PRINCETON UNIV. F'18 COS 521: ADVANCED ALGORITHM DESIGN

Homework 4: Convex Optimization (**60pts**)

Due: **Friday, December 7th, 2018, 11:59pm**

Collaboration is allowed on this problem set, but solutions must be written-up individually. Please list collaborators for each problem separately, or write "No Collaborators" if you worked alone. Collaboration is not allowed on bonus problems.

Please prepare your problem sets in LaTeX and compile to a PDF for your final submission. A LaTeX template is available on the course webpage.

§1 **(20 pts)** Review the **Portfolio Management** case study from the Lecture 16 notes and the Constant Rebalanced Portfolio (CRP) strategy for managing a stock portfolio. Download the S&P stock data from the course website, which contains price data for 1000 days of trading from 2001–2005 (in .csv or .mat format).

(a) If an investor places all their money in the best single stock, what is the maximum possible earnings over the course of the 1000 trading days? What about if an equal amount of money is initially placed in each stock? Finally, what earnings would an investor achieve using a CRP strategy that allocates an equal percentage of the portfolio to each stock?

(b) Find a CRP strategy (i.e. a fixed allocation of funds) that performs better than the best single stock over the 1000 days. Include the strategy in a text file as a comma separated list containing the percentage of funds allocated to each stock. (Hint: Implement Gradient Descent. The following might be helpful for your projection step: https://eng.ucmerced.edu/people/wwang5/papers/SimplexProj.pdf.

(c) Of course, an investor would not be able to choose the best CRP strategy in hindsight. Implement online gradient descent to find a portfolio re-balancing strategy that an investor could actually use. What percentage gain were you able to achieve? Feel free to test multiple learning rates and report your best result. In practice, the learning rate itself would be learned based on prior data, or in an online way.

For parts (b) and (c) include a short discussion of your implementation. What are your gradient updates and how were they derived? Attach all code.

§2 **(7 pts)** Consider a set of n objects (images, songs, 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 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 whether such u_i's exist (and outputs them in the event that they do).

§3 **(8 pts)** Give examples of PSD matrices A and B where:

(a) $A \succeq B$ but $A^2 \not\succeq B^2$.

(b) $\lambda_i(A) > \lambda_i(B)$ for all i, but $A \not\succeq B$. Here λ_i denotes the i^{th} largest eigenvalue of the matrix.

Extra Credit: I want to find simple examples to use in lecture notes! Provide solutions where all entries of A and B are integers and try to minimize the following measure of "simplicity": $C(A,B) = \sum_{i,j} |A_{i,j}| + \sum_{i,j} |B_{i,j}|$. More credit will be given to solutions with smaller $C(A,B)$.

§4 **(10 pts)** Describe separation oracles for the following convex sets. Your oracles should run in linear time, plus a constant number of calls to the given oracles.

(a) The set $A \cap B$ given separation oracles for A and B.

(b) The ℓ_1 ball, $\{x : \|x\|_1 \leq 1\}$. Recall that $\|x\|_1 = \sum_{i=1}^{d} |x_d|$.

(c) Any convex set A that we have a projection oracle for. I.e. we have an oracle to compute $\arg\min_{x \in A} \|x - y\|_2$ for any y.

(d) The ϵ-neighborhood, E of any convex set A:

$$E = \{x : \exists y \in A \text{ with } \|x - y\|_2 \leq \epsilon\},$$

given a projection oracle for A.

§5 **(15 pts)** The maximum cut problem asks us to cluster the nodes of a graph $G = (V, E)$ into two disjoint sets X, Y so as to maximize the number of edges between these sets:

$$\max_{X,Y} \sum_{(i,j) \in E} \mathbb{1}[(i \in X, j \in Y) \vee (i \in Y, j \in X)]$$

Consider instead clustering the nodes into **three** disjoint sets X, Y, Z. Our goal is to maximize the number of edges between different sets:

$$\max_{X,Y,Z} \sum_{(i,j) \in E} \mathbb{1}[(i \in X, j \in Y \cup Z) \vee (i \in Y, j \in X \cup Z) \vee (i \in Z, j \in X \cup Y)]$$

Design an algorithm based on SDP relaxation that solves this problem with approximation ration greater then .7.

Hint: In the Goemans-Williamson algorithm for maximum cut, we claimed that $\frac{2\theta}{\pi(1-\cos\theta)} \geq 0.878,\ \forall \theta \in [0, \pi]$. This is much easier to verify analytically (e.g. with a plot in MATLAB) than to prove formally. If similar quantities appear in your proof, feel free to bound them analytically, without proof.

Obtain the highest object value you can – partial credit will be given to any non-trivial solution, even if it obtains a weaker bound than .7. Extra credit: obtain an algorithm with approximation factor $> .8$.