Trading off Rewards and Errors in Multi-armed Bandit for Interactive Education

A. Erraqabi (Inria), A. Lazaric (Inria), M. Valko (Inria), E. Brunskill (CMU), Y.-E. Liu (CMU-EnLearn)

Motivation

Serious Games (see Liu et al. (2014))

- *Scientific discovery*: collect as much information as possible about different learning options to accurately estimate their outcome (e.g., difficulty of an exercise).
- *User experience*: provide learning options that allow to move on in the game and learn how to solve the problem (e.g., exercises with increasing difficulty).

Other Examples

- *Medical research studies*: estimate the effectiveness of different treatments and provide more effective treatments at the same time.
- *Crowdsourcing*: estimate quality of different items and encourage users to engage in the test at the same time.
- *A-B testing*: estimate value of different alternatives and maximize the CTR at the same time.

Can we trade off estimation accuracy and rewards at the same time?

Objective Function

The Multi-Armed Bandit Problem

- K arms, each characterized by a distribution ν_i of mean μ_i and variance σ_i^2
- Given an arbitrary sequence of n arms $\mathcal{I}_n = (I_1, I_2, .., I_n)$ with $T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$

$$\text{[average reward]} \quad \rho(\mathcal{I}_n) = \mathbb{E}\Big[\frac{1}{n}\sum_{t=1}^{n} X_{I_t,T_{I_t,t}}\Big] = \frac{1}{n}\sum_{i=1}^{K} T_{i,n}\mu_i$$

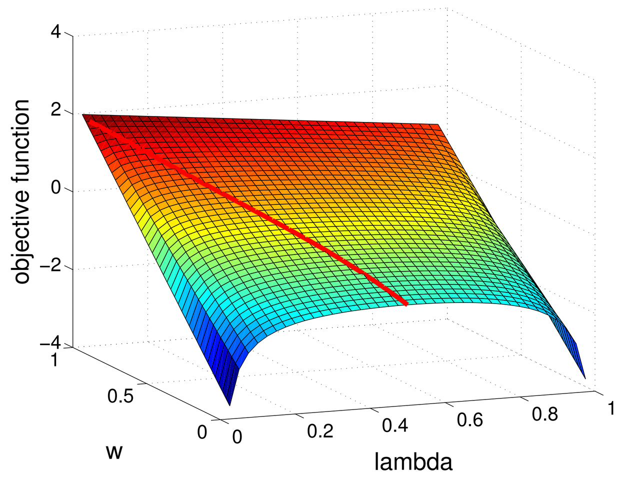
$$\text{[average error]} \quad \varepsilon(\mathcal{I}_n) = \frac{1}{K}\sum_{i=1}^{K}\sqrt{n\mathbb{E}\Big[\big(\widehat{\mu}_{i,n}-\mu_i\big)^2\Big]} = \frac{1}{K}\sum_{i=1}^{K}\sqrt{\frac{n\sigma_i^2}{T_{i,n}}}$$

- How to maximize $\rho(\mathcal{I}_n)$ and how to minimize $\varepsilon(\mathcal{I}_n)$ is the topic of previous literature Auer et al. (2002); Antos et al. (2010); Carpentier et al. (2011).

Trading off Errors and Rewards

- *Continuous relaxation*: $\boldsymbol{\lambda} \in \mathcal{D}_K$, with $\lambda_i = T_{i,n}/n$
- Given a weight parameter $w \in (0,1)$

$$f(\boldsymbol{\lambda};\{\nu_i\}_i) = w\rho(\boldsymbol{\lambda}) - (1-w)\varepsilon(\boldsymbol{\lambda})$$
$$= w\boxed{\sum_{i=1}^{K}\lambda_i\mu_i} - \frac{(1-w)}{K}\boxed{\sum_{i=1}^{K}\frac{\sigma_i}{\sqrt{\lambda_i}}}$$



- Optimal (asymptotic) solution

$$\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}\in\mathcal{D}_K} f_w(\boldsymbol{\lambda};\{\nu_i\}_i) \qquad f^* = f_w(\boldsymbol{\lambda}^*;\{\nu_i\}_i)$$

Properties

- $w = 1$ is average reward maximization, $w = 0$ is estimation error minimization
- w is a Lagrangian multiplier corresponding to a ***constrained optimization problem***
- The two terms are ***homogeneous*** in n and in magnitude unlike in Liu et al. (2014)

Lemma 1. Let $\sigma_{\max} = \max_i \sigma_i$ and $\sigma_{\min} = \min_i \sigma_i > 0$, then $f_w(\boldsymbol{\lambda};\{\nu_i\})$ is α-strongly concave in $\mathcal{D}_K$ with $\alpha = \frac{3(1-w)\sigma_{\min}}{4K}$ and it is β-smooth in $\overline{\mathcal{D}}_K$ with $\beta = \frac{3(1-w)\sigma_{\max}}{4K\lambda_{\min}^{5/2}}$.

The Learning Problem

After n steps, an algorithm $\mathcal{A}$ implemented an allocation $\widetilde{\boldsymbol{\lambda}}_n$ (i.e., $\widetilde{\lambda}_{i,n} = T_{i,n}/n$) with ***regret***

$$R_n(\widetilde{\boldsymbol{\lambda}}_n) = f^* - f_w(\widetilde{\boldsymbol{\lambda}}_n;\{\nu_i\}_i)$$

References

A. Antos, V. Grover, and Cs. Szepesvári. Active learning in heteroscedastic noise. *Theoretical Computer Science*, 411:2712–2728, June 2010.

Y.-E. Liu, T. Mandel, E. Brunskill, and Z. Popovic. Trading off scientific knowledge and user learning with multi-armed bandits. In *Proceedings of EDM*, 2014.

A. Carpentier, A. Lazaric, M. Ghavamzadeh, R. Munos, and P. Auer. Upper-confidence-bound algorithms for active learning in multi-armed bandits. In *Proceedings of ALT'11*, pages 189–203, 2011.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

Confidence-Bound Algorithms

Given estimates $\widehat{\mu}_{i,n}$, $\widehat{\sigma}_{i,n}$ of the mean and standard deviation of each arm.

Upper-confidence bound

$$f_w^{UB}(\boldsymbol{\lambda};\{\widehat{\nu}_{i,n}\}) = w\sum_{i=1}^{K}\lambda_i\Big(\widehat{\mu}_{i,n} + \sqrt{\frac{\log(1/\delta_n)}{2T_{i,n}}}\Big) - (1-w)\sum_{i=1}^{K}\frac{1}{\sqrt{\lambda_i}}\Big(\widehat{\sigma}_{i,n} - \sqrt{\frac{2\log(2/\delta_n)}{T_{i,n}}}\Big)$$

Issues: despite being optimistic in f_w, it ***fails*** for $w \to 0$ since it does not explore arms with poorly estimated low variance.

Lower-confidence bound: similar issues when $w \to 1$ since it does not explore arms with poorly estimated low mean.

Open question: how to design ***no-regret confidence-based*** algorithm for this problem.

The ForcingBalance Algorithm

Input: forcing param η, restricted simplex $\overline{\mathcal{D}}_K(\lambda_{\min})$
for $t = 1, \ldots, n$ **do**
  $U_t = \arg\min T_{i,t}$
  if $T_{U_t,t} < \eta\sqrt{t}$ **then**
    Select arm $I_t = U_t$ (*forcing*)
  else
    Compute optimal estimated allocation
    $\widehat{\boldsymbol{\lambda}}_t = \arg\max_{\boldsymbol{\lambda}\in\overline{\mathcal{D}}_K} f_w(\boldsymbol{\lambda};\{\widehat{\nu}_{i,t}\}_i)$
    Select arm (*tracking*)
    $I_t = \arg\max_{i=1,\ldots,K} \widehat{\lambda}_{i,t} - \widetilde{\lambda}_{i,t}$
  end if
  Pull arm I_t, observe $X_{I_t,t}$, update $\widehat{\nu}_{I_t}$.
end for

Intuition

- *Forcing* $\Rightarrow$ accurate $\widehat{\mu}$ and $\widehat{\sigma}$ and $\widehat{\boldsymbol{\lambda}}$
- *Tracking* $\Rightarrow$ accurate $\widetilde{\boldsymbol{\lambda}}$
- Vanishing forcing $(\sqrt{n}/n) \Rightarrow \widetilde{\boldsymbol{\lambda}} \to \boldsymbol{\lambda}^*$

Forcing parameter η

- Small η: Faster tracking, poorer estimates of $\widehat{\mu}$ and $\widehat{\sigma}$
- Large η: Slower tracking, more accurate estimates of $\widehat{\mu}$ and $\widehat{\sigma}$

Restricted simplex $(\overline{\mathcal{D}}_K, \lambda_{\min})$

- Small $\lambda_{\min}$: consistency, slow convergence
- Large $\lambda_{\min}$: potential bias, faster convergence

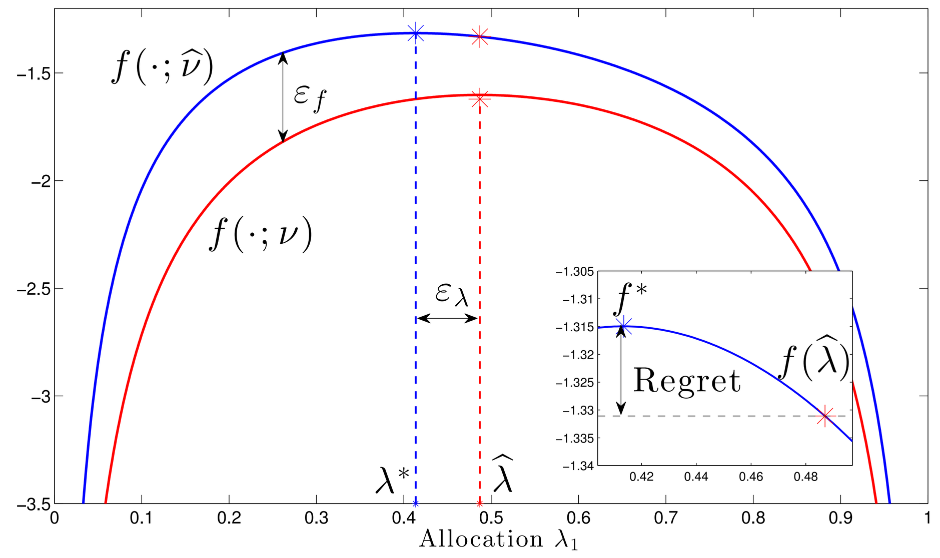
Theoretical Guarantees

Lemma 3. For any allocation $\boldsymbol{\lambda} \in \mathcal{D}_K$ and any arm $i \in [K]$,

$$|\lambda_i - \lambda_i^*| \le \sqrt{\frac{2K}{\alpha}}\sqrt{f^* - f(\boldsymbol{\lambda};\{\nu_i\})}$$

Lemma 4. For any allocation $\boldsymbol{\lambda} \in \overline{\mathcal{D}}_K$

$$f(\boldsymbol{\lambda}^*;\{\nu_i\}) - f(\boldsymbol{\lambda};\{\nu_i\}) \le \frac{3\beta}{2}\|\boldsymbol{\lambda}-\boldsymbol{\lambda}^*\|^2$$



Lemma 2. Let $\widehat{\nu}_i$ be s.t. $|\widehat{\mu}_i - \mu_i| \le \varepsilon_i^\mu$ and $|\widehat{\sigma}_i - \sigma_i| \le \varepsilon_i^\sigma$, then for any $\boldsymbol{\lambda} \in \mathcal{D}_K$

$$\big|f(\boldsymbol{\lambda};\{\nu_i\}) - f(\boldsymbol{\lambda};\{\widehat{\nu}_i\})\big| \le w\max_i \varepsilon_i^\mu + \frac{1-w}{\min_i\sqrt{\lambda_i}}\max_i \varepsilon_i^\sigma$$

Assumption 1. Let $\lambda_{\min}^* = \min_i \lambda_i^*$, we assume that $\lambda_{\min}^* \ge \lambda_{\min}$ (i.e., $\boldsymbol{\lambda}^* \in \overline{\mathcal{D}}_K$).

Theorem. Under Asm. 1, ForcingBalance with a parameter $\eta \le 21$ and a simplex $\overline{\mathcal{D}}_K$ restricted to $\lambda_{\min}$ suffers a regret

$$R_n(\widetilde{\boldsymbol{\lambda}}) \le \begin{cases} 1 & \text{if } n \le n_0 \\ 43K^{5/2}\dfrac{\beta}{\alpha}\sqrt{\dfrac{\log(2/\delta_n)}{\eta\lambda_{\min}}}\,n^{-1/4} & \text{if } n_0 < n \le n_2 \\ 153K^{5/2}\dfrac{\beta}{\alpha}\sqrt{\dfrac{\log(2/\delta_n)}{\lambda_{\min}\lambda_{\min}^*}}\,n^{-1/2} & \text{if } n > n_2, \end{cases}$$

w.p. $1-\delta$ and $n_0 = K(K\eta^2 + \eta\sqrt{K} + 1)$ and $n_2 = \frac{C}{(\lambda_{\min}^*)^8}\frac{K^{10}}{\alpha^4}\frac{\log^2(1/\delta_n)}{\lambda_{\min}^2}$.

Remarks

- *Dep. on n*: multiple phases and asymptotic performance $O(n^{-1/2})$, which illustrates the fact that ForcingBalance converges to the performance of the optimal allocation.
- *Dep. on $\lambda_{\min}$*: for $\lambda_{\min} = 0$, Asm.1 is always satisfied. It can be replaced by $\lambda_{\min}^*$ as n grows.
- *Dep. on $\lambda_{\min}^*$*: as the allocation over arms becomes more "extreme" the higher the regret.

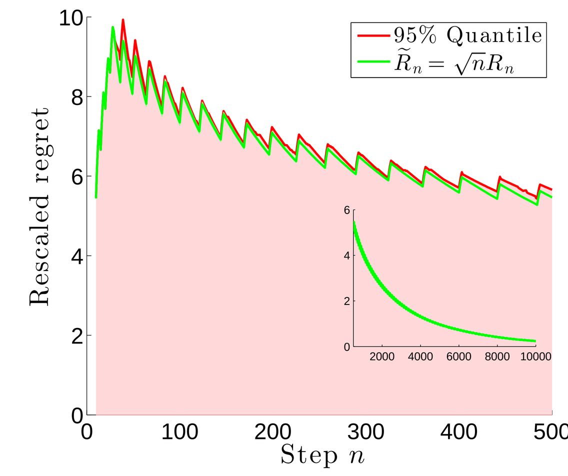
Synthetic Experiments

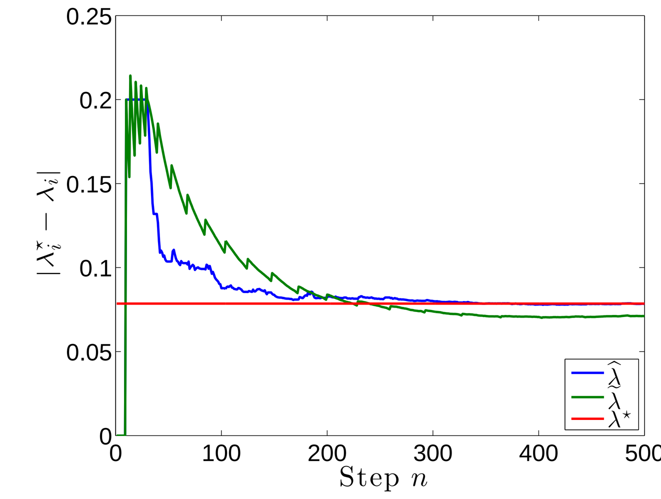
The setting.

- $K = 5$ arms, $w = 0.9$ (i.e., favor rewards over errors).
- Parameters $\eta = 1$, $\lambda_{\min} = 0$.
- Arm 4 has the largest variance and it should be pulled the most to minimize ε.
- Arm 5 has the largest reward and it should be pulled the most to maximize ρ.
- The optimal allocation $\boldsymbol{\lambda}^*$ is very unbalanced towards *arm5* and a bit on *arm4*.

	μ	σ^2	$\boldsymbol{\lambda}^*$
Arm1	1.0	0.05	0.0073
Arm2	1.5	0.1	0.01
Arm3	2.0	0.2	0.014
Arm4	4.0	**4.0**	0.0794
Arm5	**5.0**	0.5	0.8893

The results.





Rescaled regret

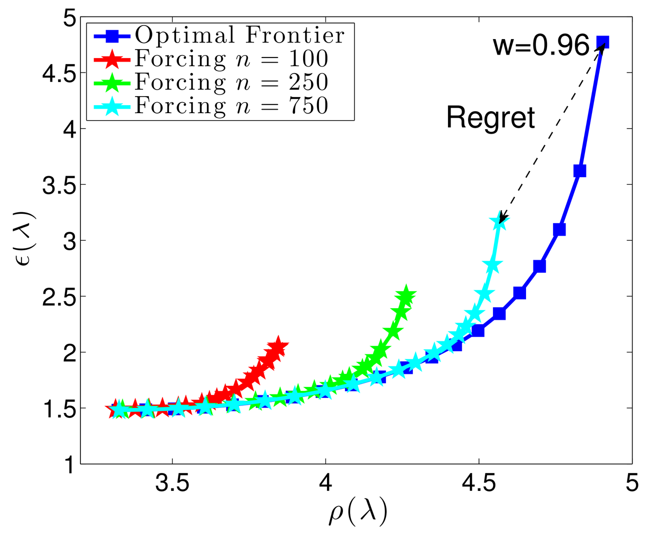
- In the first phase driven by forcing, the rescaled regret increases.
- Later the rescaled regret starts decreasing.
- Difficult to asses whether it stabilizes or it keeps decreasing (i.e., true regret $O(1/n)$?)

Tracking

- The estimated optimal allocation $\widehat{\boldsymbol{\lambda}}$ converges fast
- The empirical frequency $\widetilde{\boldsymbol{\lambda}}$ effectively tracks the estimated optimal allocation

Pareto Frontier

- Varying w from 0.01 to 0.96.
- For $w = 0$, the minimization of ε induces an optimal allocation with $\lambda_4^* = 0.41$ and $\lambda_5^* = 0.20$.
- For $w = 0.95$, the maximization of ρ induces an optimal allocation with $\lambda_4^* = 0.0484$ and $\lambda_5^* = 0.9326$.
- ForcingBalance is more effective in approaching the performance of $\boldsymbol{\lambda}^*$ for small values of w. In fact, for $w = 0$, $\lambda_{\min}^* = 0.097$, while for $w = 0.95$, $\lambda_{\min}^* = 0.004$.



Educational Experiment

The setting.

- $K = 64$ arms (2 representations of the fraction, 2 representations of the label fractions, tick marks on/off, hinting animations on/off, 4 different rates of backoff hints)
- Means and variances determined from real interaction data
- Let π^* be the true ranking and $\widehat{\pi}$ the estimated ranking

$$\text{DCG}_\pi = \sum_{k=1}^{K}\frac{\mu_{\pi(k)}}{\log(k+1)}; \quad \text{RelDCG} = \frac{\text{DCG}_{\pi^*} - \text{DCG}_{\widehat{\pi}}}{\text{DCG}_{\pi^*}}; \quad RankErr = \frac{1}{K}\sum_{i=1}^{K}|\pi^*(i) - \widehat{\pi}(i)|$$

The results.

- UCB maximizes reward ρ, GAFS minimizes errors ε, but Force is the most effective in minimizing the regret and trading off rewards and accuracy of the estimates.
- For $w = 0.95$ ForcingBalance achieves a much higher reward than GAFS without compromising the accuracy (in terms of *RelDCG* and *RankErr*).
- For $w = 0.6$ ForcingBalance still achieves the best reward among explorative algorithm but is now even more accurate in ranking performance.

Alg.	$\frac{\varepsilon(\boldsymbol{\lambda})}{\sigma^2_{\max}}$	$\frac{\rho(\boldsymbol{\lambda})}{\mu_{\max}}$	R_n	*RelDCG*	*RankErr*
$w = 0.95$					
$\boldsymbol{\lambda}^*$	6.549	0.9405	-	-	-
Force	6.708	0.9424	**1.878**	0.1871	5.935
UCB	11.03	**0.9712**	95.15	1.119	8.629
GAFS	**5.859**	0.9183	17.79	**0.1268**	**5.117**
Unif	5.861	0.9168	20.49	0.132	5.25
$w = 0.6$					
$\boldsymbol{\lambda}^*$	5.857	0.9189	-	-	-
Force	**5.859**	0.92	**0.4437**	**0.1227**	5.178
UCB	11.03	**0.9712**	1343	1.119	8.629
GAFS	**5.859**	0.9183	1.314	0.1268	**5.117**
Unif	5.861	0.9168	3.482	0.132	5.25