Algoritmos e Incerteza **(PUC-Rio INF2979, 2018.2)**

Lecture 7: Online Convex Optimization

25/09/2018

Lecturer: Marco Molinaro *Scribe: Alessandro Soares*

1 Introdução

Um algoritmo (ou jogo) de Online Convex Optimization (OCO) pode ser descrito da seguinte maneira:

Algorithm 1 Online Convex Optimization

$P \subseteq \Re^n$
for t = 1,..,T **do**
  Algoritmo escolhe um ponto no playing set ($x_t \in P$)
  Adversário escolhe função de perda f_t
  Algoritmo observa função de perda f_t
end for

Objetivo do algoritmo: $\min \sum_{t=1}^{T} f_t(x_t)$. Vamos comparar nossa função objetivo com o ótimo que escolhe apenas um x, isto é:

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

Note que o OCO nada mais é do que uma generalização do jogo dos experts. Por exemplo, basta fazer:

$$P = \Delta^n$$
$$f_t(x) = < l^t, x >$$

E obtemos o problema dos experts.

Nesta aula vamos analisar algoritmos para a resolução de problema desta classe (OCO), que generalizam os problemas vistos até então no curso.

Primeiramente vamos falar do algoritmo "Follow the Leader"(FTL) e vamos analisar o seu REGRET. Uma boa referencia do FTL, para mais detalhes ou coisas especificas, pode ser visto em [3].

Vamos ver que o FTL não é, em algum sentido, estável, e isso faz com que o mesmo tenha alguns resultados relativamente longe do ótimo. Com isso, o algoritmo Follow the Regularized Leader (FTRL) é introduzido, de forma a melhorar a estabilidade do FTL, e obter melhores resultados de Regret. [3] também é uma boa referência para o FTRL.

2 Follow the Leader

O FTL é a primeira ideia trivial que temos ao pensar num algoritmo para resolver o OCO. Pois ao avaliar a função de perda $f_1, ..., f_{t-1}$, o algoritmo pode, de forma gulosa, escolher o x que minimiza a perda considerando todas as funções de perda vistas até então. Em outras palavras:

Algorithm 2 Follow the Leader

1: x_1 é arbitrario em P
2: **for** t = 2,..,T **do**
3: $\quad x_t \in arg \min_{x_t \in P} \sum_{k=1}^{t-1} f_k(x_t)$
4: **end for**

Para exemplificar o comportamento do algoritmo num caso concreto, suponha que:

$$P = \Delta^2 \tag{2}$$
$$f_1(x) =< (0.1, 0), x > \tag{3}$$
$$f_2(x) =< (0, 1), x > \tag{4}$$
$$f_3(x) =< (1, 0), x > \tag{5}$$
$$f_4(x) =< (0, 1), x > \tag{6}$$

Soluções do FTL:

- x_1 é qualquer coisa
- Perda vista até $T - 1$: $0.1x_1 + 0x_2$. Solução: $x_2 = (0, 1)$
- Perda vista até $T - 1$: $0.1x_1 + 1x_2$. Solução: $x_3 = (1, 0)$
- Perda vista até $T - 1$: $1.1x_1 + 1x_2$. Solução $x_4 = (0, 1)$
- ...

Note que o que acontece é que o FTL vai trocando bruscamente de solução a cada iteração, pois a cada uma delas a função de perda faz com que a solução de mínimo mude entre os experts, pois eles vão trocando de quem é melhor e pior.

Agora vamos avaliar quanto custa o algoritmo com as funções de perda reais.

- $t = 1$ qualquer perda
- $t = 2$: $x_1 = 0$, $x_2 = 1$, Perda = $0x_1 + 1x_2$ = 1
- $t = 3$: $x_1 = 1$, $x_2 = 0$, Perda = $1x_2 + 0x_2$ = 1
- $t = 4$: $x_1 = 0$, $x_2 = 1$, Perda = $0x_2 + 1x_2$ = 1

- ...

Desta forma, o algoritmo teve o pior desempenho possível, com custo total $\simeq T$.

Para compararmos a performance do algoritmo, vamos considerar o "Ótimo fixo", isto é, o algoritmo offline ótimo que só pode tomar uma única decisão, neste caso a perda do mesmo seria: $OPT \simeq \frac{T}{2}$, pois escolheria um dos x e teria perda = 0 metade do tempo e perda = 1 na outra metade. Assim podemos ver que o REGRET do FTL, neste caso, é:

$$Regret_{FTL} = PERDA - OPT = T - \frac{T}{2}$$

$$Regret_{FTL} = \frac{T}{2} \tag{7}$$

Sendo assim, lembrando que trata-se de um problema genérico do jogo dos experts, a fim de comparação, o regret do MWU em função do T era na ordem de $\sqrt{T}$ (para mais detalhes, verificar em [1]), o que nos dava o resultado interessante de que na média, nosso regret em relação ao OPT era perto de 0.

Sendo assim, concluimos que o FTL na verdade tem uma performance muito baixa comparado com outros tipos de algoritmos para resolver problemas similares, sendo assim é necessário alguma outra solução melhor.

Podemos verificar que o problema do FTL é que o mesmo fica trocando muito abruptamente de solução, o que o deixa muito exposto à avaliação das funções de perda sem se preocupar com as decisões anteriores em algum sentido. Com isso em mente, vamos ver o Lemma 2.1 que mostra um bound para o Regret do FTL, com considerações sobre sua estabilidade:

Lemma 2.1. *(BTL-FTL) Regret do FTL* $\leq \sum_{t=1}^{T} (f_t(x_t) - f_t(x_{t+1}))$

Considerando o Lema, se a solução do algoritmo na instância é estável, isto é $x_{t+1} \simeq x_t \Rightarrow REGRET \cong 0$.

Claro que o Lema só vale para o algoritmo FTL, qualquer outro algoritmo que, de alguma forma, fixe a solução, o Lema não necessariamente valeria mais.

Por exemplo, vamos utilizar o Lemma num jogo onde as funções de perda são sempre iguais, i.e., $f_1 = f_2 = ... = f_T = f$. Neste caso, vamos ver como o FTL toma as decisões:

- x_1 é qualquer coisa
- $x_2 \in arg \min\limits_x f(x)$
- $x_3 \in arg \min\limits_x 2f(x)$
- De forma geral: $x_t \in arg \min\limits_x (t-1)f(x)$

Note que, como as funções se repetem, e as funções de perda só se diferem por uma constante multiplicativa, a solução de minimo é sempre igual, i.e:

$$arg \min kf(x) = arg \min f(x) \ \ \forall k \in R^+ \tag{8}$$

Sendo assim todas as decisões $x_1 = x_2 = ... = x_T$ serão iguais. E pelo Lema 2.1, o Regret é igual a zero neste caso. Esse resultado é fácil de interpretar, pois como todas as funções são iguais, por definição, estamos no tempo t minimizando as perdas até o tempo $t-1$ mas que é igual às perdas até o tempo t fora uma constante.

Demonstração do lema. Por definição do FTL que x_{t+1} seja a melhor solução para as perdas até tempo o t.

Sendo assim: $OPT = x_{T+1}$.

O Regret do FTL é então:

$$\sum_{t=1}^{T} (f_t(x_t) - f_t(OPT)) = \sum_{t=1}^{T} (f_t(x_t) - f_t(x_{T+1}))$$

Sendo assim, reescrevendo o Lema, queremos provar que:

$$\textstyle\sum_{t=1}^{T} (f_t(x_t) - f_t(x_{T+1})) \leq \sum_{t=1}^{T} (f_t(x_t) - f_t(x_{t+1}))$$

Cortando a primeira parcela ($\sum_{t=1}^{T} f_t(x_t)$), ficamos com:

$$\sum_{t=1}^{T} f_t(x_{t+1}) \leq \sum_{t=1}^{T} f_t(x_{T+1}) \tag{9}$$

Vamos terminar a prova usando indução. Então supondo que vale para $T-1$:

$$\sum_{t=1}^{T-1} f_t(x_{t+1}) \leq \sum_{t=1}^{T-1} f_t(x_T) \leq \sum_{t=1}^{T-1} f_t(x_{T+1}),$$

aonde a ultima desigualdade $\sum_{t=1}^{T-1} f_t(x_T) \leq \sum_{t=1}^{T-1} f_t(x_{T+1})$, pois a solução x_T é a ótima em $t = T$ e portanto a solução x_{T+1} é por definição mais cara em $t = T$. Agora basta somar $f_T(x_{T+1})$ em ambos os lados da expressão acima:

$$\sum_{t=1}^{T-1} f_t(x_{t+1}) + f_T(x_{T+1}) \leq \sum_{t=1}^{T-1} f_t(x_{T+1}) + f_T(x_{T+1})$$

Colocando para dentro do somatório para terminar a prova:

$$\sum_{t=1}^{T} f_t(x_{t+1}) \leq \sum_{t=1}^{T} f_t(x_{T+1})$$

$\square$

3 Follow the Regularized Leader

A principal lição do Lema 2.1 é: Caso o FTL seja estável, ele é necessariamente ótimo! Mas claro que isso nem sempre é verdade. Dito isso, é razóavel procurarmos algoritmos parecidos com o FTL, só que buscando, em algum sentido, a estabilidade do mesmo. Algumas idéias são:

- Penalizar o algoritmo por trocas de solução
- Estabilizar as decisões com uma especie de combinação convexa entre o FTL e x_{t-1}
- Regularização para evitar overfitting do passado.

Nesta aula vamos focar nesta ultima idéia de regularização. Mas como regularizar o FTL? Podemos adicionar um termo de regularização na função de minimo das perdas:

$$x_t \in arg \min_{x \in P} \left\{ \sum_{t=1}^{T-1} f_t(x) + \frac{1}{\eta}||x||^2 \right\} \tag{10}$$

A idéia é penalizar soluções muito longe da origem. Porém a regularização pode ser qualquer função α - fortemente convexa, de acordo com a Definição 1.

$$x_t \in arg \min_{x \in P} \left\{ \sum_{t=1}^{T-1} f_t(x) + \frac{1}{\eta}R(x) \right\} \tag{11}$$

Definicao 1. *f é α - fortemente convexa se* $\forall x, y \; f(y) \leq f(x) + < \nabla f(x), y - x > + \frac{\alpha}{2}||y - x||^2$

Em português, f é α - fortemente convexa se a mesma é uma função convexa mais um termo quadrático pequeno. Exemplo: $\frac{1}{2}x^2$ é 1-fortemente convexa; $\frac{1}{2}||x||_2^2 = \frac{1}{2}\sum_i x_i^2$ é 1-fortemente convexa também.

Algorithm 3 Follow the Regularized Leader

1: x_1 é escolhido
2: $R(x)$ é α - fortemente convexo
3: **for** t = 1,..,T **do**
4: $\quad x_t \in arg \min_{x_t \in P} \left\{ \sum_{k=1}^{T-1} f_k(x_t) + \frac{1}{\eta}R(x) \right\}$
5: **end for**

Vamos usar o FTRL no exemplo onde o FTL tinha ido mal, para ver se a regularização realmente ajuda.

$$P = \Delta^2$$
$$R(x) = \frac{1}{2}(x_1^2 + x_2^2)$$
$$f_1(x) = < (0.1, 0), x >$$

$$f_2(x) =< (0,1), x >$$
$$f_3(x) =< (1,0), x >$$
$$f_4(x) =< (0,1), x >$$

Soluções do FTRL, usando que $x_2 = 1 - x_1$

- $x_{t=1} = (0.5, 0.5)$, poderia ser qualquer coisa.
- $x_{t=2} = arg \min_{x_1 \in [0,1]} (0.1x_1 + 2x_1^2 - 2x_1 + 1)$ = $(0.45, 0.55)$
- $x_{t=3} = arg \min_{x_1 \in [0,1]} (0.1x_1 + 3x_1^2 - 4x_1 + 2.5) = (0.725, 0.275)$

Aqui percebemos que a solução de x_3 ainda muda o expert no qual o algoritmo "prefere", porém a transição é muito mais suave, o que ocorreu por conta da regularização.

Qual seria a sensibilidade (ou "função") do η neste caso?

- Se $\eta \to \infty \to$ FTRL = FTL (Não estável)
- Se $\eta \to 0 \to$ FTRL esqueçe a história e minimiza a regularização (não aprende)

Lemma 3.1. *Regret FTRL é* $\leq 2G\sqrt{T\frac{\Theta}{\alpha}}$ *, onde:*

- *Onde G é o Upper-Bound para o gradiente das funções de perda:* $||\nabla f_t(x)|| \leq G$ *(Mede o quão rapidamente as funções de perda crescem).*
- Θ *é o tamanho da regularização:* $\Theta \geq R(x) - R(y)\ \forall x, y$*. Logo:*

$$\Theta \geq \max_{x \in P} R(x) - \min_{x \in P} R(x)$$

- α *é o nivel de fortemente convexo das funções de regularização, que devem ser* α *- fortemente convexas.*

Este bound é, em algum sentido, o que estávamos procurando, pois tem a $\sqrt{T}$ aparecendo, ao invés de um T.

Analizando a sensibilidade do bound, se por exemplo a gente colocasse α muito grande, o Θ iria compensá-lo de alguma forma (pois as funções de perda teriam picos muito grandes), não fazendo muito sentido escolher, por exemplo, α muito grande, o que poderia parecer uma boa idéia olhando para o Regret.

Demonstração do Lema 3.1. Para provar o Lema 3.1, vamos considerar os seguintes lemas abaixo:

Lemma 3.2. *Regret FTRL* $\leq \sum_{t=1}^{T} [f_t(x_t) - f_t(x_{t+1})] + \frac{1}{\eta}\Theta$

Note que a primeira parcela do REGRET é relativo à estabilidade e a segunda parcela ao tamanho da função de regularização.

Demonstração do Lema 3.2. Usando o Lema 2.1, a ideia da prova é:

- Adicione tempo 0 $f_0(x) = \frac{1}{\eta}R(x)$
- FTRL = BTL-FTL no jogo extendendo com o tempo 0
- Aplica Lema 2.1 e desconta o f_0 pois não faz parte do jogo real.

Aplicando o Lema 2.1 ao jogo extendido:

$$REGRET \leq \sum_{t=0}^{T} (f_t(x_t) - f_t(x_{t+1}))$$

$$REGRET \leq \sum_{t=1}^{T} (f_t(x_t) - f_t(x_{t+1})) + \frac{1}{\eta}R(x_0) - \frac{1}{\eta}R(x_1)$$

$$REGRET \leq \sum_{t=1}^{T} (f_t(x_t) - f_t(x_{t+1})) + \max_{x\in P}\frac{1}{\eta}R(x) - \min_{x\in P}\frac{1}{\eta}R(x)$$

$$REGRET \leq \sum_{t=1}^{T} (f_t(x_t) - f_t(x_{t+1})) + \frac{1}{\eta}\Theta$$

□

Lemma 3.3. *Se R é α - fortemente convexa, então:*

$$f_t(x_t) - f_t(x_{t+1}) \leq \frac{\eta}{\alpha}G^2$$

Demonstração. A demonstração completa do Lema 3.3 pode ser vista em [2]. Aqui vamos mostrar para o caso especial:

$$P = \Re^n$$
$$R = \frac{1}{2}||x||_2^2 \ (\alpha = 1)$$
$$f_t(x) = < l^t, x >$$

Note que $\frac{\partial f_i}{\partial x_i} = l_i^t$, e portanto pela hipotese $||l^t|| \leq G$:

$$f_t(x_t) - f_t(x_{t+1}) = < l^t, x_t > - < l^t, x_{t+1} >$$
$$f_t(x_t) - f_t(x_{t+1}) = < l^t, x_t - x_{t+1} >$$
$$f_t(x_t) - f_t(x_{t+1}) = \|l_t\|\|x_t - x_{t+1}\|cos(\theta)$$
$$f_t(x_t) - f_t(x_{t+1}) \leq \|l_t\|\|x_t - x_{t+1}\|$$
$$f_t(x_t) - f_t(x_{t+1}) \leq G\|x_t - x_{t+1}\| \tag{12}$$

Agora vamos utilizar o algoritmo FTRL e calcular x_t explicitamente:

$$\min \underbrace{< l^1, x > + ... + < l^{t-1}, x > + \frac{1}{\eta}\frac{1}{2}||x||_2^2}_{=F}$$

Derivando e igualando a zero:

$$\nabla F(x) = 0$$

$$\frac{\partial F}{\partial x_i}(x) = (l_i^1 + .. + l_i^{t-1}) + \frac{1}{\eta} x_i = 0$$

$$x_i = -\eta \sum_{j=1}^{t-1} l_i^j$$

Escrevendo x na forma vetorial:

$$x^t = -\eta \left(\sum_{j=1}^{t-1} l^j \right)$$

$$x^{t+1} = -\eta \left(\sum_{j=1}^{t} l^j \right)$$

$$x^t - x^{t+1} = \eta \left(\sum_{j=1}^{t} l^j \right) - \eta \left(\sum_{j=1}^{t-1} l^j \right) = \eta \left(\sum_{j=1}^{t-1} l^j \right) - \eta \left(\sum_{j=1}^{t-1} l^j \right) - \eta l^t$$

$$x^t - x^{t+1} = -\eta l^t$$

$$||x^t - x^{t+1}|| = \eta ||l^t||$$

$$||x^t - x^{t+1}|| \leq \eta G \tag{13}$$

Substituindo a Equação (13) na Equação (12):

$$f_t(x_t) - f_t(x_{t+1}) \leq \frac{\eta G^2}{1} \tag{14}$$

Terminando assim a prova do Lema 3.3, para o caso especial descrito, onde $\alpha = 1$ □

Voltando à prova do Lema 3.1:

$$Regret_{FTRL} \leq \frac{1}{\alpha} T \eta G^2 + \frac{1}{\eta}\Theta$$

Agora temos que escolher η de forma a minimizar o $Regret_{FTRL}$. Para isto, basta fazer:

$$\frac{\partial}{\partial \eta} \left(\frac{1}{\alpha} T \eta G^2 + \frac{1}{\eta}\Theta \right) = 0$$

$$\frac{1}{\alpha}TG^2 - \frac{\Theta}{\eta^2} = 0$$

$$\eta = \frac{1}{G}\sqrt{\frac{\alpha\Theta}{T}}$$

$$Regret_{FTRL} \leq \frac{TG}{\alpha}\sqrt{\frac{\alpha\Theta}{T}} + \frac{G\Theta}{\sqrt{\frac{\alpha\Theta}{T}}}$$

$$Regret_{FTRL} \leq \frac{TG}{\alpha}\sqrt{\frac{\alpha\Theta}{T}} + \frac{GT}{\alpha}\sqrt{\frac{\alpha\Theta}{T}}$$

$$Regret_{FTRL} \leq 2\frac{TG}{\alpha}\sqrt{\frac{\alpha\Theta}{T}}$$

$$Regret_{FTRL} \leq 2TG\sqrt{\frac{\Theta}{\alpha T}}$$

$$Regret_{FTRL} \leq 2G\sqrt{\frac{T\Theta}{\alpha}}$$

Terminando assim, a prova do Lema 3.1. □

Agora vamos aplicar a garantia do FTRL ao problema dos experts abstrato. Para isso vamos analisar cada uma das parcelas que compõe o jogo, tais como o *playing set*, as funções de perdas e a função de regularização, e aplicar suas características no Regret do FTRL para o OCO:

- Espaço do *playing set*: $P = \Delta^m$
- Funções de perdas: $f_t(x) =< l^t, x >$
- Função de regularização: $R(x) = \frac{1}{2}||x||_2^2$ Implica num α - FC de $\alpha = 1$
- Analizando o maior gradiente das funções de perdas: $G = \nabla f_t = l^t = ||l^t|| \leq ||(1, 1, 1, 1, ..., 1)||$
- $G \leq \sqrt{\sum_{i=1}^{m} 1} \leq \sqrt{m}$
- Θ é definido como $\max R(x) - \min R(x)$
- $\Theta : \min R(x) = 0$
- $\Theta : \max R(x) = \max_{x \in \Delta^m} \frac{1}{2}||x||_2^2 = \frac{1}{2}$
- $\Theta : \frac{1}{2} - 0 = \frac{1}{2}$

$$FTRL \leq 2G\sqrt{\frac{T\Theta}{\alpha}}$$

$$FTRL \leq 2\sqrt{\frac{Tm}{2}}$$

Para esse problema dos experts, a melhor garantia com MWU é $2\sqrt{\frac{Tlog(m)}{2}}$.

A diferença entre o FTRL e o MWU nesse caso é um $\sqrt{m}$ contra um $\sqrt{log(m)}$. Apesar do MWU ser marginalmente melhor, esta tudo bem, pois conseguimos um algoritmo muito mais customizável e genérico que o MWU, com uma garantia quase que tão boa quanto.

Outra observação interessante é que o MWU pode ser derivado do FTRL, com a seguinte regularização:

$$R(x) = \sum_i x_i log(x_i) \tag{15}$$

Esse fato mostrado na Equação (15) levanta outras questões como: Como escolher a regularização ótima? Essas questões dependem de, por exemplo, a geometria do playing set P.

Referências

[1] E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

[2] B. McMahan. Follow-the-regularized-leader and mirror descent: Equivalence theorems and l1 regularization. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 525–533, 2011.

[3] S. Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.