M4: Unsupervised ML methods – Immediate applications of (spectral) linear algebra

Manikandan Narayanan

Week 8 (Sep 15-, 2025)

PRML Jul-Nov 2025 (Grads Section)

Acknowledgment of Sources

- Slides based on content from related
  - Courses:
    - IITM – Profs. Arun/Harish[HR]/Chandra[CC]/Prashanth's PRML offerings (slides, quizzes, notes, etc.), Prof. Ravi's "Intro to ML" slides – cited (e.g., **[HR]/[HG]**) in the bottom right of a slide.
    - India – NPTEL PR course by IISc Prof. PS. Sastry (slides, etc.) – cited as [PSS] in the bottom right of a slide.
  - Books:
    - PRML by **Bishop**. (content, figures, slides, etc.) – cited as **[CMB]**
    - Pattern Classification by Duda, Hart and Stork. (content, figures, etc.) – [DHS]
    - Mathematics for ML by Deisenroth, Faisal and Ong. (content, figures, etc.) – [DFO]
    - Foundations of ML by Mohri, Rostamizadeh, and Talwalkar (content, figures, slides by Mohri, etc.). – [MRT]
    - Information Theory, Inference and Learning Algorithms by MacKay – [DJM]

Outline for Module M4

- **M4.0 Context/Introduction**
- M4.1 Spectral clustering algorithm
  - M4.1.1 From data to similarity graph
  - M4.1.2 From graph connectivity to clusters
- M4.2 Conclusion & next steps

Context/Intro.: Two unsupervised ML problems

- Unsupervised ML: Recognize patterns in the dataset (set of n data points in $\mathbb{R}^d$) without any labels on the data points.

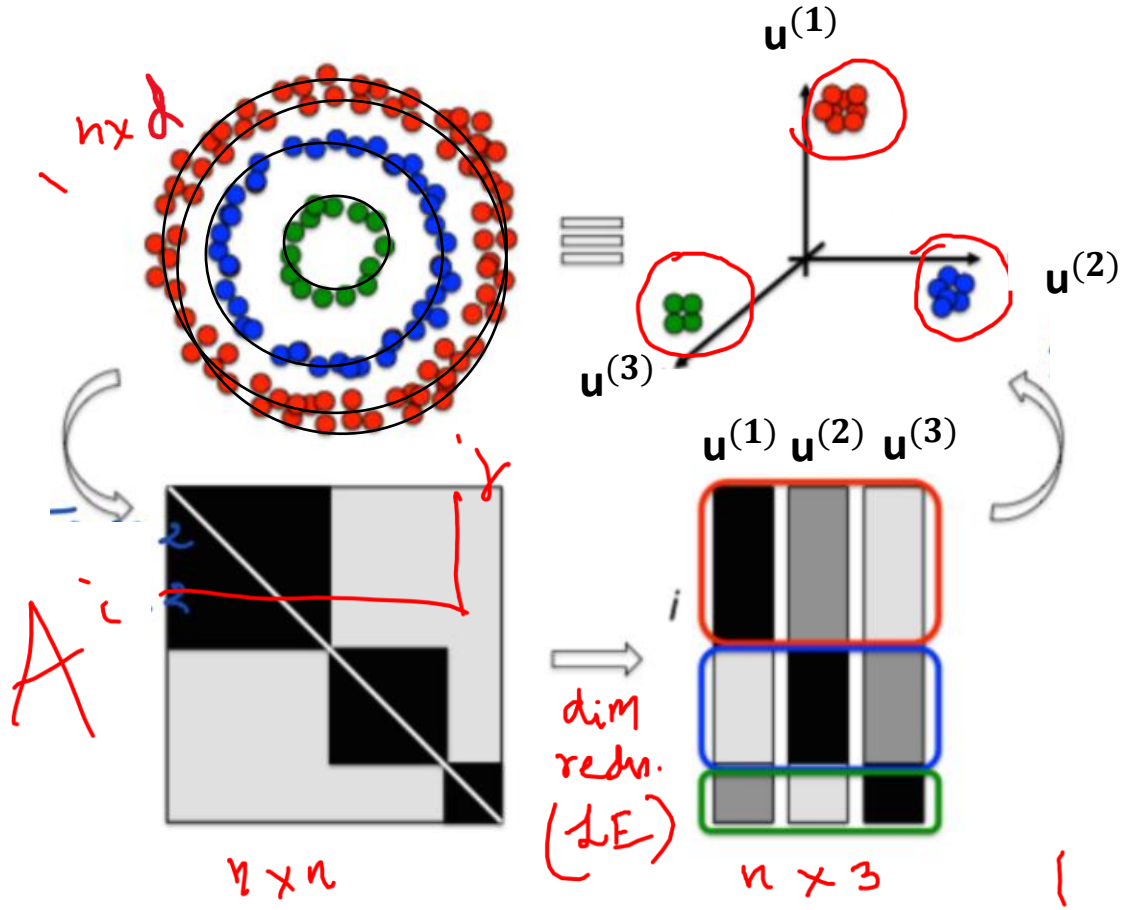
- Dimensionality reduction:
  - Transforms data points from high to low dimensions without much loss of "information", assuming the data points lie "effectively in/close to" a low-dim. manifold of the original space.
  - Many approaches possible: **PCA**, t-SNE, UMAP, **Laplacian Eigenmaps (spectral)**, etc.

- Clustering:
  - Grouping n objects into k clusters based on their similarity
  - Again, many approaches possible: **k-means**, hierarchical, **spectral**, etc.

Outline for Module M4

- M4.0 Context/Introduction
- **M4.1 Spectral clustering algorithm**
  - **M4.1.1 From data to similarity graph**
  - M4.1.2 From graph connectivity to clusters
- M4.2 Conclusion & next steps

Spectral clustering == dim. redn. + kmeans



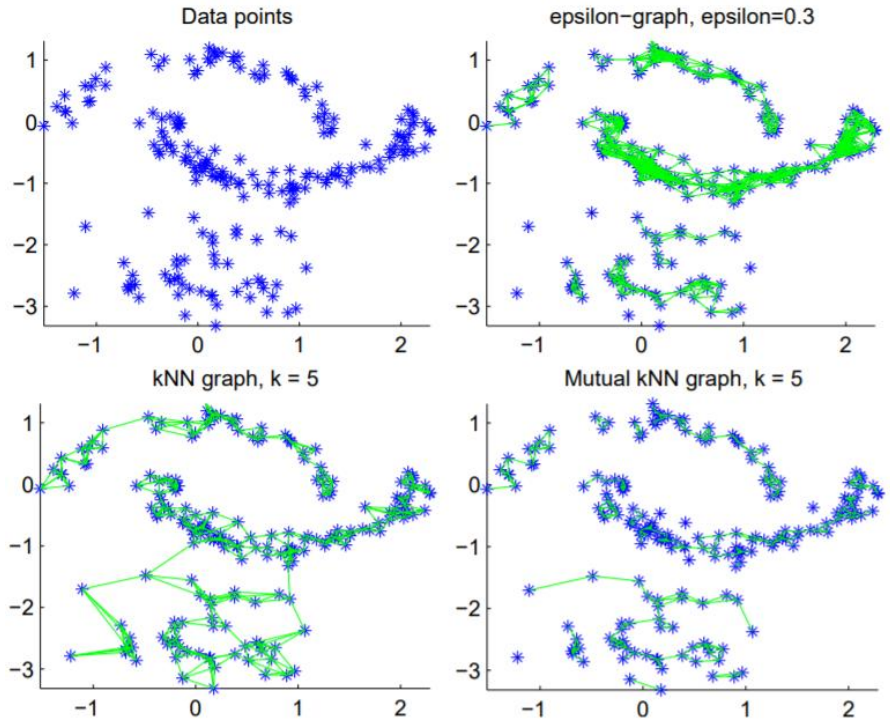
[See Appendix for defn. of convex cluster]

[HR]

Step 1: From (n x d) data matrix to (n x n) similarity-graph's adj. matrix!

- Similarity function: $s(x_i, x_j) = \exp(-\|x_i - x_j\|^2/(2\sigma^2))$

- Similarity graph: complete (all-pairs) graph, or sparse graphs

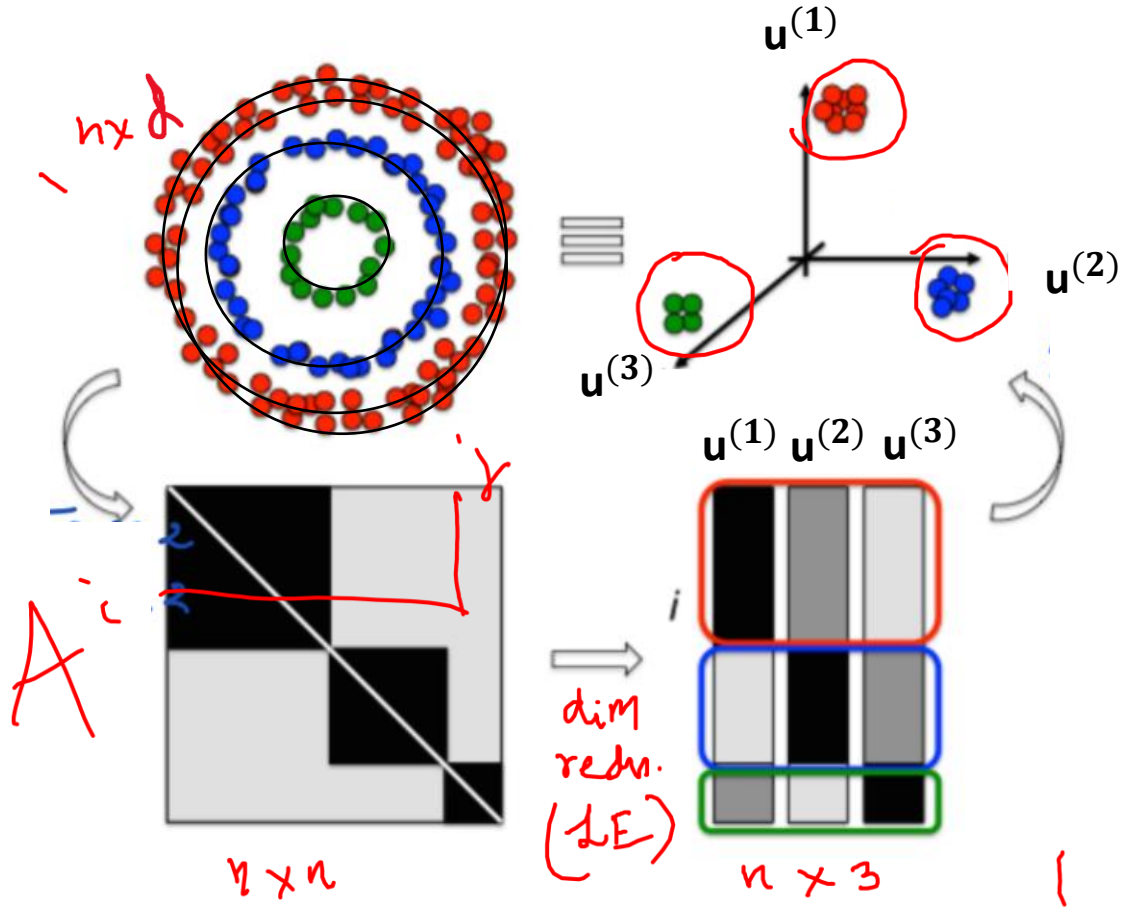


[Luxburg. Tutorial on spectral clustering -- https://arxiv.org/pdf/0711.0189.pdf]

Outline for Module M4

- M4.0 Context/Introduction
- **M4.1 Spectral clustering algorithm**
  - M4.1.1 From data to similarity graph
  - **M4.1.2 From graph connectivity to clusters**
- M4.2 Conclusion & next steps

Spectral clustering == dim. redn. + kmeans



[See Appendix for defn. of convex cluster]

[HR]

Step 2. Spectral clustering – problem defn. using graph connectivity (i.e., adjacency matrix A)

Input: $A \in \mathbb{R}_{+}^{n \times n}$: A_{ij} = Similarity between datapoint i & j

≥ 0

Output: $u \in \mathbb{R}^{n}$: A score assignment to all nodes.

cl-2 [o o o b b b]

cl-1 [a a a o o o]

~~Obj. fn: $L(u) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} (u_i - u_j)^2$

When is $L(u)$ small ?

e.g.

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

u

clust1 clust2

[a a a b b b]

X [a a a a a a]

~Obj. fn. (constrained): $\min_{u} L(u)$ s.t. $u^T u = 1$

$D = \begin{bmatrix} 3 & & 0 \\ & 3 & \\ 0 & & \ddots \end{bmatrix}$

[HR]

Laplacian Matrix (L=D-A)

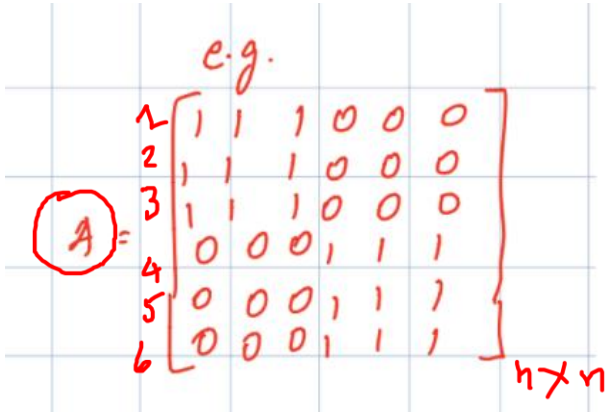
$$L(u) = \sum_{i=1}^{n} \sum_{j=1}^{n} u_i u_j (-A_{ij}) + \sum_{i=1}^{n} u_i^2 \sum_{j=1}^{n} A_{ij}$$

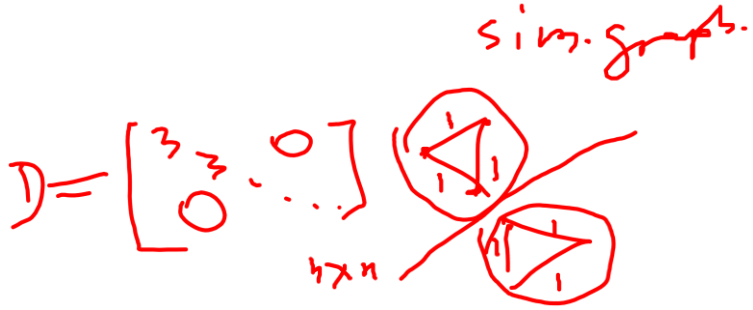
$$= u^T (D - A) u \qquad \text{where } D_{ii} = \sum_j A_{ij}$$

Laplacian matrix $L = D - A$

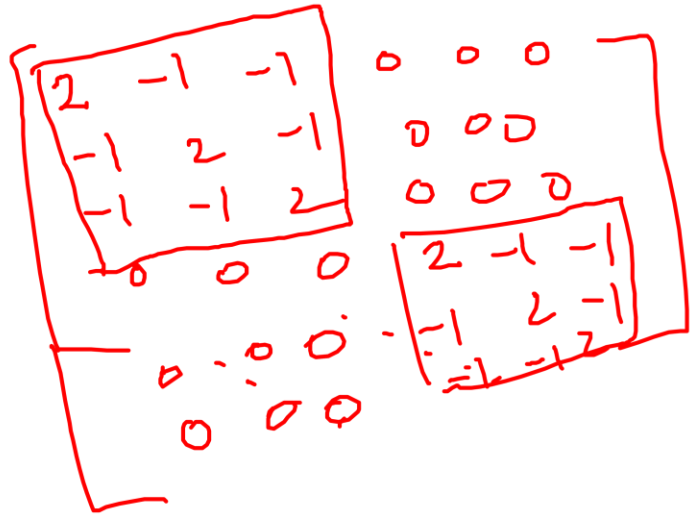
[HR]

Blank space: Example Graph Laplacian





$L = D - A =$



Background on Constrained Optimization (i): Lagrange multiplier (single constraint)

Problem: opt (min or max) $f(\boldsymbol{x})$ $\quad (\boldsymbol{x} \in \mathbb{R}^n, f: \mathbb{R}^n \to \mathbb{R})$

s.t. constraint $g(\boldsymbol{x}) = c$

FONC (First-order necessary conditions): Instead of setting gradient of $f(\boldsymbol{x})$ to zero as in unconstrained optima, set gradient of Lagrangian function $\mathcal{L}(\boldsymbol{x}, \lambda)$ to zero for constrained optima!

$$\nabla \mathcal{L}(\boldsymbol{x}, \lambda) = 0, \qquad \text{where } \mathcal{L}(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) + \lambda\big(\, c - g(\boldsymbol{x})\big); \;\; \lambda \in \mathbb{R}.$$

Opt. strategy: Equivalently, we've these (n+1) equations over (n+1) unknowns to solve, and inspect each solution to find the constrained maxima or minima.

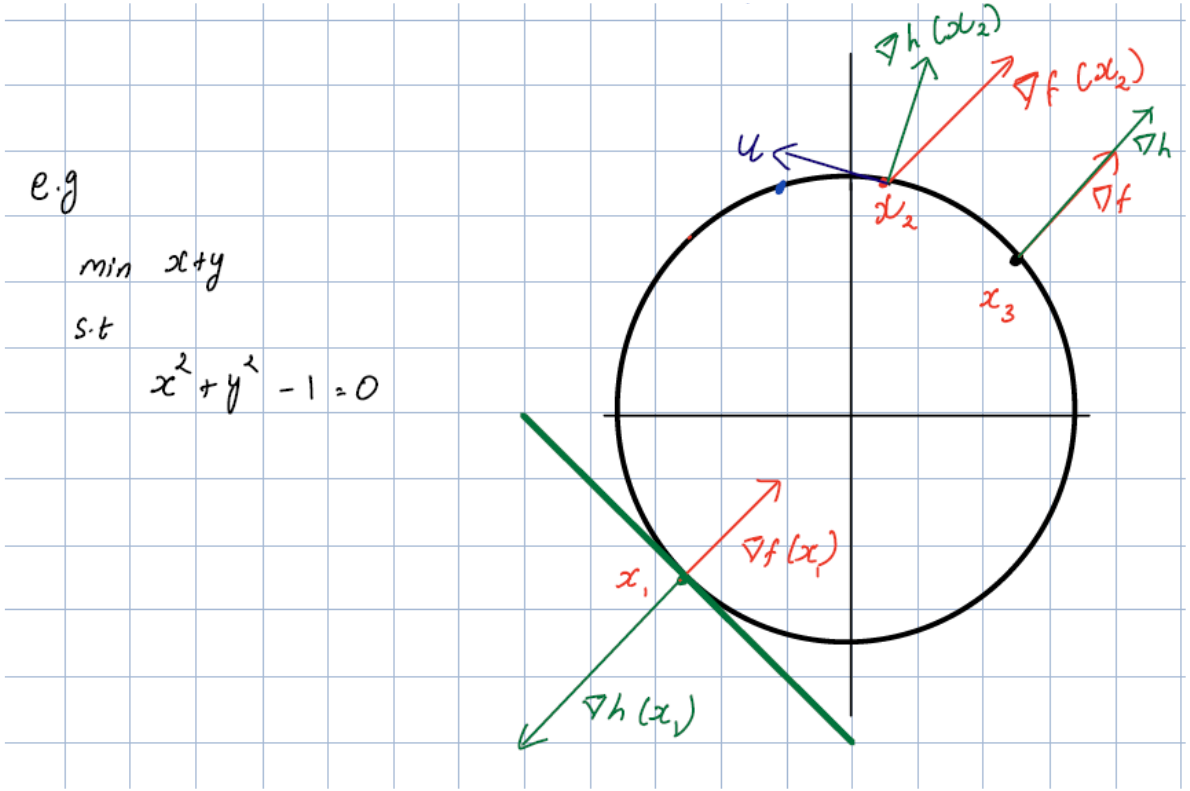
$$\nabla f(\boldsymbol{x}) = \lambda \nabla g(\boldsymbol{x}) \qquad \text{(n equations)}$$

$$g(\boldsymbol{x}) = c \qquad \text{(1 equation)}$$

Fineprint: For Lagrange multiplier to work, f, g must be smooth (continuously differentiable) and constraint qualification $\nabla g(x^*) \neq 0$ must hold. Otherwise, solution may not exist for the above (n+1) equations!

From Paul's online notes [https://tutorial.math.lamar.edu/classes/calciii/lagrangemultipliers.aspx]

Background on Constrained Optimization (i): Example



[HG]

How do you minimize $L(\boldsymbol{u}^{(1)})$ subject to unit norm constraint?

Minimize $_{\boldsymbol{u}^{(1)} \in \mathbb{R}^n}$ $\boldsymbol{u}^{(1)T} \boldsymbol{L} \boldsymbol{u}^{(1)}$ s.t. $\boldsymbol{u}^{(1)T} \boldsymbol{u}^{(1)} = 1$

Set gradient of Lagrangian: $\boldsymbol{u}^{(1)T} \boldsymbol{L}\, \boldsymbol{u}^{(1)} + \lambda_1 \left(1 - \boldsymbol{u}^{(1)T} \boldsymbol{u}^{(1)}\right)$ to zero to

$\nabla_x (x^T A x) = (A + A^T) x$

$\nabla_x (x^T x) = 2x$

get these eqns: $\boldsymbol{L}\boldsymbol{u}^{(1)} = \lambda_1 \boldsymbol{u}^{(1)}$

$\boldsymbol{u}^{(1)T} \boldsymbol{u}^{(1)} = 1$

Thus, any solution $\boldsymbol{u}^{(1)}$ is an unit-norm eigen vector of $\boldsymbol{L}$, with function value being:

$$\boldsymbol{u}^{(1)T} \boldsymbol{L} \boldsymbol{u}^{(1)} = \boldsymbol{u}^{(1)T} \lambda_1 \boldsymbol{u}^{(1)} = \lambda_1 \boldsymbol{u}^{(1)T} \boldsymbol{u}^{(1)} = \lambda_1$$

To minimize above function value, choose the solution (eigen vector) with the smallest eigen value!

[CBM]

Given that optimal $u^{(1)}$ is an eigen vector of L, what do we know about the spectra of L?

Let's look at the spectral properties of the Laplacian matrix L…

Properties of Laplacian L=D-A: Initial obsns.

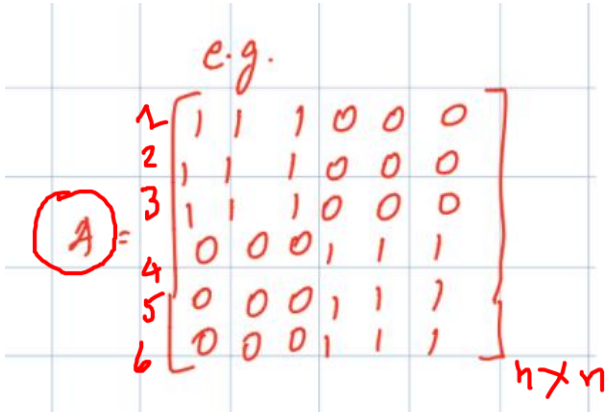
L (real) · symm.

L psd $\Rightarrow \lambda_i \geq 0$

Is $\lambda_i = 0$?

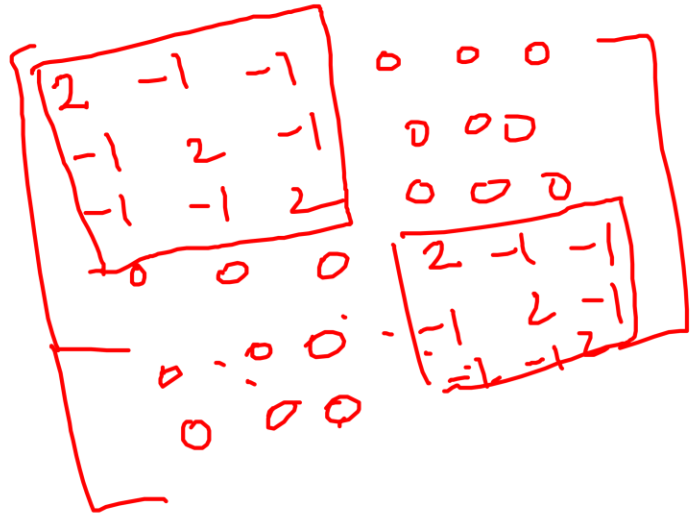
If $u = [1, 1, 1, \ldots, 1]$, then $Lu = 0$ $(\lambda_1 = 0)$

Properties of Laplacian: Toy Graph Example



$L\vec{x} = \lambda\vec{x}$

$L = D - A =$



λ

0 or 3 or:

$\vec{u}^{(1)} = [1\ 1\ 1\ 0\ 0\ 0]$ — 0

$\vec{u}^{(2)} = [0\ 0\ 0\ 1\ 1\ 1]$ — 0

$\vec{u} = [1\ 1\ 1\ 1\ 1\ 1]$ — 0

$L\vec{u}^{(1)} = 0 \cdot \vec{u}^{(1)}$

$L\vec{u}^{(2)} = 0 \cdot \vec{u}^{(2)}$

...

Properties of Laplacian L=D-A: formally

Proposition 1 (Properties of L) *The matrix L satisfies the following properties:*

1. For every vector $u \in \mathbb{R}^n$ we have

$$u^T L u = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} (u_i - u_j)^2$$

2. L is symmetric and positive semi-definite.
3. The smallest eigenvalue of L is 0, the corresponding eigenvector is the constant one vector $\mathbf{1}$.
4. L has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$.

Proposition 2 (Number of connected components and the spectrum of L) *Let G be an undirected graph with non-negative weights. Then the multiplicity k of the eigenvalue 0 of L equals the number of connected components $A_1, \ldots, A_k$ in the graph. The eigenspace of eigenvalue 0 is spanned by the indicator vectors $\mathbb{1}_{A_1}, \ldots, \mathbb{1}_{A_k}$ of those components.*

(Proof for k=1
(Then use spectrum of block diag. matrices)

[Luxburg. Tutorial on spectral clustering -- https://arxiv.org/pdf/0711.0189.pdf]

Going from one ($\boldsymbol{u}^{(1)}$) to two scores per node!

- What is the objective function and constraints with
  - one score vector?
  - two score vectors $\boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)}$?
    - (find two score vectors that minimize $\boldsymbol{u}^{(1)T}\boldsymbol{L}\boldsymbol{u}^{(1)} + \boldsymbol{u}^{(2)T}\boldsymbol{L}\boldsymbol{u}^{(2)}$ s.t. orthonormality)
  - three score vectors?
  - …
  - M score vectors?

Background on Constrained Optimization (ii): Lagrange multiplier (multiple constraints)

Problem: opt (min or max) $f(\boldsymbol{x})$ $\quad (\boldsymbol{x} \in \mathbb{R}^n, f: \mathbb{R}^n \to \mathbb{R})$

s.t. constraints $g_1(\boldsymbol{x}) = 0, \ldots g_m(\boldsymbol{x}) = 0$ $(m < n)$

FONC (First-order necessary conditions): same: set gradient of Lagrangian function $\mathcal{L}(\boldsymbol{x}, \lambda)$ to zero for constrained optima!

$$\nabla \mathcal{L}(\boldsymbol{x}, \lambda) = 0, \qquad \text{where } \mathcal{L}(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) + \textstyle\sum_i \lambda_i g_i(\boldsymbol{x}), \lambda_i \in \mathbb{R}.$$

Opt. strategy: Equivalently, we've these (n+m) equations over (n+m) unknowns to solve, and inspect each solution to find the constrained maxima or minima.

$$\nabla f(\boldsymbol{x}) = -\textstyle\sum_i \lambda_i \nabla g_i(\boldsymbol{x}) \qquad \text{(n equations)}$$

$$g_i(\boldsymbol{x}) = 0 \qquad \text{(m equations)}$$

Fineprint: For Lagrange multiplier to work, f, g_i must be smooth (continuously differentiable) and constraint qualification "$\nabla g(x^*) \neq 0$" becomes "the constraint gradient vectors $\{\nabla g_i(x^*)\}_i$ should be linearly indept.". (Otherwise, solution may not exist again).

From Paul's online notes [https://tutorial.math.lamar.edu/classes/calciii/lagrangemultipliers.aspx]

How do you minimize $L(\boldsymbol{u}^{(1)}) + L(\boldsymbol{u}^{(2)})$ subject to orthonormality constraints?

Minimize $_{\boldsymbol{u}^{(1)},\boldsymbol{u}^{(2)}\in\mathbb{R}^n}$ $f(\boldsymbol{u}^{(1)},\boldsymbol{u}^{(2)}) \coloneqq \boldsymbol{u}^{(1)T}\boldsymbol{L}\boldsymbol{u}^{(1)} + \boldsymbol{u}^{(2)T}\boldsymbol{L}\boldsymbol{u}^{(2)}$

s.t. $\boldsymbol{u}^{(1)T}\boldsymbol{u}^{(1)} = 1,\ \boldsymbol{u}^{(2)T}\boldsymbol{u}^{(2)} = 1$, and

$\boldsymbol{u}^{(1)T}\boldsymbol{u}^{(2)} = 0$ (equivalently $-2\,\boldsymbol{u}^{(1)T}\boldsymbol{u}^{(2)} = 0$ for convenience)

Set gradient of Lagrangian: $\boldsymbol{u}^{(1)T}\boldsymbol{L}\,\boldsymbol{u}^{(1)} + \boldsymbol{u}^{(2)T}\boldsymbol{L}\,\boldsymbol{u}^{(2)} +$

$\lambda_1(1 - \boldsymbol{u}^{(1)T}\boldsymbol{u}^{(1)}) + \lambda_2(1 - \boldsymbol{u}^{(2)T}\boldsymbol{u}^{(2)}) + \lambda_{12}(-2\,\boldsymbol{u}^{(1)T}\boldsymbol{u}^{(2)})$ to zero to

get these eqns:

$$\partial\boldsymbol{L}/\partial\boldsymbol{u}^{(1)} = 2\boldsymbol{L}\boldsymbol{u}^{(1)} - 2\lambda_1\boldsymbol{u}^{(1)} - 2\lambda_{12}\boldsymbol{u}^{(2)} = 0 \quad \text{-(1)}$$

$$\partial\boldsymbol{L}/\partial\boldsymbol{u}^{(2)} = 2\boldsymbol{L}\boldsymbol{u}^{(2)} - 2\lambda_2\boldsymbol{u}^{(2)} - 2\lambda_{12}\boldsymbol{u}^{(1)} = 0 \quad \text{-(2)}$$

$$\boldsymbol{u}^{(1)T}\boldsymbol{u}^{(1)} = 1,\ \boldsymbol{u}^{(2)T}\boldsymbol{u}^{(2)} = 1,\ \boldsymbol{u}^{(1)T}\boldsymbol{u}^{(2)} = 0 \quad \text{-(3)}$$

One solution is given by: $\lambda_{12} = 0$, and $\boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)}$ being any two orthonormal eigen vectors of $\boldsymbol{L}$, with function value f being the sum of their corresp. eigen values: $\lambda_1 + \lambda_2$.

Thus to minimize f, choose the two orthonormal eigen vectors of $\boldsymbol{L}$ with the smallest two eigen values (incl. multiplicity)!

[CBM]

Final check: Qn. What about other solutions where $\lambda_{12} \neq 0$?
Ans. They can be converted to a solution with $\lambda_{12} = 0$ without changing the function value!

Let $\boldsymbol{U}_{nx2} = [\boldsymbol{u}^{(1)}\, \boldsymbol{u}^{(2)}]$ and $H = \begin{bmatrix} \lambda_1 & \lambda_{12} \\ \lambda_{12} & \lambda_2 \end{bmatrix}$.

Then, Eqns. (1)-(3) can be rewritten in matrix notation as:

$$L\boldsymbol{U} = \boldsymbol{U}H \qquad \text{-(4)}$$

$$\boldsymbol{U}^T\boldsymbol{U} = I \text{ (identity matrix)} \qquad \text{-(5)}$$

, and the function value can be written as a matrix trace: $f(\boldsymbol{U}) = \boldsymbol{u}^{(1)T}\boldsymbol{L}\boldsymbol{u}^{(1)} + \boldsymbol{u}^{(2)T}\boldsymbol{L}\boldsymbol{u}^{(2)} = Tr(\boldsymbol{U}^{\mathrm{T}}L\boldsymbol{U})$ -(6)

Fact: Any solution ($\boldsymbol{U} = \widehat{\boldsymbol{U}}, H = \widehat{H}$) to eqns. (4)-(5) with $\widehat{H}$ a non-diagonal matrix (i.e., $\lambda_{12} \neq 0$) can be converted to a solution ($\boldsymbol{U} = \widehat{\boldsymbol{U}}Q, H = \Lambda$) with Λ a diagonal matrix (i.e., $\lambda_{12} = 0$) and the same function value (i.e., $f\left(\widehat{\boldsymbol{U}}Q\right) = f(\widehat{\boldsymbol{U}})$).

Here, Q, Λ can be computed from orthogonal diagonalization of the symmetric matrix $\widehat{H}$ as $\widehat{H} = Q\Lambda Q^T$.

Proof of Fact: Given that $(\widehat{\boldsymbol{U}}, \widehat{H})$ satisfies Eqns. (4)-(5), we first show that $(\widehat{\boldsymbol{U}}Q, \Lambda)$ also satisfies Eqns. (4)-(5) below.

- $\mathrm{L}\widehat{\boldsymbol{U}} = \widehat{\boldsymbol{U}}\widehat{\mathrm{H}} \Rightarrow \mathrm{L}\widehat{\boldsymbol{U}} = \widehat{\boldsymbol{U}}Q\Lambda Q^T \Rightarrow \mathrm{L}(\widehat{\boldsymbol{U}}Q) = (\widehat{\boldsymbol{U}}Q)\Lambda$
- $\left(\widehat{\boldsymbol{U}}Q\right)^T \widehat{\boldsymbol{U}}Q = Q^T\widehat{\boldsymbol{U}}^{\boldsymbol{T}}\widehat{\boldsymbol{U}}Q = Q^TQ = I$

Now, the function value (6) of any solution is: $f(\boldsymbol{U}) = Tr(\boldsymbol{U}^{\mathrm{T}}L\boldsymbol{U}) = Tr(\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U}H)$ (by Eqn (4)) $= Tr(H)$ (by Eqn (5)). Therefore,

- $f\left(\widehat{\boldsymbol{U}}Q\right) = Tr(\Lambda)$, and $f\left(\widehat{\boldsymbol{U}}\right) = Tr\left(\widehat{H}\right)$. But, $\mathrm{Tr}(\Lambda) = Tr\left(\widehat{H}\right)$ because the sum of eig.vals. of any matrix is its trace.

Going from two $(\boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)})$ to multiple scores per node!

- What is the objective function and constraints with
  - one score vector?
  - two score vectors?
  - three score vectors?
  - …
  - M score vectors $\boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)}, \ldots, u^{(M)}$?
    - (find M score vectors that minimize $\sum_k \boldsymbol{u}^{(k)T} \boldsymbol{L} \boldsymbol{u}^{(k)}$ s.t. orthonormality)

How do you minimize $\sum_k L\left(\boldsymbol{u}^{(k)}\right)$ subject to orthonormality constraints? Outline of answer (without proof)

Let $\boldsymbol{U}_{n\text{x}M} = \left[\boldsymbol{u}^{(1)}\ \boldsymbol{u}^{(2)} \ldots \boldsymbol{u}^{(M)}\right]$, and Lagrange multipliers $H_{M\text{x}M} = \{\lambda_{ij}\}_{i,j=1,\ldots,M}$.

Problem is to Minimize $f(\boldsymbol{U}) = \sum_{k=1}^{M} \boldsymbol{u}^{(k)T}\boldsymbol{L}\boldsymbol{u}^{(k)} = Tr(\boldsymbol{U}^{\mathrm{T}}L\boldsymbol{U})$

s.t. $\boldsymbol{U}^T\boldsymbol{U} = I$

Setting gradient of Lagrangian $Tr(\boldsymbol{U}^{\mathrm{T}}L\boldsymbol{U}) + Tr(\boldsymbol{H}\left(I - \boldsymbol{U}^T\boldsymbol{U}\right))$ to zero yields:

$$L\boldsymbol{U} = \boldsymbol{U}H \qquad \text{-(7)}$$

$$\boldsymbol{U}^T\boldsymbol{U} = I \text{ (identity matrix)} \qquad \text{-(8)}$$

Eqns. (7)-(8) can be solved and constrained minima found as in the M=2 case! Specifically,

- H can be taken to be a symmetric matrix, and further assumed wlog (without loss of generality) to be diagonal.
- Then, any solution should be M orthonormal eigen vectors of L.
- Minimizer found by taking the eigen vectors corresp. to the M smallest eigen values.

[related to Exercise 12.2 in [CMB]

In other words, solution using M=(a+1)th smallest eigen values (and corresp. orthonormal eigen vectors)

Let $u^0, u^1 \cdots u^a$ be the eigen vectors corresponding to $\lambda_0, \lambda_1, \cdots \lambda_a$ the $k+1$ smallest eigen-values of L.

Then $\sum_{i,j} A_{ij} (u_i^k - u_j^k)^2 = u^{kT} L u^k = \lambda_k$

Thus if λ_k is small, then the score assigned by u^k is such that Large A_{ij} => Small $|u_i^k - u_j^k|$

[HR]

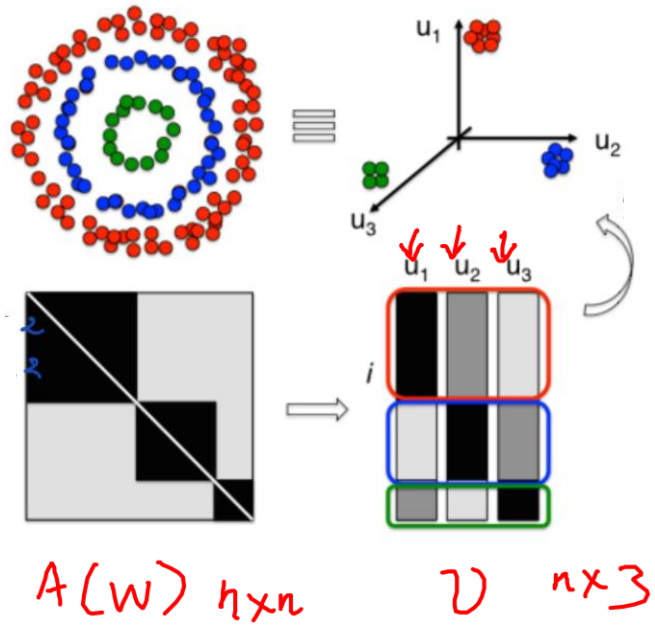
Spectral clustering algorithm:

Unnormalized spectral clustering

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

- Construct a similarity graph by one of the ways described in Section 2. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L.
- **Compute the first k eigenvectors $u_1, \ldots, u_k$ of L.**
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns.
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U.
- Cluster the points $(y_i)_{i=1,\ldots,n}$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, \ldots, C_k$.

Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{j | \, y_j \in C_i\}$.



[Luxburg. Tutorial on spectral clustering -- https://arxiv.org/pdf/0711.0189.pdf]

Conclusion and Looking ahead

- Linear Algebra has some immediate applications in Unsupervised ML.
  - We have looked at spectral-related applications (for both dim. redn. and clustering), by inspecting the eigen-values/vectors (spectra) of the Laplacian matrix.
  - Spectral clustering code on toy examples:
    https://colab.research.google.com/drive/1b3AEjJ2_QseWJrGBwWb1x_08OVBXwZS3?usp=sharing .

- Next steps:
  - Further applications of Linear Algebra:
    - Dim. redn (contd. appn. of spectra of another matrix) – PCA
    - Supervised ML (appn. of projection in linear algebra) – Linear regression

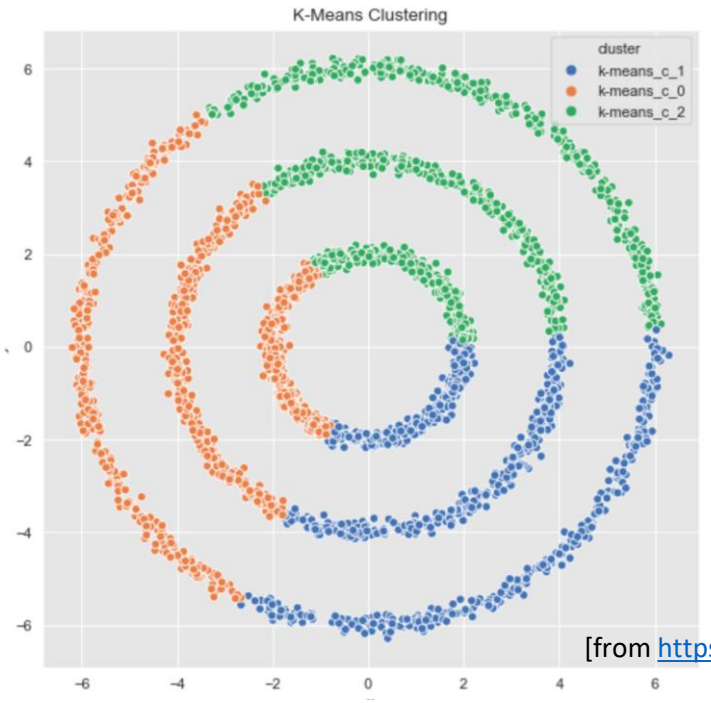
Thank you!

- Appendix follows

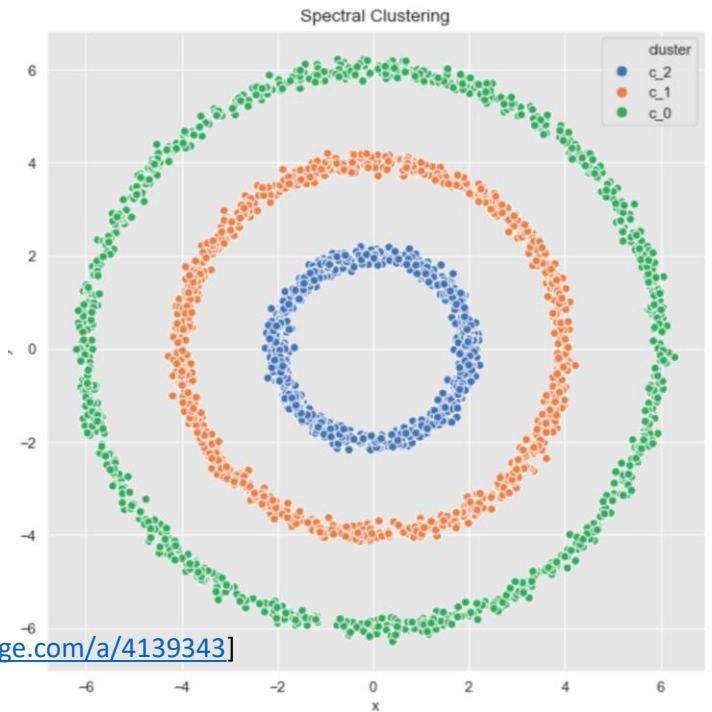
Defn. of convexity of a cluster

- **Observation:**
  - The clusters returned by (hard) k-means algorithm are each convex.
  - So k-means algo. won't work on a dataset where some "ground-truth" clusters are not convex.
- **Definition:**
  - Consider a dataset $D = \{x_1, x_2, \ldots, x_n\}$ with each $x_i \in \mathbb{R}^d$, and a clustering of this dataset.
  - A cluster C in this clustering is convex if for all pairs of points $x_i, x_j \in C$, any data point $x_k \in D$ that lie in the line connecting x_i, x_j also belongs to the same cluster C (i.e., $\forall\, x_k \in D$ that can be expressed as $\lambda x_i + (1-\lambda)x_j$ for some $\lambda \in [0,1]$, $x_k \in C$ also).
- **Examples:**

Convex clusters of D



Non-convex clusters (green and orange) of D



[from https://math.stackexchange.com/a/4139343]

Some final thoughts on Clustering and other Clustering algorithms!

Clustering: a popular PR task

- Human brains are good at grouping objects based on their similarity. How do we automate it?
  - Group N objects into K groups
  - Paradigms: **K-means** clustering – **centroid-based** clustering; **Hierarchical** clustering – **nested/stratified clustering** with top-down or bottom-up approaches; **Spectral clustering** – **graph connectivity-based (in-syllabus, already seen)**, etc.

- Why is a good clustering popular? It
  - allows exploratory data analysis (unsupervised ML)
  - has predictive power
  - allows lossy compression
  - can reveal interesting outliers

- Our approach in these lectures
  - Focus on spectral clustering to illustrate applications of Linear Algebra in clustering and dim. redn. (as well as the connection between graph-connectivity and linear algebra)
  - Say a few words about k-means and hierarchical clustering, which you may've already seen in other courses, unlike spectral clustering

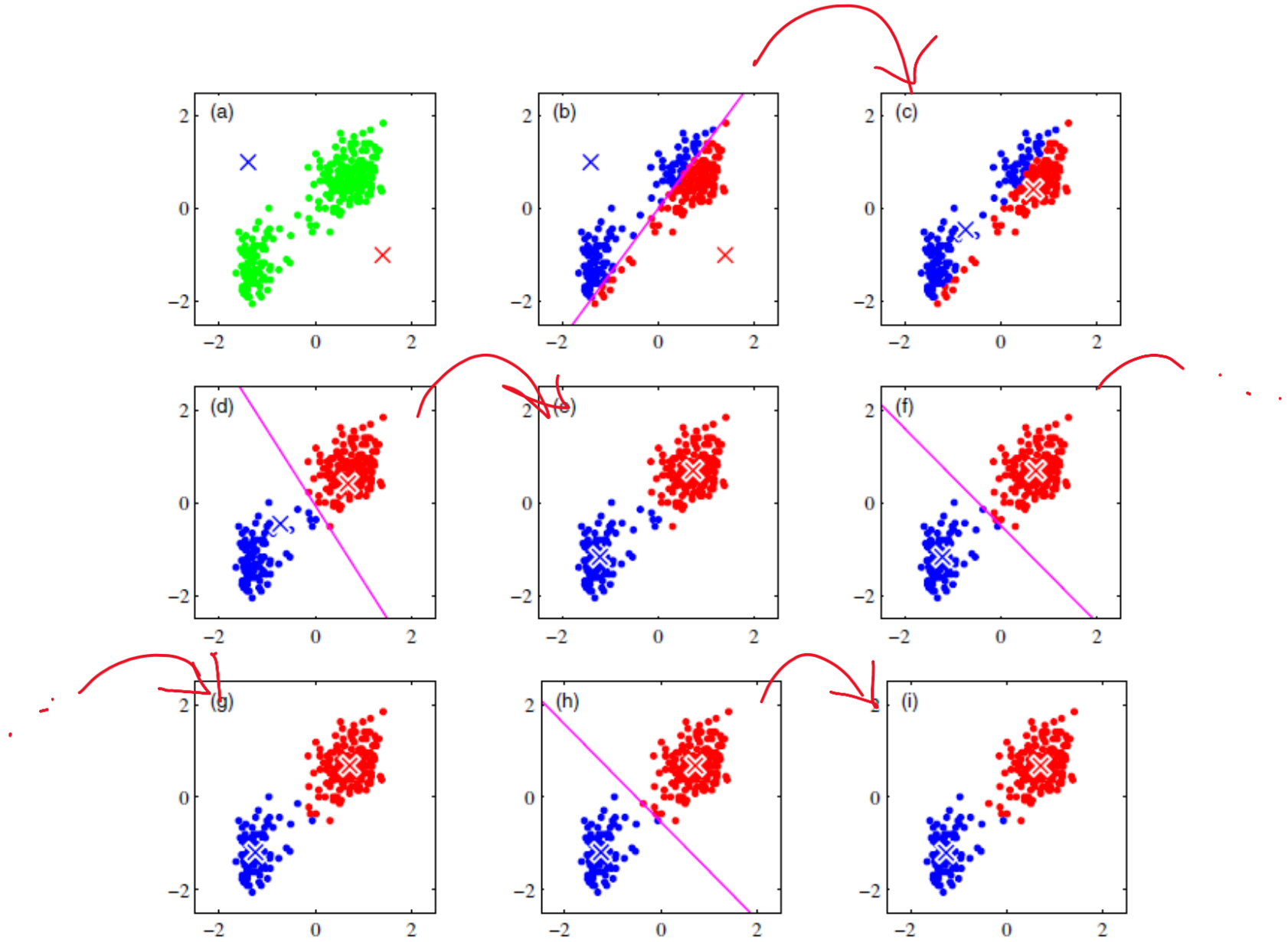
K-means clustering (in one slide! actually three :)

Problem: Find K cluster centers that minimize the sum of squared distance of each data point to the nearest cluster center.

Algorithm: Starting with K centers initialized in some way, iterate until convergence:

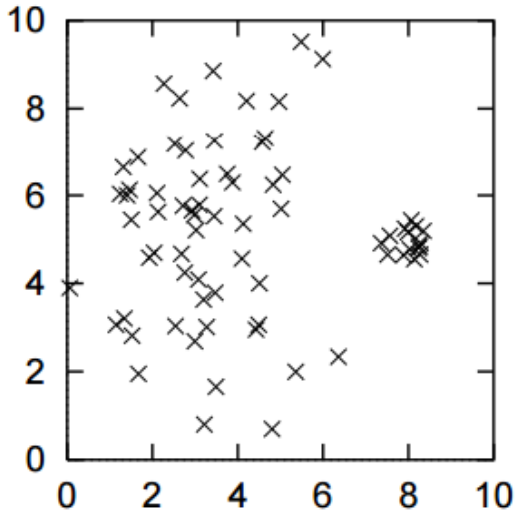
a) ***[Centers -> Clusters] Assign*** each data point to the nearest center.
b) ***[Clusters -> Centers] Update*** centers to sample means of data points they are responsible for.

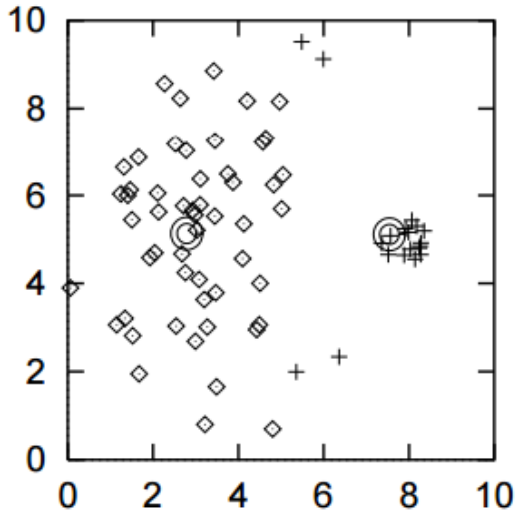
[DJM]

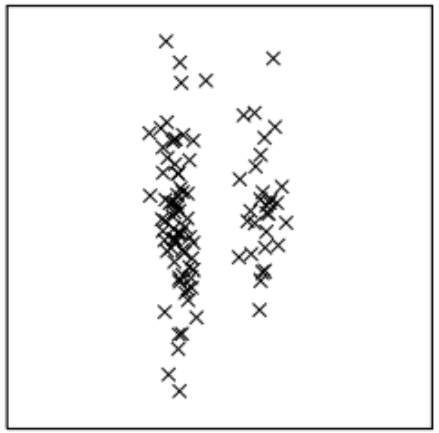


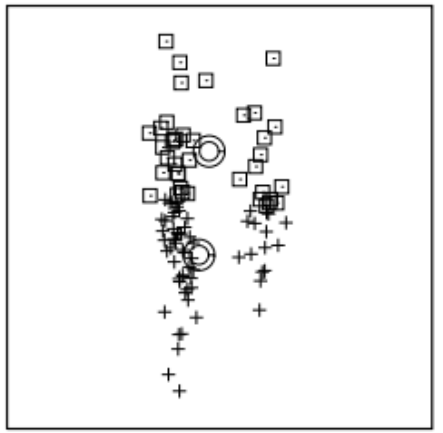
[CMB]

(hard) K-means pitfalls (and solutions?)









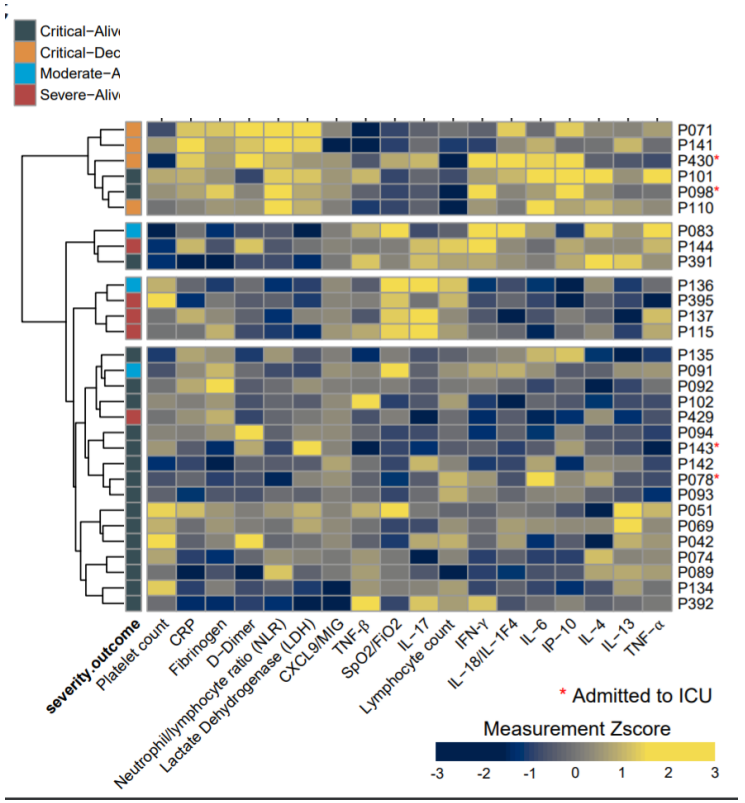
[DJM]

Hierarchical clustering (in one slide)

- Stratified clustering of objects, represented as a tree or dendrogram
  - very popular in heatmap display of data matrix.
  - E.g., in COVID bioinformatics shown to right!

- Two approaches to hierarchical clustering:
  - repeated bottom-up groupings of closest pair of objects – **agglomerative**
  - repeated top-down splitting of objects into groups – **divisive**

- In agglomerative, how do you construct the tree?
  - Convert the dataset into an adjacency or distance matrix as in Spectral clustering!
  - Aggregate the most similar pair of clusters/objects into one cluster, and iterate!



(Image: [Liu, Martins, Lau, et al. *Cell*. 2021])

Clustering: Overall Summary & Final Notes

- **Clustering** is a popular PR task that groups similar objects into clusters.
  - Unsupervised ML problem as it doesn't require any label information on the objects represented as datapoints.
  - Many algorithms available; the ones we saw were based on pairwise similarity/connectivity relations inferred from the datapoints: **K-means, Hierarchical, and Spectral**.

- In practice, evaluation of a clustering must be done via internal or external performance metrics:
  - **Internal:** how consistent is your clustering? E.g., Silhouette coefficient, Dunn Index
  - **External:** how well does your clustering match ground-truth labelling? E.g., Rand Index, F1-score.