https://github.com/tedinburgh/ads2023

Graph-based clustering and density-based clustering

Tom Edinburgh
te269

Today: more clustering

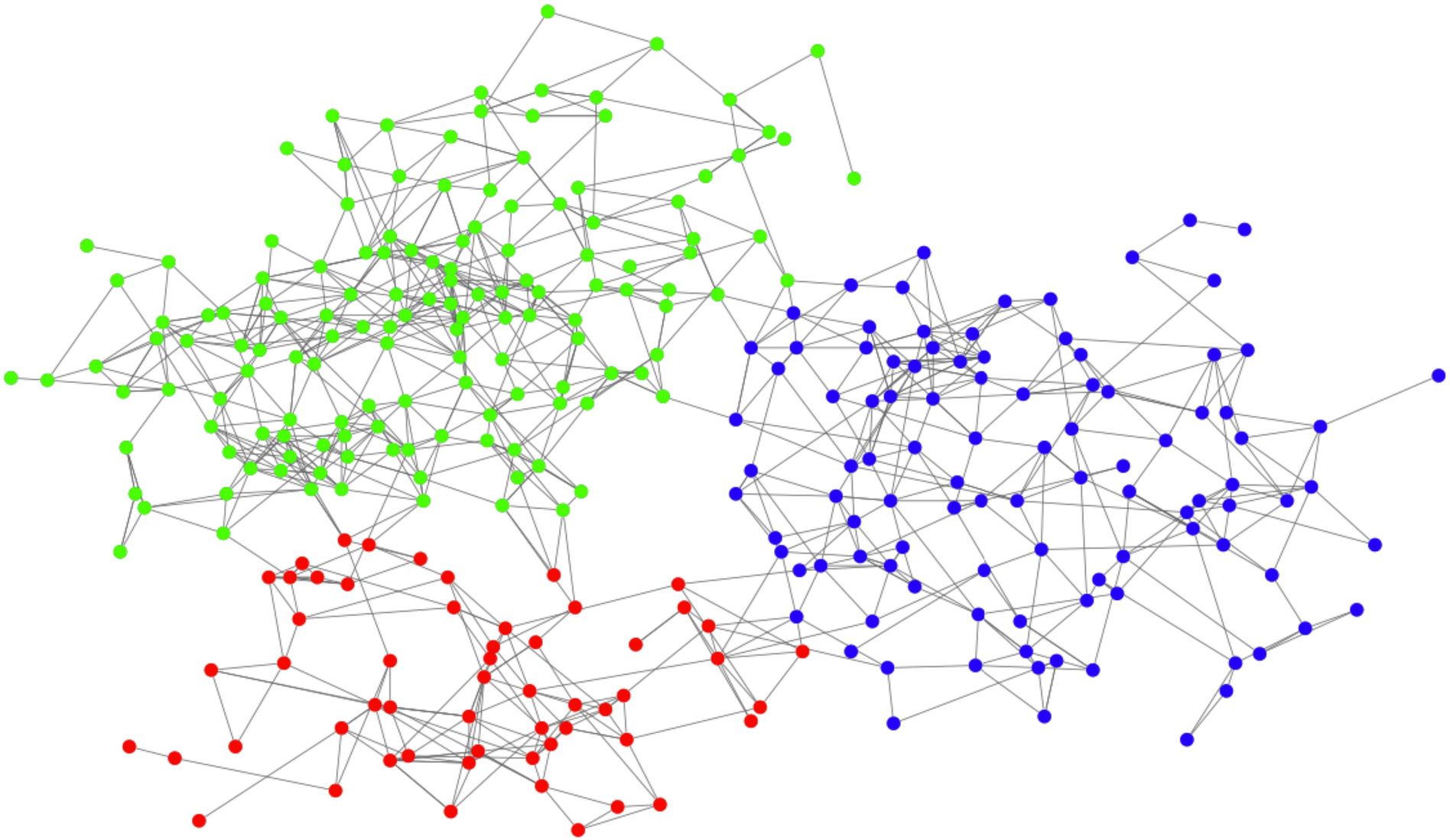
- Spectral clustering
- Graph-based clustering
- Density-based clustering
- Outliers (continued next time)
- Questions: halfway through, at the end, or by email (te269)

Resources

- Slides adapted from:
  - Ethan Fetaya/James Lucas/Emad Andrews, Toronto
  - Andrew Ng, Stanford
  - Thomas Sauerwald, Cambridge
  - Akshay Krithnamurthy, UMass

- Resources for spectral clustering:
  - A Tutorial on Spectral Clustering, Ulrike von Luxburg, Max Planck Institute

Similarity graphs



Similarity graphs

$$X = \begin{pmatrix} X_1 & X_2 & \cdots & X_p \end{pmatrix} = \begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

- Suppose we have some notion of similarity s_{ij} between all pairs of data points x_i and x_j
- Two points are connected if the similarity s_{ij} is over some threshold
- This defines undirected graph $G = (V, E)$, vertices v_i represent observations x_i

This can just be labelled $1, \ldots, n$, and their location doesn't really mean anything

Similarity graphs

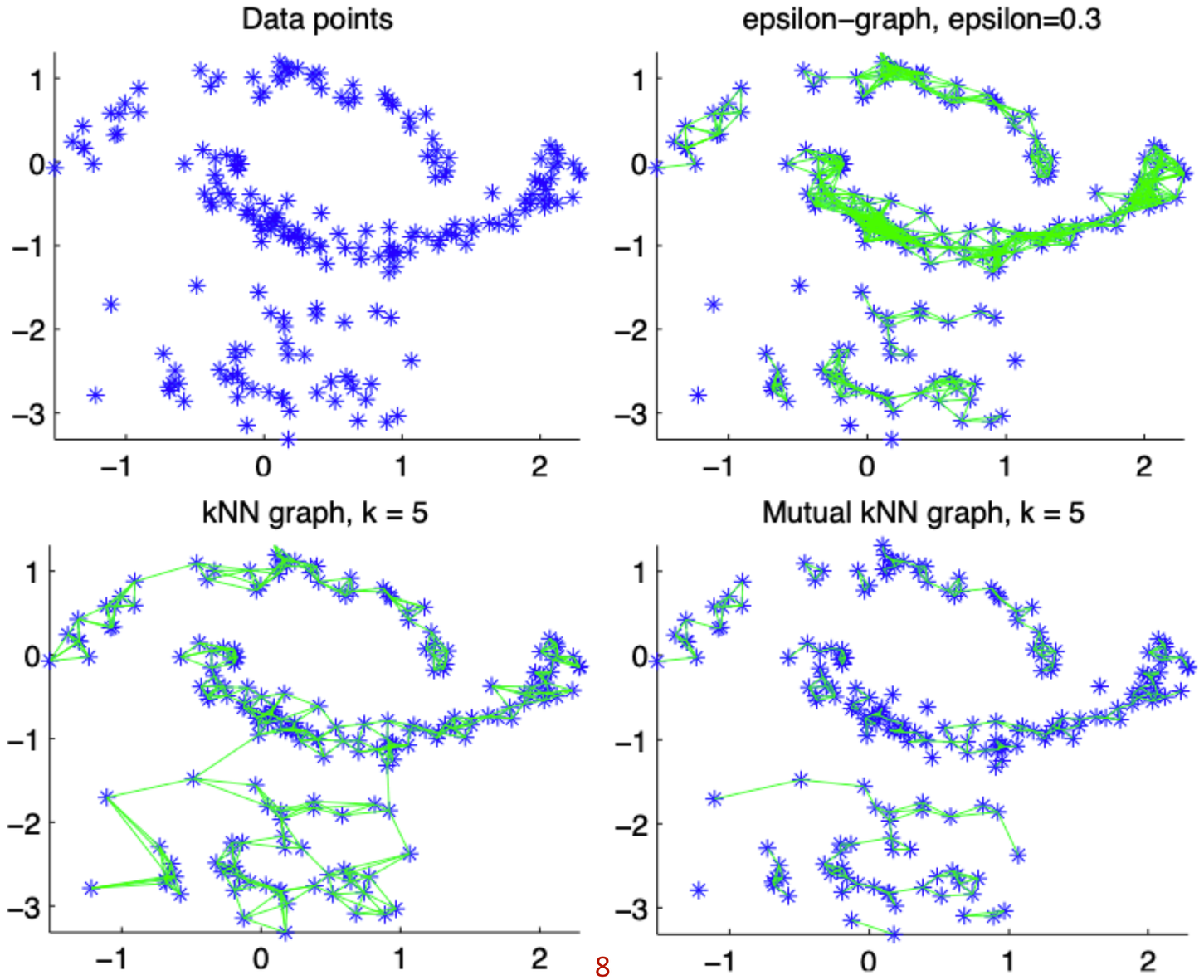
$$X = \begin{pmatrix} X_1 & X_2 & \cdots & X_p \end{pmatrix} = \begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

- Suppose we have some notion of similarity s_{ij} between all pairs of data points x_i and x_j
- The type of data itself is largely irrelevant, once we've defined the similarity (e.g. it could be categorical, continuous, quantitative, qualitative)
- The goal is to model local neighbourhood relationships within the graph network

Examples of similarity graphs

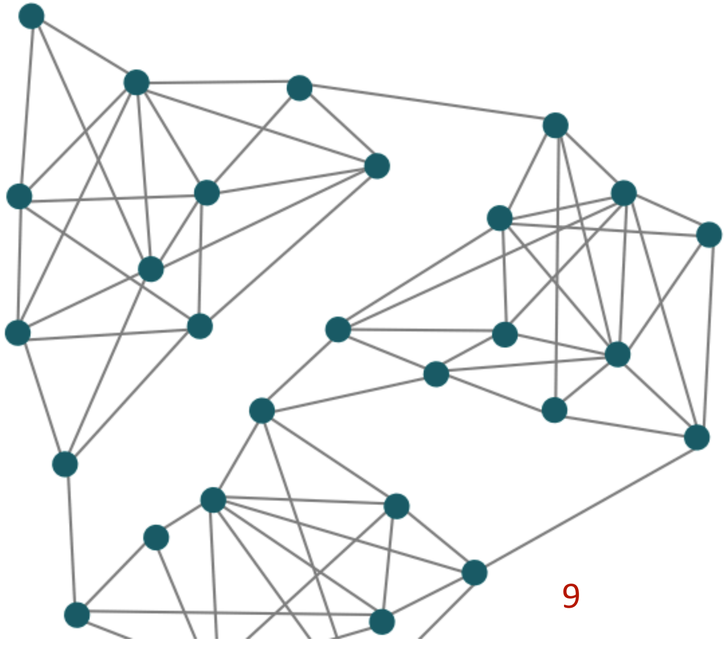
- How do we define similarity to construct the graph network?
- There are various options! E.g.
  - ϵ-neighbourhood graph: connect points whose pairwise distance is $< \epsilon$
  - k-nearest neighbour graph: connect v_i and v_j if one of these vertices is among the k-nearest neighbours of the other
  - Mutual k-nearest neighbour graph: connect v_i and v_j if both vertices are among the k-nearest neighbours of the other
  - Fully connected (weighted) graph: each vertex connected to all others

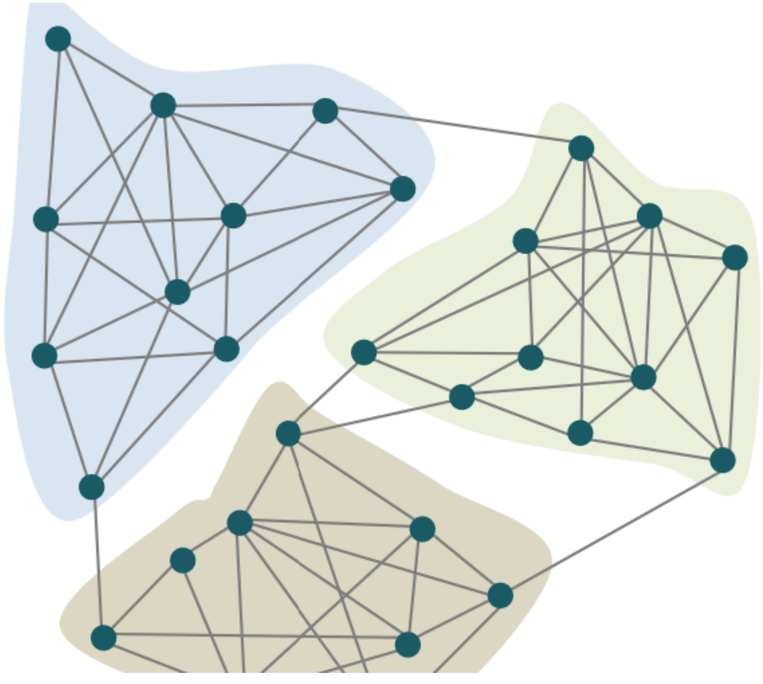
Examples of similarity graphs



Graph-based clustering

- The idea for graph-based clustering is to partition the graph into subgraphs (pieces), so that vertices within each piece have more connections among each other (on average) than with vertices in other pieces
- There are some theoretical results that connect this approach to the Laplacian spectral methods





Normalised cuts

- For two subsets of vertices, A and B, of a weighted graph $G = (V, E, w)$:
  - $W(A, B) = \sum_{i \in A, j \in B} w_{ij}$
  - $\text{ncut}(A, B) = \frac{w(A, B)}{w(A, V)} + \frac{w(A, B)}{w(B, V)}$, $\text{nassoc}(A, B) = \frac{w(A, A)}{w(A, V)} + \frac{w(B, B)}{w(B, V)}$
- We want to cut the graph into two distinct non-overlapping pieces, S and $\bar{S}$
- $\text{ncut}(S, \bar{S})$ measures the similarity between pieces, $\text{nassoc}(S, \bar{S})$ measures the total similarity of vertices within the same part

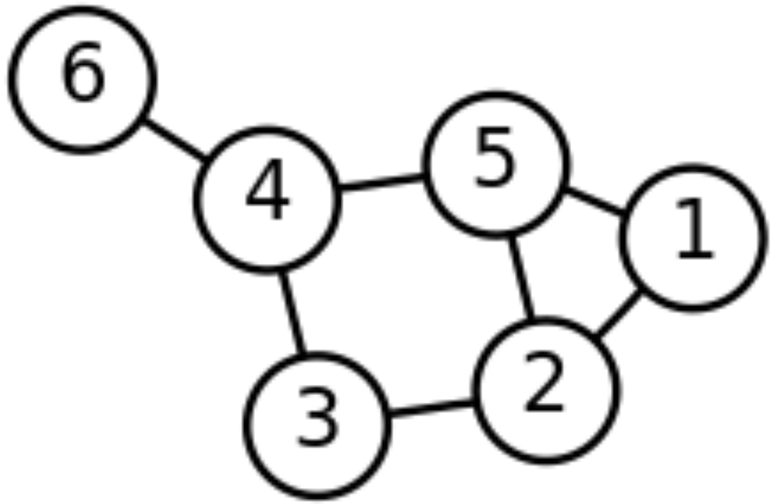
HCS algorithm

- Seek highly connected subgraphs
- Repeatedly cut smaller and smaller subgraphs until each is highly connected
- This is a recursive algorithm (e.g. cut into two subgraphs and then apply HCS to each, until convergence)
- At each step, perform a minimum cut (i.e. identify the minimum set of edges whose removal disconnects the graph)
- In a highly connected graph of size m, each vertex must have degree $\geq m/2$ and the diameter of the graph (the longest path between any two nodes) is at most 2

Returning to spectral clustering

$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{\{1, 2\}, \{1, 5\}, \{2, 5\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{4, 6\}\}$$



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

Similarity graphs: unweighted graphs

- Edge $\{i, j\} \in E$ if vertices v_i and v_j are connected (s_{ij} above some threshold)
- Unweighted adjacency A is $n \times n$ matrix, with $A_{ij} = \begin{cases} 1 & \text{if } \{i, j\} \in E \\ 0 & \text{otherwise} \end{cases}$
- Spectral properties of A contain information about clusters within the graph
- We use the Laplacian instead because it has some useful properties (e.g. positive semi-definite), while retaining information about clusters
- The normalised Laplacian ($L_s = I - D^{-1/2}AD^{-1/2}$ or $L_r = I - D^{-1}A$) can improve results (spectral clustering using L sometimes don't converge well)

Similarity graphs: weighted graphs

- Graph $G = (V, E, w)$, edge $\{i, j\} \in E$ has weight w_{ij} (if connected)
- E.g. Gaussian similarity $w_{ij} = \exp(-\|x_i - x_j\|^2/(2\sigma^2))$ for hyperparameter σ^2
- Weighted adjacency A is $n \times n$ matrix, with $A_{ij} = \begin{cases} w_{ij} & \text{if } \{i, j\} \in E \\ 0 & \text{otherwise} \end{cases}$
- We usually require undirected graphs, with $w_{ij} = w_{ji}$
- The degree of v_i is $d_i = \sum_j w_{ij}$ and Laplacian $L_{ij} = \begin{cases} d_i & \text{if } i = j \\ -w_{ij} & \text{if } \{i, j\} \in E \\ 0 & \text{otherwise} \end{cases}$

Spectral clustering: basic algorithm (last lecture)

1. Calculate the normalised Laplacian
2. Calculate the eigenvalues and eigenvectors
3. Form a matrix U of K eigenvectors corresponding to the K smallest non-zero eigenvalues
4. U is an $n \times K$ matrix, the i^{th} row defines features of the network graph node i
5. Cluster the graph nodes based on these features, using e.g. k-means

- Why? Spectral properties of the graph contains information about clustering

Spectral clustering

- Idea: transform x_i (vector of length p) to y_i (vector of length k) using eigenvectors of the Laplacian matrix
- The clusters are more easily distinguished from one another in this representation
- Form a $n \times K$ matrix U of the K eigenvectors corresponding to the smallest non-trivial eigenvalues of the Laplacian ($\lambda_1 = 0$ has eigenvector $u_1 = 1$)
- Denote the i^{th} row y_i, this represents observation x_i / vertex v_i and cluster y_i using k-means (or another basic clustering method)
- If we take just one eigenvector (called the Fiedler vector), then y_i is a scalar variable, and we can cluster the data by $y_i \geq 0$ or $y_i < 0$

Implementing spectral clustering

- What similarity function should you use?
- What type of similarity graph and how do you define connectedness?
- How many eigenvectors do you include?
- How many clusters?
- Should you use unnormalised or normalised Laplacian?

Normalised cuts (again)

- We want to find $\min_{S,\bar{S}}$ ncut$(S, \bar{S})$
- This turns out to be equivalent to the constrained optimisation problem $\min_y y^T D^{-1/2} L D^{-1/2} y$, subject to $y^T D^{1/2} 1 = 0$, $y^T D y = \sum_{i=1}^{n} d_i$ and

$$y_i = \begin{cases} c & v_i \in S \\ -1/c & v_i \in \bar{S} \end{cases} \quad \text{where } c = \sqrt{\frac{\sum_{i \in \bar{S}} d_i}{\sum_{i \in S} d_i}}$$

- This is very difficult to solve

Normalised cuts (again)

- We want to find $\min_{S,\bar{S}} \text{ncut}(S, \bar{S})$
- Instead, if we relax the final condition, and consider the constrained optimisation problem $\min_y y^T D^{-1/2} L D^{-1/2} y$, subject to

$$y^T D^{1/2} 1 = 0, \ \|y\|^2 = \sum_{i=1}^{n} d_i$$

- The solution to this is the eigenvector corresponding to the second eigenvalue of the normalised Laplacian L_d
- We can extend this to multiple cuts (and similarly relax some constraints) in order to recover multiple eigenvectors from spectral clustering

Conductance

- The conductance of a graph measures how ‘well-knit’ the graph is
- This measures the quality of a spectral clustering (the conductance of a cluster should be low)
- The conductance of a cut is $\phi(S) = \frac{\sum_{i \in S, j \in \bar{S}} w_{ij}}{\min\left(\sum_{i \in S, j \in V} w_{ij}, \sum_{i \in \bar{S}, j \in V} w_{ij}\right)}$
- The conductance of the graph is $\phi(G) = \min_{S \subseteq G} \phi(S)$ $\phi(G) = 0$ if G is disconnected
- For a d-regular undirected graph, $\lambda_2/2 \leq \phi(G) \leq \sqrt{2\lambda_2}$ (Cheever’s inequality)

Eigenvalue of Laplacian

Conductance

- When we are performing spectral clustering using only one eigenvector u (corresponding to λ_2) of the Laplacian
- If order the components $u_1 \leq u_2 \leq \ldots \leq u_n$, then perform a cut $(\{1,\ldots,k\}, \{k+1,\ldots,n\})$ so that this cut has the the smallest conductance
- Then the clusters S and $\bar{S}$ are such that $\phi(S) \leq \sqrt{2\lambda_2} \leq 2\sqrt{\phi(G)}$
- i.e. we have guarantees on the amount of improvement in the clustering quality, when performing just one cut

Questions?

Density-based clustering

- Density-based spatial clustering of applications with noise (DBSCAN)
- Introduced by Ester, Kriegel, Sander and Xu in 1996
- A very popular non-parameter clustering algorithm

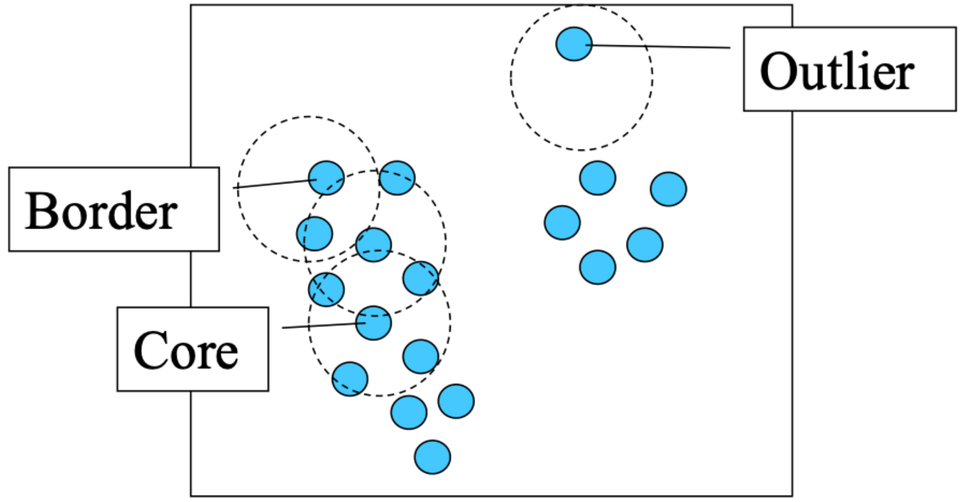
DBSCAN

- Basic idea: cluster together points that are closely packed together and mark points in low-density regions as outliers
- Each point is one of the following:
  - Core point
  - Directly reachable from a core point
  - Reachable from a core point
  - An outlier (not reachable from any other point)
- A core point forms a cluster with all points (core or non-core) that are reachable from it, and each cluster contains at least one core point

We need to define what these mean!

DBSCAN - some definitions

- Parameters: minPts and ϵ (the radius of a neighbourhood around each point)
- p is a **core point** if there are at least minPts within the ϵ neighbourhood around p
- q is **directly reachable from** p if q is within the ϵ neighbourhood around the core point p. If q is not a core point itself, it is a **border point**
- q is **reachable from** p if there is a path $p_1, \ldots, p_n$ of core points where
  - each p_{k+1} is directly reachable from p_k,
  - p_1 is directly reachable from p,
  - q is directly reachable from p_n
- All other points are **outliers**



DBSCAN - some definitions

- p and q are **density-connected** if both p and q are reachable from some core point o
- Reachability is not symmetric (core points can reach non-core points, but not vice versa)
- But density-connected is symmetric
- All points within a cluster are mutually density-connected

DBSCAN - basic algorithm

- For each point p,
  - If p is not yet classified, then
    - If p is a core point (i.e. it has at least minPts within its ϵ-neighbourhood), then it forms a new cluster, and all points that are reachable from p join this cluster
    - otherwise, p is classed as noise
- p may be initially classed as noise because it is a non-core point, but if it is reachable, it will eventually be re-classed within some cluster

DBSCAN

- Pros:
  - No need to pre-specify the number of clusters
  - Can find arbitrarily shaped clusters
  - Requires at least minPts connected to a core, so single-link connectedness within a cluster is reduced
  - Robust to outliers (and can identify outliers!)

DBSCAN

- Cons:
  - Depends on the distance metric (we'll discuss the curse of dimensionality again next week)
  - Cannot cluster datasets that contain large differences in densities (ϵ is universal to all clusters)
  - Sensitive to parameter choices and it may be challenging to find a meaningful neighbourhood radius ϵ

Outliers

- What is an outlier?
- When might outliers be important?
- What is the difference between noise and outliers?

Outliers

- Outliers are data points that are considerably different from the remainder of the data
- Naturally occurring outliers do occur but are relatively rare
- They are usually either important or a nuisance (e.g. rare diseases, decimal errors)
- Label error, e.g. images of dogs but with a few cats included by accident
- Noise is generally not very interesting (not unusual values)

How can we identify outliers?

- Model-based:
  - Outliers are points that don’t fit the model very well or distort the model
  - Points far away from cluster centres or small clusters may be outliers
- Data-based:
  - Identify directly from the data without a model e.g. density-based

- What assumptions might we make about outliers?
- How do outliers relate to statistical significance/hypothesis testing?

Local outlier factor

- This is based on local density, similar concepts to to DBSCAN
- Identify points that have a much lower density, these are outliers
- LOF uses k-nearest neighbour distances rather than ϵ-neighbourhoods
- The reachability distance between points p and q is

  $\text{rd}_k(p, q) = \max(r_k(q), d(p, q))$, where r_k is the distance from q to its k^{th} -nearest neighbour
- The local reachability density of p is the average reachability distance of p from its neighbours (from, not to)

Local outlier factor

- The local outlier factor (LOF) compares the local reachability density (LRD) of p to the LRD of its k-nearest neighbours
- An LOF approximately 1 means a similar density to the neighbours
- An LOF < 1 means a higher density than the neighbours (an **inlier**)
- An LOF > 1 means a lower density than the neighbours (an **outlier**)

Questions?

- Feel free to email me at te269@cam.ac.uk

Next time

- Clustering
  - Outliers
  - Clustering evaluation and hyperparameters (e.g. number of clusters)
  - Pros and cons
  - Consensus clustering