COS 521: Advanced Algorithms
Homework 4

Zachary Hervieux-Moore

Friday 16th December, 2016

Exercise 1: Consider a set of n objects (images, sonds, etc.) and suppose somebody has designed a *distance* function $d(\cdot)$ among them where $d(i,j)$ is the distance between objects i and j. We are trying to find a geometric realization of these distances. Of course, exact realization may be impossible and we are willing to tolerate a factor a factor 2 approximation. We want n vectors $u_1, u_2, \ldots, u_n$ such that $d(i,j) \leq |u_i - u_j|_2 \leq 2d(i,j)$ for all pairs i, j. Describe a polynomial-time algorithm that determines such u_i's exist.

Answer: We first note that $d(\cdot)$ is a metric. So, $d(i,j) \geq 0$. Thus, we can rewrite the inequality above as

$$d(i,j)^2 \leq \langle u_i, u_i \rangle - 2\langle u_i, u_j \rangle + \langle u_j, u_j \rangle \leq 4d(i,j)^2$$

We now define the matrix M to be the Gramian matrix of the u_i's. That is, $M_{ij} = \langle u_i, u_j \rangle$. We now write an SDP as

$$\begin{gathered} \min_{i,j} \sum M_{ii} - 2M_{ij} + M_{jj} \\ M \succeq 0 \\ d(i,j)^2 \leq M_{ii} - 2M_{ij} + M_{jj} \\ M_{ii} - 2M_{ij} + M_{jj} \leq 4d(i,j)^2 \end{gathered}$$

Notice that the inequalities are linear. Also, $M = X^T X$ and so $y^T M y = y^T X^T X y = \|Xy\|_2^2 \geq 0$. Thus, M is positive semidefinite. Thus it is feasible since the first constraint is always satisfied and we can just pick M_{ii} to be large enough that the second constraint is satisfied while not breaking the third constraint. To extract the u_i's from M, simply take the Cholesky decomposition of M which yields $M = U^T U$ and the columns of U are the vectors.

Exercise 2: The course webpage links to a grayscale photo. Interpret it as an $n \times m$ matrix and run SVD on it. What is the value of k such that a rank k approximation gives a reasonable approximation (visually) to the image? What value of k gives an approximation that looks high quality to your eyes? Attach both pictures and code. (In matlab you need mat2gray function). Extra credit: Try to explain from first principles why SVD works for image compression at all.

Answer: For a decent approximation, $k = 50$ worked reasonably. It is still quite fuzzy. However, the compression resulted in a size of 10% of the original size. The image is shown below.

For a high quality image, $k = 100$ worked well and left only few artifacts. The compression resulted in a size of 20% of the original. The image is shown below.

To explain why SVD compresses at all, we turn it into an optimization problem

$$\min_{\widehat{I}} \|I - \widehat{I}\|_F \text{ s.t. } \operatorname{rank}(\widehat{I}) \leq k$$

That is, we want to minimize the total squared error between the two matrices (images) subject to constraining the rank of the approximation. By the "Best rank k approximation" theorem in class, the solution is precisely the SVD decomposition restricted to the k largest singular values. This shows that SVD solves the optimization problem above. Since images have lots of inherent linear depedence, the eigenvalues actually decrease in size quite fast. This allows for dropping many singular values with little loss to the image. Note that without this dependence, there could be no compression.

The Matlab code is attached below.

```
image = imread('image.jpg');
image = mat2gray(image);
[w, h] = size(image);

k = 50;
[U,S,V] = svd(image);

% Now approximate U,S,V with only k largest entries
U_approx = U(1:w,1:k);
S_approx = S(1:k,1:k);
```

```
V_approx = V(1:h,1:k);

image_approx = U_approx*S_approx*V_approx';

compression = (w*k + k*k + h*k)/(w*h)

figure
imshow(image_approx)
```

Exercise 3: Suppose we have a set of n images and for some multiset E of image pairs we have been told whether they are *similar* (denoted +edges in E) or *dissimilar* (denoted -edges). These ratings were generated by different users and may not be mutually consistent (in fact the same pair may be rated as + as well as -). We wish to *partition* them into clusters $S_1, S_2, \ldots$ so as to maximise:

(# of +edges that lie within clusters) + (# of -edges that lie between clusters)

Show that the following SDP is an upperbound on this, where $w^+(ij)$ and $w^-(ij)$ are the number of times pair i, j has been rated + and - respectively.

$$\max \sum_{(i,j)\in E} w^+(ij)(x_i \cdot x_j) + w^-(ij)(1 - x_i \cdot x_j)$$
$$|x_i|_2^2 = 1 \quad \forall i$$
$$x_i \cdot x_j \geq 0 \quad \forall i \neq j$$

Answer: We first note that there can only be at most n clusters. Thus, let us denote v_i as the cluster that element x_i is in. Where $v_i = e_j$ where $j \in [n]$ is the cluster. We show that these v_i's satisfy the SDP. Clearly $|e_i|_2^2 = 1$ and we know that $e_i \cdot e_j = 0$. Thus, we have the constraints satisfied. Finally, this achieves the optimum since the solution will be

$$\sum_{k=1}^{n} \left[\sum_{(i,j)\in S_k}^{n} w^+(ij) \sum_{(i,j)\notin S_k} w^-(ij) \right]$$

Which is precisely the maximum we wish to get.

Exercise 4: For the problem in the previous question, describe a clustering into 4 clusters that achieves an objective value 0.75 times the SDP value. (Hint: Use Goemans-Williamson style rounding but with two random hyperplanes instead of one. You may need a quick matlab calculation just like GW).

Answer: Following Goemans-Williamson, use 2 random hyperplanes to split into 4 clusters. Let θ_{ij} denote the angle between u_i and u_j. The probability that they will be in the same cluster is $(1 - \theta_{ij}/\pi)^2$. Thus, the expected objective value in the previous problem becomes

$$\sum_{(i,j)\in E} w^+(ij)\left(1-\frac{\theta_{ij}}{\pi}\right)^2 + w^-(ij)\left(1-\left(1-\frac{\theta_{ij}}{\pi}\right)^2\right)$$

The value of the optimal objective will also be

$$\sum_{(i,j)\in E} w^+(ij)cos(\theta_{ij}) + w^-(ij)\left(1-(1-cos(\theta_{ij})\right)$$

Thus, the ration of the optimal and expected is

$$\frac{\left(1-\frac{\theta_{ij}}{\pi}\right)^2}{cos(\theta_{ij})}$$
$$\frac{1-\left(1-\frac{\theta_{ij}}{\pi}\right)^2}{1-cos(\theta_{ij})}$$

Note that $\theta_{ij} \in [0, \pi/2]$. Doing a Wolfram minimization, the lower bound on these ratios are

$$\frac{\left(1-\frac{\theta_{ij}}{\pi}\right)^2}{cos(\theta_{ij})} \geq 0.790$$
$$\frac{1-\left(1-\frac{\theta_{ij}}{\pi}\right)^2}{1-cos(\theta_{ij})} \geq 0.75$$

Thus, we have achieved 0.75OPT.

Exercise 5: Suppose you are given m halfspaces in $\mathbb{R}^n$ with rational coefficients. Describe a polynomial-time algorithm to find the largest *sphere* that is contained inside the polyhedron defined by these halfspaces.

Answer: We first suppose the halfspaces are of the form

$$\langle a_i, x \rangle + b_i \geq 0$$

We now suppose that the center of the sphere is located at x_0. Then the distance to the i^{th} halfspace is

$$\frac{|\langle a_i, x_0 \rangle + b_i|}{|a_i|}$$

Notice that the way we defined the halfspace makes the numerator positive. Thus, the distance is

$$\frac{\langle a_i, x_0 \rangle + b_i}{|a_i|}$$

We now define the LP

$$\begin{aligned} &\max r \\ \text{s.t. } r \leq &\frac{\langle a_i, x_0 \rangle + b_i}{|a_i|} \quad \forall i \end{aligned}$$

Which has a polynomial time solution.

Exercise 6: Let f be an n-variate convex function such that for every x, every eigenvalue of $\nabla^2 f(x)$ lies in $[m, M]$. Show that the optimum value of f is lowerbounded by $f(x) - \frac{1}{2m}|\nabla f(x)|_2^2$ and upperbounded by $f(x) + \frac{1}{2M}|\nabla f(x)|_2^2$ where x is any point. In other words, if the gradient at x is small, then the value of f at x is near optimal. (Hint: By the mean value theorem, $f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$, where z is some point on the line segment joining x, y).

Answer: Using the hint, we have that $f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$. Furthermore, we have that

$$\frac{m}{2}|y-x|^2 \leq \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x) \leq \frac{M}{2}|y-x|^2$$

Which follows from the fact that m and M are the minimum and maximum eigenvalues respectively (consider that the matrix is a span of its eigenvectors, clearly, minimizing or maximizing will restrict $y-x$ to the smallest or largest eigenvalue). Thus, we have

$$f(y) \geq f(x) + \nabla f(x)^T(y-x) + \frac{m}{2}|y-x|^2$$

Let $y = x^*$ be the optimal value of f. So

$$f(x^*) \geq f(x) + \nabla f(x)^T(x^*-x) + \frac{m}{2}|x^*-x|^2$$

Right side is quadratic in $x^* - x$ which is minimized at $x^* - x = \nabla f(x)/m$. Plugging this in yields

$$f(x^*) \geq f(x) - \frac{\nabla f(x)^2}{m} + \frac{m}{2}\frac{\nabla f(x)^2}{m^2}$$
$$f(x^*) \geq f(x) - \frac{\nabla f(x)^2}{m} + \frac{\nabla f(x)^2}{2m}$$
$$f(x^*) \geq f(x) - \frac{1}{2m}\nabla f(x)^2)$$

Now for the upper bound, we have

$$f(x^*) \leq f(x) + \nabla f(x)^T(x^*-x) + \frac{M}{2}|x^*-x|^2$$

Right side is quadratic in $x^* - x$ which is minimized at $x^* - x = \nabla f(x)/M$. Plugging this in yields

$$f(x^*) \leq f(x) - \frac{\nabla f(x)^2}{M} + \frac{M}{2}\frac{\nabla f(x)^2}{M^2}$$
$$f(x^*) \leq f(x) - \frac{\nabla f(x)^2}{M} + \frac{\nabla f(x)^2}{2M}$$
$$f(x^*) \leq f(x) - \frac{1}{2M}\nabla f(x)^2)$$

Exercise 7: (Yet another convex minimization based algorithm for submodular minimization). In Lecture 16, we saw that one can minimize arbitrary submodular function by minimizing the convex Lovasz extension of the function over the solid hypercube. This exercise presents another convex minimization based approach for the problem. This approach gives a fast-in-practice algorithm for submodular minimization used in many machine learning applications.

Recall the notation from Lecture 16: Let $f : 2^{[n]} \to \mathbb{R}$ be a submodular function with $f(\emptyset) = 0$ (recall te notation used : f is a *set* $[n] = \{1, 2, \ldots, n\}$ of size n). The base polyhedron of f, $\mathcal{B}_f$ is the convex set in $\mathbb{R}^n$ defined by the inequality constraints:

$$x \in \mathcal{B}_f \Leftrightarrow \{\sum_{i \in S} x_i \leq f(S) \forall S \subseteq [n]; \sum_{i \in n} x_i = f([n])\}$$

a) Prove the "dumbbell lemma": Suppose $J \subseteq N$ is such that $f(J) \leq f(K)$ for any $K \supseteq J$ or $K \subseteq J$. Then $f(J) = \min_{S \subseteq N} f(S)$.

b) Let x^* be the optimal solution to $\min_{x \in \mathcal{B}_f} \|x\|_2^2$ and set $J = \{i | x_i^* < 0\}$. Prove that $f(J) = \min_{S \subseteq [n]} f(S)$. (Hint: Show that J gives a "tight" inequality in $\mathcal{B}_f$, i.e., $\sum_{j \in J} x_j^* = f(J)$ and use the dumbbell lemma).

c) Show that there's an efficient separation oracle $\mathcal{B}_f$ that uses only an evaluation oracle for f. Conclude that one can thus use the ellipsoid algorithm to minimize an arbitrary submodular function.

In practice, instead of the ellipsoid method, one uses an iterative procedure known as *Wolfe's method* for implementing the algorithm suggested by b) above.

Answer:

a) From Lemma 1 of Lecture 16, a submodular function satisfies

$$f(S) + f(J) \geq f(S \cap J) + f(S \cup J)$$

This is lower bounded by $2f(J)$ by our assumption that $f(J) \leq f(K)$ for any $K \supseteq J$ or $K \subseteq J$. Thus, we have

$$\begin{aligned} f(S) + f(J) \geq f(S \cap J) + f(S \cup J) &\geq 2f(J) \\ \iff f(S) &\geq f(J) \end{aligned}$$

Equivalently, $f(J) = \min_{S \subset N} f(S)$.

b) We have $x^* = \min_{x \in \mathcal{B}_f} \|x\|_2^2$. Then we have

$$x^*(K) = \sum_{k \in K} x_k^* \quad \forall K$$

Now, we reorder the indices to make the analysis easier.

$$x_1^* \geq x_2^* \geq \ldots \geq x_n^*$$

Since $x^* \in \mathcal{B}_f$, we have $x^*(J) = f(J)$. Now have for all $K \subseteq J$, we have that

$$f(K) \geq x^*(K) \geq x^*(J) = f(J)$$

since all terms of $x^*(J)$ are negative and so removing terms will make the sum greater. It also follows that

$$f(K) \geq x^*(K) \geq x^*(J) = f(J)$$

Since anything that contains J has positive terms. Hence, J satisfies the conditions of the dumbbell lemma and so we conclude that $f(J) = \min_{S \subset N} f(S)$.

c) There exists an efficient separation oracle for $\mathcal{B}_f$ by starting at a vertex, say $(f(1), 0, \ldots, 0)$, and use a greedy algorithm to find the closest vertex to x. Call it v^*. Then check whether or not the point is contained in the halfspaces of v^* and its neighbours. If it is, then it is contained in $\mathcal{B}_f$. If not, a separating hyperplane is $\frac{v^*+x}{2}$ with slope equal to any of the slopes of v^* and its neighbours. Using this separation oracle, one can use the ellipsoid method to find the minimum of any submodular function.

Exercise 8: (Tensor Power Iteration, Updated). In Lecture 12, we saw the power method for computing the eigenvector of a real symmetric matrix $M \in \mathbb{R}^{n \times n}$ corresponding to its largest singular value (assuming that there's some gap between the largest and the second largest singular values).

Now, consider the 3 tensor $T = \sum_{i \leq k} \lambda_i \cdot a_i \otimes a_i \otimes a_i$ for $a_i \in \mathbb{R}^n$ satisfying $\|a_i\|^2 = 1$ for each i and $\langle a_i, a_j \rangle = 0$ whenever $i \neq j$. In Lecture 19, we saw Jenrich's algorithm that gives us a method to computer the a_i's. This problem explores the analog of the power method for computing some single component of the tensor T.

We first need a notion of "tensor-vector multiplication". For any x, define $T \cdot x$ as the vector $z \in \mathbb{R}^n$ such that $z_i = \sum_{j,k \in n} T_{i,j,k} x_j x_k$. In the *tensor power method*, we start with a random vector $x_0 \in \mathbb{R}^n$ satisfying $\|x_0\|_2 = 1$ and repeatedly compute $x_i = \frac{T \cdot x_{i-1}}{|T \cdot x_{i-1}|_2}$ for $i = 1, 2, \ldots, r$.

a) Show that for large enough t, x_t converges to some a_i.

b) Give a tight estimate of the minimum t required to ensure $\|x_t - a_i\|_2 \leq \epsilon$ in terms of λ_i's, n, ϵ, and the initial choice x_0.

While we understand the power method for matrices very well, theoretical analyses of tensor power iteration is known only in very special cases (even though the method works well in practice in very general situations).

Answer: The following is a summary of the paper "Tensor Decompositions for Learning Latent Variable Models" by Anandkumar et al. We will prove both parts at once. First thing is to order $|\lambda_1 a_1^T x_0| \geq |\lambda_2 a_2^T x_0| \geq \ldots \geq |\lambda_k a_k^T x_0|$. We now define the following

$$c_i = a_i^T x_0$$
$$\tilde{x}_t = T \cdot x_{t-1}$$

Then it is clear that $x_t = \frac{\tilde{x}_t}{|\tilde{x}_t|_2}$. We also have $\tilde{x}_t = \sum_{i=1}^k \lambda_i^{2^t - 1} c_i^{2^t} a_i$. Which can be shown by induction by

$$\tilde{x}_{t+1} = \sum_{i=1}^k \lambda_i (a_i^T \tilde{x}_t)^2 a_i = \sum_{i=1}^k \lambda_i (\lambda_i^{2^t - 1} c_i^{2^t})^2 a_i$$
$$= \sum_{i=1}^k \lambda_i^{2^{t+1} - 1} c_i^{2^{t+1}} a_i$$

Now, we have the following

$$\|a_1 - x_t\|_2^2 = 1 - 2a_1^T x_t + 1 = 2(1 - a_1^T x_t) \leq 2((1 - (a_1^T x_t)^2) \qquad (\star)$$

Everything is setup and so now we prove what is required,

$$\begin{aligned}
1 - (a_1^T x_t)^2 &= 1 - \frac{(a_1^T \tilde{x}_t)^2}{\|\tilde{x}_t\|_2^2} \\
&= 1 - \frac{\lambda_1^{2^{t+1}-2} c_1^{2^{t+1}}}{\sum_{i=1}^k \lambda_i^{2^{t+1}-2} c_i^{2^{t+1}}} \\
&\leq \frac{\sum_{i=2}^k \lambda_i^{2^{t+1}-2} c_i^{2^{t+1}}}{\sum_{i=1}^k \lambda_i^{2^{t+1}-2} c_i^{2^{t+1}}} \\
&\leq \lambda_1^2 \sum_{i=2}^k \lambda_i^{-2} \left| \frac{\lambda_2 c_2}{\lambda_1 c_1} \right|^{2^{t+1}}
\end{aligned}$$

Putting this together with $(\star)$, we have the result

$$\|a_1 - x_t\|_2^2 \leq 2\lambda_1^2 \sum_{i=2}^k \lambda_i^{-2} \left| \frac{\lambda_2 c_2}{\lambda_1 c_1} \right|^{2^{t+1}}$$

Which we conclude that x_t converges to a_1 quadratically.