Introduction to Reinforcement Learning

A mini course @ HCMUS, Vietnam

Lectures 7-9 (cont'd)

Long Tran-Thanh

long.tran-thanh@warwick.ac.uk

University of Warwick, UK

Adversarial Online Learning

Stochastic vs. Adversarial

Adversarial to stochastic:

- Any algorithm with low regret in adversarial setting can also produce low regret in stochastic setting
- Claim: regret in stochastic setting is smaller than regret in adversarial setting
- Implication 1: if adversarial regret is bounded above by a sub-linear bound -> also the stochastic bound
- Implication 2: The other direction is not true

(Adversarial) Online Optimisation

A generalised version of adversarial bandits

Recall the model of classical multi-armed bandit problem:

- Time step *t*: opponent chooses value r(t,i) for each arm *i* from K arms (indexed by 1..K), player chooses arm *i(t)*
  - Stochastic: r(t,i) ~ D(i) (drawn iid); adversarial: r(t,i) arbitrarily chosen

(Adversarial) Online Optimisation

A generalised version of adversarial bandits

Recall the model of classical multi-armed bandit problem:

- Time step *t*: opponent chooses value v(t,i) for each arm *i* from K arms (indexed by 1..K), player chooses arm *i(t)*
  - Stochastic: v(t,i) ~ D(i) (drawn iid); adversarial: v(t,i) arbitrarily chosen

More general model:

- X: action space/space of possible arms (e.g., classical bandit as X = {1,2,..K} set of integers)
- Time step *t*: opponent chooses function $f_t : \mathbb{X} \to \mathbb{R}$
- Player chooses $x \in \mathbb{X}$, and receives $f_t(x)$
- Goal: minimise regret $\max_x \sum_{t=1}^{T} f_t(x) - \sum_{t=1}^{T} f_t(x_{A(t)})$ (for expected regret, take the expectation)

Online Optimisation: Formulation

- X: action space/space of possible arms (e.g., classical bandit as X = {1,2,..K} set of integers)
- Time step *t*: opponent chooses function $f_t : \mathbb{X} \to \mathbb{R}$
- Player chooses $x \in \mathbb{X}$, and receives $f_t(x)$
- Goal: minimise regret $\max_x \sum_{t=1}^{T} f_t(x) - \sum_{t=1}^{T} f_t(x_{A(t)})$

Online Optimisation: Formulation

- X: action space/space of possible arms (e.g., classical bandit as X = {1,2,..K} set of integers)
- Time step *t*: opponent chooses function $f_t : \mathbb{X} \rightarrow \mathbb{R}$
- Player chooses $x \in \mathbb{X}$, and receives $f_t(x)$
- Goal: minimise regret $\max_x \sum_{t=1}^{T} f_t(x) - \sum_{t=1}^{T} f_t(x_{A(t)})$

How to identify good (i.e., no-regret) A(t)?

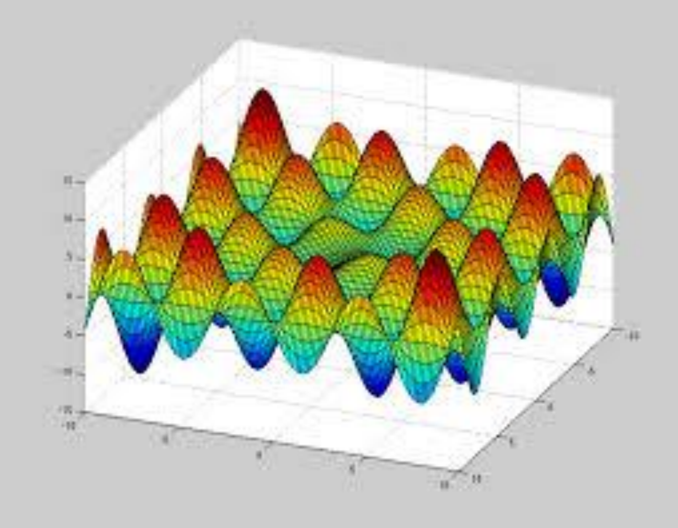
- Degree of feedback (A(t) depends on this)
  - Full information feedback: at the end of each round *t,* we see the whole f_t (we will know its value for each x)
  - Bandit feedback: at the end of each round *t,* we can only see the value of chosen x

Can We Always Efficiently Solve These Problems?

- Efficiency = computational (i.e., can we solve in polynomial time?)

Can We Always Efficiently Solve These Problems?

- Efficiency = computational (i.e., can we solve in polynomial time?)
- Answer: highly depends on the f functions



- When f is highly non-convex/non-linear, even with the full knowledge of f in advance, we cannot solve the problem efficiently (i.e., cannot find the global optimum in polynomial time)

Hopeful Cases

Discrete models:

- Small number of arms (trivial - not interesting)
- Arms with structures (e.g., combinatorial)

Continuous models:

- Linear reward function
- Convex
- Non-convex unimodal
- Non-convex + smoothness conditions (stochastic only)

Combinatorial Bandits

- X: action space – subset of binary vectors: $\mathbb{X} \subseteq \{0,1\}^K$
- Each arm: $x_t = [0, 1, 0, 1, ...0, 0]$
- Adversary chooses $v_t = [v_{t,1}, v_{t,2}, \ldots, v_{t,K}]$ at the beginning of each time step *t* (before we choose)
- Our reward: $f_t(x_t) = v_t \cdot x_t = \sum_{k=1}^{K} v_{t,k} x_{t,k}$
- Regret: $\max_x \sum_{t=1}^{T} f_t(x) - \sum_{t=1}^{T} f_t(x_{A(t)}) \;= \max_x \sum_t v_t \cdot x - \sum_t v_t \cdot x_{A(t)}$
- Goal: put an efficient bound on the expectation of the regret

Combinatorial Bandits

- X: action space – subset of binary vectors: $\mathbb{X} \subseteq \{0,1\}^K$
- Each arm: $x_t = [0,1,0,1,...0,0]$
- Adversary chooses $v_t = [v_{t,1}, v_{t,2}, \ldots, v_{t,K}]$ at the beginning of each time step *t* (before we choose)
- Our reward: $f_t(x_t) = v_t \cdot x_t = \sum_{k=1}^{K} v_{t,k} x_{t,k}$
- Regret: $\max_x \sum_{t=1}^{T} f_t(x) - \sum_{t=1}^{T} f_t(x_{A(t)}) = \max_x \sum_t v_t \cdot x - \sum_t v_t \cdot x_{A(t)}$
- Goal: put an efficient bound on the expectation of the regret
- **Semi-bandit feedback** : modification on feedback - we can see not only f_t(x_t), but all the v_(t,i)*x(t,i) values as well (i.e., each entry of the dot product)

Combinatorial Bandits - Algorithms

- Idea: use FPL as a basis

Parameter: $\eta \in \mathbb{R}^+, M \in \mathbb{Z}^+, \gamma \in [0,1]$;

Initialize the estimated reward $\widehat{r} = 0 \in R^n$;
Pick the set of exploration strategies $E = \{v_1, ..., v_n\}$ such that target i is protected in pure strategy v_i.
for t=1,...,T **do**
  Sample $flag \in \{0,1\}$ such that $flag = 0$ with prob. γ;
  if $flag = 0$ **then**
    Let v_t be a uniform randomly sampled strategy from E;
  else
    Draw $z_i \sim \exp(\eta)$ independently for $i \in [n]$ and let $z = (z_1, ..., z_n)$;
    Let $v_t = \arg\max_{v \in \mathcal{V}}\{v \cdot (\widehat{r} + z)\}$;
  end if
  Adversary picks $r_t \in [0,1]^n$ and defender plays v_t.
  Run GR$(\eta, M, \widehat{r}, t)$: estimate $\frac{1}{p_{t,i}}$ as $K(t,i)$;
  Update $\widehat{r}(i) \leftarrow \widehat{r}(i) + K(t,i) r_{t,i} \mathbb{I}(t,i)$; where $\mathbb{I}(t,i) = 1$ for i satisfying $v_{t,i} = 1$; $\mathbb{I}(t,i) = 0$ otherwise;
end for

Extra epsilon-greedy style exploration

Extra part: geometric re-sampling

Combinatorial Bandits – Algorithms (cont’d)

Algorithm 1 The GR Algorithm

Input: $\eta \in \mathbb{R}^+, M \in \mathbb{Z}^+, \widehat{r} \in \mathbb{R}^n, t \in \mathbb{N}$;
Output: $K(t) := \{K(t,1), \ldots, K(t,n)\} \in \mathbb{Z}^n$

Initialize $\forall i \in [n] : K(t,i) = 0, k = 1$;
for k=1,2,...,M **do**
    Repeat step 4 $\sim$ 10 in Algorithm 2 once just to produce $\widetilde{v}$ as a simulation of v_t.
    for all $i \in [n]$ **do**
        if $k < M$ **and** $\widetilde{v}_i = 1$ **and** $K(t,i) = 0$ **then**
            Set $K(t,i) = k$;
        else if $k = M$ **and** $K(t,i) = 0$ **then**
            Set $K(t,i) = M$;
        end if
    end for
    if $K(t,i) > 0$ for all $i \in [n]$, **then break;**
end for

Further readings:

- Neu & Bartok (2015): http://cs.bme.hu/~gergo/files/NB16.pdf
- Xu et al. (2016): https://eprints.soton.ac.uk/387256/1/main.pdf

Combinatorial Bandits – Regret Analysis

Theorem 1. *The total expected regret of FPL with geometric resampling satisfies*

$$R_n \leq \frac{m(\log d + 1)}{\eta} + 2\eta m d T + \frac{dT}{eM}$$

under semi-bandit information. In particular, setting $\eta = \sqrt{(\log d + 1)/(dT)}$ *and* $M \geq \sqrt{dT}/\left(em\sqrt{2(\log d + 1)}\right)$, *the regret can be upper bounded as*

$$R_n \leq 3m\sqrt{2dT(\log d + 1)}.$$

Special Case: Stochastic Combinatorial Bandits

Stochastic version of the combinatorial bandit

- $V_{t,i} \sim D_{i}$ iid

Special Case: Stochastic Combinatorial Bandits

Stochastic version of the combinatorial bandit

- V_{t,i} ~ D_{i} iid

Idea: use a combinatorial version of UCB

Algorithm 1: Learning with Linear Rewards (LLR)

// INITIALIZATION
If $\max_{\mathbf{a}} |\mathcal{A}_{\mathbf{a}}|$ is known, let $L = \max_{\mathbf{a}} |\mathcal{A}_{\mathbf{a}}|$; else, $L = N$;
for $p = 1$ to N **do**
    $n = p$;
    Play any action $\mathbf{a}$ such that $p \in \mathcal{A}_{\mathbf{a}}$;
    Update $(\hat{\theta}_i)_{1\times N}$, $(m_i)_{1\times N}$ accordingly;
end for
// MAIN LOOP
while 1 **do**
    $n = n + 1$;
    Play an action $\mathbf{a}$ which solves the maximization problem

$$\mathbf{a} = \arg\max_{\mathbf{a}\in\mathcal{F}} \sum_{i\in\mathcal{A}_{\mathbf{a}}} a_i \left(\hat{\theta}_i + \sqrt{\frac{(L+1)\ln n}{m_i}} \right); \qquad (4)$$

    Update $(\hat{\theta}_i)_{1\times N}$, $(m_i)_{1\times N}$ accordingly;
end while

Special Case: Stochastic Combinatorial Bandits (cont'd)

Theorem: the expected regret of LLR is at most O(N^4 lnT)

Further reading: Gai et al. (2012). Combinatorial Network Optimization With Unknown Variables: Multi-Armed Bandits With Linear Rewards and Individual Observations.

Link: http://anrg.usc.edu/www/papers/TON-Jan2012.pdf

Combinatorial Optimisation with Full Information

Similar to the combinatorial bandit model, but now we see all the v_k values, not just those which have x_{t,k} = 1 (semi-bandit), or only the sum (bandit feedback)

Many good algorithms, but the main idea is to take a good bandit algorithm, and then just update all the reward estimate of ALL the arms /actions.

For example: we don't need the GR part in FPL

Parameter: $\eta \in \mathbb{R}^+, M \in \mathbb{Z}^+, \gamma \in [0,1]$;

Initialize the estimated reward $\hat{r} = 0 \in R^n$;

Pick the set of exploration strategies $E = \{v_1, ..., v_n\}$ such that target i is protected in pure strategy v_i.

for t=1,...,T **do**

  Sample $flag \in \{0,1\}$ such that $flag = 0$ with prob. γ;

  if $flag = 0$ **then**

    Let v_t be a uniform randomly sampled strategy from E;

  else

    Draw $z_i \sim \exp(\eta)$ independently for $i \in [n]$ and let $z = (z_1, ..., z_n)$;

    Let $v_t = \arg\max_{v \in \mathcal{V}}\{v \cdot (\hat{r} + z)\}$;

  end if

Update every r at each t

Linear Bandits

- X: action space – subset of **real-number** vectors: $\mathbb{X} \subseteq \mathbb{R}^K$
- Each arm: $x_t = \{x_{t,1}, \ldots, x_{t,K}\}$
- Adversary chooses $v_t = [v_{t,1}, v_{t,2}, \ldots, v_{t,K}]$ at the beginning of each time step *t* (before we choose)
- Our reward: $f_t(x_t) = v_t \cdot x_t = \sum_{k=1}^{K} v_{t,k} x_{t,k}$
- Regret: $\max_x \sum_{t=1}^{T} f_t(x) - \sum_{t=1}^{T} f_t(x_{A(t)}) = \max_x \sum_t v_t \cdot x - \sum_t v_t \cdot x_{A(t)}$
- Goal: put an efficient bound on the expectation of the regret

Linear Bandits – Algorithms

Idea: if X is a **finite set** of real-numbered vectors -> use Exp3 style algorithms

- Pseudo reward: $\hat{r}_t(x) = \dfrac{f_t(x)\mathbb{I}(x = x_{A(t)})}{p_t(x)}$

- Probability of pulling x: $\hat{p}_{t+1}(x) = \dfrac{\exp\{-\eta \sum_{\tau=1}^{t} \hat{r}_\tau(x)\}}{\sum_{x' \in \mathbb{X}} \exp\{-\eta \sum_{\tau=1}^{t} \hat{r}_\tau(x')\}}$

$$p_{t+1}(x) = (1-\gamma)\hat{p}_{t+1}(x) + \boxed{\gamma\pi(x)}$$

Uniform distribution over the (combinatorial) vector space

Exploitation

Exploration

Linear Bandits – Algorithms

Bounded-loss assumption: for any t and action x: $v_t \cdot x \leq 1$

Theorem: expected regret is at most $2\sqrt{3KT \log |\mathbb{X}|}$

What if X is continuous?

- Idea: discretise the space of X -> use the previous version of Exp3
- Extra regret from fineness of discretisation level: $O(K\sqrt{T \ln T})$

Can we avoid discretisation?

Online Convex Optimisation

- X: action space – **convex subset** of real-numbered vectors: $\mathbb{X} \subseteq \mathbb{R}^K$
- Each arm: $x_t = \{x_{t,1}, \ldots, x_{t,K}\}$
- Adversary **chooses convex function** $f_t(x)$ at the beginning of each time step *t* (before we choose)
- Our reward: $f_t(x_t)$
- Regret: $\max_x \sum_{t=1}^{T} f_t(x) - \sum_{t=1}^{T} f_t(x_{A(t)})$
- Goal: put an efficient bound on the expectation of the regret

Full Information Feedback – Algorithm

Full information feedback: we see the whole function f_{t} at the end of each t

Online gradient descent algorithm (Zinkevich, 2001)

1: Input: convex set $\mathcal{K}$, T, $\mathbf{x}_1 \in \mathcal{K}$, step sizes $\{\eta_t\}$
2: **for** $t = 1$ to T **do**
3: Play $\mathbf{x}_t$ and observe cost $f_t(\mathbf{x}_t)$.
4: Update and project:

$$\mathbf{y}_{t+1} = \mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)$$
$$\boxed{\mathbf{x}_{t+1} = \prod_{\mathcal{K}}(\mathbf{y}_{t+1})}$$

Projection step: keeps x_(t+1) within the action space X

5: **end for**

Note that we choose x_{t+1} solely based on f_t (so we only use the previous function to choose the next action (!!)

Full Information Feedback – Regret Analysis

D: diameter of X (the largest distance between 2 points in X)

G: upper bound on the norm of any gradients in f_t

Theorem 3.1. Online gradient descent with step sizes $\{\eta_t = \frac{D}{G\sqrt{t}},\ t \in [T]\}$ guarantees the following for all $T \geq 1$:

$$\text{regret}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}^\star \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}^\star) \leq \frac{3}{2} GD\sqrt{T}$$

Further reading: Elad Hazan (2015). Introduction to Online Convex Optimization

Link: http://ocobook.cs.princeton.edu/OCObook.pdf

Bandit Feedback – Algorithms

Bandit feedback: we only see the value of f_t(x_t) at the end of each t (i.e., the value of chosen point)

Idea: reduce this problem back to full information case

E.g., OGD from Zinkevich

1: Input: convex set $\mathcal{K} \subset \mathbb{R}^n$, first $\boxed{\text{order online algorithm}}$ $\mathcal{A}$.
2: Let $\mathbf{x}_1 = \mathcal{A}(\emptyset)$.
3: **for** $t = 1$ to T **do**
4: $\quad$ $\boxed{\text{Generate distribution } \mathcal{D}_t, \text{ sample } \mathbf{y}_t \sim \mathcal{D}_t \text{ with } \mathbf{E}[\mathbf{y}_t] = \mathbf{x}_t.}$
5: $\quad$ Play $\mathbf{y}_t$.
6: $\quad$ $\boxed{\text{Observe } f_t(\mathbf{y}_t), \text{ generate } \mathbf{g}_t \text{ with } \mathbf{E}[\mathbf{g}_t] = \nabla f_t(\mathbf{x}_t).}$
7: $\quad$ Let $\mathbf{x}_{t+1} = \mathcal{A}(\mathbf{g}_1, ..., \mathbf{g}_t)$.
8: **end for**

How?

Bandit Feedback – Algorithms

Bandit feedback: we only see the value of f_t(x_t) at the end of each t (i.e., the value of chosen point)

Idea: reduce this problem back to full information case

E.g., OGD from Zinkevich

Input: convex set $\mathcal{K} \subset \mathbb{R}^n$, first order online algorithm $\mathcal{A}$.
Let $\mathbf{x}_1 = \mathcal{A}(\emptyset)$.
for $t = 1$ to T **do**
  Generate distribution $\mathcal{D}_t$, sample $\mathbf{y}_t \sim \mathcal{D}_t$ with $\mathbf{E}[\mathbf{y}_t] = \mathbf{x}_t$.
  Play $\mathbf{y}_t$.
  Observe $f_t(\mathbf{y}_t)$, generate $\mathbf{g}_t$ with $\mathbf{E}[\mathbf{g}_t] = \nabla f_t(\mathbf{x}_t)$.
  Let $\mathbf{x}_{t+1} = \mathcal{A}(\mathbf{g}_1, ..., \mathbf{g}_t)$.
end for

How?

Answer: remember finite differences from policy gradient?