COS 445 - PSet 3

Due online Monday, March 29th at 11:59 pm.

Instructions:

- Some problems will be marked as no collaboration problems. This is to make sure you have experience solving a problem start-to-finish by yourself in preparation for the midterms/final. You cannot collaborate with other students or the Internet for these problems (you may still use the referenced sources and lecture notes). You may ask the course staff clarifying questions, but we will generally not give hints.
- Submit your solution to each problem as a **separate PDF** to codePost. Please make sure you're uploading the correct PDFs![1] If you collaborated with other students, or consulted an outside resource, submit a (very brief) collaboration statement as well. Please anonymize your submission, although there are no repercussions if you forget.
- This cheatsheet gives problem solving tips, and guidelines for a "good proof" or "partial progress": `http://www.cs.princeton.edu/~smattw/Teaching/cheatsheet445.pdf`.
- Please reference the course collaboration policy here: `http://www.cs.princeton.edu/~smattw/Teaching/infosheet445sp21.pdf`.

[1] We will assign a minor deduction if we need to maneuver around the wrong PDFs. Please also note that depending on if/how you use Overleaf, you may need to recompile your solutions in between downloads to get the right files.

Problem 1: Repeated Prisoner's Dilemma (50 points)

Consider the following payoff matrix for prisoner's dilemma. The left number in each entry denotes the payoff for Alice, and the right denotes the payoff for Bob. The left two strategies denote Alice's actions, and the top two strategies denote Bob's.

(Alice, Bob)	Cooperate	Defect
Cooperate	(2,2)	(0,5)
Defect	(5,0)	(1,1)

In this problem, we'll consider the case where Alice and Bob repeatedly play prisoner's dilemma for 1000 rounds. That is, for all t from 1 to 1000, Alice and Bob each choose to Cooperate or Defect. Based on their actions, they each receive some payoff according to the payoff matrix above. When deciding which strategy to use in round t, Alice and Bob each know the entire history of decisions made by the other player (and theirself) for the first $t-1$ rounds. We'll call this game Repeated Prisoner's Dilemma.

Observe that a complete strategy for Repeated Prisoner's Dilemma must decide for all t, as a function of the history from the first $t-1$ rounds, which action to play in round t.

Part a: No Dominant Strategies (10 points)

Prove that neither Alice nor Bob has a dominant strategy in Repeated Prisoner's Dilemma.

Hint: Make sure you've understood exactly what a strategy is for Repeated Prisoner's Dilemma, and what it would mean for that strategy to be dominant.

Part b: Iterated Deletion of Dominated Strategies (20 points)

Recall that iterated deletion of weakly dominated strategies repeatedly finds a strategy for some player that is weakly dominated and deletes that strategy (possibly updating which strategies of the other player are weakly dominated), and terminates when both players have no weakly dominated strategies remaining. Recall also that this process is not well-defined, because it may terminate differently depending on how you execute this process (i.e. how you choose which strategies to delete).

Prove that there exists an execution of iterated deletion of weakly dominated strategies that terminates with Alice and Bob each only having a single remaining strategy (and state that strategy).

Hint: Again, make sure you've understood exactly what a strategy is for Repeated Prisoner's Dilemma, and shown that every strategy (except the remaining one) must be deleted.

Part c: Tit-for-Tat (20 points)

Prove that if Alice plays the "Tit-for-Tat" strategy (defined below), then no matter what Bob does, Bob's total payoff after 1000 rounds is at least as large as Alice's.

Definition 1 (Tit-for-Tat) *The Tit-for-Tat strategy plays Cooperate in round one. In any round $t > 1$, it copies its opponent's strategy from round $t-1$.*

Problem 2: Limited Information Scoring Rules (50 points)

In this problem you'll examine whether it's necessary that the predictor be allowed to report their entire belief and not just properties of their belief, even when you only care about one property of their belief. In all parts below, X is a random variable supported on $\mathbb{R}$ (this just means that X is always a real number).

Hint 1: You may use the following fact without proof: the derivative of $\mathbb{E}_{x \leftarrow X}[f(x, y)]$ with respect to y is equal to $\mathbb{E}_{x \leftarrow X}[\frac{\partial f(x,y)}{\partial y}]$. Here, recall that $\mathbb{E}_{x \leftarrow X}[f(x)]$ means "take the expectation of $f(x)$, when x is drawn according to distribution X."

Hint 2: For part of this problem, you may find yourself wanting to show that the unique maximum of a function occurs at a place where the derivative is undefined. Here you should remember that in order to prove a point is the unique maximum, you need to show that it is strictly larger than all points to its left (in which case, you only need to consider the left derivative), and also that it is strictly larger than all points to its right (in which case you only need to consider the right derivative).

Part a (15 points)

Let also X have a unique median: there exists a unique real number $M(X)$ such that $\Pr[X \geq M(X)] \geq 1/2$ and $\Pr[X \leq M(X)] \geq 1/2$. You are interested only in learning $M(X)$, the median of X.

Design a function S such that for all X, the unique argmaximum (over y) of $\mathbb{E}_{x \leftarrow X}[S(y, x)]$ is $y = M(X)$.[2] That is, design a proper scoring rule which takes as input only the reported median of X, and strictly incentivizes the predictor to report the true median of X (and prove that it is proper).

Part b (15 points)

Let also $\mathbb{E}[X]$ be finite (not $\pm\infty$). Now, you are interested in learning only the expected value of X, $\mathbb{E}[X]$. Design a function S such that for all X, the unique argmaximum (over y) of $\mathbb{E}_{x \leftarrow X}[S(y, x)]$ is $y = \mathbb{E}[X]$. That is, design a proper scoring rule which takes as input only the reported mean of X, and strictly incentivizes the predictor to report the true mean of X (and prove that it is proper).

Part c (20 points)

Finally, you are interested in learning only the variance of X, $\mathbb{E}[X^2] - (\mathbb{E}[X])^2$. Prove that there does not exist any function S such that for all X, the unique argmaximum (over y) of $\mathbb{E}_{x \leftarrow X}[S(y, x)]$ is $y = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$. That is, prove that no scoring rule which takes only as input the reported variance of X can strictly incentivize the predictor to report the true variance of X.

Hint: Assume for contradiction that such an S existed. Consider two random variables X and Y with the same variance, and a third distribution Z which is equal to X with probability $1/2$ and Y with probability $1/2$. Assuming that S succeeds for X and Y, what can you say about $\arg\max_y\{\mathbb{E}_{z \leftarrow Z}[S(y, z)]\}$?

[2] The argmaximum is the value y maximizing $\mathbb{E}_{x \leftarrow X}[S(y, x)]$.

Problem 3: Noisy Cascades (60 points)

Recall the model of information cascades from class: I have a bag containing some number of colored balls. With probability 1/2 the bag contains 2 red balls and 1 blue ball, and with probability 1/2 the bag contains 2 blue balls and 1 red ball. One by one, people observe a ball randomly selected from the bag, and guess (based on this observation and all prior guesses) whether the bag has more red balls or blue balls.

In this problem we'll examine a variant of this game. Assume that all players are perfectly rational and guess to maximize their probability of being correct, and know that all other players are perfectly rational and are guessing to maximize their probability of being correct. If the player ever believes that both cases are equally likely, assume that they tie-break to guess the color of their own draw.

The difference between this problem and lecture is that each person's guess is revealed noisily; that is, with probability α, each player's true guess is replaced by a uniformly random color before it is announced to the other players (and everyone knows α, and everyone knows that everyone knows α). But each player is aware of their own draw, without noise.

To be extra clear: if player one truly guesses blue, then this is turned into a revealed guess of blue with probability $1 - \alpha/2$ and a revealed guess of red with probability $\alpha/2$. The other players withness this revealed guess, and have no other information about the true guess.

Note: Most parts of this problem involve non-trivial calculations. You should both "show your work" for any non-trivial calculations, and also state in text why your calculations are correct (e.g. if you apply Bayes' rule, you should state which equality follows from Bayes' rule. You do not need to argue why your algebra/formula manipulation is correct).

Part a (10 points)

Assume that we have not yet reached an information cascade (that is, assume that so far every player's true guess is equal to their draw from the urn). Let $S = \langle G_1, \ldots, G_k \rangle$ be an ordered sequence of revealed guesses, where each G_i is either red or blue, and let exactly B of the k revealed guesses be blue, and $R = k - B$ of the revealed guesses be red. What is the probability, as a function of α, conditioned on the bag containing two red balls and one blue ball, that you will observe exactly the sequence S of guesses?

Hint: First, try to compute the probability that a particular revealed guess G_i is red, conditioned on the bag containing two red balls and one blue ball. As a sanity check, make sure your formula is correct when $\alpha = 1$ and $\alpha = 0$.

Part b (10 points)

Imagine that you have seen exactly the sequence S of revealed guesses before you, where S has B blue revealed guesses, and R red revealed guesses, and you yourself draw a red ball from the urn. Conditioned on this, what is the probability, as a function of α, that the bag contains two red balls and one blue ball (again, assuming that all true guesses before you are equal to that player's draw)?

Hint: Use Bayes' rule. Part a helps you find the probability of seeing S (plus you draw a red) conditioned on there being two red balls and one blue ball.

Part c (15 points)

As a function of α, how many more red revealed guesses than blue revealed guesses (or vice versa) would you have to see before you decide to ignore your own draw (again, assuming that all true guesses before you copied the color of their draw)?

Use this to briefly argue that, for all $\alpha < 1$, a cascade will eventually happen. Also briefly argue that when $\alpha = 1$, a cascade will never happen.

Hint: The math will be simpler if you try to argue that you believe it is more likely the urn is red versus blue, rather than trying to argue that the probability that the urn is red is $> 1/2$.

Part d (10 points)

When $\alpha < 1$, let $S = \langle G_1, \ldots, G_k \rangle$ be an ordered sequence of revealed guesses so that a red cascade begins right after revealed guess k (that is, the $(k+1)^{st}$ player will ignore their draw). Let S' be the ordered sequence S, but with all reds and blues flipped (that is, all reds are now blue, and blues are now reds).

As a function of α, what is $\Pr[S|\text{two reds one blue}]/\Pr[S'|\text{two reds one blue}]$?

Part e (15 points)

When $\alpha < 1$, as a function of α, what is the probability that the eventual cascade is correct?

Extra Credit: Another Algorithm for Nash

Recall that extra credit is not directly added to your PSet scores, but will contribute to your participation grade. Some extra credits are **quite** challenging and will contribute significantly.

For this problem, you may collaborate with any students. You may not consult course resources or external resources. In this problem we will guide you through the proof of a well-known result, so you should not copy the proof from one of the course texts (nor should you try to find a proof from external sources). You must follow the guide below (and not provide an alternative proof).

Consider any symmetric two-player game. That is, $p_1(x, y) = p_2(y, x)$ for all x, y. Consider also the following system of inequalities. We'll refer to this as the Lemke-Howson Polytope.

- Variables $x_1, \ldots, x_n$.
- (Non-negativity) $x_i \geq 0$ for all i.
- (Responsiveness) $\sum_j x_j p_1(i, j) \leq 1$.

Part a

Say that an action i is covered in $\vec{x}$ if either the non-negativity constraint is tight (i.e. $x_i = 0$) or the Responsiveness constraint is tight (i.e. $\sum_j x_j p_1(i, j) = 1$), or both. Prove that if $\vec{x}$ is inside the Lemke-Howson polytope, and $\vec{x} \neq 0$, and all actions i are covered in $\vec{x}$, then $\vec{x}/|\vec{x}|_1$ is a symmetric Nash equilibrium (that is, $\vec{x}/|\vec{x}|_1$ is a best response to itself).

Part b

For this part, you should assume that any set of n equations taken above have a solution, and that this solution is unique.

The Lemke-Howson algorithm starts from the point $\vec{0}$ and repeatedly pivots. That is, the current point will always have exactly n tight constraints. The pivot will pick one of these constraints and "relax" it (keeping the other $n - 1$ tight). A new constraint will become tight, and this will be the new point (you do not need to prove that this procedure is well-defined).

From $\vec{0}$, the pivot rule simply picks an arbitrary tight constraint to relax (let's say $x_1 = 0$). This causes a new constraint to become tight. If it's the 1st Responsiveness constraint, then by Part a we've found a Nash and are done! If not, then we have exactly one double-covered action. That is, there is some action i such that the non-negativity and Responsiveness constraints are both tight. We pick the non-negativity constraint for i to relax next.

In general, for our current point $\vec{x} \neq \vec{0}$, if there is no double-covered action we terminate (and hope that it's a Nash and not back at $\vec{0}$). If there's a double-covered action, it's because we just made one of the constraints for i tight. So relax the other one and continue.

Prove that the Lemke-Howson algorithm will never revisit a vertex $\vec{y}$ without first revisiting the origin. You may use without proof the fact that if $\vec{z}$ pivots to $\vec{w}$ when constraint C is relaxed, causing constraint D to become tight, then $\vec{w}$ pivots to $\vec{z}$ when constraint D is relaxed, causing constraint C to become tight.

Part c

Prove that the Lemke-Howson algorithm cannot ever return to $\vec{0}$. Conclude that the Lemke-Howson algorithm finds a Nash after at most $\binom{2n}{n}$ pivots.
Hint: You may want to prove that the algorithm terminates as soon as 1 becomes covered.