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. Here is the code of our receive_reward function with the updates of the covariance matrix. The action is then chosen as the maximizer of the inner product between the estimated $\hat{\theta}$ and the arms.

```
    def receive_reward(self, chosen_arm, reward):
        """
        update the internal quantities required to estimate the
           parameter theta using least squares
        """
        # update inverse covariance matrix
        self.cov += np.outer(chosen_arm, chosen_arm) # update the
           covariance matrix
        self.invcov = pinv(self.cov) # update the inverse
           covariance matrix

        # update b_t
        self.b_t += reward * chosen_arm

        self.hat_theta = np.inner(self.invcov, self.b_t) # update
           the least square estimate
        self.t += 1
```

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.

2 Problem 2: LinUCB and LinTS

1. For the implementation of LinUCB, we have implemented the function $\beta(t, \delta)$ and directly computed the upper confidence born from the course. The arm selected is the one that max-

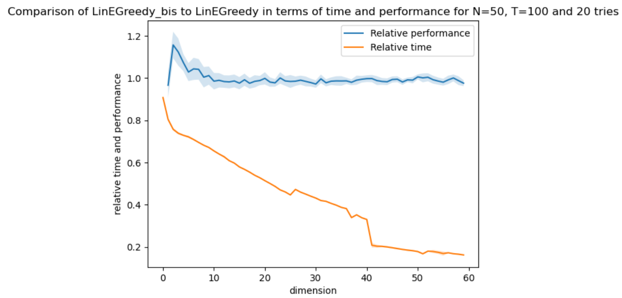

Figure 1: Comparison of the performances and time of execution of LinearEpsilonGreedy and the LinearEpsilonGreedy bis, with N=50, T=200 and 20 tries.

imises the following quantity:

2. $x^T \hat{\theta}_t^\lambda + \|x\|_{(B_t^\lambda)^{-1}} \beta(t, \delta)$.

3. The prior on θ^* is that it follows the law $\mathcal{N}(0, \Sigma^{-1})$.

   Let us define :
   $\Sigma_t^{-1} = \Sigma^{-1} + \frac{\sum_{t=1}^T A_t^T A_t}{\sigma^2}$

   and $\mu_t = \Sigma_t \times (\frac{\sum_{t=1}^T A_t^T r_t}{\sigma^2})$
   The posterior on θ^* is $\mathcal{N}(\mu_t, \Sigma_t)$

4. q3

3 Appendix

Proof of the posterior distribution of θ^* in LinTS

Let us note A_t, the chosen action at time t and $Y_t =< A_t, \theta^* > +\epsilon_t$ the reward.

We directly have that, $Y_t \sim \mathcal{N}(A_t(\theta^*)^T, \sigma^2)$.

Thanks to Bayes rule, we have :

$\mathbb{P}(\theta^*|Y_1, ..., Y_t, A_1, ..., A_t) = \mathbb{P}(Y_1, ..., Y_t, A_1, ..., A_t|\theta^*) \times \frac{\mathbb{P}(\theta^*)}{\mathbb{P}(Y_1, ..., Y_t, A_1, ..., A_t)}$
The denominator is a constant with respect to θ^* so :
$\mathbb{P}(\theta^*|Y_1, ..., Y_t, A_1, ..., A_t) \propto \mathbb{P}(Y_1, ..., Y_t, A_1, ..., A_t|\theta^*) \times \mathbb{P}(\theta^*)$
But, $\theta^* \sim \mathcal{N}(0, \Sigma)$ and
$\mathbb{P}(Y_1, ..., Y_t, A_1, ..., A_t|\theta^*) = \Pi_{t=1}^T \frac{1}{\sqrt{2\pi}} exp(\frac{-1}{2\sigma^2}(A_t \theta^{*T} - r_t)^2) = (\frac{1}{\sqrt{2\pi}})^T \times exp(\sum_{t=1}^T \frac{-1}{2\sigma^2}(A_t \theta^{*T} - r_t)^2)$
θ^* and $(Y_1, ..., Y_t, A_1, ..., A_t|\theta^*)$ follow normal distributions so $\theta^*|Y_1, ..., Y_t, A_1, ..., A_t$ also follows a normal distribution. We finally just have to identify the mean (μ_t) and the covariance matrix (Σ_t) of this distribution.

$$
\begin{aligned}
log(\mathbb{P}((\theta^*|Y_1, ..., Y_t, A_1, ..., A_t))) &= C + \sum_{t=1}^{T} \frac{-1}{2\sigma^2}(A_t\theta^{*T} - r_t)^2 - \frac{1}{2}(\theta^{*T}\Sigma^{-1}\theta^*) && (1)\\
&= \frac{-1}{2}((\sum_{t=1}^{T} \frac{(A_t^T\theta^{*T}\theta^* A_t - 2r_t A_t\theta^{*T} + r_t^2)}{\sigma^2}) + \theta^{*T}\Sigma^{-1}\theta^*) + C && (2)\\
&= C + \theta^{*T}(\frac{1}{\sigma^2}\sum_{t=1}^{T} A_t^T A_t + \Sigma^{-1})\theta^* - 2\theta^{*T}(\frac{\sum_{t=1}^{T} A_t r_t}{\sigma^2}) - (\frac{r_t}{\sigma})^2 && (3)\\
& && (4)
\end{aligned}
$$

We finally identify Σ_t thanks to the quadratic term in θ^* :

$\Sigma_t^{-1} = \Sigma^{-1} + \frac{\sum_{t=1}^{T} A_t^T A_t}{\sigma^2}$

and we have directly $\mu_t = \Sigma_t \times (\frac{\sum_{t=1}^{T} A_t^T r_t}{\sigma^2})$.