Trabalho 2 - Algoritmos e Incerteza
Entrega: 7 de Novembro

Marco Molinaro

Teoria 1: OCO via Experts

Vamos mostrar que podemos resolver Online Convex Optimization (OCO) utilizando apenas Experts Abstrato.

Considere então uma instância do OCO com playing set convexo $P \subseteq \mathbb{R}^d$ e funções de perda convexas $f_1, \dots, f_T$. Assuma o seguinte:

1. O conjunto P está contido em $[-M, M]^d$.*

2. As funções f_t não mudam de valor muito rápido: existe um valor L tal que se os vetores $x, y \in [-M, M]^d$ diferem de no máximo ε em cada coordenada (i.e., $|x_i - y_i| \leq \varepsilon$ pra todo $i \in \{1, 2, \dots, d\}$), então $|f_t(x) - f_t(y)| \leq \varepsilon L$.†

3. As funções de perda f_t tem valores entre $[0, 1]$.‡

Seja $\bar{P}$ a discretização de P dada por pontos em um grid d-dimensional de espaçamento ε, ou seja, $\bar{P} = P \cap \{-M, -M+\varepsilon, -M+2\varepsilon, \dots, M-\varepsilon, M\}^d$. Considere o algoritmo ExpertOCO que simplesmente trata cada ponto em $\bar{P}$ como um expert. Ou seja, ao invés de tentar jogar pontos em P "quase tão bons quanto o melhor ponto" em P, ExpertOCO tenta jogar pontos em $\bar{P}$ "quase tão bons quanto o melhor ponto" em $\bar{P}$. Para simplificar, vamos assumir a existência de um algoritmo SingleExpert para experts que em cada tempo t escolhe somente um expert i_t (ao invés de uma distribuição sobre os experts) mas com mesma garantia do MWU:

$$\sum_{t=1}^{T} \ell_{i_t}^t \leq \min_i \sum_{t=1}^{T} \ell_i^t + 2\sqrt{T \log(\#\text{experts})}, \tag{1}$$

onde ℓ_i^t é a perda do expert i no tempo t. O algoritmo ExpertOCO é então o seguinte:

ExpertOCO - Inicializa SingleExpert com um expert para cada ponto em $\bar{P}$ - No tempo t: 1. Obtém o expert $x_t \in \bar{P}$ escolhido por SingleExpert para essa iteração 2. Joga o ponto x_t 3. Observa a função de perda f_t (e toma perda $f_t(x_t)$) 4. Calcula a perda $\ell_x^t := f_t(x)$ para cada expert $x \in \bar{P}$ 5. Passa essas perdas para SingleExpert, para que ele se atualize (i.e., se fosse o MWU, essas perdas seriam utilizadas para atualizar os pesos dos experts).

*Isso é essencialmente equivalente a dizer que o θ na garantia do FTRL é no máximo $\sqrt{d}M$.

†Essa condição é chamada de Lipschitz com relação à norma ℓ_∞. Ela é satisfeita se, por exemplo, o G da garantia do FTRL é no máximo $\frac{L}{\sqrt{d}}$.

‡Dadas as hipóteses anteriores, podemos modificar funções de perdas genéricas multiplicando-as por um fator pequeno e "shiftando" elas para satisfazermos que os novos valores estão em $[0, 1]$. Porém, isso altera o regret obtido (temos que "desfazer" essas modificações para obter o regret para as funções originais; não vamos entrar nisso.

Queremos provar que esse algoritmo tem baixo regret no jogo OCO, ou seja,

$$\sum_{t=1}^{T} f_t(x_t) \leq \min_{x \in P} f_t(x) + \text{Regret}.$$

Note que as perdas de interesse são do jogo OCO, e que ainda estamos querendo nos comparar com a melhor solução em P (e não em $\bar{P}$).

Questão 1. Seja

$$\text{OPT} = \min_{x \in P} \sum_{t=1}^{T} f_t(x)$$

o OPT para o problema OCO, e seja

$$\overline{\text{OPT}} = \min_{x \in \bar{P}} \sum_{t=1}^{T} f_t(x)$$

a melhor solução dentre os pontos do conjunto discretizado $\bar{P}$. Utilizando a hipótese de não-mudança-rápida das funções de perda f_t, argumente que

$$\overline{\text{OPT}} \leq \text{OPT} + \varepsilon L T,$$

ou seja, não perdemos muito valor por apenas considerar pontos na discretização $\bar{P}$. (Obs: Sejam x^* e $\bar{x}$ os pontos ótimos referentes a OPT e $\overline{\text{OPT}}$ respectivamente. Note que **não** necessariamente temos $|\bar{x}_i - x_i^*| \leq \varepsilon$ para toda coordenada.)

Questão 2. Utilize a questão anterior mais a garantia de SingleExpert dada em (1) para provar que

$$\sum_{t=1}^{T} f_t(x_t) \leq \text{OPT} + 2\sqrt{Td \log(\frac{2M}{\varepsilon})} + \varepsilon L T.$$

Questão 3. Note então que setando $\varepsilon = \frac{1}{TL}$ obtemos um regret para o jogo OCO (sob as hipóteses acima) com garantia

$$\sum_{t=1}^{T} f_t(x_t) \leq \text{OPT} + 2\sqrt{Td \log(2MTL)} + 1,$$

bem semelhante à do FTRL.

Observação 1 *Note que:*

1. *Essa redução de OCO para experts funciona mesmo quando o playing set e as funções de perda não são convexas. Portanto, o que ganhamos em assumir convexidade são algoritmos cujo regret tem dependência um pouco melhor nos parâmetros do problema (em particular, em T).*

2. *Essa ideia de discretização + experts foi usada (com mais outras ideias) por Dani et al. e Hazan-Li para obter as melhores garantias da época para a versão bandit de OCO.*

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, OCO, bandit, etc.) pra fazer a classificação online de emails em spam/não spam. Um bom ponto de partida podem ser as aplicacoes que discutimos em detalhe 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.