Trabalho 2 - Algoritmos e Incerteza
Entrega: 13 de Novembro

Marco Molinaro

Teoria 1: Redução OCO para OLO

Nesse exercício mostraremos passo a passo como utilizar algoritmos de Online Linear Optimization (OLO) para resolver problemas mais gerais de Online Convex Optimization (OCO). Vamos relembrar esses problemas brevemente:

OLO: temos playing set $P \subseteq \mathbb{R}^d$ arbitrário; no tempo t jogamos $p^t \in P$, e após isso vemos a função de perda linear $p \mapsto \langle \ell^t, p\rangle = \sum_i \ell^t_i p_i$; a perda que tomamos é então $\langle \ell^t, p^t\rangle$. O *regret* do algoritmo é

$$[\text{total loss algo}] - [\text{loss best fixed action}] = \sum_t \langle \ell^t, p^t\rangle - \min_{p\in P} \sum_t \langle \ell^t, p\rangle.$$

OCO: temos playing set $P \subseteq \mathbb{R}^d$ arbitrário; no tempo t jogamos $p^t \in P$, e após isso vemos a função de perda $f_t : \mathbb{R}^d \to \mathbb{R}$ **convexa**; a perda que tomamos é então $f_t(p^t)$. O *regret* do algoritmo é

$$[\text{total loss algo}] - [\text{loss best fixed action}] = \sum_t f_t(p^t) - \min_{p\in P} \sum_t f_t(p).$$

Você não precisa saber nada sobre funções convexas ou mesmo gradientes, e pode trabalhar com esse objetos "sintaticamente". A única propriedade desses objetos que precisamos é a seguinte. Considere uma função convexa $f : \mathbb{R}^d \to \mathbb{R}$ qualquer que seja diferenciável (vamos assumir que todas as função que aparecem nesse exercício são diferenciáveis). Dado um ponto $q \in \mathbb{R}^d$, considere a função afim $L_q[f] : \mathbb{R}^d \to \mathbb{R}$ dada por

$$L_q[f](p) = f(q) + \langle \nabla f(q), p - q\rangle \quad (= \langle \nabla f(q), p\rangle + f(q) - \langle f(q), q\rangle);$$

note que o termo $f(q) - \langle f(q), q\rangle$ é constante, i.e., independente de p.

Propriedade de convexidade: $L_q[f](p) \le f(p)$ para todo $p \in \mathbb{R}^d$.

(Para visualizar: em $d = 1$ podemos desenhe uma função convexa f; $L_q[f]$ é a reta tangente ao grafo da função no ponto $(q, f(q))$. Nesse caso, $L_q[f]$ é também a aproximação de f dada pela série de Taylor de ordem 1 no ponto q.)

Questão 1. Considere o problema Online *Affine* Optimization (OAO), onde a função de perda no tempo t tem a forma $p \mapsto \langle \ell^t, p\rangle + c_t$, onde $\ell^t \in \mathbb{R}^d$ e $c_t \in \mathbb{R}$ (ou seja, essa função é afim, ao invés de linear). Considere qualquer algoritmo $\mathcal{A}$ para OLO com regret R. Verifique que usar essa algoritmo em Online Affine Optimization diretamente (ou seja, ignorando os c_t's) também da regret R nesse modelo.

Questão 2. Considere então qualquer algoritmo $\mathcal{A}$ para OAO em playing set P com regret R. Considere então o seguinte algoritmo $\mathcal{A}'$ para OCO em playing set P:

- No tempo 1 $\mathcal{A}'$ joga p^1 como $\mathcal{A}$. Ao ver a função de perda f_1, $\mathcal{A}'$ define a função afim $L_{p^1}[f_1]$ e a alimenta ao algoritmo $\mathcal{A}$ (que a usa para atualizar p^1 e obter o próximo ponto p^2).
- No tempo 2, $\mathcal{A}'$ joga o p^2 atualizado pelo $\mathcal{A}$. Novamente, ao ver a função de perda f_2, o algoritmo $\mathcal{A}'$ define a função afim $L_{p^2}[f_2]$ e a alimenta ao algoritmo $\mathcal{A}$ (que a usa para atualizar p^2 e obter o próximo ponto p^3).
- ...

Ou seja, temos o jogo OCO "real" com funções de perda f_t's e construímos o jogo OAO "fake" com perdas $L_{p^t}[f_t]$; o algoritmo $\mathcal{A}'$ então joga o jogo OCO utilizando as ações do algoritmo $\mathcal{A}$ no jogo OAO.

Prove que a perda total de $\mathcal{A}'$ é igual a perda total de $\mathcal{A}$ em seus respectivos jogos, ou seja, $\sum_t f_t(p^t) = \sum_t L_{p^t}[f_t](p^t)$.

Prove também que a melhor solução fixa no jogo OCO tem perda total maior que a melhor solução fixa no jogo OAO, ou seja,

$$\min_{p\in P} \sum_t f_t(p) \geq \min_{p\in P} \sum_t L_{p^t}[f_t](p).$$

Juntas, essas afirmações mostram que o regret do algoritmo $\mathcal{A}'$ no jogo OCO é no máximo o regret do algoritmo $\mathcal{A}$ no jogo OAO

Questão 3. Ou seja, utilizando as reduções das questões anteriores, mostramos como utilizar qualquer algoritmo de OLO pra jogar bem o jogo OCO.

Considere o algoritmo FTRL com regularização $\mathcal{R} = \frac{1}{2}\|.\|_2^2$ **para o jogo OLO** (chame esse algoritmo de $\bar{\mathcal{A}}$). Suponha que temos um jogo OCO onde P tem diâmetro D ou seja, para todo $x, y \in P$ temos $\|x-y\|_2 \leq D$, e todas a funções f_t satisfaçam $\|\nabla f_t(p)\|_2 \leq G$ para todos os pontos $p \in P$. Se utilizarmos o algoritmo $\bar{\mathcal{A}}$ para jogar esse jogo OCO (seguindo as reduções acima), qual é o regret obtido nesse jogo OCO?

Observação 1 *Pode-se mostrar que o algoritmo obtido para o jogo OCO é* exatamente *o Online Gradient Descent.*

Teoria 2: Contextual Experts

Considere a seguinte extensão do Problema de Experts Abstratos: Temos um conjunto $\{1, 2, \ldots, m\}$ de ações possíveis (e.g., aumentar/diminuir/manter nosso nível de produção). Temos também um conjunto de **contextos** $\{1, 2, \ldots, k\}$ (e.g., informação extra sobre condição atual do mundo) que podem ser utilizados para auxiliar na escolha das ações. O protocolo do jogo é: no instante t temos (em ordem)

1. Vemos o contexto $c_t \in \{1, \ldots, k\}$ do dia
2. Temos que escolher uma distribuição $p^t \in \Delta^m$ sobre as ações m
3. Vemos o vetor de perdas $\ell^t \in [0, 1]^m$ das ações.

O objetivo é minimizar a nossa perda total $\sum_{t=1}^T \langle p^t, \ell^t \rangle$. Mas agora queremos competir contra a **melhor politica de escolha de ação baseada nos contextos**, ou seja, na melhor função $\pi : \{1, \ldots, k\} \to \{1, \ldots, m\}$ que diz pra cada contexto c qual é a ação $\pi(c)$ que deve ser tomada. Mais

precisamente, seja Π o conjunto de todas as funções/politicas $\pi : \{1, \ldots, k\} \to \{1, \ldots, m\}$; queremos minimizar o regret

$$Regret := \underbrace{\sum_{t=1}^{T} \langle p^t, \ell^t \rangle}_{\text{nossa perda total}} - \underbrace{\min_{\pi \in \Pi} \sum_{t=1}^{T} \ell^t_{\pi(c_t)}}_{\text{perda total da melhor politica}} .$$

Note que o Problema dos Experts Abstrato é um caso especial desse problema quando o conjunto de contextos é vazio (ou só tem um elemento). Note também que o conjunto de funções/politicas Π tem m^k elementos $\{\pi_1, \pi_2, \ldots, \pi_{m^k}\}$ (experimente com $k = 2$, por exemplo).

Questão única: Utilize como caixa preta algum dos algoritmo visto em sala para obter no problema acima regret cuja dependência em T é $\sqrt{T}$ (terá também dependência em outros parâmetros). **Demonstre** que seu algoritmo obtém o regret desejado; em particular, tome cuidado em explicitar a distribuição p^t sobre as **ações** $\{1, \ldots, m\}$ jogada pelo seu algoritmo no tempo t.

Prática

Use algum modelo de Online Learning ([abstract] experts, OLO, OCO, bandit, etc.) pra fazer a classificação online de emails em spam/not spam. Um bom ponto de partida pode ser o exemplo de modelagem de "predição com contexto" feito em aula. Teste seu algoritmo na base `https://archive.ics.uci.edu/ml/datasets/spambase`.

Descreva "formalmente" como você modelou o problema (ou seja, qual o conjunto de ações possíveis, funções de perdas, etc.) Comente sobre qualquer hipótese feita para justificar o modelo escolhido, e teste-a pra ver se o dataset a satisfaz.

Reporte a performance do seu algoritmo.