Ronak Mehta
Dr. Joshua Vogelstein
TA: Eric Bridgeford

Neuro Data Design
Seeded Graph Matching and Vertex Nomination
Reading Notes

Seeded Graph Matching

In the case of graph matching, we are given two graphs and wish to find a bijection between the graphs such that the number of edge disagreements is minimized. Seeded graph matching is the problem when part of the task is complete. That is, a few of the nodes are already aligned.

Quadratic Assignment Problem

Graph matching itself is a special case of a Quadratic Assignment Problem (QAP). For two real $n \times n$ matrices A and B, we wish to minimize trace(APB^TP^T) subject to $P \in \mathcal{P}$, where $\mathcal{P}$ is the set of permutation matrices. We want to minimize the amount of weight between facilities (A) times the distance between locations (B) via assigning the facilities to locations. We can relax this problem slightly by letting P lie in the polytopic region of $\mathcal{D} = \{P \in \mathbb{R}^{n \times n} : P^T\mathbf{1} = P\mathbf{1} = \mathbf{1}, P \succeq 0\}$, easing the optimization problem to the point where we can use continuous feasible region optimization algorithms.

Fast Approximate Quadratic Programming for Graph Matching

Now, given adjacency matrices $A$and B, we want to solve our original problems. Considering edges as either weights or distances of 1 unit, we might want to maximize the amount of weight and distance, as we want the most 1s in each matrix to be multiplied by each other after permutation. This is equivalent to:

$$\begin{aligned} &\max \operatorname{trace}(APB^TP^T) \\ &\min \; -\operatorname{trace}(APB^TP^T) \end{aligned}$$

Algebraically manipulating min $|AP - PB|_F$ yields the same result. Because P may not exactly be a permutation matrix in this continuous case, we must solve this optimization problem with known techniques and then project onto the permutation matrix space $\mathcal{P}$ for an approximate solution.

Adjustments Given Seeds

In the seeded setting, we now say that A and B are adjacency matrices of $n + m$ vertices, with m seeds (nodes with a known bijection between the graphs), and n other vertices. Then, we look to solve max trace$(A^T(I_m \oplus P)B(I_m \oplus P^T))$, where $P \in \mathbb{R}^{n \times n}$ is from the relaxed $\mathcal{D}$ from before, and $\oplus$ is the direct sum. So, we order the first m vertices by their known matching. We then apply the Frank-Wolfe algorithm, which linearly approximates our quadratic objective function about the current value of the decision variable, and then repeatedly solves the linear program at this approximation. This becomes a linear assignment problem. At convergence of the Frank-Wolfe Algorithm, we have an doubly stochastic matrix P that we can project onto $\mathcal{P}$ in order to achieve our approximate solution.

In the case that we have more vertices in one graph, we can pad the smaller with isolated vertices and label their matches as extraneous. After adjusting the definition of the adjacency matrices and objective function slightly, this works to solve the same problem.

Vertex Nomination via SGM

Vertex nomination is a problem arising from a stochastic block model, where one block is of particular interest but the blocks of the realized graph are unobserved. We wish to create a nomination list that "ranks" the nodes' affinity for membership in this block.

Relationship to SGM

Here, rather than assigning vertices to blocks, we can assign them to nearby vertices via SGM. However, as SGM will usually find local optima and suggest a variety of different matchings based on different initializations in various simulations, we can produce an empirical probability distribution over vertices $[n]$ of "matching probability" for each vertex i, with π_j being the probability of matching with vertex j. By considering vertices in a small spacial neighborhood around vertex i, such as in a connectome, we can reduce the support of this distribution, creating a nomination list that a subject matter expert or future algorithms can link more efficiently.