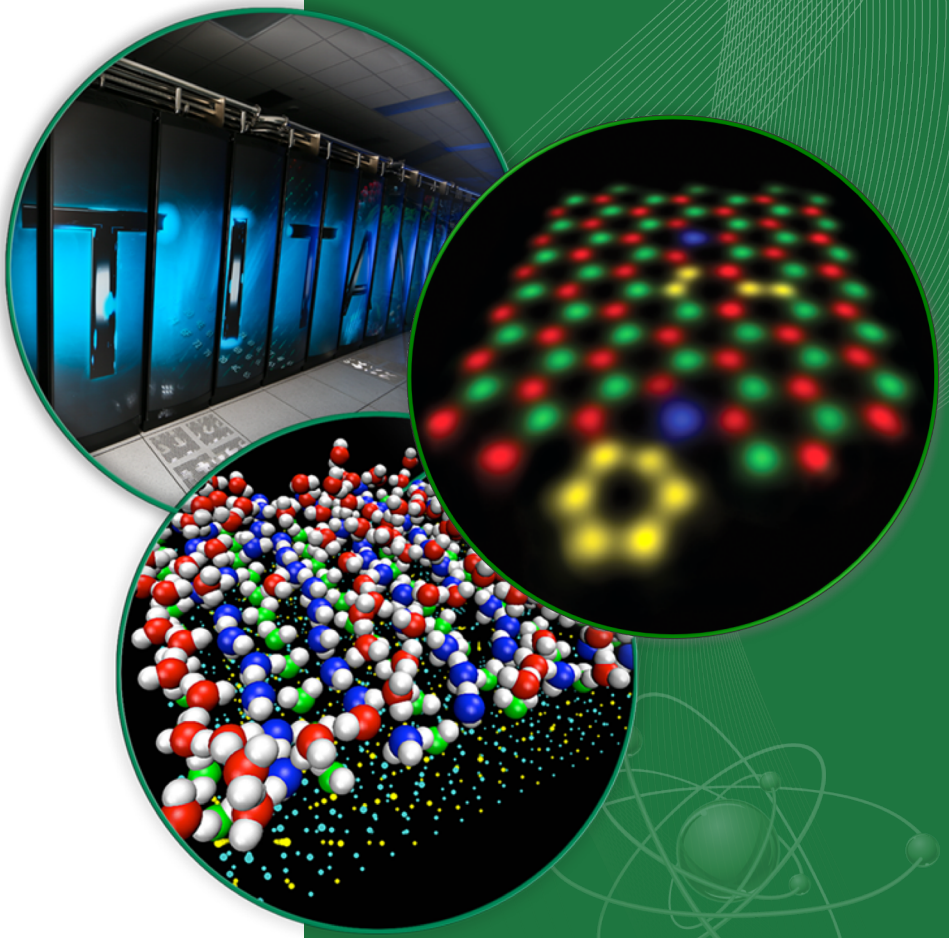
X16 - Multivariate Methods and Image-processing for Quantitative Microscopy

Spectral Unmixing Methods



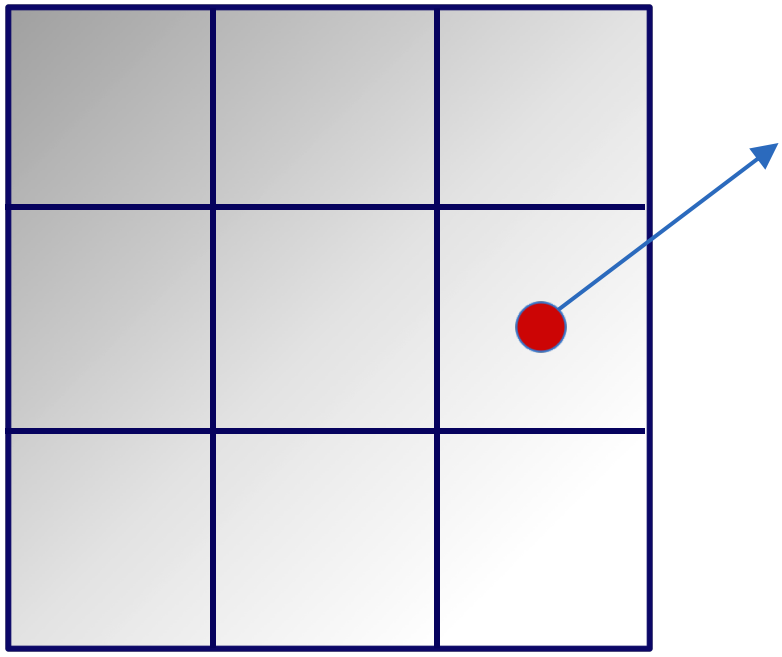
Rama Vasudevan

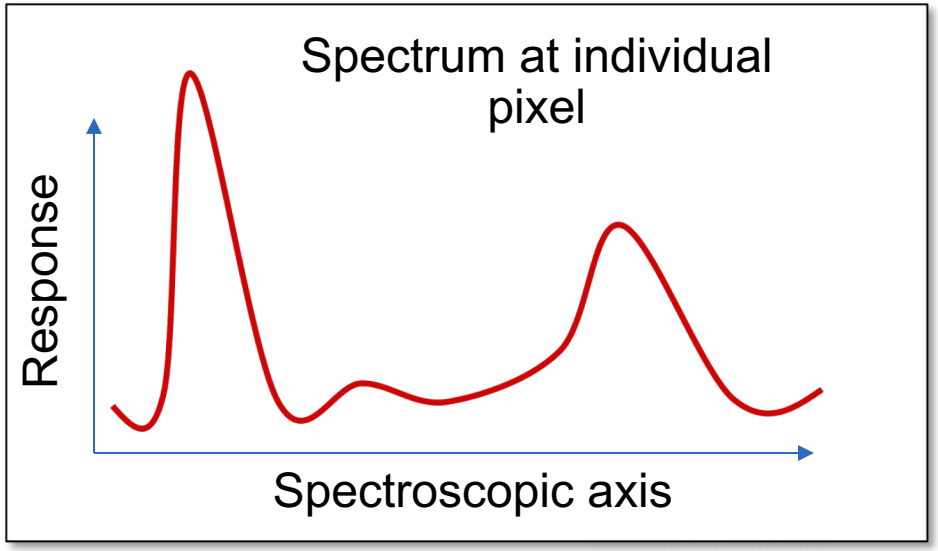
CNMS User Meeting Workshop
Oak Ridge National Laboratory
August 13th 2018

Spectral Data

- Many imaging modalities have spectroscopy components
- By capturing spectra across an image, it allows us to correlate features of the spectra with specific regions of the sample

Image (MxN pixels)



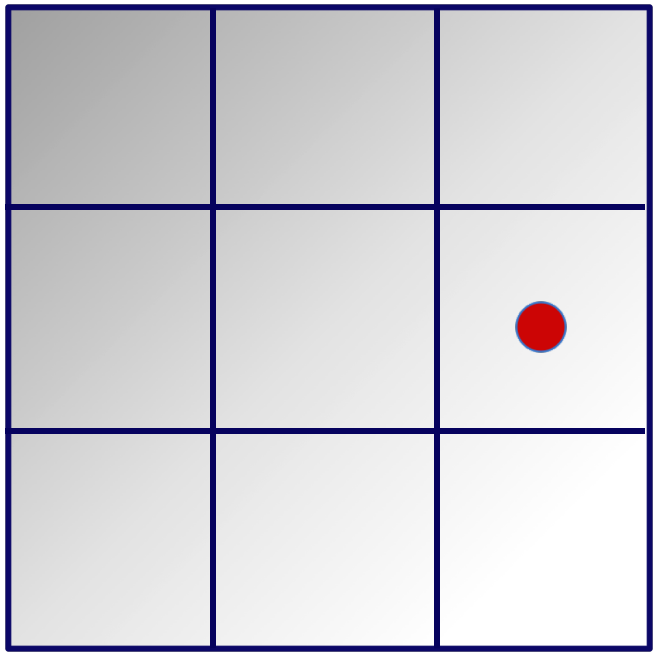


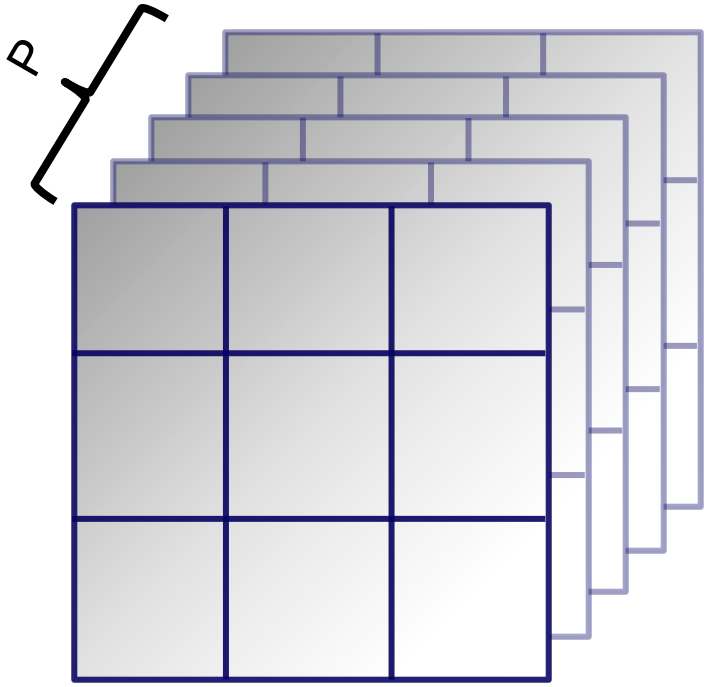
M (rows) x N (columns) x P (spectral points)

Spectral Data

- Many imaging modalities have spectroscopy components
- By capturing spectra across an image, it allows us to correlate features of the spectra with specific regions of the sample

Image (MxN pixels)





(Repeat MxN matrix P times to make 3D cube)

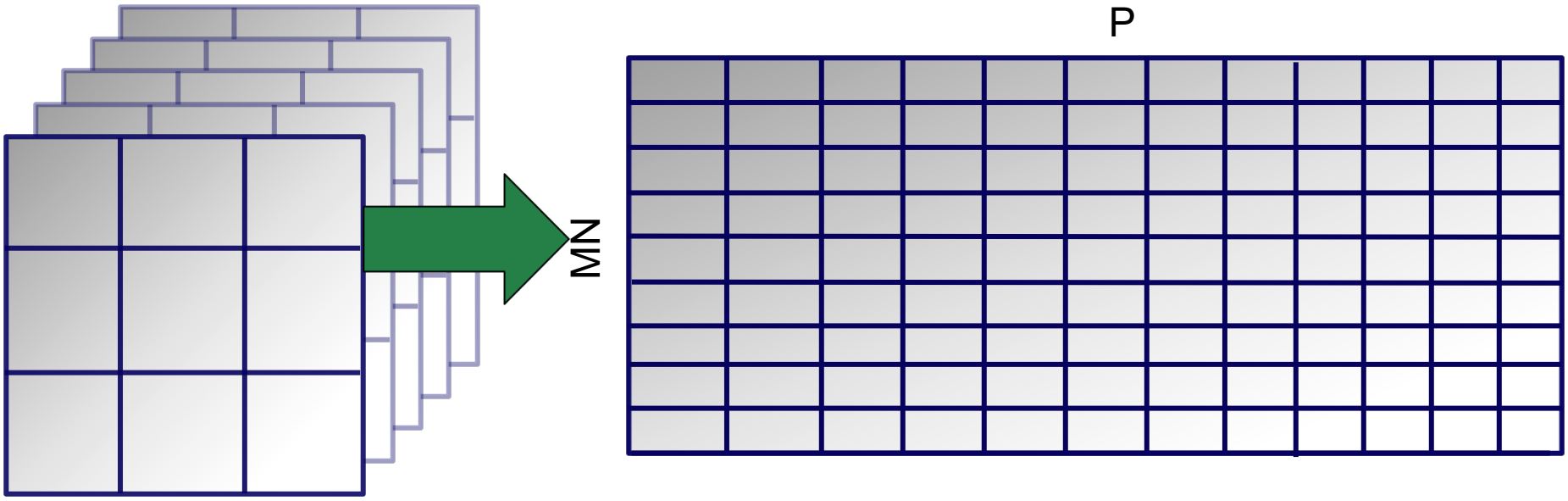
Spectral Data

- Examples include EELS, micro Raman imaging, force-distance curve maps in AFM, Scanning tunneling spectroscopy, etc.
- Usually, the 3D matrix is converted to a 2D matrix

M (rows) x N (columns) x P (spectral points)

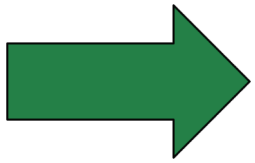
= (MN) x P

Pixels x spectral points



Matrix Factorization

- We now have a 2D matrix (call it **X**) of size (MN,P)
- If **X** is 'small' (e.g., (10,10)) we can comprehend it easily. But if the size of X is (2500, 32), then this becomes more difficult
- The number of spectral points **P** is often termed the "number of dimensions".
- To see trends in our data, we want to visualize them in some manner. But the 3D data cube is big, so what can we do?



Matrix Factorization

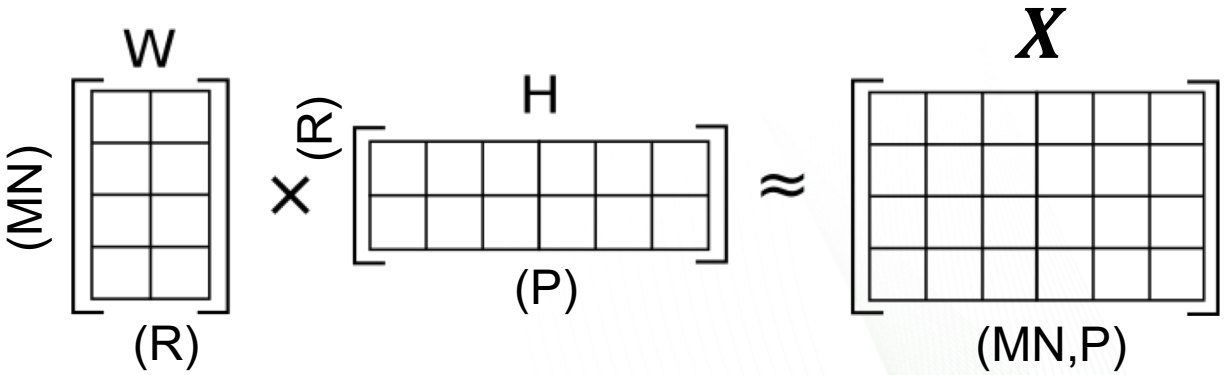
Dimensionality Reduction

- The reason it is difficult to visualize is because there are too many spectral points *P* per pixel. What we need is to reduce the dimensionality, while preserving most of the data structure.
- What if we could write our matrix **X** (size MN,P) as

$$\boldsymbol{X} \approx \boldsymbol{WH}$$

Where W is size (MN,R)
H is size (R,P)

R <P



Dimensionality Reduction

- The reason it is difficult to visualize is because there are too many spectral points *P* per pixel. What we need is to reduce the dimensionality, while preserving most of the data structure.
- Alternative viewpoint: Represent the spectrum at pixel z by a linear expansion

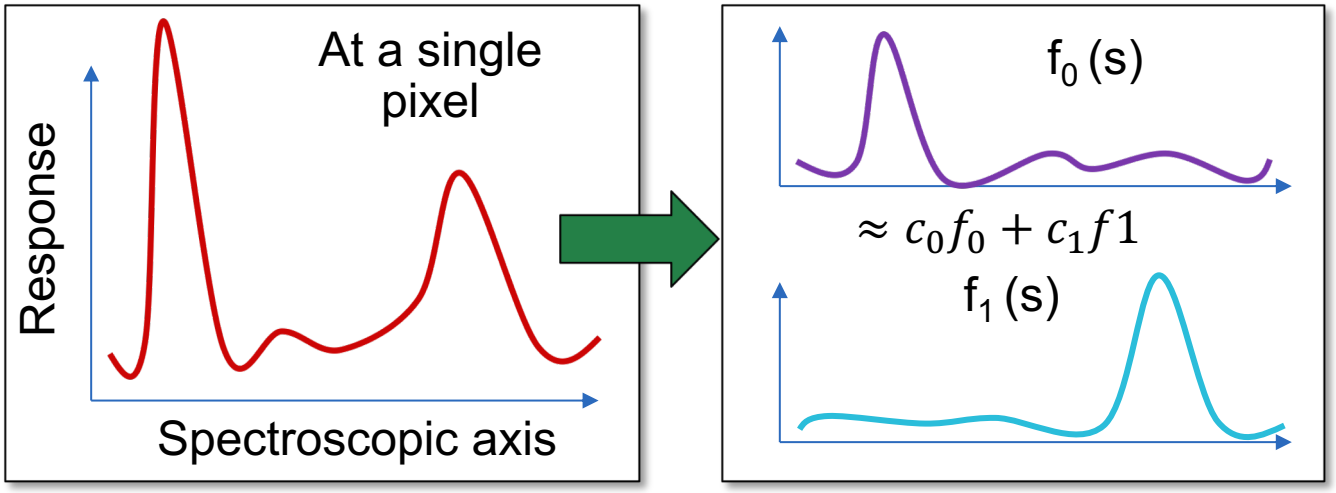
$$f(z_i, s) = c_0(z_i)f_0(s) + c_1(z_i)f_1(s) + \ \ldots + c_n(z)f_n(s)$$

$$f(z, s) = \sum_{i=1}^{MN} c_i(z_i)f_i(s)$$

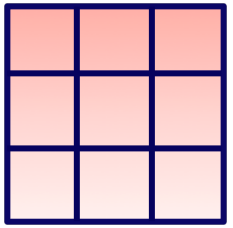
The task boils down to determining the functions $f_n(s)$, and determining the coefficients c_n

Dimensionality Reduction

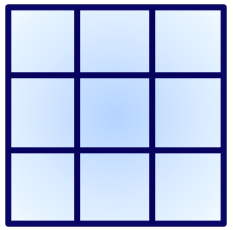
- The reason it is difficult to visualize is because there are too many spectral points P per pixel. What we need is to reduce the dimensionality, while preserving most of the data structure.
- Alternative viewpoint: Represent the spectrum at pixel z by a linear expansion



$c_0(x, y)$



$c_1(x, y)$



(1) Singular Value Decomposition

$$X = USV^T$$

Note: In the full decomposition, r = n. Practically, r<<n

X is of size (m,n)
U is of size (m,r)
S is of size (r,r)
V is of size (n,r)

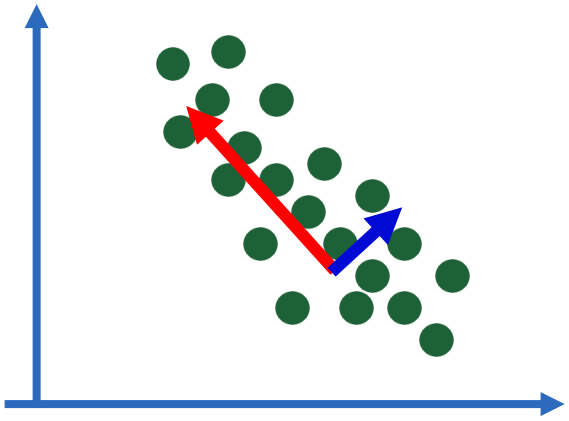
The columns of U,V form the basis
S scales them, in descending order

By the SVD theorem, it is always possible to decompose real matrix **X** into USV^T where

U,S,V: unique
U,V: column orthonormal
- $U^TU = I$, $V^TV = I$
- Columns are orthogonal unit vectors

S: diagonal
- Positive, sorted in descending order

Finding the main 'directions' in the data



(Jupyter Notebook)

Example on using SVD on an EELS dataset, via pycroscopy's do_svd() algorithm.

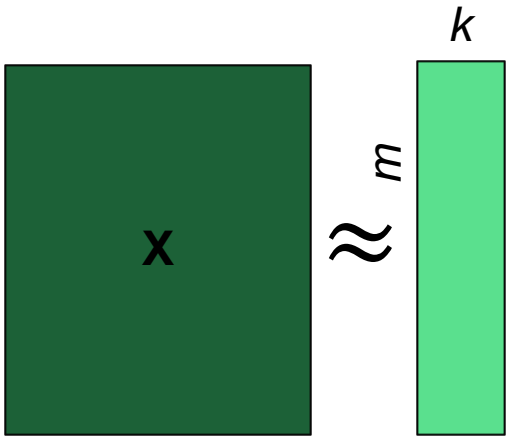
Non-negative Matrix Factorization

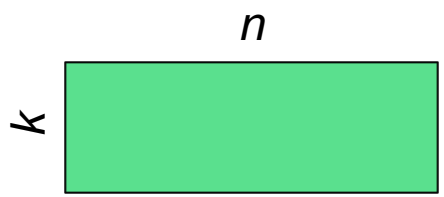
- SVD is an extremely powerful tool to visualize data and observe trends, but it has weaknesses. Mainly, it often produces highly unphysical behavior, for instance negative intensity values in spectra where only positive values are physically realizable
- Therefore, we can turn to other methods, such as non-negative matrix factorization approaches

Matrix Form

$$\boldsymbol{X} \approx \boldsymbol{WH}$$

Here, we enforce that X, W and H are all non-negative matrices





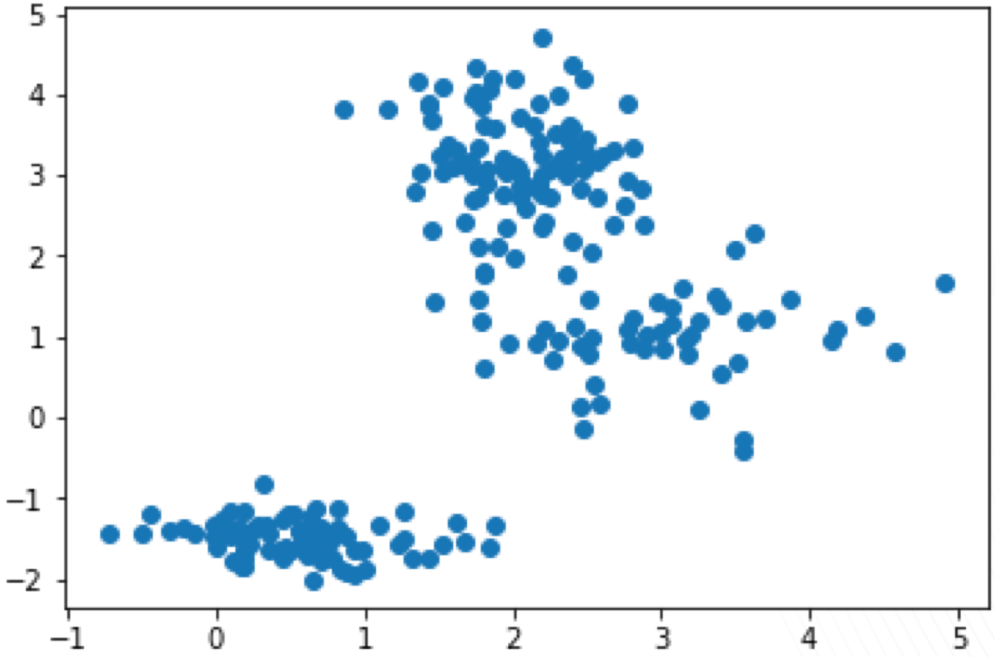
k is called 'low-rank'. The positivity constraint often helps to make the factor matrices more interpretable.

(Jupyter Notebook)

Example on using NMF on an EELS spectroscopic dataset

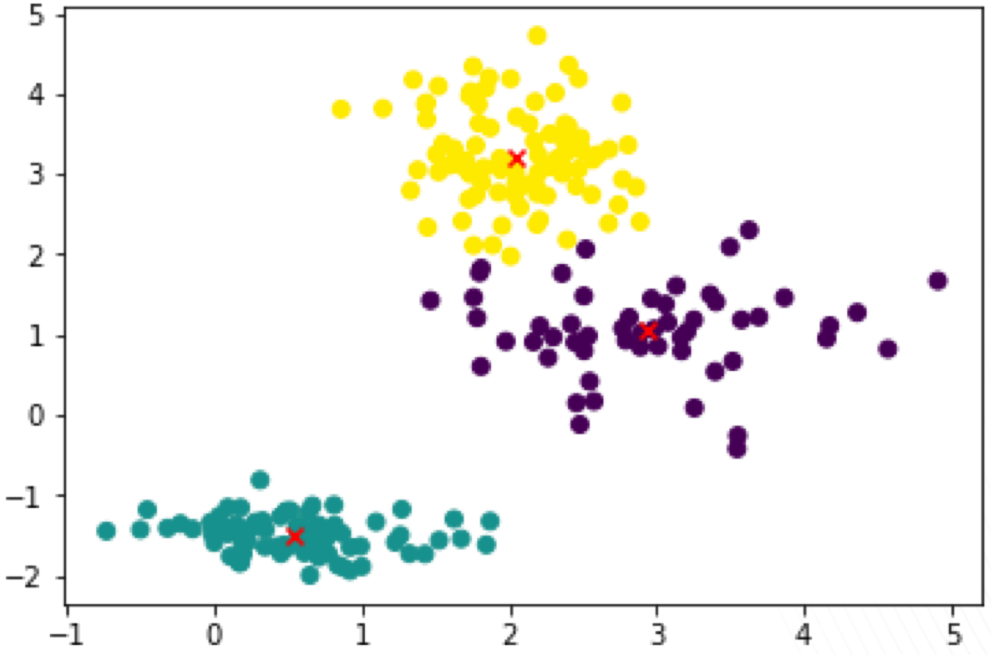
K-Means Clustering

- SVD, NMF and ICA are decompositions. But sometimes, we don't want to decompose our signal, but just group them into 'alike' sets.
- This is termed 'clustering'. The easiest and most widely used method is the k-means algorithm



K-Means Clustering

- SVD, NMF and ICA are decompositions. But sometimes, we don't want to decompose our signal, but just group them into 'alike' sets.
- This is termed 'clustering'. The easiest and most widely used method is the k-means algorithm



K-Means Clustering

- SVD, NMF and ICA are decompositions. But sometimes, we don't want to decompose our signal, but just group them into 'alike' sets.
- This is termed 'clustering'. The easiest and most widely used method is the k-means algorithm

K-means Clustering algorithm, to separate data (x_1,x_2,…,x_n) into *k* clusters

$$\arg\min_{\boldsymbol{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}\in S_i} \left|\left|\mathbf{x} - \boldsymbol{\mu}_i\right|\right|^2$$ where μ_i is the mean of points in S_i

(Determine **S** = {S_1,S_2,…S_k}, such that within cluster sum of squares is minimized)

(Jupyter Notebook)

Example on using Kmeans on a EELS Dataset.