In this simulation study, I compared the performance of two bandit algorithms.

Algorithm: Risk-aware Bandit Algorithm

Input parameters: q, h, τ

Initialization:
For all $i \in [K]$, set

$$\mathcal{T}_{i,0} = \mathcal{S}_{i,0} = \emptyset, \quad \hat{\beta}_{\mathcal{T}_{i,0}} = \hat{\beta}_{\mathcal{S}_{i,0}} = 0 \in \mathbb{R}^d$$

For $t \in [T]$:

1. Observe $X_t \sim \mathcal{P}_{\mathcal{X}}$.
2. **If** $t \in \mathcal{T}_i$:
   - Set $\pi_t \leftarrow i$ (forced-sampling).

   Else:
   - Compute suboptimal arm set:

   $$\hat{\mathcal{K}}_{\text{sub}} \leftarrow \left\{ i \in [K] \mid X_t^\top \hat{\beta}_{\mathcal{T}_{i,t-1}} \geq \max_{j \in [K]} X_t^\top \hat{\beta}_{\mathcal{T}_{j,t-1}} - \frac{h}{2} \right\}$$

   - Select arm:

   $$\pi_t \leftarrow \arg\max_{i \in \hat{\mathcal{K}}_{\text{sub}}} X_t^\top \hat{\beta}_{\mathcal{S}_{i,t-1}}$$

3. Update the all-sample set:

$$\mathcal{S}_{i,t} \leftarrow \mathcal{S}_{i,t-1} \cup \{t\}$$

4. Observe reward:

$$Y_t = X_t^\top \beta_{\pi_t} + \epsilon_{\pi_t,t}$$

5. **If** $t \in \mathcal{T}_i$:
   - Update using forced-sample set:

   $$\hat{\beta}_{\mathcal{T}_{i,t}} \leftarrow \arg\min_{\beta \in \mathbb{R}^d} \frac{1}{|\mathcal{T}_{i,t}|} \sum_{r \in \mathcal{T}_{i,t}} \rho_\tau (Y_r - X_r^\top \beta)$$

Else:

- Update using all-sample set:

$$\hat{\beta}_{\mathcal{S}_{i,t}} \leftarrow \arg\min_{\beta \in \mathbb{R}^d} \frac{1}{|\mathcal{S}_{i,t}|} \sum_{r \in \mathcal{S}_{i,t}} \rho_\tau(Y_r - X_r^\top \beta)$$

OLS bandit uses OLS update in the beta updating stage.

Why these DGPs? Why these sample sizes/conditions?

Truncated normal contextual vector : Following the theoretical part of Online Decision Making with High-Dimensional Covariates, we require the l-2 norm of contextual vector is bounded.

Two different uniformly distributed beta : With assumption 2, we expect the arms are "distinguishable" with high probability. Uniform distribution can be easily generated with some overlapping.

Heavy-tailed error : We expect the quantile update have robustness property compared with OLS update.

Sample size : Time frame T may grow large, but up to some point, the proportion of forced-sampling exploration becomes too small compared with the total number of samples.

How did you ensure your simulation design was fair and unbiased?

I tried several different degrees of freedom for the error student t-distribution to avoid cherry-picking.

What are the limitations of your simulation study?

I did not explore the impact of different β distribution and also the high-dimensional cases.

What scenarios did you not include, and why might they matter?

When the target focuses on different quantile level, the results may differ a lot. However, I do not have a good benchmark algorith to compare with. It can be applied for conservative decision-making.

How do your results inform practice or theory?

It aligns with the theory that a quantile estimator in this case provides a more robust and more accurate β estimation. But this can be inferred from the quantile regression property.

What would you investigate next if you had more time/resources?

I may want to work on high-dimensional setting simulation, as the main contribution of the paper was based on high-dimensional contextual data.

Which aspects of the implementation were most challenging?

Modularize my code is definitely a pain. When I write them in functions, I get confused about which one should be called first. Also, the structure must be designed more carefully compared with working directly with interface notebook. Making the results reproducible -- adding a random seed in each part and creating a makefile are quite challenging.

How confident are you in your results? What could undermine that confidence?

I feel quite confident in my simulation results.