COMS 4774 Unsupervised Learning Fall 2024
Problem Set #3

Brian Chen - bc2924@columbia.edu

October 30, 2024

Problem 1

Similarity Search in High Dimensions via Hashing by Gionis, Indyk and Motwani.

Nearest neighbor search naively requires a linear scan of all other points. This is obviously slow, so we tend to use faster versions that gives us an *approximate* nearest neighbor e.g. through k-d trees. However, in high dimensions k-d trees becomes almost as inefficient as a linear scan. LSH instead is a similarity search based on hashing, where closer points have a higher chance of collision after the hash. Their LSH algorithm allows for an approximate nearest neighbor to be calculated in $O(dn^{1/1+\epsilon})$ time, which is sublinear for any $\epsilon > 0$.

First we formalize the problem. Let l_p^d be the normed Euclidean space R^d under the l_p norm.

Let H^d be the Hamming metric space of dimension d, e.g. the space of binary vectors of length d under the hamming metric, and let $d_H(p, q)$ be the Hamming distance of p, q (e.g. the number of bits that are different in the binary representation of p and q). The nearest neighbor search problem is as such.

Definition 1. The *nearest neighbor search problem* is given a set P of points in a normed space l_p^d, process P such that you can find the point in P that is closest to a query point q.

We work with the approximate nearest neighbor search problem.

Definition 2. The *approximate nearest neighbor search problem* is given a set P of points in a normed space l_p^d, process P such that you can find a point in P that is at most at distance $(1 + \epsilon)d(p, q)$ away, where p is the true nearest neighbor.

The algorithm makes two assumptions. First, that we use the l_1 norm. Second, that all coordinates in P are positive integers. Note that (1) is a pretty easy assumption to make, because there's no reason to assume that this would not be the case. We can ensure (2) by adding some arbitrarily large number to all of the coordinates (effectively shifting the origin) until all coordinates are positive, and multiplying all by some large scaling factor to make them all integer. Note that this means the minimum distance between any two points is now 1.

Now we define the algorithm itself. Let C be the largest coordinate of all points in P. Then we can embed P into the Hamming space of dimension $H^{d'}$ with $d' = Cd$. This is done for every point $p = (x_1, \ldots, x_d)$ with the embedding:

$$v(p) = \text{Unary}_C(x_1) \ldots \text{Unary}_C(x_d)$$

with $\text{Unary}_C(x)$ the unary representation of x, e.g. x ones followed by $C - x$ zeros.

Fact 1: For any points $p, q \in P$, with coordinates in the set $\{1, \ldots, C\}$,

$$d_1(p, q) = d_H(v(p), v(q))$$

e.g. the embedding preserves the distance between points.

Now we need to choose our hashing function. Choose l random subsets $I_1, \ldots, I_l$ of coordinates in $\{1, \ldots, d'\}$. Let $p_{|I}$ be the projection of a point p onto one of these subsets, e.g. choose the coordinates of p in I and concatenate them together.

Clearly, this will cause some collisions because when you get rid of some coordinates, different points may become similar points. Let $g_j(p) = p_{|I}$. Then, when we hash, we **store every point $p \in P$ into one of l buckets.**

Thus the preprocessing step involves

- Choosing some l random subsets - Hashing all of the points into these buckets

and the querying step (finding the k-NN) for a query point q involves

- Searching all of the bucket $g_1(q), \ldots, g_l(q)$ until you get enough points - or until you have seen all l points in the bucket

We need to consider how to choose the set of subsets I_j. Note that each I_j for $j = \{1, \ldots, l\}$ involves a choice of k elements from $\{1, \ldots, d'\}$ sampled randomly with replacement.

Optimally, we want to choose k to maximize probability that points 'close' to p are hashed into the same bucket, and points 'far' from p are not hashed into the same bucket.

Definition 3. Let $D(\cdot, \cdot)$ be a distance function for a set of points S,Slet $\mathcal{B}(p, r)$ be the set of elements in S such that $D(p, q) < r$. Then, a family $\mathcal{H}$ of functions from $S \to U$ is (r_1, r_2, p_1, p_2)-sensitive if

- if $p \in \mathcal{B}(q, r_1)$, then $\Pr[h(q) = h(p)] \geq p_1$ - if $p \notin \mathcal{B}(q, r_2)$, then $\Pr[h(q) = h(p)] \leq p_2$

where h is a function in $\mathcal{H}$. For the LSH algorithm to be effective, we need $r_1 < r_2$ and $p_1 > p_2$.

Fact 2: The family of projections onto one coordinate is locality-sensitive if $D(\cdot, \cdot)$ is the Hamming distance. THis is true because if we let S be the d'-dimensional Hamming cube, and $D(p, q) = d_H(p, q)$. Then $\forall r, \epsilon > 0$, the family $\mathcal{H}_{d'} =$ make all coordinates b_i for $i = 1, \ldots, d'$ is $\left(r, r(1+\epsilon), 1 - \frac{r}{d'}, 1 - \frac{r(1+\epsilon)}{d'}\right)$-sensitive.

Now we define the (r, ϵ) neighbor problem.

Definition 4. The (r, ϵ) neighbor problem is to determine whether $\exists$ point p within a distance $r_1 = r$ of q, or if all points are at least a distance $r_2 = r(1 + \epsilon)$ away from q.

Theorem 1. *LSH algorithm solves this problem for proper choice of k and l.*

The theorem is true if the following two properties hold:

1. If $\exists\ p$ s.t. $p \in \mathcal{B}(q, r)$, e.g. if p is within distance r of q, then there is some hashing bucket that contains both p and q: $\exists\ j$ such that $g_j(p) = g_j(q)$

2. The total number of blocks pointed to by q and containing only points from P' is less than cl.

Let $\mathcal{H}$ be a (r_1, r_2, p_1, p_2)-sensitive family, and let $\rho = \frac{\ln(1/p_1)}{\ln(1/p_2)}$. Then we have

Theorem 2. *Setting $k = \log_{1/p_2}(n/B)$ and $l = (n/B)^\rho$ guarantees that both properties hold with probability at least $\frac{1}{2} - \frac{1}{e} \geq 0.132$.*

Proof. Let the probability of 1 be $P1$ and the probability of 2 be $P2$. We want to show that both of these are large.

Assume there exists some point p^* within r_1 of q, e.g. $p^* \in \mathcal{B}(q, r_1)$.

Then for any point $p \notin \mathcal{B}(q, r_2)$, the probability that p and q get hashed into the same bucket is at most $p_2^k = \frac{B}{n}$, because every hash of the k hashes we chose needs to result in the same bucket. The expected number of buckets allocated for g_j which contain exclusively points from P' does not exceed 2. The expected number of such blocks allocated for all of g_j is at most $2l$.

Similarly, for any point $p \in \mathcal{B}(q, r_2)$, the probability that p and q are hashed into the same bucket is lower bounded by $p_1^k = p_1^{\log_{1/p_2}(n/B)} = (n/B)^{-\rho}$

The probability, then, that NO bucket contains any similar point is $(1 - p_1^k)^l$ which surprisingly, for the judicious choice of $l = (\frac{n}{B})^\rho$, $= \frac{1}{e}$.

Therefore the probability that both properties hold is $1 - [(1 - P_1) + (1 - P_2)] \geq \frac{1}{2} - \frac{1}{e}$ $\square$

Now that we have this probability guarantee, let's plug in the values for the LSH family for the Hamming metric, which we know is $\left(r, r(1 + \epsilon), 1 - \frac{r}{d'}, 1 - \frac{r(1+\epsilon)}{d'}\right)$-sensitive as per Fact 2.

We have

$$\begin{aligned}
\rho &= \frac{\ln(1 - r/d')}{\ln(1 - (1 + \epsilon)r/d')} \\
&= \frac{\ln(1 - r/d')^{d'/r}}{\ln(1 - (1 + \epsilon)r/d')^{d'/r}} \\
&= \frac{U}{L}
\end{aligned}$$

where we want to bound U from below and L from above to upper bound ρ. Noting the inequalities $(1-(1+\epsilon)r/d')^{d'/r} < e^{1+\epsilon}$ and $(1-\frac{r}{d'})^{d'/r} > e^{-1}(1-\frac{1}{d'/r})$, we can prove that

$$\frac{U}{L} < 1/(1+\epsilon) - \ln(1-1/\ln n)$$

and therefore

$$\rho \leq \frac{1}{1+\epsilon}$$

A lower bound on the distortion of embedding planar metrics into Euclidean space by Newman, Rabinovich.

The paper explores a family of graphs that are unable to be embedded into Euclidean space with distortion less than $\Omega(\sqrt{\log n})$. This places an overall lower bound on the embeddability of planar graphs into Euclidean spaces.

Definition 5. The *distortion* of an embedding $f : S \to R$ between metric spaces (S, μ) and (R, δ) is given by

$$\text{distr}(f) = \max_{x,y \in S} \frac{\delta(f(x), f(y))}{\mu(x,y)} \cdot \max_{x,y \in S} \frac{\mu(x,y)}{\delta(f(x), f(y))}$$

e.g. the product between the maximum expansion of two points and the maximum contraction of two points after the embedding. Defining the smallest possible distortion of an embedding μ into real Euclidean and l_1 spaces respectively as $c_2(\mu), c_1(\mu)$, it can be show than that metrics on trees have $c_1(\mu) = 1$ and $c_2(\mu) = P(\sqrt{\log \log n})$. The paper aims to show some results on "series parallel trees", which are essentially trees defined recursively by combining vertices in series or in parallel.

Define a family of graphs in this way. G_0 is a single edge. G_i is a refinement of G_{i-1} obtained by replacing all edges of G_{i-1} with two aprallel edges, with each edge half the length of the original. In this sense, the geodetic metric is identical between the two graphs.

Theorem 3. *If μ is the geodetic metric of G_k, then*

$$c_2(\mu) \geq \sqrt{k+1}$$

Proof. WLOG, we can assume μ is non expanding since we can always scale up the Euclidean embedding. Let $\alpha = \min \frac{\|f(v)-f(u)\|_2}{\mu(v,u)}$. Trivially,

$$\|f(a) - f(c)\| \leq \sqrt{1-(k-i)\alpha^2} \cdot \mu(a,c)$$

for $i = k$. Now assume this holds for $i+1$, and inductively apply it to i. For any edge in G_i, it is related to five other edges: the four surrounding edges in G_{i+1} and its anti-edge that shares no vertices. By the indictive assumption, the images of the other edges are at most $\sqrt{1-(k-i-1)\alpha^2} \cdot 2^{-(i+1)}$ apart. By definition of α, the anti-edge is at least $\alpha \cdot 2^{-2(i+1)}$ apart. We get

$$4 \cdot 2^{-2(i+1)} \cdot [1-(k-i-1)\alpha^2] \geq \alpha^2 \cdot 2^{-2i} + \|f(a) - f(c)\|^2$$

and therefore

$$\|f(a) - f(c)\|^2 \leq [1 - (k-i)\alpha^2] \cdot 2^{-2i} = [1 - (k - i - \alpha^2] \cdot \mu^2(a, c)$$

Therefore, for the edge in G_0, the images of the vertices are at least α apart, and at most $\sqrt{1 - k \cdot \alpha^2}$ apart, giving us

$$\alpha \leq \frac{1}{\sqrt{k+1}}$$

□

Theorem 4. *In any embedding f of H_k into Euclidean space which does not expand the edges, there exists an edge in $E(H_k)$ whose length is contracted by f by at least a factor of $\sqrt{k+1}$.*

Proof. Proceeds similarly to before. Let H be constructed iteratively like before. Then every new edge introduce new edges between each of the four vertices each of length 2^{-1} and a new anti-edge. H_{i-1} embeds isometrically into H_i, but H_i is also still planar with the same number of vertices as in G_k. Thus applying the above yields the result. □

Problem 2

We immediately notice that embedding each input vector of dimension d to an output vector of dimension 2^d implies that we should look for an embedding involving subsets of the initial d dimensions. Specifically, for input vector x, let the i-th component of the embedding be defined as

$$f(x)_i = \sum_{i \in S} x_i - \sum_{i \notin S} x_i$$

where $S \subseteq \{1, 2, \ldots, d\}$ subsets of the initial d coordinates. Since $|S| = 2^d$ this is a 2^d dimensional embedding.

The resultant 2^d dimensional vector is an isometric embedding of x under l_1.

$$\|f(x)\|_\infty = \max_{i \in S} \left| \sum_{i \in S} x_i - \sum_{i \notin S} x_i \right|$$

$$= \max_{i \in S} \left| 2 \sum_{i \in S} x_i - \sum_{i=1}^{d} x_i \right|$$

If $x_i \geq 0 \ \ \forall i$, then since the sum is indexed only over choices of S, we clearly maximize when $|S| = d$. Then we get

$$\|f(x)\|_\infty = \left| 2 \sum_{i=1}^{d} x_i - \sum_{i=1}^{d} x_i \right|$$

$$= \left| \sum_{i=1}^{d} x_i \right|$$

$$= \|x\|_1$$

Otherwise, if there are some negative components in x, we choose S such that $i \in S$ if $x_i \geq 0$. Then

$$\|f(x)\|_\infty = \left| \sum_{\text{positive components of } x} x_i - \sum_{\text{negative components of } x} x_i \right|$$

$$= \|x\|_1$$

so either way the norms are preserved.

Problem 3

Part (a)

Q_d has 2^d vertices corresponding to all of the ways to enumerate d coin tosses. For each of the 2^d vertices, there are d adjacent vertices corresponding to every way we can flip one of the bits of each of the d-dimensional vertices. Since this double counts the number of vertices, we have

$$\boxed{\rho^2(Q_d) = \frac{d \cdot 2^d}{2}}$$

For K_{2^d}, we note that $\rho(u, v) = k$ where u, v differ by k bits. There are $\binom{d}{k}$ vertices that differ by k bits, so

$$\rho^2(K_{2^d}) = \frac{1}{2} \sum_{u \in K_{2^d}} \sum_{k=1}^{d} \binom{d}{k} k^2$$

$$= 2^{d-1} \sum_{k=1}^{d} \binom{d}{k} k^2$$

$$= 2^{d-1} \cdot (d^2 - d) 2^{d-1}$$

$$= \boxed{(d^2 - d) 2^{2d-2}}$$

Part (b)

Note that Q_d defines a regular graph of degree d, for each of the neighboring vertices that is adjacent to each vertex. By definition the Laplacian is

$$L(Q_d) = D(Q_d) - A(Q_d)$$

We are interested in

$$\sigma^2(Q_d) = \sum_{u,v \in Q_d} \|f(u) - f(v)\|_2^2$$

where $f : Q_d \to l_2^m$. We get

$$\sigma^2(Q_d) = \sum_{u,v \in Q_d} \|f(u) - f(v)\|_2^2$$

Since $m = 1$, this is just

$$\sigma^2(Q_d) = \sum_{u,v \in Q_d} (f(u) - f(v))^2$$

Now from previous problem sets we know that we can express this directly using the graph laplacian.

$$\boxed{\sigma^2(Q_d) = f^\mathsf{T} L(Q_d) f}$$

Identical logic can be applied for K_{2^d} which yields

$$\boxed{\sigma^2(K_{2^d}) = f^\mathsf{T} L(K_{2^d}) f}$$

Part (c)

By the rayleigh quotient for the laplacian, we know that $f^\mathsf{T} L f$ is bounded below by the smallest nontrivial eigenvalue of L. Specifically, we have

$$\begin{aligned}
\sigma^2(Q_d) &= f^\mathsf{T} L(Q_d) f \\
&= \sum_i^n \lambda_i c_i^2 \\
&\geq 2 \cdot \sum_{i=2}^n c_i^2 \\
&\geq 2 \cdot \|f\|^2
\end{aligned}$$

since 2 is the smallest nonzero eigenvalue, so any combination of the coefficients of the decomposition of f weighted by 2 provides a lower bound on $\sigma^2(Q_d)$.

Part (d)

We have

$$\begin{aligned}
\sigma^2(K_{2^d}) &= f^\mathsf{T} L(K_{2^d}) f \\
&= (2^d - 1) \sum_{i=1}^{2^d} f_i^2 - \sum_{i \neq j} f_i f_j \\
&= (2^d - 1) \sum_{i=1}^{2^d} f_i^2 - \sum_i^{2^d} f_i^2 \\
&= \boxed{2^d \cdot \|f\|_2^2}
\end{aligned}$$

where we use the fact that $\sum_i^{2^d} f_i = 0$ since we assume f is zero centered, so the second summation simplifies to just a sum over i.

Part (e)

In l_2^m, the Pythagorean theorem tells us that for any set of orthogonal vectors $v_1, \ldots, v_m \in l_m^2$,

$$\|v_1 + \cdots + v_m\|_2^2 = \|v_1\|_2^2 + \cdots + \|v_1\|_2^2$$

Given that embedding just into $m = 1$ already requires a distortion of $D = \Omega(\sqrt{d})$, clearly increasing the number of input vectors to some arbitrary requires the preservation of m total (orthogonal) distances because of the Pythagorean theorem. Therefore, we are incurring an additional cost of $\Omega(\sqrt{d})$ for each additional orthogonal direction, for a total distortion of $D = \Omega(m\sqrt{d}) = \Omega(\sqrt{d})$.

Problem 4

Part (i)

Choose $q = \log n$. Then we have

$$D = 2 \log n - 1 = O(\log n)$$

and

$$\begin{aligned} d &= O(\log n \; n^{1/\log n} \log n) \\ &= O(\log^2 n \cdot e) \\ &= O(\log^2 n) \end{aligned}$$

Therefore for this choice of q we get an embedding into $l_\infty^{O(\log n)}$ with distortion $O(\log n)$. Via Bourgain embedding, we know we can go from l_∞ to l_2 with an additional distortion cost of $O(\log n)$, where the embedding dimension remains unchanged. Therefore, we can embed first into $l_\infty^{O(\log n)}$ with distortion $O(\log n)$, and then embed into l_2, for a total distortion $D = O(\log^2 n)$ and dimension $d = O(\log^2 n)$.

Part (ii)

Following the hint, we proceed via induction on the number of nodes in the tree.

Base case: a tree with two nodes and edge length e can trivially be embedded into l_1 as you can just put the two nodes at a separation of e under l_1.

Inductive step: assume that for any tree with k nodes that there exists an embedding into l_1. Consider a tree with $k+1$ nodes: choose any leaf node v from the tree and remove it. By the inductive hypothesis, this tree can be embedded into l_1. Then we can add v back by placing it at a distance $d(v, v_{\text{neighbor}})$, from v_{neighbor}, the original neighbor node of v before we removed it. Since v is a leaf node, it only needs to maintain an edge distance e from this neighboring node, so we can just place it at a point in l_1 that is e away from v_{neighbor}. Therefore, the embedding of the $k+1$ points is still isometric to the original tree.

Problem 5

Let $B = S \cap \{x : \|x\| = 1\}$ be the unit sphere in S. Then we note that linearity of the map means that any point in S can be written as a scaled point on the unit sphere B, e.g. $p = ru \ \forall p \in S$ for some $r \geq 0$ and $u \in B$. Therefore, we just need to find an approximate preservation of interpoint distances on this sphere, as we are guaranteed to be able to scale them to the entirety of S.

Now we create a δ-net N of the unit sphere B. We know from previous problem sets that the size of this covering for the k-dimensional unit sphere is $O((1/\delta)^k)$. This net by definition satisfies

$$\forall u \in B, \exists p \in N \ \ s.t. \ \|u - v\| \leq \delta$$

Now we apply the given lemma to this finite subset of S. Specifically, for the set of $n = O((D/\delta)^k)$ points in the cover, we have for $d \geq \Omega(\frac{k}{\epsilon^2} \ln(D/\delta))$ that with probability at least $3/4$ over the choice of a random $d \times D$ matrix G that

$$(1-\epsilon)^2\|x_i - x_j\|^2 \leq \|Gx_i - Gx_j\|^2 \leq (1+\epsilon)^2\|x_i - x_j\|^2$$

For $u, v \in S$, let

$$w = \frac{u - v}{\|u - v\|}$$

where $w \in B$ because it's a unit vector in S. Let $p \in N$ be the closest net point to w in B. Then

$$\|w - p\| \leq \delta$$

Then we have

$$\begin{aligned}
\|Gu - Gv\| &= \|u - v\| \cdot \|Gw\| \\
&= \|u - v\| \cdot \|Gp + G(w - p)\| \\
&\leq \|u - v\|(\|Gp\| + \|G(w - p)\|) \\
&\leq \|u - v\| \left[1 + \delta + (1 + \delta)\delta\right] \\
&\leq (1 + O(\delta))\|u - v\|
\end{aligned}$$

since $\|Gp\| \leq 1 + \delta$ by the lemma given. Therefore,

$$\|Gu - Gv\| \leq (1 + O(\delta))\|u - v\|$$

Identical logic for the lower bound yields

$$\|Gu - Gv\| \geq \|u - v\|(\|Gp\| - \|G(w - p)\|) \geq \|u - v\|((1 - \delta) - (1 + \delta)\delta) \geq (1 - O(\delta))\|u - v\|$$

so we have the full inequality for ϵ, since $\epsilon \propto \delta$.

$$\boxed{(1-\epsilon)\|u - v\| \leq \|Gu - Gv\| \leq (1+\epsilon)\|u - v\|}$$

Problem 6

Part (i)

The Rayleigh quotient of the adjacency matrix A is

$$R = \frac{x^\mathsf{T} A x}{x^\mathsf{T} x}$$

where the max eigenvalue is

$$\lambda_{\max}(A) = \max_{x \neq 0} \frac{x^\mathsf{T} A x}{x^\mathsf{T} x}$$

Let $x = \mathbf{1}$, the all ones vector. Then we have

$$\frac{x^\mathsf{T} A x}{x^\mathsf{T} x} = \frac{\mathbf{1}^\mathsf{T} A \mathbf{1}}{\mathbf{1}^\mathsf{T} \mathbf{1}}$$

$$= \frac{\sum_{i,j} A_{ij}}{n}$$

Since $A_{ij} = 1$ only if there is an edge between i, j, $\sum_{i,j} A_{ij} = 2|E|$. If G is a clique, then there are $\binom{n}{2}$ edges. Therefore we have

$$\lambda_{\max}(A) = \frac{2\frac{n(n-1)}{2}}{n}$$

$$= \boxed{n-1}$$

In the opposite direction, suppose G' is not a clique. Let the adjacency matrix of G' be A'. Then

$$\frac{x^\mathsf{T} A' x}{x^\mathsf{T} x} = \frac{\mathbf{1}^\mathsf{T} A' \mathbf{1}}{\mathbf{1}^\mathsf{T} \mathbf{1}}$$

$$= \frac{\sum_{i,j} A'_{ij}}{n}$$

$$= \frac{2|E'|}{n}$$

where $|E'| < |E|$ since a clique is by definition maximally connected. Therefore, the max eigenvalue is necessarily smaller.

Part (ii)

To show that it is in NP, we need a polynomial time verifier. Given v, we can calculate if it is m-sparse in linear time by counting the number of nonzero entries in v and seeing if it

is $\leq m$. We can also compute matrix multiplication in polynomial time, so we can perform the $v^\mathsf{T} Av$ calculation in polynomial time which allows us to check the condition $v^\mathsf{T} Av \geq M$. Therefore, we have a polynomial time verifier.

To show that it is NP-hard and thus that the overall problem is NP-complete, we show a reduction from the Clique problem to sparse PCA.

Given a graph $G = (V, E)$ with n vertices, construct an adjacency matrix A such that $A_{ij} = 1$ if $i \neq j$ and $ij \in E$, and $= 0$ otherwise. Set the target sparsity $m = k$, the size of the clique we are looking for. Also set the target explained variance $M = k$.

Create the m-sparse vector v that has exactly k nonzero entries that act as indicators for the clique. Specifically, let the indices of the selected vertices be $v_1, \ldots, v_k$. Then we construct $v = \frac{1}{\sqrt{k}}[1\ 1 \ldots 00 \ldots]^\mathsf{T}$ with the first k entries being 1, and the rest 0. Now we can check the quadratic form $v^\mathsf{T} Av$. We would have

$$v^\mathsf{T} Av = \sum_{1 \leq i,j \leq k} A_{ij}$$

the sum of the indicated vertices. If there is a clique of size k in the graph, then the submatrix of A corresponding to the selected vertices will be complete. That would mean that the quadratic form would sum to $\binom{k}{2}$.

Thus, if the graph contains a clique of size k, then there exists a sparse vector with exactly k nonzero entries such that $v^\mathsf{T} Av \geq M$. Conversely, if there is no size k clique in the graph, then no selection of k vertices would yield a quadratic form $v^\mathsf{T} Av \geq M$ since there would not be enough edges in the selected vertices to satisfy the condition of the explained variance.

Problem 7

Part (i)

If E is a connected graph, then there exists a path between any two points in E. We know that the distances between each point are finite/bounded by definition. Therefore, if there was some point y_i placed at "infinity", then since the graph is connected then there must be some point connected to y_i (call it y_k) such that $d(y_i, y_k) = \infty$. However we know that the distances are constrained by $d(x_i, x_j) = \|y_i - y_j\|_2$, which is finite by assumption. Therefore there cannot be any y_i such that the objective diverges since the distance constraint enforces its finiteness.

Part (ii)

The Gram matrix for $Y \in \mathbb{R}^{N \times n}$ is

$$G = YY^\mathsf{T}$$

where $G_{ij} = y_i \cdot y_j$. Note that

$$\|y_i - y_j\|_2^2 = \|y_i\|_2^2 - 2(y_i \cdot y_j) + \|y_j\|_2^2 = G_{ii} + G_{jj} - 2G_{ij}$$

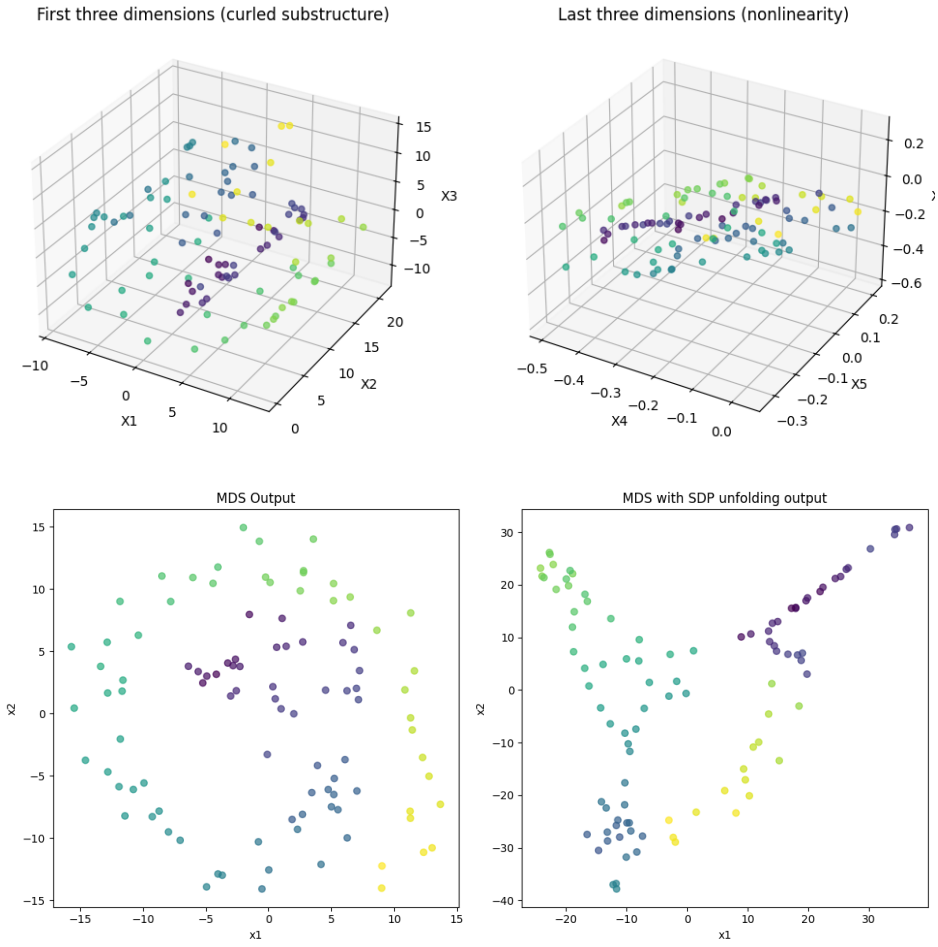
which means we can rewrite the objective function as

$$\begin{aligned}
\frac{1}{2N} \sum_{i,j \in [N]} \|y_i - y_j\|_2^2 &= \frac{1}{2N} \sum_{i,j \in [N]} (G_{ii} + G_{jj} - 2G_{ij}) \\
&= \frac{1}{2N} \left(2N \ \text{trace}(G) - 2 \ \text{sum}(G)\right) \\
&= \text{trace}(G) - \frac{1}{N}\text{sum}(G)
\end{aligned}$$

The gram matrix needs to be positive semidefinite as well. Therefore, we have the full form of the SDP:

$$\begin{aligned}
\text{maximize} \quad & \text{trace}(G) - \frac{1}{N}\text{sum}(G) \\
\text{subject to} \quad & G_{ii} + G_{jj} - 2G_{ij} = d(x_i, x_j)^2 \qquad \forall (i,j) \in E \\
& G \succeq 0 \qquad i = 1, \ldots, m
\end{aligned}$$

Part (iii)


First three dimensions (curled substructure)
X1
X2
X3
Last three dimensions (nonlinearity)
X4
X5
MDS Output
x1
x2
MDS with SDP unfolding output
x1
x2