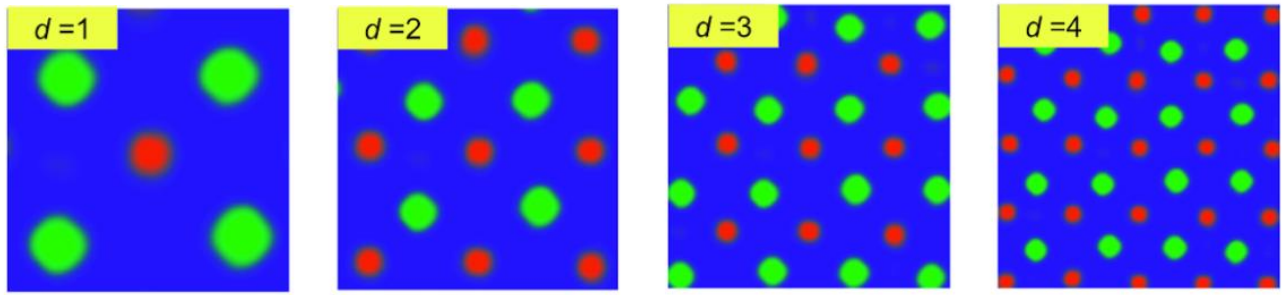
- Defines physical inferential biases and allows to introduce prior knowledge
- Determines the physical meaning of the analysis
- Establishes the analysis pipeline

Local crystallography: FerroNET

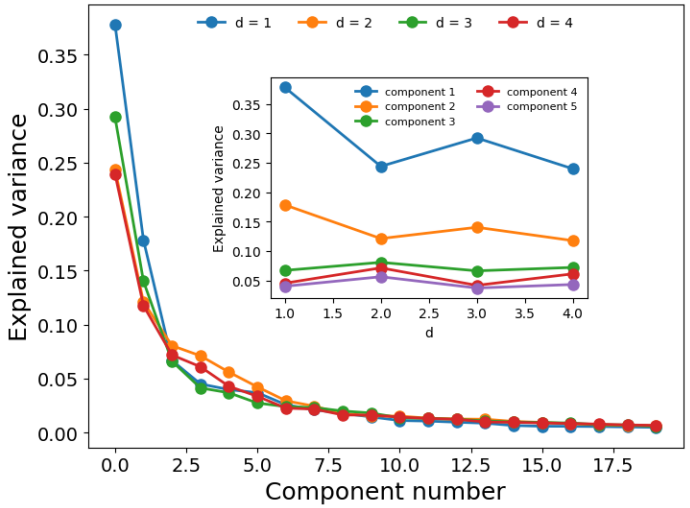
Experimental (LBFO film)



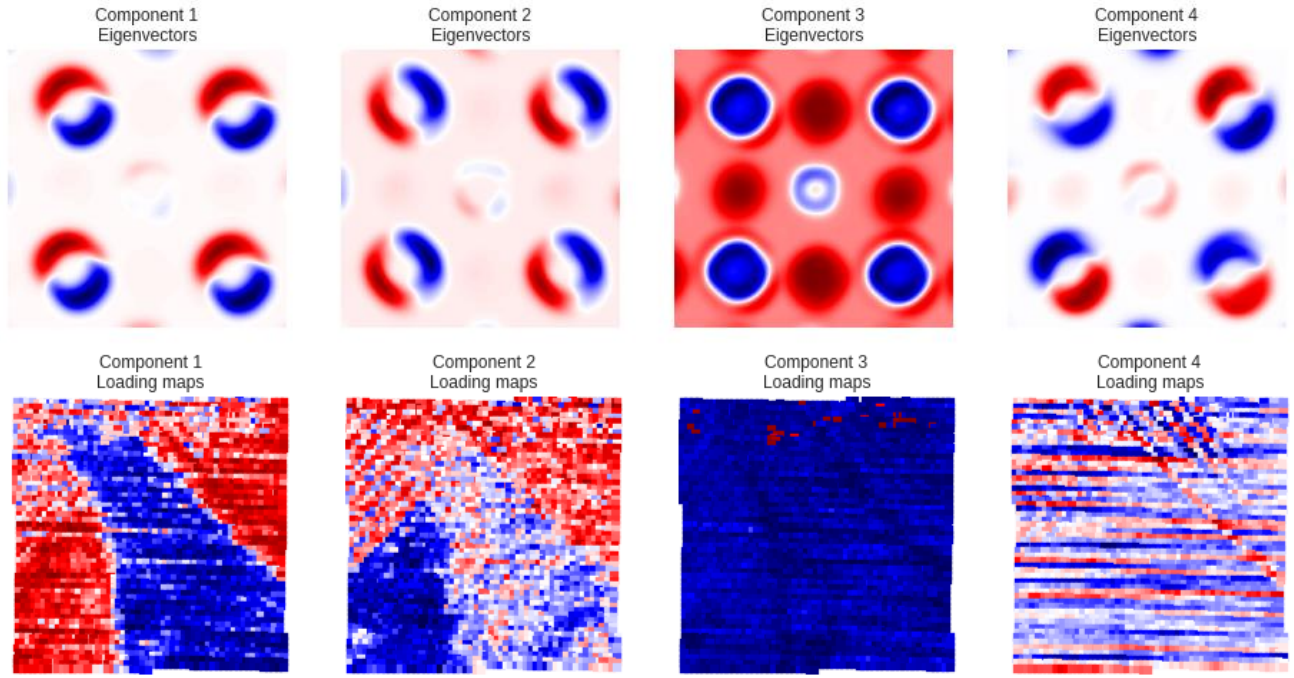
Building blocks (from neural network output)



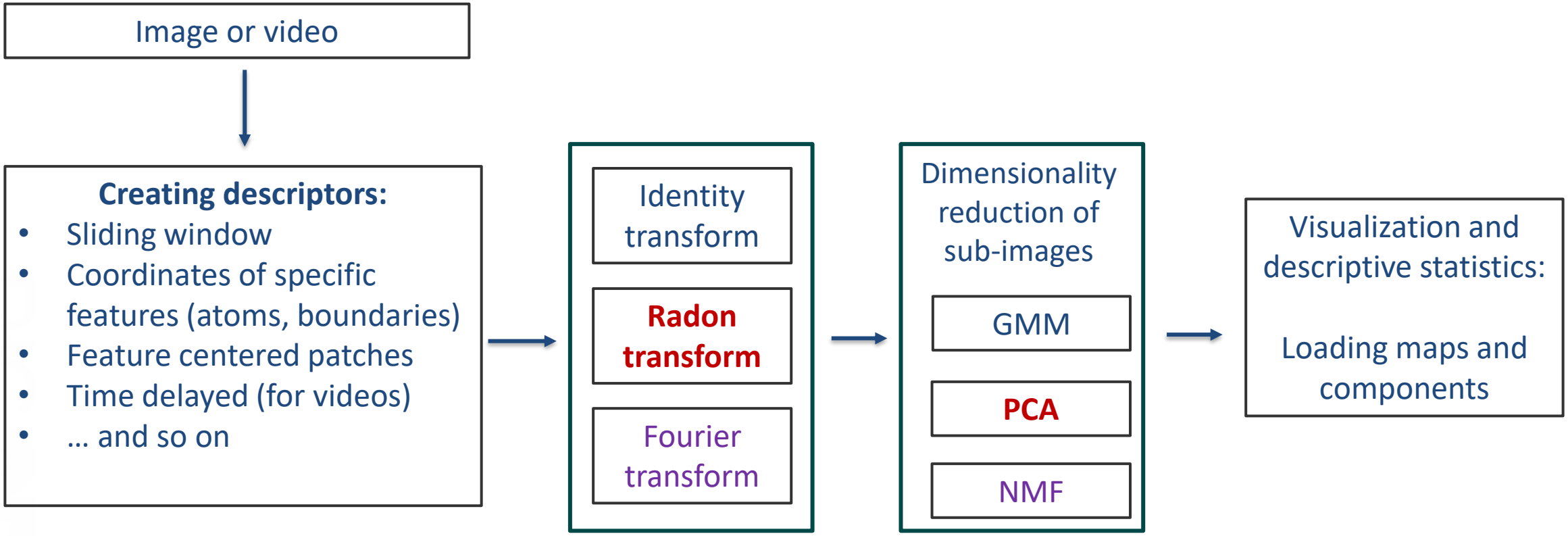
Information Content



PCA eigenvectors and loading maps



Example of analysis pipeline



Pipelines are defined to

- Make analysis traceable, repeatable, explainable, and transferable
- Allow for hyperparameter tuning and optimization
- Efficiently use the memory

How general should you be: depends on applications

Day 3_FerroicBlocks_mockup_paper_v3a.ipynb