9. Graphs in Machine Learning (Graph-based Semi-supervised learning)

COMP0078: Supervised Learning

Mark Herbster

2 December 2019

University College London
Department of Computer Science

SL-GM-19v1

Today's Plan

Graph-based Semi-Supervised Learning

- Motivating Semi-Supervised Learning
- The Graph Laplacian
- Spectral Clustering
- Graph-based Semi-supervised Learning
- Understanding the Laplacian Interpolation (effective resistance)
- Generalising the Laplacian (p-Laplacian)

Part I

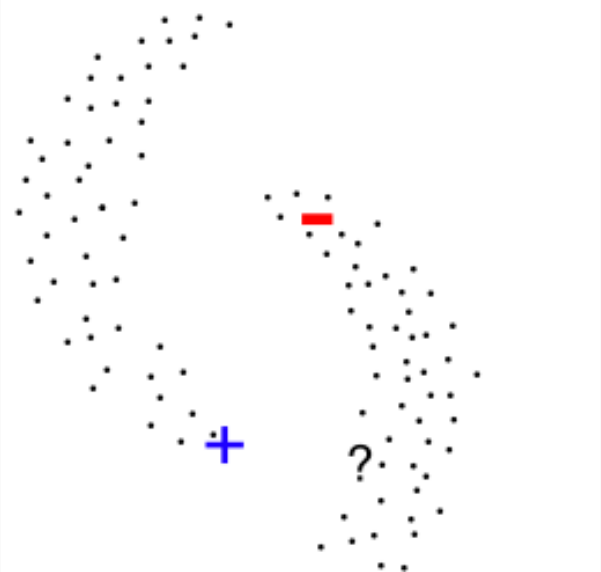
Motivation

"To learn from both labeled and unlabled data"

Motivation (Learning)

The hypothesis of SSL is that the structure of the input space may be exploited to improve learning for example.

- **[Cluster hypothesis]:** If points are in the same cluster, they are likely to be of the same class.
- **[Manifold hypothesis]:** The (high-dimensional) data lie on a low-dimensional manifold.



Thought problem

Problem

There exists two Gaussian distributions and a hyperplane. Points are generated by either distribution and labelled according to which side of the hyperplane they fall.

Will unlabelled data help?

1. No known relationship between the hyperplane and the inputs ("x"'s).
2. The hyperplane is known to separate the "centers" of the gaussians.

Motivation (Data)

Unlabeled may be "cheap" wrt to labeled the data. For example,

- We have a collection of news stories – yet only a few are labeled with respect to content
- We have a great deal of collected voice (eg siri, google voice); transcription slow and expensive
- In drug design we have many sample molecules, each experiment to test for certain properties, may require days of labwork and considerable expense.

Recall: Supervised learning

- Typical goal
  - Given data (pattern,target) $\mathcal{S} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\}$
    infer a function f such that $f(\mathbf{x}_i) \approx y_i$
    for future data $\mathcal{S}' = \{(\mathbf{x}_{\ell+1}, y_{\ell+1}), (\mathbf{x}_{\ell+2}, y_{\ell+2}), \ldots\}$.
  - Classification : $y \in \{-1, +1\}$; Regression : $y \in \mathbb{R}$
- Algorithms
  1. Linear Regression
  2. Neural Networks
  3. Decision Trees
  4. Support Vector Machines

Recall:Unsupervised learning

- Typical goal
  - Given data (patterns) $\mathcal{S} = \{\mathbf{x}_1, \ldots, \mathbf{x}_\ell\}$

    Model the data.
  - For example (clustering)

    Give a partition/function of $f : x \to \{1, \ldots, k\}$ of $\mathcal{S}$

    Such that $f(\mathbf{x}_i) = f(\mathbf{x}_j) \quad \Rightarrow \quad \mathbf{x}_i \approx \mathbf{x}_j$

    $f(\mathbf{x}_i) \neq f(\mathbf{x}_j) \quad \Rightarrow \quad \mathbf{x}_i \not\approx \mathbf{x}_j$
- Algorithms
  1. Clustering (k-means)
  2. Dimensionality Reduction (PCA)

Semi-supervised Learning

- Typical goal
  - Given data $\mathcal{S} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell), \mathbf{x}_{\ell+n}, \ldots, \mathbf{x}_n\}$
    infer a function f such that $f(\mathbf{x}_i) \approx y_i$
- Idea
  - Combine supervised and unsupervised learning.
- Algorithms
  1. Co-Training
  2. Graph-based Semi-Supervised Learning [today]
  3. Semi-supervised Support Vector Machines

Inductive and transductive semi-supervised learning

Definition: Inductive semi-supervised learning

Given a training sample,

$$\{(\mathbf{x}_i, y_i)\}_{i=1}^{\ell}, (\mathbf{x}_j)_{j=\ell+1}^{n}\}$$

an *inductive* learner learns a function $f : X \to Y$, with the aim that "$f(x) \approx y$", **for all**, $x \in X$.

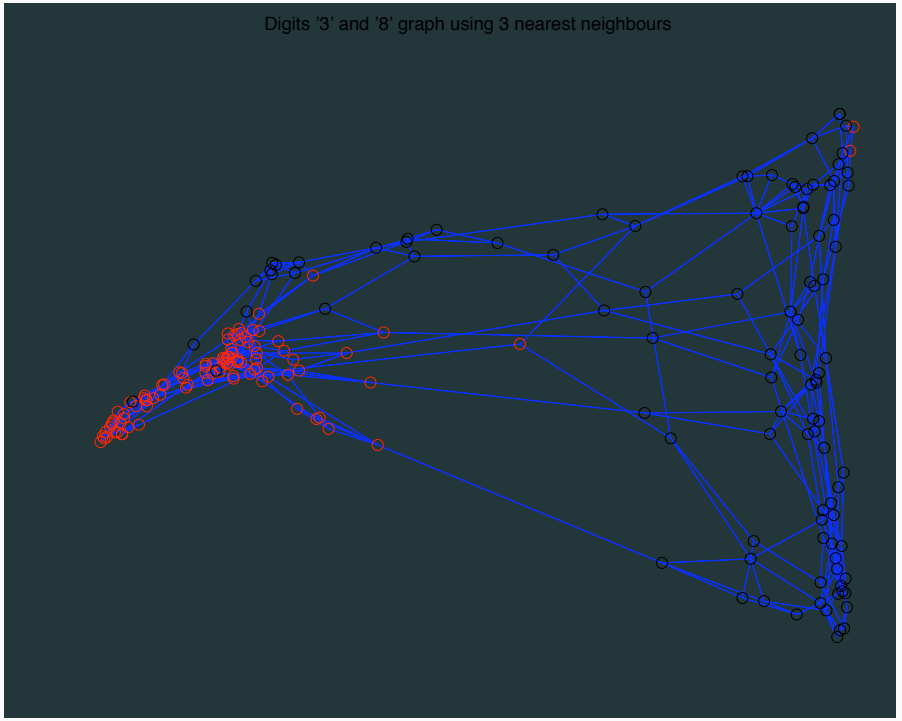
Definition: Transductive semi-supervised learning

Given a training sample,

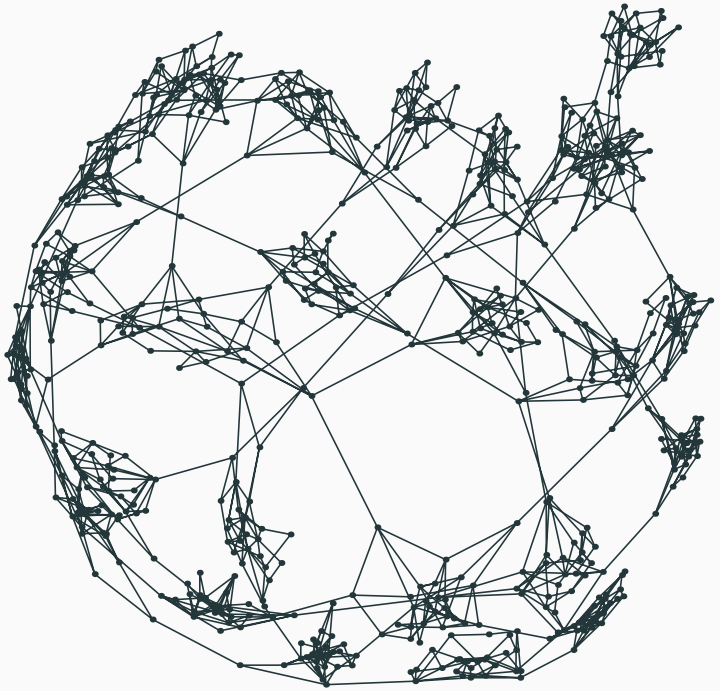
$$\{(\mathbf{x}_i, y_i)\}_{i=1}^{\ell}, (\mathbf{x}_j)_{j=\ell+1}^{n}\}$$

a transductive learner learns a function "$f(x) \approx y$" only on the **unlabelled** data $f : X_U \to Y$ (with $X_U = (\mathbf{x}_j)_{j=\ell+1}^{n}$).

Graph Examples – 1

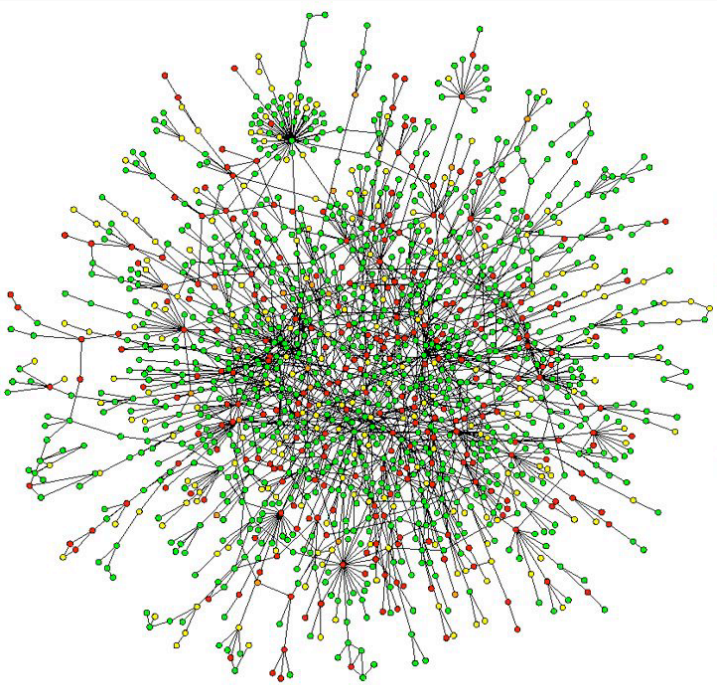


USPS digits 3 and 8

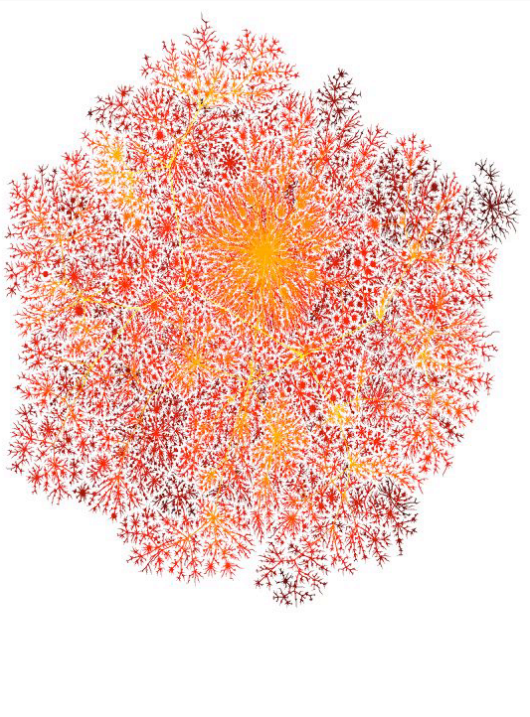


Random graph $G^2_{k-\text{out}}(26; 2)$

Graph Examples – 2

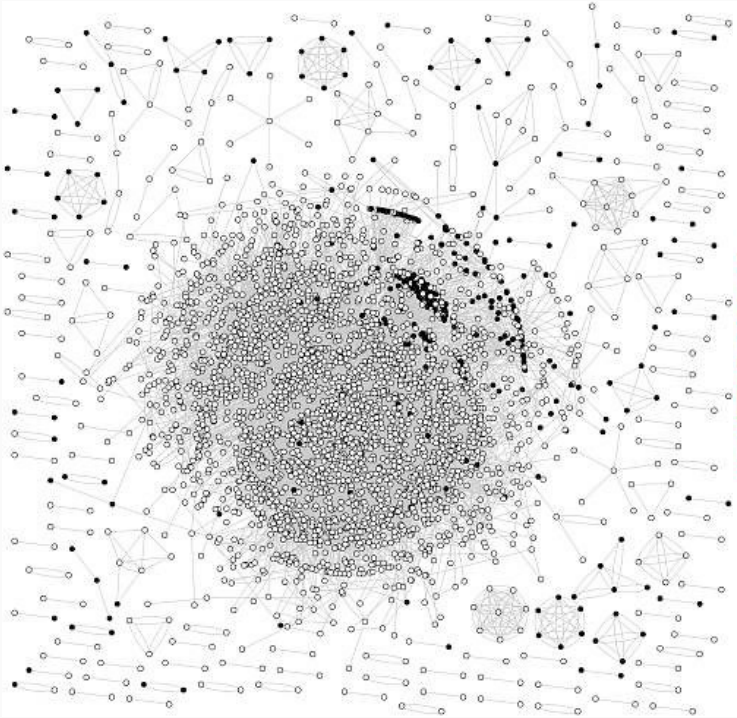


Yeast protein network



Internet hosts

Graph Examples – 3



Web Spam

Twitter Social Network

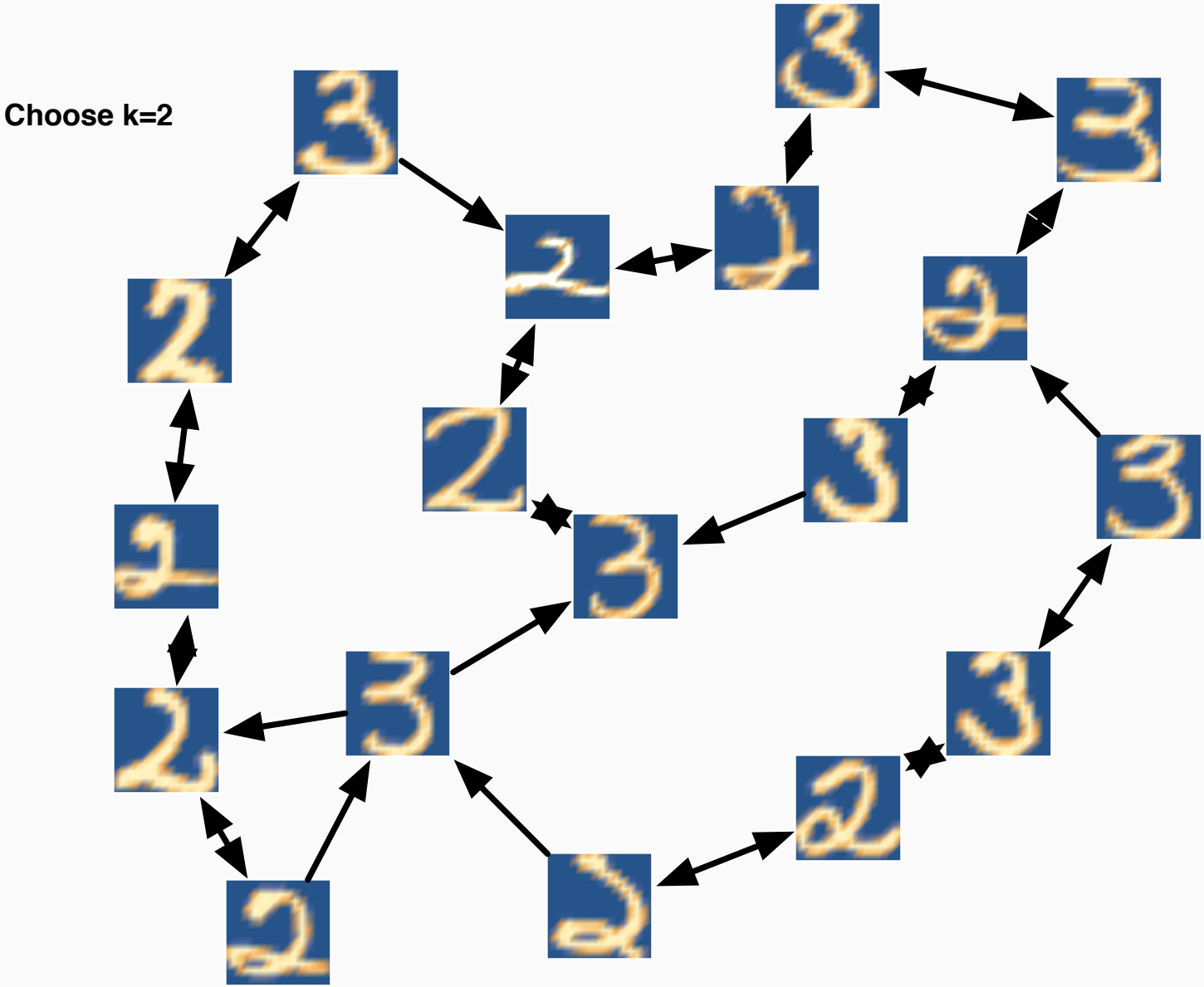
Building graphs

Where do graphs come from?

- Intrinsic: (protein interaction network, social network)
- Built (using a similiarity metric $d(p, q)$):
  1. `k-Nearest neighbors:` Each vertex connects to k nearest neighbors
  2. `ϵ-ball:` Every point is connected to a every other within ϵ distance
  3. `Tree-based`: Build a MST ("Minimum spanning tree") or a SPT ("Shortest paths tree")
  4. `Weighted graph:` The edge between p and q is weighted $e^{-cd(p,q)}$
  5. `Combo:` The first three methods may be combined with weights.

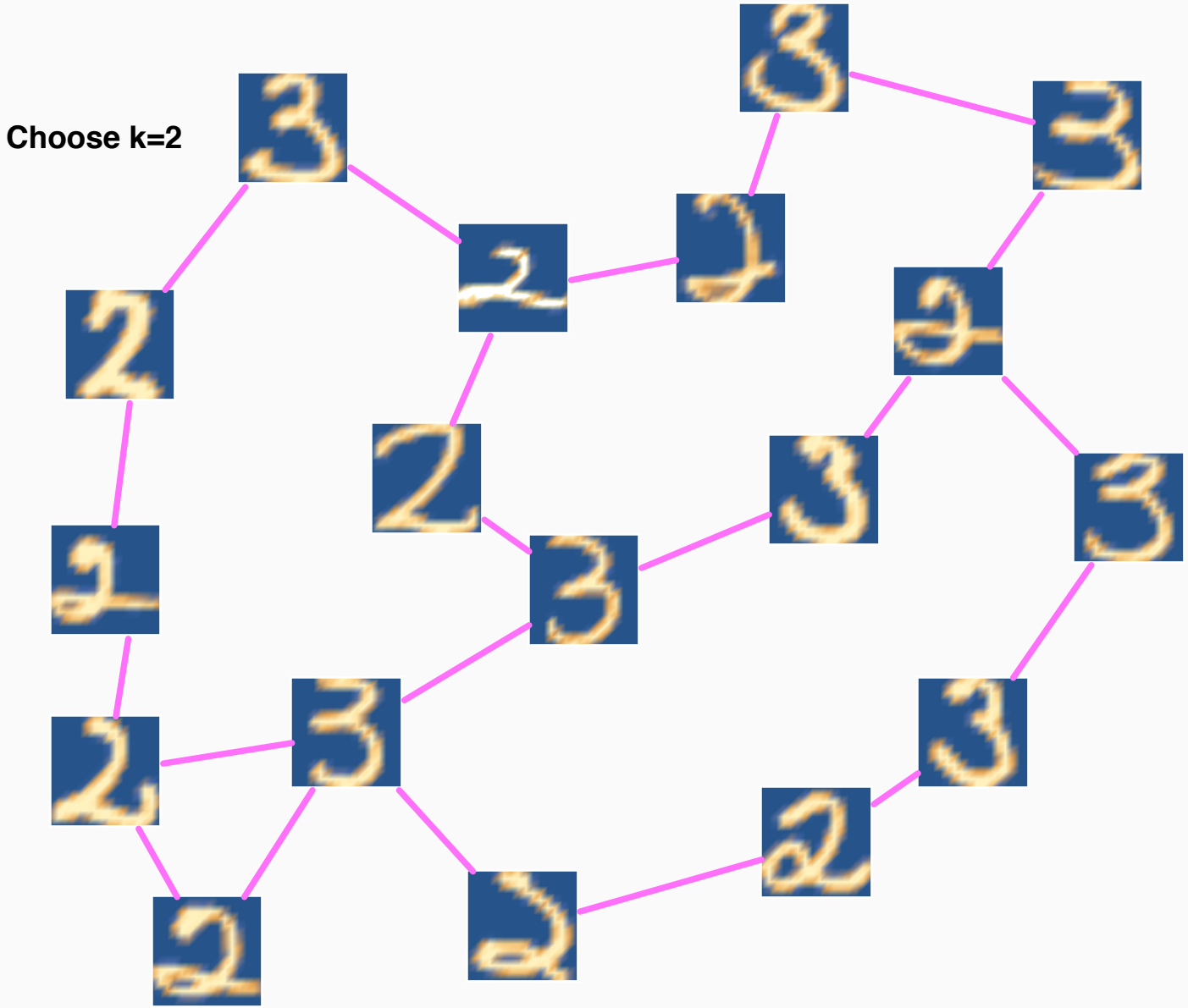
Comment: The quality of predictions depend on the quality of the graph. A continuing area of research.

Example 2-nn (Step 1)



A 2-nn graph (step 1)

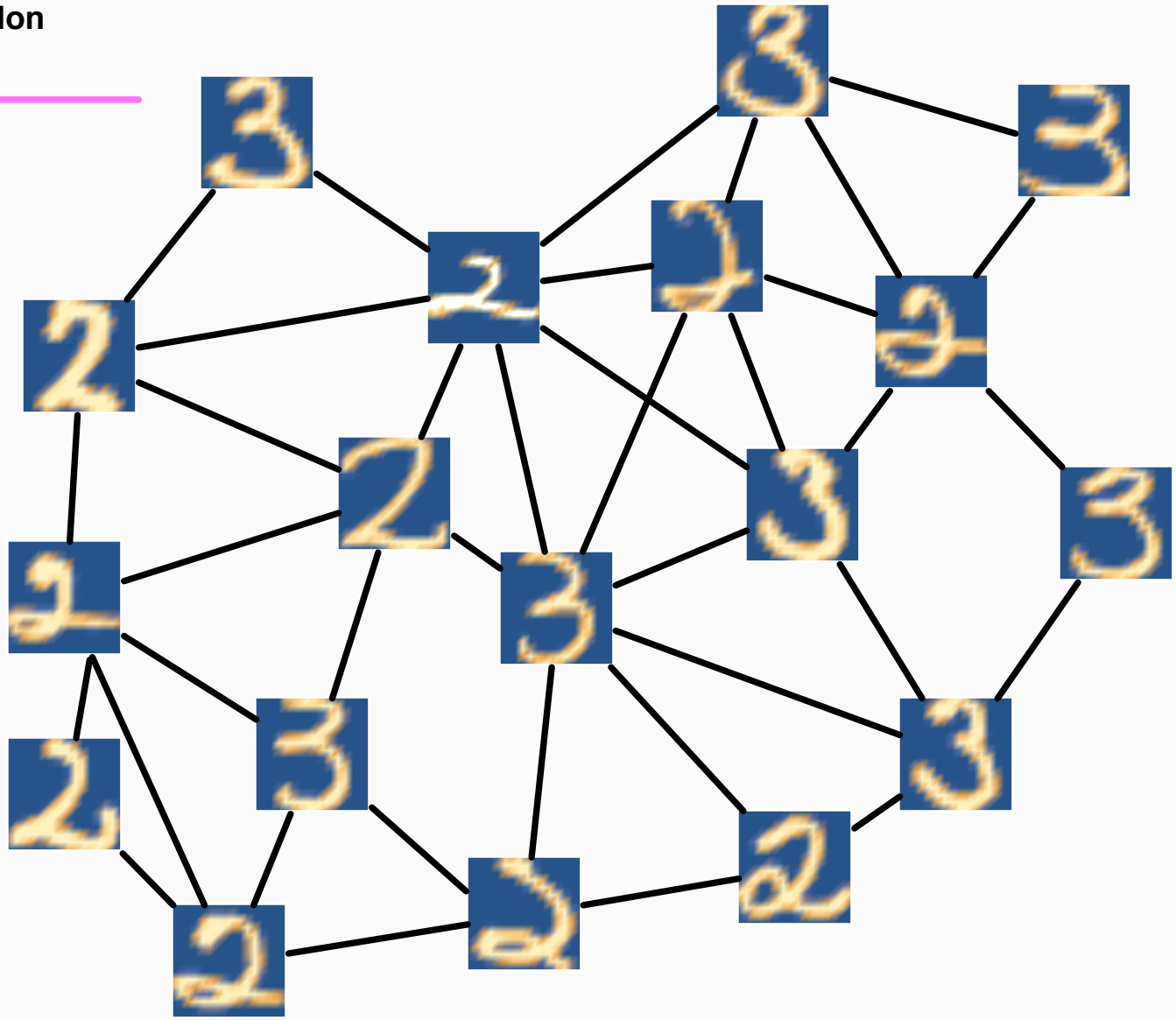
Example 2-nn (Step 2)



A 2-nn graph (step 2)

Example ϵ-ball

Choose Epsilon



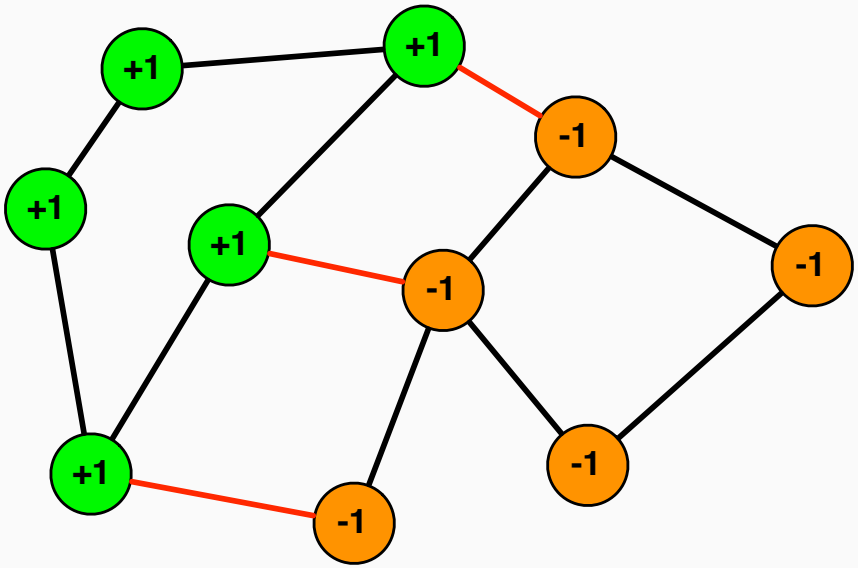
An ϵ-ball graph

Part II

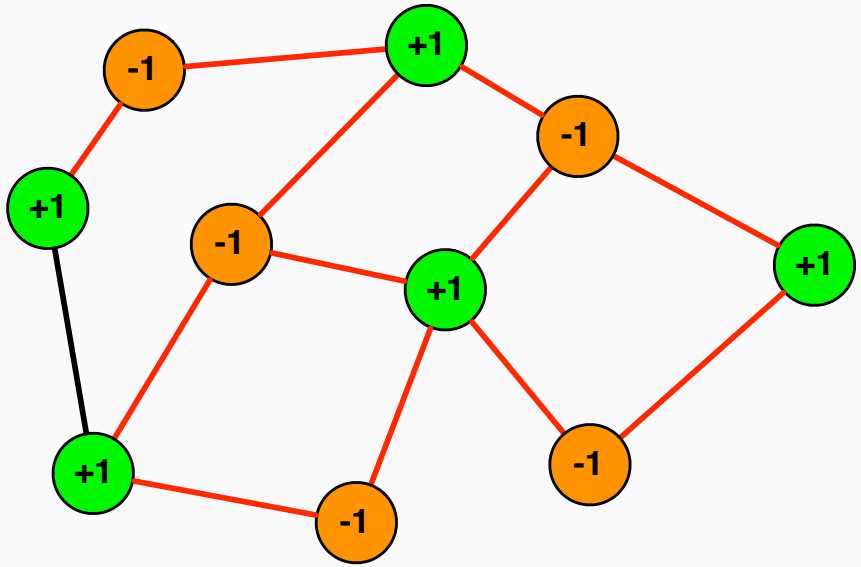
The graph Laplacian

Complexity of a Graph "Classifiers"

- A "Complexity" $\Phi_{\mathcal{G}}(\mathbf{u})$ is associated with a labeling $\mathbf{u} \in \{-1, 1\}^n$



$\Phi_{\mathcal{G}}(\mathbf{u}) = 3$



$\Phi_{\mathcal{G}}(\mathbf{u}) = 12$

The graph Laplacian (matrix) will allow to extend this discrete measure of compexity/smoothness to real values.

The Graph Laplacian

Definition

The graph Laplacian is $\mathbf{L}(G) := D - W$ where W is the (symmetric non-negatively weighted) adjacency matrix and $\mathbf{D} := \mathrm{diag}(d_1, \ldots, d_n)$ is diagonal matrix of (weighted) degrees where $d_v := \sum_{i \neq v} w_{vi}$.

Associated (semi) inner product and (semi) norm

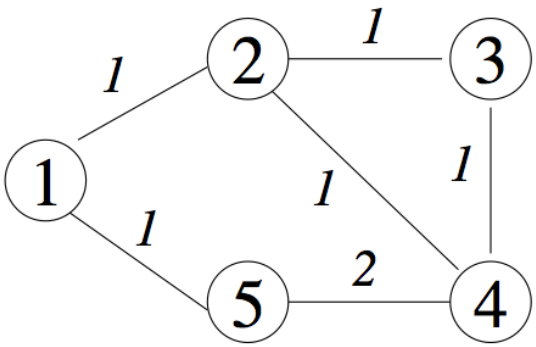
- The laplacian $\mathbf{L}(G)$ is positive semi-definite.
- Define

$$\begin{aligned} \langle \mathbf{u}, \mathbf{v} \rangle_G &:= \mathbf{u}^\top \mathbf{L} \mathbf{v} \\ \|\mathbf{u}\|_G^2 &:= \langle \mathbf{u}, \mathbf{u} \rangle_G = \sum_{(i,j) \in E(G)} w_{ij} (u_i - u_j)^2 \end{aligned}$$

- Define $\mathbb{S}_n := \{\mathbf{u} : \sum_{i=1}^n u_i = 0\}$; assume G is connected
- For $\mathbf{u} \in \mathbb{R}^n$ ($\mathbf{u} \in \mathbb{S}_n$) we've defined (semi) norm and a (semi)inner product.
- Observe that the constant vector $\mathbf{1}$ is the eigenvector with smallest eigenvalue=0.
- In a connected graph $\mathbf{1}$ is the only eigenvector with eigenvalue 0.
- The associated kernel: $\mathbf{K_G}(i,j) = \mathbf{e}_i^\top \mathbf{L}^+ \mathbf{e}_j = L_{ij}^+$ (pseudoinverse)
- Observe that $\phi : [n] \to \mathbb{S}_n$ where $\phi(i) := \mathbf{e}_i^\top \mathbf{L}^+$. is feature for the kernel $\mathbf{L}^+$.
- Checking

$$\langle \phi(i), \phi(j) \rangle = \langle \mathbf{e}_i^\top \mathbf{L}^+, \mathbf{e}_j^\top \mathbf{L}^+ \rangle_G = \mathbf{e}_i^\top \mathbf{L}^+ \mathbf{L} \mathbf{e}_j^\top \mathbf{L}^+ = \mathbf{e}_i^\top \mathbf{L}^+ \mathbf{L} \mathbf{L}^+ \mathbf{e}_j = \mathbf{e}_i^\top \mathbf{L}^+ \mathbf{e}_j = L_{ij}^+$$

Laplacian Example



$$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 3 & -1 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & -1 & -1 & 4 & -2 \\ -1 & 0 & 0 & -2 & 3 \end{pmatrix}$$

Figure 1. A Graph on five vertices and its Laplacian matrix. The weights of edges are indicated by the numbers next to them. All edges have weight 1, except for the edge between vertices 4 and 5 which has weight 2.

Example from *Algorithms, Graph Theory, and Linear Equations in Laplacian Matrices* (D. Spielman 2010)

The *weights* on the edges represent the strength of a connection. A larger weight indicates a strong connection. If the weight is 0 there is then no edge.

Checking that $\mathbf{u}^\top L\mathbf{u} = \sum_{i<j} w_{ij}(u_i - u_j)^2$

$$
\begin{aligned}
\|\mathbf{u}\|_G^2 &:= \langle \mathbf{u}, \mathbf{u} \rangle_G \\
&= \mathbf{u}^\top (D - W)\mathbf{u} \\
&= \sum_{i,j} D_{ij} u_i u_j - \sum_{i,j} w_{ij} u_i u_j \\
&= \sum_i d_i u_i^2 - 2 \sum_{i<j} w_{ij} u_i u_j \\
&= \sum_i (\sum_{j \neq i} w_{ij}) u_i^2 - 2 \sum_{i<j} w_{ij} u_i u_j \\
&= 2 \sum_{i<j} w_{ij} (u_i^2) - 2 \sum_{i<j} w_{ij} u_i u_j \\
&= \sum_{i<j} w_{ij} (u_i^2 + u_j^2) - 2 \sum_{i<j} w_{ij} u_i u_j \\
&= \sum_{i<j} w_{ij} (u_i - u_j)^2
\end{aligned}
$$

Part III

Spectral Clustering

Spectral Clustering

To divide the graph into natural clusters without labels.

Objectives

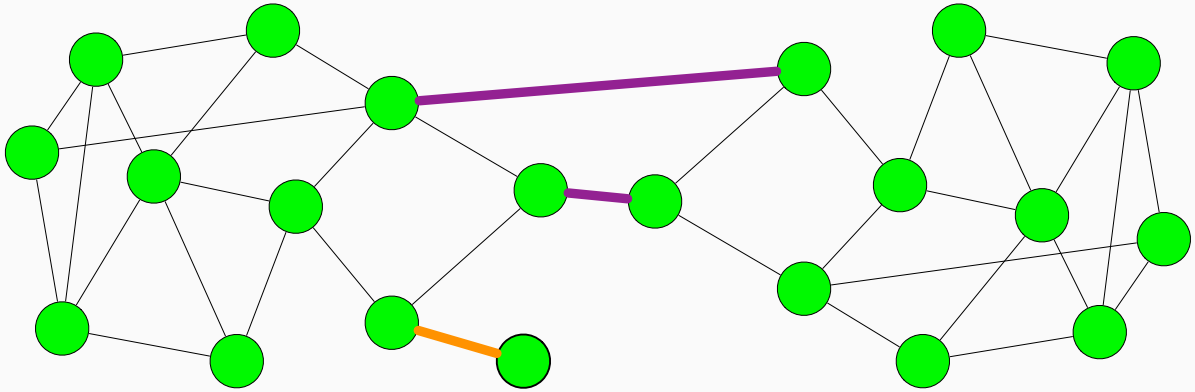
Mincut:

$$\text{cut}(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

RatioCut:

$$\text{RatioCut}(A, B) = \text{cut}(A, B) \left(\frac{1}{|A|} + \frac{1}{|B|} \right)$$

Example



The **Mincut** and the **RatioCut**.

Observations

1. The Mincut can be computed efficiently in $O(n^3)$ time
2. The RatioCut is NP-hard
3. The mincut is not useful for clustering as it does not capture the idea of balanced clusters unlike RatioCut.
4. Both objectives can be generalised to K class case.
5. A third objective called *normalised cuts* is also used commonly in practice. Approximated in much the same way as the BalancedCut but with a "normalised Laplacian" for space reasons we omit.

- Since we cannot compute exactly and efficiently we will approximate!

Relaxing Balanced Cuts – 1

For simplicity we restrict the problem before relaxing.

Problem: Perfectly Balanced Cuts

$$\min_{A,B:|A|=|B|} \text{cut}(A,B)$$

Still NP-Hard.

We will represent a clustering by a vector $\mathbf{u} \in \{-1,1\}^n$ so that

$$u_i := \begin{cases} 1 & i \in A \\ -1 & i \in B \end{cases}$$

Observe that now,

$$\text{cut}(A,B) = \frac{1}{4}\sum_{i<j} w_{ij}(u_i - u_j)^2 = \frac{1}{4}\mathbf{u}^\top \mathbf{L}\mathbf{u}$$

and that

$$|A| = |B| \implies \sum_{i=1}^{n} u_i = 0 \implies \mathbf{u} \perp \mathbf{1}$$

Perfectly Balanced Cuts (Objective rewritten)

$$\min_{\mathbf{u}} \mathbf{u}^\top \mathbf{L}\mathbf{u} \text{ s.t. } \quad \mathbf{u} \in \{-1,1\}^n, \quad \mathbf{u} \perp \mathbf{1}, \quad \|\mathbf{u}\|^2 = n$$

The term $\|\mathbf{u}\|^2 = n$ is redundant. (Why?) we introduce as it will be useful for our relaxation of the problem.

Relaxing Balanced Cuts – 2

Relaxed Balanced Cuts (Objective rewritten)

$$\min_{\mathbf{u}} \mathbf{u}^\top \mathbf{L}\mathbf{u} \text{ s.t.} \quad \mathbf{u} \in \mathbb{R}^n, \quad \mathbf{u} \perp \mathbf{1}, \quad \|\mathbf{u}\|^2 = n \tag{1}$$

Relaxing $\mathbf{u}$ to the reals, leads to an efficiently solvable problem. The key is the following *variational characterisation* of eigenvalues.

Generalised Rayleigh-Ritz

If $\lambda_1 \leq \cdots \leq \lambda_n$ are the eigenvalues of a real-valued symmetric matrix $\mathbf{M}$ and $\mathbf{v}_1, \ldots, \mathbf{v}_n$ are the corresponding eigenvectors then we have the following,

$$\begin{aligned} \lambda_k &= \min_{\mathbf{u} \neq \mathbf{0}} \frac{\mathbf{u}^\top \mathbf{M}\mathbf{u}}{\mathbf{u}^\top \mathbf{u}} : \mathbf{u} \perp \mathbf{v}_1, \ldots, \mathbf{v}_{k-1} \\ &= \min_{\mathbf{u} \neq \mathbf{0}} \mathbf{u}^\top \mathbf{M}\mathbf{u} : \mathbf{u} \perp \mathbf{v}_1, \ldots, \mathbf{v}_{k-1}, \quad \mathbf{u}^\top \mathbf{u} = 1 \end{aligned}$$

Observe therefore that since $\mathbf{1}$ is the first eigenvector of the Laplacian that the second eigenvector (up to scaling by a constant) is then the solution of (1).

Relaxing Balanced Cuts – 3

In fact the second eigenvector is also a relaxed solution to balanced cuts without the constraint $|A| = |B|$. Recall,

$$\text{RatioCut}(A, B) = \text{cut}(A, B)\left(\frac{1}{|A|} + \frac{1}{|B|}\right)$$

Now instead represent a clustering by a vector $\mathbf{u}$ so that

$$u_i := \begin{cases} \sqrt{\frac{|B|}{|A|}} & i \in A \\ -\sqrt{\frac{|A|}{|B|}} & i \in B \end{cases}$$

Observe that now,

$$\mathbf{u}^\top \mathbf{L}\mathbf{u} = \left(\frac{|B|}{|A|} + \frac{|A|}{|B|} + 2\right)\text{cut}(A, B) = (|A| + |B|)\text{RatioCut}(A, B)\,,$$

and thus with $\mathbf{u}$ relaxed then (1) is also the solution to relaxed RatioCut.

In practice

We have a real-valued solution we need to convert to a discrete solution. Simplest is to use $\text{sign}\,\mathbf{u}$ in practice there are a number of heuristics that tend to work better based around finding a "natural gap" in the "middle" of the range u_i values. Likewise we may wish to extend the objective to include K classses. See [L07] for details.

Part IV

The graph Laplacian

Building graph classifiers

Algorithmic frameworks

- Minimum cut
- Laplacian
- p-Laplacian
- Kernel (Laplacian-based) [Not discussed, many graph kernels serve as alternatives to $\mathbf{L}^+$.]
- Markov Random Field (Ising Model) [Not discussed]

Recall: Regularization

As we have defined a "complexity" (here a semi-norm), we may now immediately apply regularization.

Idea

We obtain our predictor $f^* \in \mathcal{F}$,

$$f^* := \operatorname{argmin}_{f \in \mathcal{F}} \quad \text{error}(\text{“training data”}, f) + \lambda \text{complexity}(f)$$

Example: Ridge regression

$$\mathbf{w}^* := \arg\min_{\mathbf{w}} \sum_{i=1}^{m} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 + \lambda \sum_{\ell=1}^{n} w_\ell^2 \equiv (\mathbf{y} - X\mathbf{w})^\top (\mathbf{y} - X\mathbf{w}) + \lambda \mathbf{w}^\top \mathbf{w}$$

Interpolation

Today we simplify and focus on the **interpolation** case.

1. We assume there exists at least one $f^* \in \mathcal{F}$ that predicts perfectly on the training examples.
2. We select the zero-error f^* with minimum complexity.

Alternately,

$$f_\lambda := \text{argmin}_{f \in \mathcal{F}} \quad \text{error}(\text{“training data”}, f) + \lambda \, \text{complexity}(f)$$

and set

$$f^* := \lim_{\lambda \to 0} f_\lambda$$

Part V

Minimum cut transduction

Minimum-cut

The minimum cut method:

- For each edge we may also introduce weights $(w_e)_{e \in \mathbf{E}(\mathcal{G})} \subset [0, \infty)$
- Given a set of labeled vertices $(v \in (V(\mathcal{G}), y \in \{-1, 1\})$

$$\{(v_{i_1}, y_1), (v_{i_2}, y_2), \ldots (v_{i_\ell}, y_\ell)\}$$

- The hypothesis vector is given by

$$\hat{\mathbf{u}}_t = \underset{\mathbf{u} \in \{-1,1\}^n}{\arg\min} \left\{ \sum_{(i,j) \in E(\mathcal{G})} : w_{ij} |u_i - u_j| : u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell \right\}$$

- This may be solve by a "min-cut"/"max-flow" algorithm in cubic time or by linear programming.
- Introduced by Blum and Chawla in 2001. First graph-based SSL algorithm
- Inspired many other algorithms but not popular in practice since it has tendency to produced unbalanced labelings. The same issue

Minimum cut (Observations)

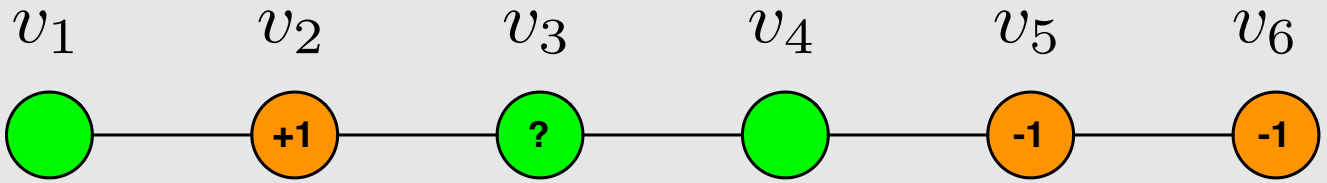
Observations

- The objective's minima is unchanged when relaxed to real values,

$$\min_{\mathbf{u}\in\{-1,1\}^n}\{\sum_{(i,j)\in E(\mathcal{G})} : w_{ij}|u_i - u_j| : u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell\} =$$

$$\min_{\mathbf{u}\in\mathbb{R}^n}\{\sum_{(i,j)\in E(\mathcal{G})} : w_{ij}|u_i - u_j| : u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell\}$$

- The problem is *ill-posed* (no unique solution). E.g., how should we label v_3,



Part VI

Laplacian-based transduction

Laplacian-based

Laplacian-based method:

- Given a weighted graph with labeled vertices ($y \in \{-1, 1\}$)

$$\{(v_{i_1}, y_1), (v_{i_2}, y_2), \ldots (v_{i_\ell}, y_\ell)\}$$

- The interpolant vector is given by

$$\bar{\mathbf{u}}_t = \arg\min_{\mathbf{u} \in \mathbb{R}^n} \{ \sum_{(i,j) \in E(\mathcal{G})} : w_{ij} |u_i - u_j|^2 : u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell \}$$

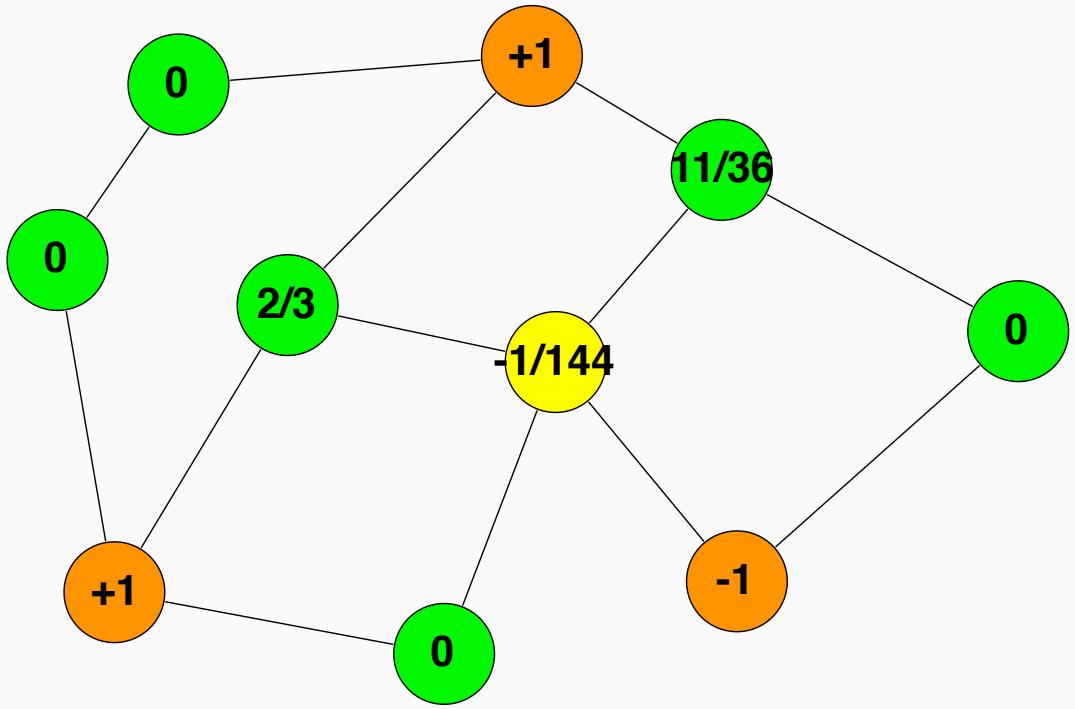
- To predict at vertex v use $\hat{u}_v := \text{sign}(\bar{u}_v)$.
- May be optimized in cubic time by solving a linear system of equations.
- It is known by many names not limited to *harmonic minimization*, *label propagation*, *Laplacian interpolated regularization*.
- Variations and embellishments on the above account for a large portion of Graph-based SSL. This method has performed well on a

Optimisation by Consensus for Laplacian interpolation

- The computation is determined by a network of nodes.
- Each nodes' computation is
  1. Simple
  2. Local (depends only on neighbors)
  3. Parallel
  4. Asynchronous
- Fault tolerant
- Two types of nodes fixed and free
- Network topology identified with topology of data

Consensus Algorithm (Relaxation)

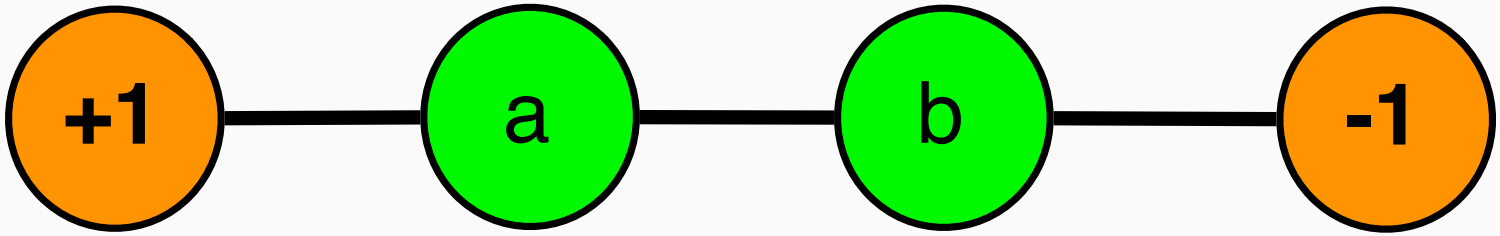
1. Select a free vertex i (Green) uniformly at random
2. Average neighbors $u'(i) = \frac{\sum_{j \in \Gamma(i)} u_j}{d_i}$
3. Goto 1



- For a weighted graph use a weighted mean (via W)

Four Node Problem

Exercise: Consider the graph,



1. What do nodes a and b converge to?
2. Why? Give justification.

Consensus Convergence Proof (Step 1)

Sketch

1. Define "energy" of a labelling u as the squared-norm induced by the Laplacian
2. Denote $\mathbf{u}$ before selection after selection $\mathbf{u}'$
3. Observe $\|\mathbf{u}\|^2 \geq \|\mathbf{u}'\|^2 \geq \|\mathbf{u}''\|^2 \geq \ldots \geq 0$ Why?!?
4. Hence convergence in "energy"

Proof – Comments

Exercise (difficult [technical]):

- Suppose $\bar{\mathbf{u}}$ denotes the minimum.
- Claim: $\mathbf{u}^{(t)} \to \bar{\mathbf{u}}$ as $t \to \infty$.

Observations

Theorem (harmonic solution)

$$\bar{\mathbf{u}}_t = \arg\min_{\mathbf{u}\in\mathbb{R}^n}\{\sum_{i<j} : w_{ij}|u_i - u_j|^2 : u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell\}$$

implies

$$\bar{u}_i = \frac{\sum_{j=1}^n w_{ij} u_j}{d_i} \qquad (i = \ell+1, \ldots, n)$$

$$\bar{u}_i = y_i \qquad (i = 1, \ldots, \ell)$$

- Consensus solves of a Linear System
- Consensus is a type of "coordinate" descent
- Many "efficient" methods to solve linear systems
- Why consensus?
  1. Simple
  2. Parallel

Part VII

Interpreting Laplacian-based transduction

Explanations

Why does it work? Types of explanations? Justifications?

1. As a resistive network
2. As a random walk on a network
3. Under certain conditions the graph-Laplacian may approximate the Reimannian Laplacian of the data Manifold
4. Via various predictive performance guarantees

Connectivity: Is the shared idea in "1" and "2". The densities of inputs points are reflected in the connectivity of the built graph, which modifies the "*ambient*" metric of the inputs.

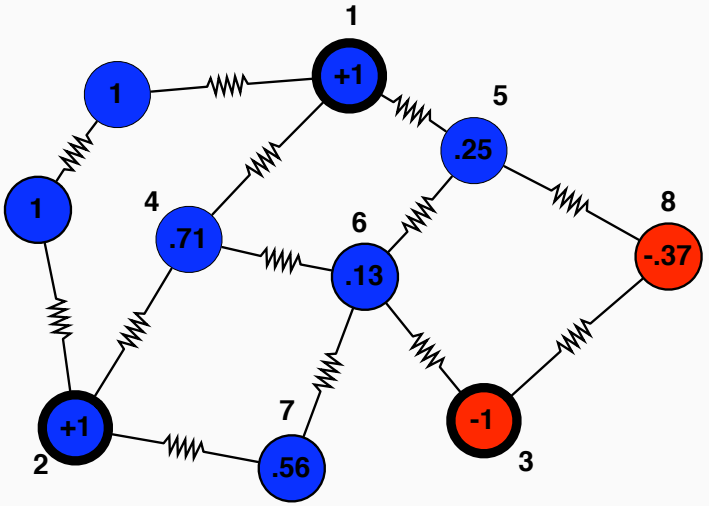
Graph to resistive network

- Identify graph with a resistive network
- Labels are voltage constraints; weights on correspond to inverse resistances.
- Power of a labeling

$$P(\mathbf{u}) := \|\mathbf{u}\|_{\mathcal{G}}^2 = \sum_{(i,j)\in E(\mathcal{G})} w_{ij}|u_i - u_j|^2 = \mathbf{u}^\top L\mathbf{u}$$

- Label by minimizing power

$$\bar{\mathbf{u}} = \arg\min_{\mathbf{u}\in\mathbb{R}^n}\{P(\mathbf{u}) : u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell\}$$

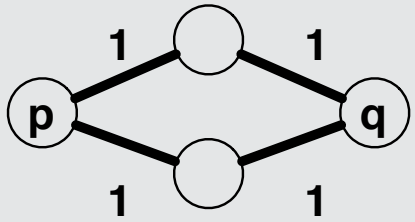


Effective Resistance – 1

Definition

For a pair of vertices the effective resistance between these vertices is the resistance of the total network when a voltage source is connected across them, i.e., it is the voltage difference needed to induce a *unit* current flow between between a pair of vertices.

Example



- *Geodesic distance* is 2 from v_p to v_q
- *Effective resistance* is 1 from v_p to v_q

Effective Resistance – 2 (Kirchoff's Circuit Laws)

Junction rule

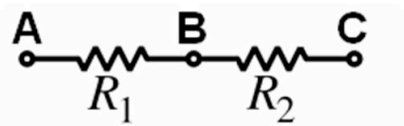
$$\sum I_{in} = \sum I_{out}$$

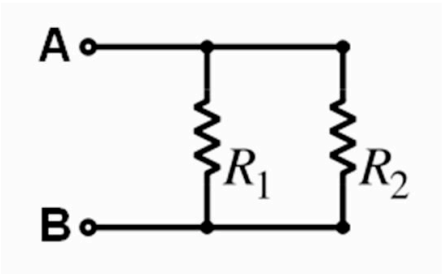
Effective resistance

$$R_{eff} = \frac{V_{tot}}{I_{tot}}$$

Loop rule

$$\sum V_{loop} = 0$$





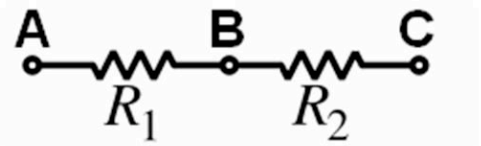
Effective Resistance – 3 (Series Circuit)

Series circuit:

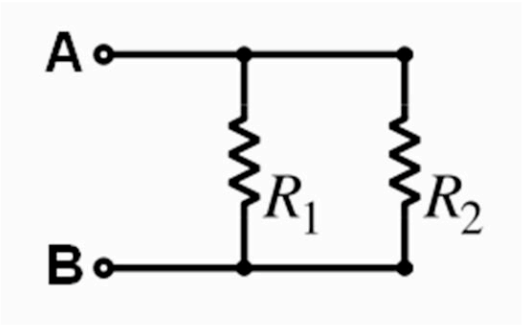
$$\begin{aligned} V_{tot} &= V_1 + V_2 \\ I_{tot} &= I_1 = I_2 \\ R_{eff} &= \frac{V_1 + V_2}{I_{tot}} \\ &= \frac{V_1}{I_1} + \frac{V_2}{I_2} \\ &= R_1 + R_2 \end{aligned}$$

Generalises to:

$$R_{eff} = R_1 + R_2 + \cdots + R_n$$



Effective Resistance – 4 (Parallel Circuit)



Parallel circuit:

$$
\begin{aligned}
V_{tot} &= V_1 = V_2 \\
I_{tot} &= I_1 + I_2 \\
R_{eff} &= \frac{V_{tot}}{I_1 + I_2} \\
&= \frac{V_1}{\frac{V_1}{R_1} + \frac{V_2}{R_2}} \\
&= \frac{V_1}{V_1(\frac{1}{R_1} + \frac{1}{R_2})} \\
\Rightarrow \frac{1}{R_{eff}} &= \frac{1}{R_1} + \frac{1}{R_2}
\end{aligned}
$$

Generalises to:

$$\frac{1}{R_{eff}} = \frac{1}{R_1} + \frac{1}{R_2} + \cdots + \frac{1}{R_n}$$

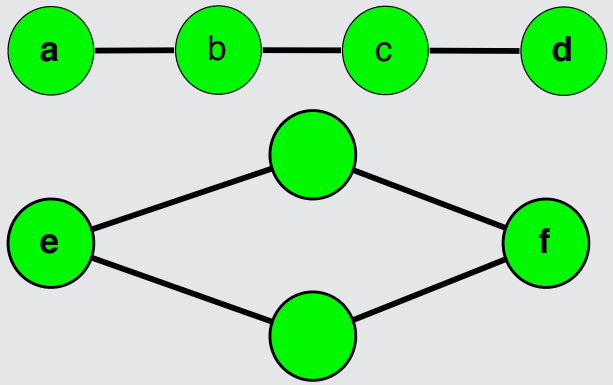
Effective Resistance – 6

(Computation via optimization): One may prove that

$$r_{\mathbf{G}}(p,q) = \frac{1}{\min_{\mathbf{u}\in\mathbb{R}^n}\{\|\mathbf{u}\|_{\mathbf{G}}^2 : u_p = 1, u_q = 0\}}$$

Exercise

1. Compute the effective resistance between a and d
2. Compute the effective resistance between a and c
3. Compute the effective resistance between e and f
4. Challenge: Use non-unit resistors on the edges



Effective Resistance – 7 (Kernel)

In fact the **effective resistance** is the squared distance induced by the kernel $\mathbf{L}^+$.

Theorem [KR93]

Effective resistance between $\mathbf{v}_p$ and $\mathbf{v}_q$ is

$$r_{\mathbf{G}}(p,q) = L^+_{pp} + L^+_{qq} - 2L^+_{pq}$$

Proof – 1

Lemma

Let $\mathbf{x} \in \mathcal{H}$ then

$$\|\mathbf{x}\|^{-2} = \min_{\mathbf{w} \in \mathcal{H}} \left\{ \|\mathbf{w}\|^2 : \langle \mathbf{w}, \mathbf{x} \rangle = 1 \right\}.$$

Proof Of Lemma

Observe that $\frac{\mathbf{x}}{\|\mathbf{x}\|^2}$ is a feasible solution to the minimization.

Suppose $\mathbf{v} = \frac{\mathbf{x}}{\|\mathbf{x}\|^2} + \alpha \mathbf{z}$ is a feasible solution with $\|\mathbf{v}\|^2 < \|\mathbf{x}\|^{-2}$.

Since $\langle \mathbf{v}, \mathbf{x} \rangle = 1$ then $\langle \mathbf{z}, \mathbf{x} \rangle = 0$. Hence

$$\|\mathbf{v}\|^2 = \|\mathbf{x}\|^{-2} + 2\alpha \|\mathbf{x}\|^{-2} \langle \mathbf{z}, \mathbf{x} \rangle + \|\mathbf{z}\|^2 = \|\mathbf{x}\|^{-2} + \|\mathbf{z}\|^2$$

which contradicts the supposition.

Proof – 2

Proof of [KR93]

Recall,

$$r_{\mathbf{G}}(p,q) = \frac{1}{\min_{\mathbf{u}\in\mathbb{R}^n}\{\|\mathbf{u}\|_{\mathbf{G}}^2 : u_p = 1, u_q = 0\}},$$

Observe that

$$\begin{aligned}\min_{\mathbf{u}\in\mathbb{R}^n}\{\|\mathbf{u}\|_{\mathbf{G}}^2 : u_p = 1, u_q = 0\} &= \min_{\mathbf{u}\in\mathbb{R}^n}\{\|\mathbf{u}\|_{\mathbf{G}}^2 : u_p - u_q = 1\} \\ &= \min_{\mathbf{u}\in\mathbb{S}_n}\{\|\mathbf{u}\|_{\mathbf{G}}^2 : u_p - u_q = 1\} \\ &= \min_{\mathbf{u}\in\mathbb{S}_n}\{\|\mathbf{u}\|_{\mathbf{G}}^2 : \langle \mathbf{e}_p^\top\mathbf{L}^+ - \mathbf{e}_q^\top\mathbf{L}^+, \mathbf{u}\rangle_{\mathbf{G}} = 1\}\end{aligned}$$

The first two equalities follow from the fact that any $c \neq 0$ that $\|\mathbf{u}\|_{\mathbf{G}}^2 = \|\mathbf{u} + c\mathbf{1}\|_{\mathbf{G}}^2$ and the third follows since

$$\langle \mathbf{e}_p^\top\mathbf{L}^+, \mathbf{u}\rangle_{\mathbf{G}} = \mathbf{e}_p^\top\mathbf{L}^+\mathbf{L}\mathbf{u} = u_p$$

for any $\mathbf{u} \in \mathbb{S}_n$. Now using the Lemma on the final equality we have

$$r_{\mathbf{G}}(p,q) = \|\mathbf{e}_p^\top\mathbf{L}^+ - \mathbf{e}_q^\top\mathbf{L}^+\|_{\mathbf{G}}^2 = (\mathbf{e}_p^\top\mathbf{L}^+ - \mathbf{e}_q^\top\mathbf{L}^+)\mathbf{L}(\mathbf{e}_p^\top\mathbf{L}^+ - \mathbf{e}_q^\top\mathbf{L}^+)^\top = L_{pp}^+ + L_{qq}^+ - 2L_{pq}^+.$$

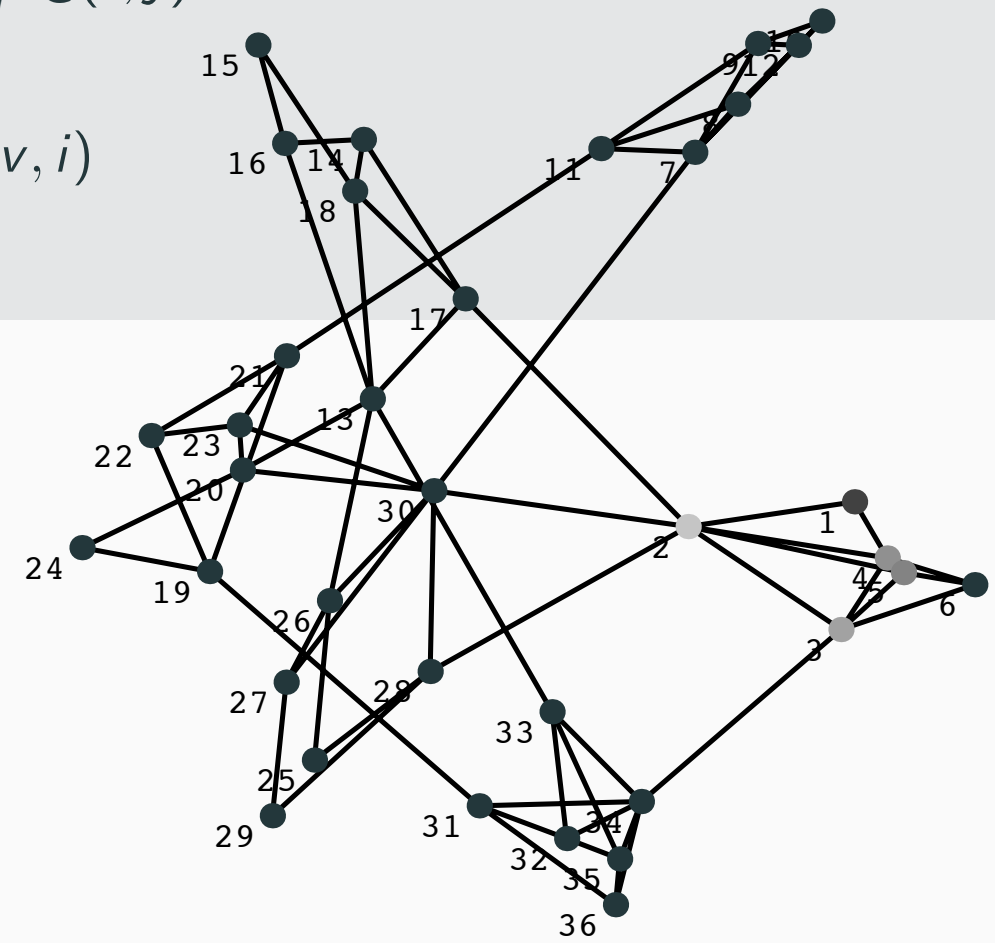
□

Effective resistance 8 – (Graph connectivity)

The kernel L^+, "contains" the following connectivity information.

Inverse Connectivities

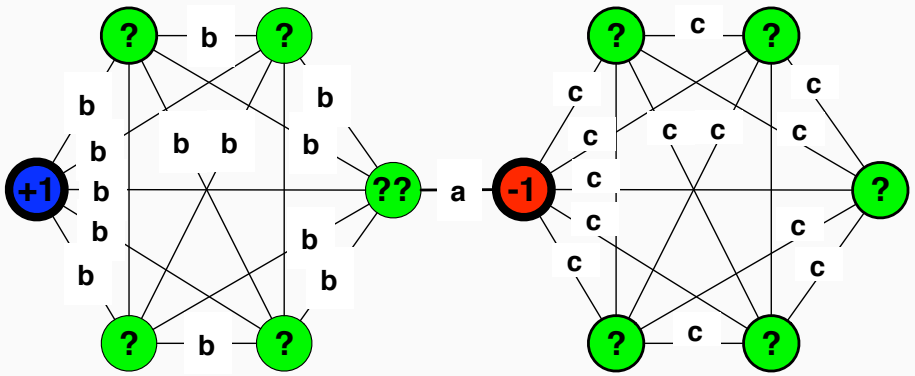
- Graph : $R_{\text{tot}} := \sum_{i>j} r_{\mathbf{G}}(i,j)$
- Vertex :
  1. $R(v) := \sum_{i=1}^{n} r_{\mathbf{G}}(v,i)$
  2. $L^+_{vv} = \frac{R(v)}{n} - \frac{R_{\text{tot}}}{n^2}$



The "Intuition"

From "distance" to "resistance"

The base global "distance" $d(p, q)$ is *adapted* according to the empirical distances via the effective resistance $r(p, q)$.



Two $m-$ *cliques* connected by an edge with distances $a < b$

- What is the label at "??"
- Effective resistance between vertices within a clique whose edge-resistance are $\leq d$ is $r_{\text{m-clique}} \leq \frac{2d}{m}$
- If $\frac{2b}{m} < a$ then label "$+1$"
- Conclusion: labelings respect cluster structure

Random walk on graphs – 1

Definition

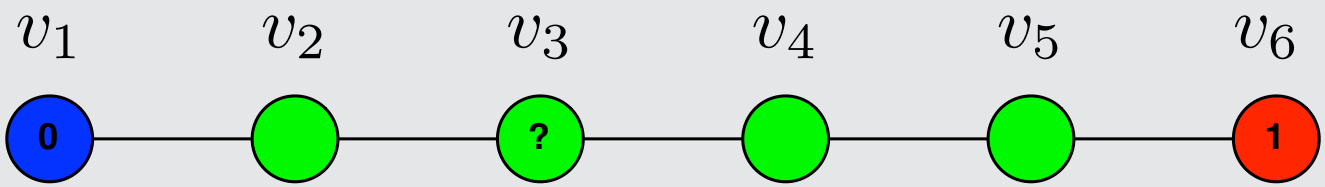
The **transition probability** P_{ij} of going from vertex i to vertex j on a "step" is

$$P_{ij} := \frac{w_{ij}}{\sum_i w_{ij}}$$

where W is the symmetric weighted adjacency matrix (L is the associated Laplacian).

Example

Consider the following path graph with unit weights on edges. What is the probability that we reach vertex labeled "1" before we reach the vertex labeled "0"?



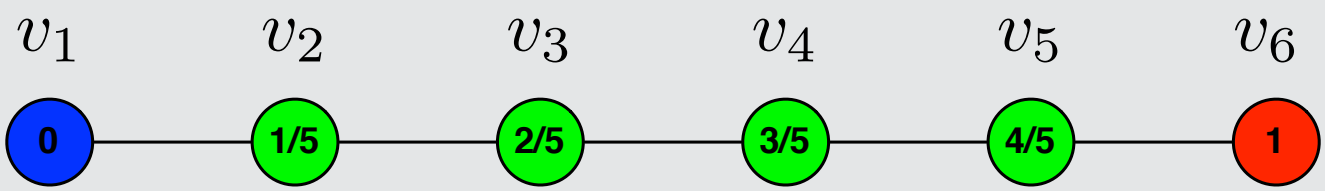
Random walk on graphs – 2

Observation

Let p_i denote the probability that from vertex i we reach the vertex labeled "1" before we reach the vertex labeled "0."

Claim

For vertices $v_2, \ldots, v_5$ we have $p_i = \frac{p_{i-1}+p_{i+1}}{2}$. Why?



Observe that the boundary conditions are met for vertices v_1 and v_6.

Random walk on graphs – 3

Theorem

Given an an unweighted graph G with adjacency matrix W, a set of vertices V_0 labeled "0", and a set vertices V_1 labeled "1" and let p_i denote the probability that a random walked started a vertex i reaches a vertex in V_1 before reaching a vertex in V_0. Then

$$\mathbf{p} = \arg\min_{\mathbf{p}\in\mathbb{R}^n}\{\sum_{i<j} : w_{ij}|p_i - p_j|^2 : (p_k = 0)_{v_k\in V_0}, (p_k = 1)_{v_k\in V_0}\}$$

Proof. The equations and the boundary conditions that determine the random walk are isomorphic to the boundary conditions and equations of label propagation.

Summary. The label propagation solution has natural interpretations in terms of voltages and random walks.

Random walk on graphs – 4

We also have the following result connecting effective resistance to commute time.

Definition

The commute time $c_{\mathcal{G}}(i,j)$ between vertex i and vertex j is the *expected* time to travel from i to j and back again.

Theorem

The commute time $c_{\mathcal{G}}(i,j)$ between i and j is the twice the total degree times the effective resistance.

$$c_{\mathcal{G}}(i,j) := r_{\mathcal{G}}(i,j) \sum_{i,j}^{n} w_{ij} \, .$$

Part VIII

Generalising Laplacian-based transduction

Overview

- We generalize the Laplacian norm to a p-Laplacian norm
- Motivation
  1. Aim to control the sparsity of the cut in analogy to lasso
  2. Tradeoff the weighting of cut-size verse geodesic distance
- A generalized p-network theory follows from a definition of p-energy
- When $p = 1$ we have a "pipe" network, $p = 2$ an electrical network, $p \to \infty$ "shortest path network"

Generalising to the p-Laplacian

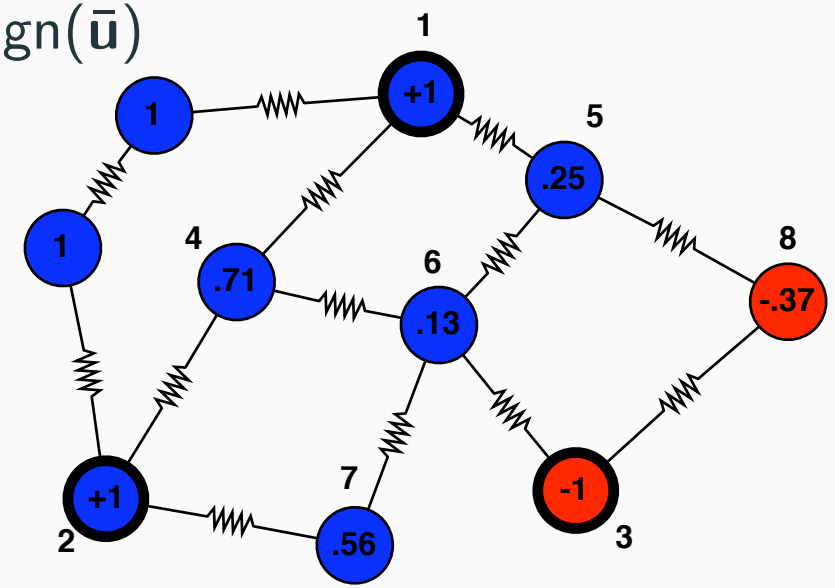
- Identify graph with a p-resistive network
- Labels are voltage constraints.
- Redefine energy (power) as

$$P_p(\mathbf{u}) := \|\mathbf{u}\|_{\mathcal{G},p}^p = \sum_{(i,j)\in E(\mathbf{G})} w_{ij}|u_i - u_j|^p$$

- Label by minimizing energy

$$\bar{\mathbf{u}} = \arg\min_{\mathbf{u}\in\mathbb{R}^n}\{P_p(\mathbf{u}) : u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell\}$$

- Predict with $\operatorname{sign}(\bar{\mathbf{u}})$



Properties of p-Resistive Networks

Analogues of usual "electric network" theory now follow ...

1. Kirchoff's laws
2. Ohm's law
3. Conservation of energy principle
4. Rayleigh's monotonicity principle
5. Black box principle (2-port)
6. The "rules" of resistors in parallel and in series
7. Triangle Inequality

We will derive some of these.

Kirchoff's current law – 1

Definitions

1. Edge Resistance : $\pi_{ij} = \frac{1}{w_{ij}}$
2. Voltage constraints : $\mathcal{S} := (u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell)$
3. Constrained vertices : $V_{\mathcal{S}} := \{i_1, \ldots, i_\ell\}$.
4. (Key Definition) : Energy with respect to $\mathcal{S}$

$$P(\mathcal{S}) := \min_{\mathbf{u} \in \mathbb{R}^n} \{P(\mathbf{u}) : u_{i_1} = y_1, \ldots, u_{i_\ell} = y_\ell\}. \tag{2}$$

Hence at the minimum we have

$$\frac{\partial P(\mathbf{u})}{\partial u_i} = 0 \qquad v_i \notin V_{\mathcal{S}}$$

$$\sum_{j:j\sim i} \frac{|u_i - u_j|^{p-1} \mathrm{sgn}(u_i - u_j)}{\pi_{ij}} = 0 \qquad v_i \notin V_{\mathcal{S}}. \tag{3}$$

Kirchoff's current law – 2

Definitions

The *current* from vertex v_i to v_j of a network

$$I_{ij}(\mathcal{S}) := \frac{|u_i - u_j|^{p-1}\mathrm{sgn}(u_i - u_j)}{\pi_{ij}} \tag{4}$$

if $p = 2$ this is *Ohm's law*.

The *net current* from vertex v_i is defined as

$$I_i := \sum_{j:j\sim i} I_{ij} \,.$$

We see that (3) is *Kirchoff's current law* for $\boldsymbol{I}$

$$0 = I_i \quad v_i \notin V_{\mathcal{S}} \tag{5}$$

I.e., the conservation of flow follows from the definition of energy (2)

p-resistance

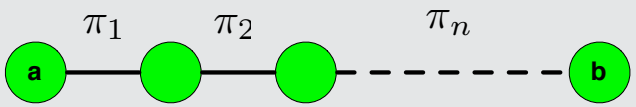
Definition

The (effective) p-resistance between any two vertices a and b is

$$r_p(a,b) = \left[\min_{\mathbf{u}\in\mathbb{R}^n}\{P_p(\mathbf{u}) : u_a = 1, u_b = 0\}\right]^{-1}.$$

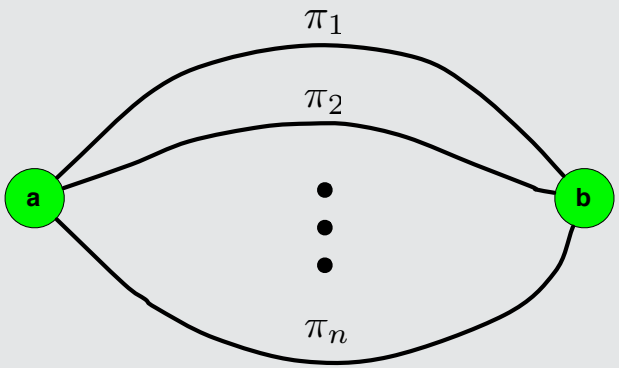
Resistors in series

$$r_{\mathcal{G},p}(a,b) = \left(\sum_{i=1}^{n} \pi_i^{\frac{1}{p-1}}\right)^{p-1}$$



Resistors in parallel

$$r_{\mathcal{G},p}(a,b) = \left(\sum_{i=1}^{n} \frac{1}{\pi_i}\right)^{-1}$$



Note if $p = 1$ then $r_{\mathcal{G},1}(a,b) = \frac{1}{\text{“mincut between } a \text{ and } b\text{”}}$

p-resistance trade-off

Idea : We can gain insight to behaviour of interpolation in a normed space by under understanding the metric induced by the norm. The p-resistance is the metric induced by the p-Laplacian norm. To understand the metric, we consider the simplified cases corresponding to the serial and parallel laws.

Recall p-norm

$$\|\mathbf{x}\|_p := (\sum_{i=1}^{n} x_i^p)^{\frac{1}{p}} \qquad p \in [1, \infty)$$

Observation

1. For resistors in series the "law" is $(\frac{1}{p-1})$-norm on the edge resistances.
2. For resistors in parallel the "law" is -1 – "pseudo-norm" on the edge resistances (triangle inequality fails).
3. The parallel law is independent of p.

Comparing p-resistances

The key cases

- $[p = 1]$ Serial law is a ∞-norm (pipe network) – max of resistances
- $[p = 2]$ Serial law is a 1-norm (electric network) – sum of resistances
- $[p = \infty]$ Serial law is a 0-"pseudo-norm" (geodesic network) – number of resistors

p-Resistive Triangle inequality [technical]

Theorem

Given a graph $\mathcal{G}$ and vertices v_a, v_b, and v_c then

$$r_{\mathcal{G},p}(a,c) \leq \left(r_{\mathcal{G},p}(a,b)^{\frac{1}{p-1}} + r_{\mathcal{G},p}(b,c)^{\frac{1}{p-1}} \right)^{p-1} \qquad p \in [1,\infty)$$

thus

$$r_{\mathcal{G},1}(a,c) \leq \max(r_{\mathcal{G},1}(a,b), r_{\mathcal{G},1}(b,c)) \qquad (p=1)$$

p-Resistance interpretation

- Generalises the mincut ($p = 1$) and harmonic solution ($p = 2$)
- As $p \to \infty$ solution directly connected to geodesic path length ([technical])
- More generally the effective resistance *emphasizes* balances "connectivity" ($p \to 1$) with geodesic path length as $p \to \infty$
- The case $p = 2$ is convenient from an optimization perspective and likewise can combine nicely kernel techniques

Part IX

Mistake Bounds for interpolation with the p-Laplacian

Model : predicting the labeling of a graph

- Aim: learn a function $\mathbf{u} : V \to \{-1, +1\}$ corresponding to a labeling of a graph $\mathcal{G} = (V, E)$ and $V = \{1, \ldots, n\}$.
- Learning proceeds in trials

for $t = 1, \ldots, \ell$ **do**

1. Nature selects $v_t \in V$
2. Learner predicts $\hat{y}_t \in \{-1, +1\}$
3. Nature selects $y_t \in \{-1, +1\}$
4. If $\hat{y}_t \neq y_t$ then mistakes $=$ mistakes $+ 1$

- Learner's goal: *minimize* mistakes
- Bound: mistakes $\leq$ *f(complexity*($\mathbf{u}$), *structure*($\mathcal{G}$))

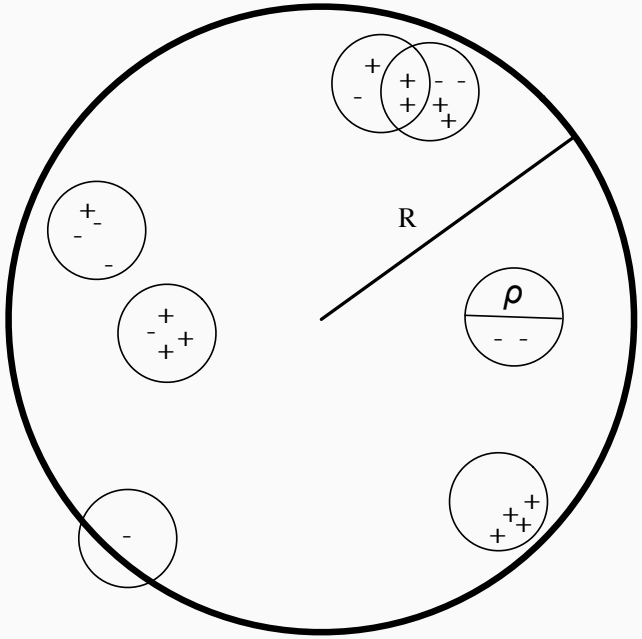
Minimum p-Seminorm Interpolation

Recall that our algorithm works as follows.

Minimum p-seminorm Interpolation

Input: $\{(i_t, y_t)\}_{t=1}^{\ell} \subseteq \{1, 2, \ldots, n\} \times \{-1, 1\}$.
for $t = 1, \ldots, \ell$ **do**
  Receive query: $i_t \in \{1, \ldots, n\}$
  Predict: $\hat{y}_{i_t} = \text{sign}(\mathbf{e}_{i_t}^\top \mathbf{w}_t) = \text{sign}(w_{t,i_t})$
  Receive label: y_{i_t}
  if $\hat{y}_{i_t} = y_{i_t}$ **then** mistakes = mistakes + 1
  $\mathbf{w}_{t+1} = \arg\min_{\mathbf{u} \in \mathbb{R}^n} \{P_p(\mathbf{u}) : u_{i_1} = y_{i_1}, \ldots, u_{i_t} = y_{i_t}\}$

Covering



Input space X of radius R with cover number $\mathcal{N}_\rho = 7$.

- Bounds to be dependent on structure of input space X.
- Novikoff is only dependent on X through radius R.
- Expectation is that a typical ambient input space is only sparsely populated (cf manifold/cluster hypotheses).
- Algorithm will depend on the cover of X.

Mistake Bound

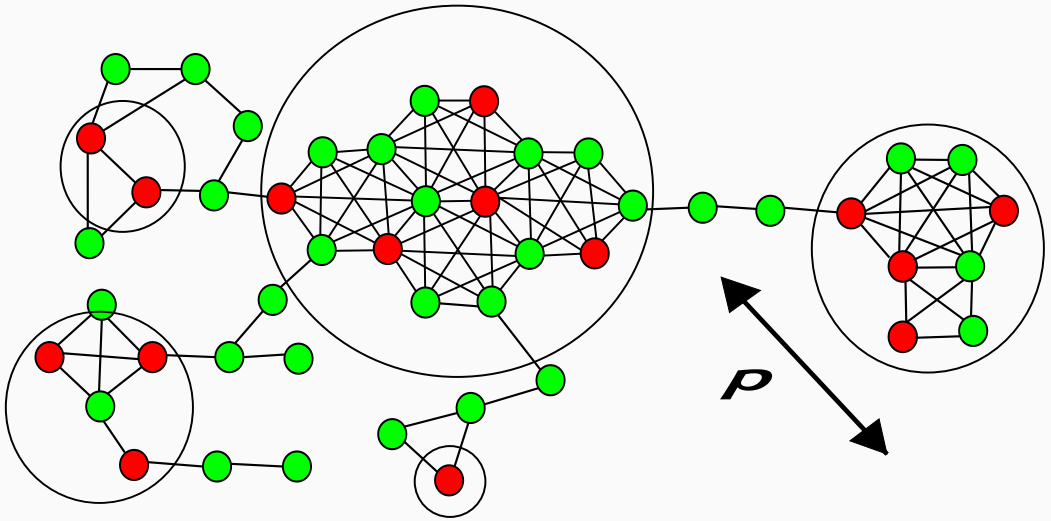
Theorem

Given $p \in (1, 2]$ then

$$M \leq \mathcal{N}_\rho + \frac{[\rho \times P_p(\mathbf{u})]^{\frac{2}{p}}}{p-1}$$

for all $0 < \rho$, and for all consistent $\mathbf{u} \in \mathbb{R}^n$.

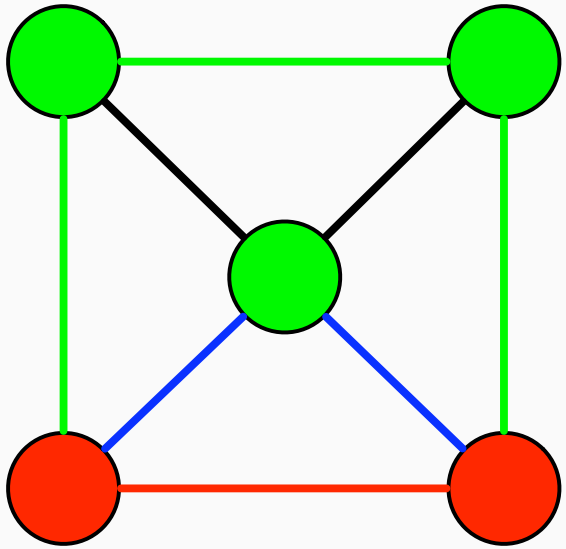
- Relative to any resistive cover w.r.t. r_p
- Both number of covering sets $\mathcal{N}_\rho$ and resistance ρ



Wide Diameter (Tuning)

Definition

A graph has wide diameter (k, Δ_k) if there exist k edge disjoint paths of length no more than Δ_k between all pairs of vertices.



Connectivity: $k = 1$, Wide Diameter: $\Delta_1 = 2$

Connectivity: $k = 2$, Wide Diameter: $\Delta_2 = 2$

Connectivity: $k = 3$, Wide Diameter: $\Delta_3 = 3$

- The wide diameter is used as a basis for upper bounding p-resistance with the resistors in parallel and series laws.

Graph prediction

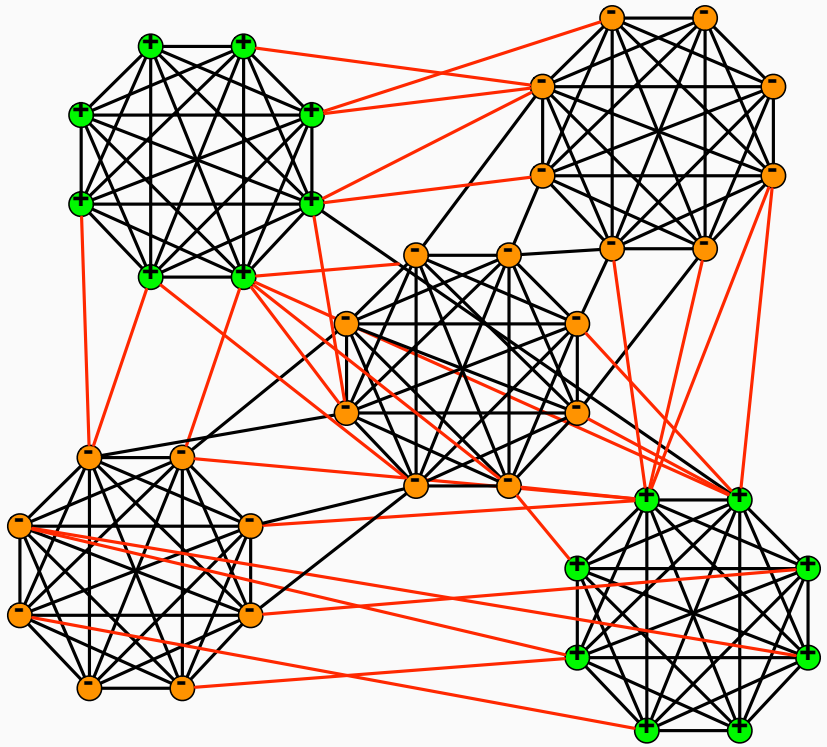
Theorem

The number of mistakes, M, incurred by minimum p-seminorm interpolation with $p := \frac{c}{c-1}$ is bounded by

$$M \leq \begin{cases} N + \frac{4e^2\Phi^2(\mathbf{u})[\log(\Delta_k)+\log(\frac{k}{\Phi(\mathbf{u})})-1]}{k^2} & \frac{k\Delta_k}{\Phi(\mathbf{u})} > e^2, \quad p \in (1,2) \\ N + \frac{4\Phi(\mathbf{u})\Delta_k}{k} & \frac{k\Delta_k}{\Phi(\mathbf{u})} \leq e^2, \quad p = 2 \end{cases}$$

where $c = \max(\log[\frac{k\Delta_k}{\Phi(\mathbf{u})}], 2)$ and $V_1 \cup \ldots \cup V_N = V_{\mathcal{G}}$ is any vertex set partition with induced subgraphs $G_1, \ldots, G_N$ of maximum wide diameter $\Delta_k := \max\{\Delta_k(G_i) : i = 1, \ldots, N\}$, $\Phi(\mathbf{u})$ is the "cutsize" of $\mathbf{u} \in \{-1, 1\}^n$ any consistent labelling.

Example: Prototypical Clusters



$c \times$ m-cliques with ℓ cut edges

- Cover: $N = c$, Energy: $P_p(\mathbf{u}) = \ell \times 2^p$
- Connectivity: $k = m - 1$, Wide Diameter: $\Delta_{m-1} = 2$
- When $p = 2$ then $M \leq c + \frac{8\ell}{m-1}$

Part X

Large Scale Graph-Based Semi-Supervised Learning

SSL in Gigantic Image Collections – 1 [FWT09]

Problem: Even building the graph typically requires quadratic time compute all pairwise distances – will not scale well to the millions. Something more clever is needed! Sketch follows.

1. Use the completed weighted graph $w_{ij} = \exp(-c\|\mathbf{x}_i - \mathbf{x}_j\|^2)$ as the weighted edges of the Laplacian (note this will never be explicitly calculated)
2. The authors observe that solving $\mathbf{u}^\top L\mathbf{u}$ subject to soft constraints may be approximating solved by using the few eigenvectors of the Laplacian
3. In the limit of large data rather than finding eigenvectors, instead estimate eigenfunctions given an estimate of the distribution of the data
4. If one further assumes that the density (with a moderate size d)
$$p(\mathbf{s}) = p(s_1)p(s_2)\dots p(s_d)$$
then we can use PCA to find to find a rotation and estimate each dimension separably
5. Each separate dimension of the density is then estimated by counting elements in the bins of a histograms
6. Computation time will depend only linearly on the number of data points

SSL in Gigantic Image Collections – 2

Data: 79,302,017 images, 445,954 labeled from 386 keywords.

Re-ranking images from 4 keywords in an 80 million image dataset, using 3 labeled pairs for each keyword. Rows from top: "Japanese spaniel", "airbus", "ostrich", "auto". From L to R, the columns show the original image order, results of nearest-neighbors and the results of our eigenfunction approach. By

Part XI

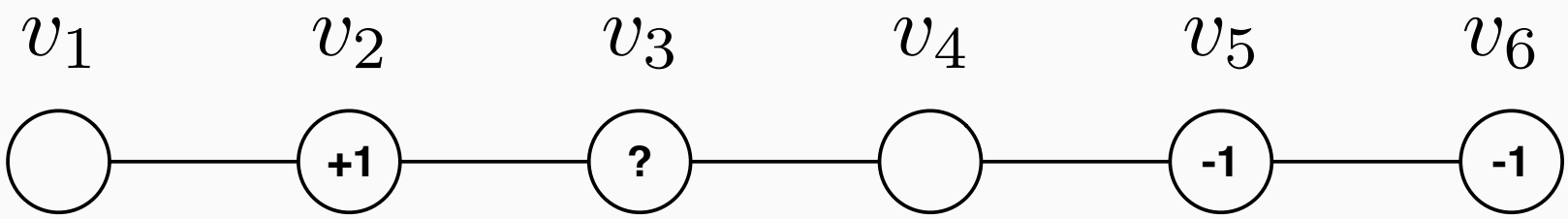
Conclusion

Discussion

- Laplacian-based transduction is versatile
- The graph laplacian may be easily combined with other kernel-based techniques (eg: kernel-Knn, Kernel-PCA)
- There are a large variety of graph "kernels" beyond the Laplacian pseudo-inverse
- Such kernel may be combined in a variety with other kernels of ways allowing the generalization from "transduction" to "induction."
- Some research issues
  1. Building graphs
  2. Time efficiency – cubic is expensive in practice
  3. Understanding how unlabeled data improves prediction

Problems – 1

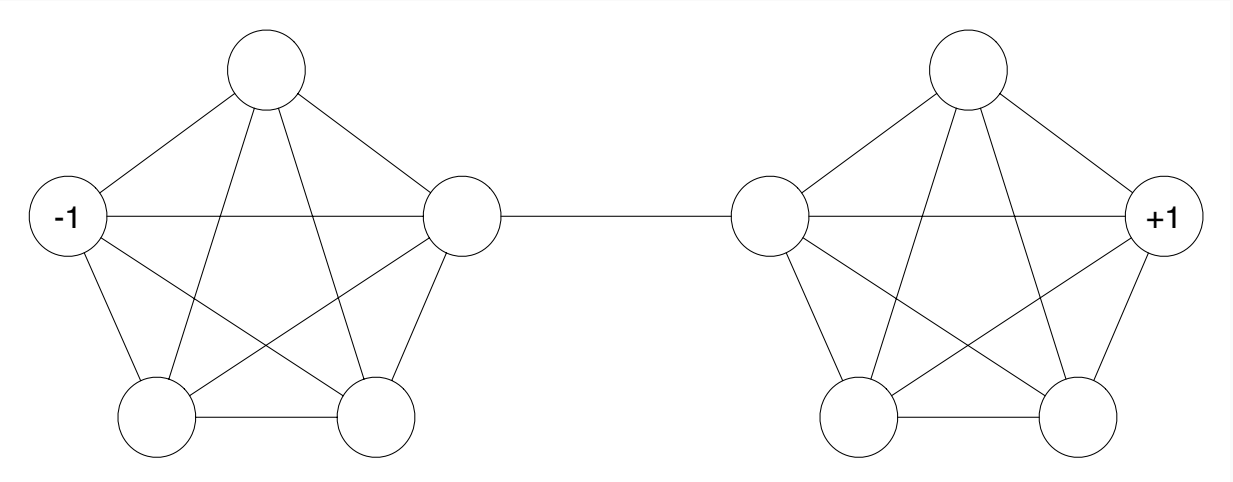
1. Describe, in brief, the differences between semi-supervised learning, supervised learning and unsupervised learning.
2. Given the unlabeled data points $((-2,0),(-1,-1),(0,0),(1,1),(2,0))$ draw the 2-nearest neighbor graph with respect to the euclidean distance.
3. Give the graph Laplacian for the dataset above.
4. In the labelled graph illustrated below,



   i What is a value of the minimum cut solution at vertex v_3?
   ii What is the value of the harmonic energy solution at vertex v_3?

Problems – 2

1. Consider the following illustration of the parameterized graph $\mathcal{D}_m$ ($m = 5$ in illustration) that consists of two m-cliques connected by a single bridge edge. The first clique contains a node labelled with a "-1" while the second clique contains a node labelled with a "$+1$".



2. Consider the labelling of the graph $\mathcal{D}_m$ as resulting from harmonic energy minimisation. Now consider the labelling of $\mathcal{D}_m$ as $m \to \infty$.
   i What is the labelling in the limit?
   ii Provide a justification of your answer to i.

3. **[Hard]:** Prove the first observation on page 35

Problems – 3

1. **[Hard]:** Prove the following inequality for the resistance diameter

$$\max_{i,j,\in\in[n]} r_{\mathbf{G}}(i,j) \leq \max_{k\in[n]} L^{+}k$$

2. **[Hard]:** Prove the equality connecting the effective resistance to the kernel $\mathbf{L}^{+}$ on page 55
3. **[Hard]:** For an unweighted graph prove,

$$\sum_{(i,j)\in\mathcal{G}} r_{\mathbf{G}}(i,j) = n-1$$

   This is trivial for trees. More difficult for generic graphs.
4. **[Hard]:** In our presentation we only consider equality constraints for interpolation instead of inequality constraints as with an SVM. This problem suggests a justification for this procedure.
   Let $\mathbf{L}$ be a graph Laplacian for an n-vertex graph. Consider the kernel (p.d. matrix) $\mathbf{K} := (\mathbf{L} + \epsilon \boldsymbol{I})^{-1}$ where $\boldsymbol{I}$ is the identity matrix and $\epsilon > 0$ is any positive constant. Prove that for any $\mathbf{x} \in [n]^{\ell}$ and any $\mathbf{y} \in \{-1, 1\}^{\ell}$ that

$$\operatorname*{arg\,min}_{\mathbf{u}\in\mathbb{R}^n : u_{x_1}=y_1,\ldots,u_{x_\ell}=y_\ell} \mathbf{u}^{\top}\mathbf{K}^{-1}\mathbf{u} = \operatorname*{arg\,min}_{\mathbf{u}\in\mathbb{R}^n : u_{x_1}y_1\geq 1,\ldots,u_{x_\ell}y_\ell\geq 1} \mathbf{u}^{\top}\mathbf{K}^{-1}\mathbf{u}\,.$$

   Think about the above property in terms of "support vectors" what does this imply for this set of Laplacian-based kernels.
   Argue that there exists a positive definite $n \times n$ matrix $\mathbf{K}$ for which this property does not hold.
5. **[Technical Hard]:** Do the problem on page 42

Suggested Readings

For graph-based semi-supervised learning, please see Chapters 1,2 and 5 from *Introduction to Semi-Supervised Learning*, Xiaojin Zhu and Andrew Goldberg; Morgan and Claypool (2009) `[Note: pdf available for download]`

For spectral clustering the following tutorial is very good *A Tutorial on Spectral Clustering*, Ulrike Von Luxberg (2007).

The presentation of spectral clustering is closely based on the material in lecture 3 in Michal Valko's course on Graphs in Machine Learning. The full lecture notes (slides) has many additional applications of graph-based concepts to machine learning.

References

1. [BC01] A. Blum and S. Chawla *Learning from Labeled and Unlabeled Data Using Graph Mincuts*, 2001
2. [BMN04] M. Belkin, I. Matveeva, and P. Niyogi. *Regularization and semi-supervised learning on large graphs*, 2004
3. [FWT09] B. Fergus, Y. Weiss, and A. Torralba Semi-supervised Learning in Gigantic Image Collections, 2009
4. [HL09] M. Herbster and G. Lever. Predicting the Labelling of a Graph via Minimum p-Seminorm Interpolation, 2009
5. [L07] U. Luxburg, *A Tutorial on Spectral Clustering*, 2007
6. [ZGL03] X. Zhu, Z. Ghahramani, and J. Lafferty. *Semi-supervised learning using gaussian fields and harmonic functions*, 2003