Last class: community detection in SBM $G(n, p, q)$. **Spectral Clustering:**

Second top eigenvector $V_2(A)$ of the adjacency matrix $A$ coefficients of $V_2(A)$ cluster, predict communities:

We proved guarantees.

If $p = \frac{a}{n}$, $q = \frac{b}{n}$, $(a-b)^2 > C(a+b)\log n$, 99% accuracy with 99% probability.

1. Other approaches to community detection for SBM:

   Instead of the adjacency matrix $A$, one can use:

   (a) **graph Laplacian** $L = I - D^{1/2} A D^{1/2}$ [Levinia-V 2017]

   (b) **non-backtracking matrix** [Mossel-Neeman-Sly 2018, Massoulie 2015]

   (c) **semidefinite relaxations** [Abbe-Bandeira-Hall 2015, Guedon-V 2016]

   Advantage: removes log factor. Guarantees:

   - If $(a-b)^2 < 2(a+b)$, no method works [MNS]
   - If $(a+b)^2 > 2(a+b)$, >50% accuracy [MNS, M] (better than a random guess)
   - If $(a+b)^2 > C(a+b)$, 99% accuracy [MNS, LLV, GV]
   - If $\sqrt{a} - \sqrt{b} > 2\sqrt{\log n}$, 100% accuracy [ABN]
Methods based on semidefinite relaxation:

\[
\max \{ \sum_{i=1}^{n} A_i x_i y_i : x_i, y_i = \pm 1, \sum_{i=1}^{n} x_i y_i = 0 \}
\]

SDP relaxation?

\[
\begin{align*}
\max \{ & \sum_{i=1}^{n} A_i x_i y_i : x_i, y_i = \pm 1, \\
& \sum_{i=1}^{n} x_i y_i = 0 \}
\end{align*}
\]

NP-hard.

Disadvantages:
Slow.

Theoretical guarantees:
99% accuracy [Guedon-V2016]

Based on Grothendieck Ineq.

Let \( x \in \{-1, 1\}^n \) encode community membership.

Here \( x_i = y_i \Rightarrow x_i = \pm 1 \)

If communities have same size

\( \Rightarrow \) edges within communities

\( \Rightarrow \) edges across communities

\( \Rightarrow \) poly.

\( \Rightarrow \) NP-hard.

Polynomial time.

Theoretical guarantees:
99% accuracy [Guedon-V2016]

52\% accuracy [Abe-Bandeira-2015]
Beyond SBM

• More than 2 communities?

Include more eigenvectors of $A$, e.g., $k=3$:

$\Rightarrow$ represent the graph $G$ as the set of points $\{x_1, \ldots, x_n\}$ in $\mathbb{R}^k$.

• Cluster using e.g., K-means.

• Or just visualize the graph.

$\Rightarrow$ graph $\xrightarrow{\text{spectral embedding}} \mathbb{R}^k$.

Vice versa:

Data in $\mathbb{R}^d \rightarrow \text{graph}$

- Data $\{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ $\rightarrow \text{neighborhood graph}$:
  - Connect $x_i, x_j$ if $\|x_i - x_j\| \leq \varepsilon$ ($\varepsilon$-graph)
  - Or connect $x_i$ to $k$ nearest neighbors ($k$-NN graph)

Data points

kNN graph, $k = 5$

epsilon-graph, epsilon=0.3

Graph theory $\leftrightarrow$ Euclidean geometry

Particularly:

$x_1, \ldots, x_n \in \mathbb{R}^d \rightarrow \text{graph} \rightarrow x_1, \ldots, x_n \in \mathbb{R}^k$
Example [Kanervsk - Kaiser' 2009]:

Figure 10: Population density in the region of Lausanne. The values are normalized to the maximum value of density in the region.

Figure 11: Clusters obtained with k-means method.

Figure 13: Clusters obtained with spectral clustering + k-means.
4. An alternative to spectral embedding: multidimensional scaling (MDS)

Embeds a finite metric space \((M, d) \rightarrow \mathbb{R}^k\)

\[
\min_{x_1, \ldots, x_n \in \mathbb{R}^k} \sum_{i,j=1}^{n} (d(x_i, x_j) - ||y_i - y_j||_2)^2
\]

- **Pro**: general
- **Con**: non-convex (\(\Rightarrow\) no theory)

In particular:

5. Graph as a metric space

- The "graph distance" between vertices = length of the shortest path.

Example:

\[
\begin{align*}
\text{graph} & \overset{\text{MDS}}{\rightarrow} \mathbb{R}^k \\
\end{align*}
\]

Putting all together gives
⑥ **ISOMAP**

- **Input:** $x_1, \ldots, x_n \in \mathbb{R}^d$
- *construct a neighborhood graph* $(K$-$NN)$
  - compute the graph distance $d(x_i, x_j) \forall i, j$
- *apply MDS*
- **Output:** $y_1, \ldots, y_n \in \mathbb{R}^k$ (e.g. $k=2$)

**Example:** Isomap unrolls a Swiss roll [Tenenbaum et al. 2000]

- **Data**
- **Graph**
- **Embedding**

**Graph distance**
$\text{MDS + kernels} = \text{SNE} \quad (\text{stochastic neighbor embedding})$:

Embed $x_1, \ldots, x_n \in \mathbb{R}^n \rightarrow y_1, \ldots, y_n \in \mathbb{R}^k$ via

$$
\min_{y_1, \ldots, y_n \in \mathbb{R}^k} \sum_{i=1}^{n} \left[ \exp \left( - \frac{||x_i - x_j||_2}{2\sigma^2} \right) - \exp \left( - \frac{||y_i - y_j||_2}{2\sigma^2} \right) \right]^2.
$$

(\ast)

- **Advantage:** cares about local geometry, in $\mathcal{O}(n)$

- **Probabilistic view:** consider $p_{ij}, q_{ij}$ as probabilities (normalize them to have $\sum p_{ij} = \sum q_{ij} = 1$).

- Instead of the objective function (\ast), minimize the KL divergence

  $\text{KL}(P \parallel Q) := \sum_{ij=1}^{n} p_{ij} \log \frac{p_{ij}}{q_{ij}}$

- Heuristically, $\text{KL}(P \parallel Q) =$ amount of information in data ($P$) that is not explained by the model ($Q$)

  Rigorously, Shannon Min:

  $\text{KL}(P \parallel Q) = \mathbb{E} \# \text{bits to encode samples from } P \text{ using code optimized for } Q$

  $- \mathbb{E} \# \text{bits to encode samples from } P \text{ using code optimized for } Q$.

- KL objective function is more forgiving than sum-of-squares.
In the low-dimensional space, replace the Gaussian distribution by Cauchy distribution (more forgiving):

\[ p_{ij} = \exp\left( -\frac{||x_i - x_j||^2}{2\sigma^2} \right), \]

\[ q_{ij} = \frac{1}{1 + ||y_i - y_j||_2^2} \]

\[ \min D(D||Q) \Rightarrow \text{t-SNE} \]

Example:

MNIST dataset:

Visualization by PCA:

(a) Visualization by Isomap.

(a) Visualization by t-SNE.

[van der Maarten-Hilton 2008]
Example: COIL dataset = images of rotated objects:

(a) Example images of twenty categories

(b) Part example images of one category

(c) Visualization by Isomap.

(a) Visualization by t-SNE.

• No theory so far.
• $D(P \parallel Q)$ is minimized by gradient descent.

\[
\frac{\partial E}{\partial y_i} = 4 \sum_{j=1}^{n} (p_{ij} - q_{ij}) q_{ij} (y_i - y_j)
\]

\[\Rightarrow y_i \text{ is attracted or repelled by each other point } y_j.\]

• Can replace optimization by an explicit attraction-repulsion rule.

\[\textbf{UMAP} \quad [\text{McInnes et al. 2018}]\]

\begin{align*}
\text{MNIST data (umap-explorer) } & \quad \text{COIL data:}
\end{align*}
What do numbers look like?

\[ \forall n \in \mathbb{N} \rightarrow \text{binary vector (prime factors)} \]

\[ n = 110 \rightarrow (1, 1, 0, 1, 0, 0, \ldots) \]

The first million integers visualized by UMAP [Williamson 2018]
Fundamental questions:

- Is this all for real (galaxies etc.) or fake, a creation of the method?
- Theory?
- What is a good visualization? Is there "the best one"?
- What are clusters? How many? (e.g. types of cancer)