Toolkit for TCS	**Feb-June 2021**
HW : MWUM and Spectral Methods	
Out: May 6, 2021	**Due:** May 24, 2021

(1) Please take time to write clear and concise solutions. You are STRONGLY *encouraged to submit LaTeXed solutions.*
(2) Collaboration is OK, but please write your answers yourself, and include in your answers the names of EVERYONE *you collaborated with and* ALL *references other than class notes you consulted.*

1. **[From Satish Rao's homework assignments]** (6 points)

   In the experts framework, we compared an online algorithm that was allowed to pick among n (expert) strategies, with the best choice of a single expert in hindsight. In this question we first introduce a different measure of performance of the online algorithm, called *regret*. The regret is how much worse the online player does than the best offline player. i.e. if the best expert suffers a loss of L while the expected loss of the online player is $\mathbf{E}[L(A)]$, then the regret is $\mathbf{E}[L(A)] - L$.

   Assume that the game lasts for T rounds, and the loss in each round is in the interval $[0, 1]$.

   (a) (3 points) Show that the regret suffered by the experts algorithm (with appropriate parameters) compared to the best expert is $O\left(\sqrt{T \cdot \log n}\right)$.

   (b) (3 points) How important was it that the online player was allowed to switch between experts, while the offline player had to stick to a single expert? To answer let us consider the regret suffered by the online player compared to an offline player who is allowed to switch between experts at every step. Construct an example where L^*, the loss suffered by the offline player is 0, while the expected regret of A is at least $T \cdot (1 - 1/n)$.

2. **[Lower bound of 2 for any deterministic algorithm]** (3 points)

   In lecture, we showed that the *weighted-majority (WM_η)* algorithm satisfies the following bound. For any positive integer T, let $m_i^{(T)}$ be the number of mistakes made by expert i upto step T and $M^{(T)}$ be the number of mistakes made by WM_η algorithm with parameter $\eta \in (0, 1/2]$. Then, for any expery $i \in [n]$, we have

   $$M^{(T)} \leq 2(1+\eta) \cdot m_i^{(T)} + \frac{2 \ln n}{\eta}.$$

   In other words, the WM_η algorithm makes no more than approximately *twice* the number of mistakes made by any expert. Show that no *deterministic* algorithm can guarantee a perfomance better than twice the number of mistakes made by the best expert.

3. **[ε-Nets & Hitting Sets (modified from Anupam Gupta and Ryan O'Donnell's psets)]** (7 points)

   Let $(U, \mathcal{F})$ be a set system, where the universe U is of size N (i.e, $|U| = n$), and $\mathcal{F}$ is a collection of m sets $\{S_1, S_2, \ldots, S_m\}$ where each $S_i \subseteq U$. Define the following two quantities:

   Hitting Set: The set $H \subseteq U$ is a hitting set for $\mathcal{F}$ if $H \cap S_i \neq \emptyset$ for all $i \in [m]$. Let c be the size of the smallest hitting set for $\mathcal{F}$; note that this is NP-hard to compute.

   ε-net: Given non-negative weights w_e for each element $e \in U$, define the weight of a set A as $w(A) = \sum_{e \in A} w_e$. A set $N \subseteq U$ is an ε-net for $(\mathcal{F}, w)$ if for all sets S_i such that $w(S_i) \geq \varepsilon \cdot w(U)$, we have $N \cap S_i \neq \emptyset$. (In other words, N hits all the "heavy-weight" sets.)

   (a) (2 points) For any parameter $\varepsilon > 0$, give a polynomial-time algorithm that given any set system $(U, \mathcal{F})$ with m sets and a weight function $w : U \to \mathbb{R}_{\geq 0}$ finds an ε-net of size at most $O((\log m)/\varepsilon)$.

   Hint: Check if a a greedy algorithm (or even a randomized algorithm) works.

   (b) (4 points) In this part, we will use the above algorithm for ε-nets to construct a hitting set that is not too large compared the size of the optimal hitting set.

   Suppose you are given an algorithm E_ε that when given a set family $(U, \mathcal{F})$ and an associated weight function w, finds an ε-net of size $T(\varepsilon, m)$. (This could be, for instance the algorithm designed in part 3a).

Consider the following algorithm that uses E_ε to compute a hitting set for $\mathcal{F}$:

Data: $(U, \mathcal{F})$ - a set system
Result: A hitting set H
Set $w(e) \leftarrow 1$ for all $e \in U$;
repeat
  Use algorithm E_ε to find an ε-net H for $(\mathcal{F}, w)$;
  Let S be any set in $\mathcal{F}$ such that $S \cap H = \emptyset$. If no such set exists, set $S \leftarrow \emptyset$;
  for *all elements* $e \in S$ **do**
    $w(e) \leftarrow 2 \cdot w(e)$ (i.e., double its weight);
  end
until H *is a hitting set for* $\mathcal{F}$;
Ouput H;

Algorithm 1: Hitting Set Algorithm From ε-nets

Let c be the size of an optimal hitting set S^* of $(U, \mathcal{F})$ (note that it is NP-hard to determine this quantity). Show that if $\varepsilon < 2^{1/c} - 1$, then the above algorithm terminates within

$$\frac{\log_2\left(\frac{n}{c}\right)}{\frac{1}{c} - \log_2(1+\varepsilon)}$$

iterations of the return loop at which point it outputs a hitting set of size at most $T(\varepsilon, m)$.

One such choice of ε is $1/2c$, which gives a bound of $O(c \cdot \log(n/c))$ on the number of iterations.

Hint: Compare the weight $w(U)$ of the universe U and the weight $w(S^*)$ of the optimal hitting set S^* at the end of each iteration of the return loop.

(c) (1 point) The above algorithm requires knowledge of the optimal hitting set size c. Use a doubling argument or otherwise, to give an $O(\log m)$-approximation algorithm of the size of the optimal hitting set size

4. **[bipartite]** (3 points)

Let A be the adjacency matrix of an n-vertex undirected *connected* graph G and $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n$ be the n eigenvalues of A arranged in non-increasing order. In lecture, we had shown that $\mu_1 > -\mu_n$.

(a) (2 points) Prove that if $\mu_1 = -\mu_n$, then the graph is bipartite.
(b) (1 point) Prove that if the graph is bipartite, then $\mu_1 = -\mu_n$.

5. **[Colouring using largest eigenvector]** (3 points)

Let A be the adjacency matrix of an undirected graph G, and let v be its top eigenvector with eigenvalue κ. Note that $\kappa \geq 0$ and v can be so chosen as to have non-negative entries. Let us assume further that we write v so that its entries are arranged in descending order (so that $v_1 \leq v_2 \leq \cdots \leq v_n$). Note that this induces an ordering on the vertices of G.

Consider now the following colouring procedure, which is based on the above order. We start with an empty list L of colours. We then process the vertices $u_1, u_2, \ldots, u_n$ in order, and for any given i, construct the set S of the colours assigned to neighbors u_j of u_i with $j < i$. If $S = L$, then we create a new color c, set $L = L \cup \{c\}$ and assign the colour c to u_i. Otherwise, if $L \setminus S$ is non-empty, we choose a color from $L \setminus S$ (according to some pre-defined choice rule) and assign it to u_i. Note that this procedure produces a proper coloring of the graph.

How large can L can be at the end of the algorithm? Show that your bound is tight by giving an appropriate example.

6. **[Hall's drawing of graphs]** (3 points)

Let $L_G = D_G - A_G$ be the Laplacian of an undirected graph $G = (V, E)$ with eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ and corresponding eigenvectots $\Psi_1 = \mathbb{1}/\sqrt{n}, \Psi_2, \ldots, \Psi_n$. For any positive integer $k < n$, Let $x_1, \ldots, x_k$ be orthonormal vectors that are all orthogonal to $\mathbb{1}$. Then prove that

$$\sum_{i=1}^{k} \langle x_i, L_G x_i \rangle \geq \sum_{i=2}^{k+1} \lambda_i,$$

and this inequality is tight only when $\langle x, \Psi_j \rangle = 0$ for all j such that $\lambda_j > \lambda_{k+1}$.