Stochastic Linear Bandits An Empirical Study

Students: Alexis Marouani, Grégoire Béchade,
Lecturer: Claire Vernade, Contact me on Slack if anything looks weird, or find my email on my website

1 Problem 1: Linear epsilon greedy

1. q1
2. q2
3. According to the documentation of numpy, the complexity of the pinv function is $O(min(nm^2, n^2m))$. In our problem, the matrix is squared, of size d so the complexity is $O(d^3)$. This can create problems when facing high-dimensional problems. We have therefore decided to implement a class LinearEpsilonGreedybis, in which we have changed the estimation of $\hat{theta}$. Instead of estimating θ through the least square estimator, we decided to estimate it through this estimator: $\hat{\theta} = \sum_{t=1}^{T} \langle \theta, A_t \rangle A_t$. We didn't manage to find theoretical guarantees about the expected value of this estimator, as $\mathbb{E}(\hat{\theta}) = \sum_{t=1}^{T} \mathbb{E}(\langle \theta, A_t \rangle A_t)$, which can't be precised without assumptions on the distribution of A_t. However, we have tested it on different problems, and it seems to obtain the same results as the one obtained with the least square estimator. Computing $\hat{\theta}$ has a complexity in $0(d)$, as we only have to compute scalars products of d-vectors. The figure 1 underlines the gain in computational time, while the performances are the same.

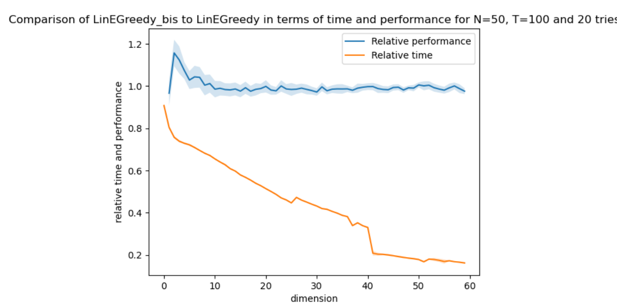

Figure 1: Comparison of the performances and rime of execution of LinearEpsilonGreedy and the LinearEpsilonGreedy bis, with N=50, T=200 and 20 tries.

2 Problem 2: LinUCB and LinTS

1. q1
2. q2
3. q3