Multi-Agent Systems: Homework Assignment 4 – Group 13

J.J. Haas F. Huang F.D. Brunet de Rochebrune A. Vozikis

2023-12-06

4 Fictitious Play

A mixed Nash equilibrium for the game has been found by running the fictional play method. Following a series of iterations and strategy adjustments depending on the opponent's past actions, the algorithm eventually arrived at the following mixed strategies:

- Player A's mixed strategy: [0,0.459,0.541], meaning player A should play strategy B with a probability of 45.9% and strategy C with a probability of 54.1
- Player B's mixed strategy: [0,0.691,0.309,0], meaning player B should play strategy Y with a probability of 69.1% and strategy Z with a probability of 30.9

This means that, with a chance of 69.1%, player B should play strategy Y, and with a probability of 30.9%, player Z. The mixed Nash equilibrium of the game, as determined by the fake play algorithm, is represented by these tactics. It's crucial to remember that these odds are the outcome of repeatedly simulating the game and, in order to provide stability and convergence, average the tactics throughout the last 10% of iterations.

5 Monte Carlo Simulation

Question 1

The estimated mean value of $E(\cos^2(X))$, where X is a standard normal variable $X\ N(0, 1)$, is approximately 0.5674. The uncertainty on this result, quantified as the standard error of the mean, is approximately 0.00035.

This means that the average value of $\cos^2(X)$ over a very large number of samples drawn from a standard normal distribution is close to 0.5674, with a very small margin of error, indicating a high level of precision in the estimate.

Question 2

To determine whether the observed correlation of 0.3 between the hyperparameter A and the score S is significant, we can use Monte Carlo (MC) simulation to compute a simulated p-value. The p-value will tell us the probability of observing a correlation as extreme as 0.3 (or more extreme) under the null hypothesis that there is no actual correlation between A and S.

The computed p-value from the Monte Carlo simulation is approximately 0.39968. Since the p-value is greater than the common significance thresholnd of 0.05. it suggests that the observed correlation of 0.3 could very well be due to chance.

Question 3

$$f(x) = \mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$
$$g(x) = \mathcal{N}(x|\nu, \tau^2) = \frac{1}{(2\pi\tau^2)^{1/2}} \exp\left\{-\frac{1}{2\tau^2}(x-\nu)^2\right\}$$

$$\begin{aligned}
\log\frac{f(x)}{g(x)} &= \log\frac{\frac{1}{(2\pi\sigma^2)^{1/2}}\exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}}{\frac{1}{(2\pi\tau^2)^{1/2}}\exp\left\{-\frac{1}{2\tau^2}(x-\nu)^2\right\}} \\
&= \log\frac{\tau\exp-\frac{(x-\mu)^2}{2\sigma^2}}{\sigma\exp-\frac{(x-\nu)^2}{2\tau^2}} \\
&= \log\frac{\tau}{\sigma} - \frac{(x-\mu)^2}{2\sigma^2} + \frac{(x-\nu)^2}{2\tau^2} \\
&= \left(\log\frac{\tau}{\sigma} - \frac{\mu^2}{2\sigma^2} + \frac{\nu^2}{2\tau^2}\right) + \left(\frac{2\mu}{2\sigma^2} - \frac{2\nu}{2\tau^2}\right)x + \left(-\frac{1}{2\sigma^2} + \frac{1}{2\tau^2}\right)x^2
\end{aligned}$$

$$\begin{aligned}
KF(f \parallel g) &= \int_{-\infty}^{\infty} f(x)\log\frac{f(x)}{g(x)} \\
&= \int_{-\infty}^{\infty}\left\{\left(\log\frac{\tau}{\sigma} - \frac{\mu^2}{2\sigma^2} + \frac{\nu^2}{2\tau^2}\right)f(x) + \left(\frac{2\mu}{2\sigma^2} - \frac{2\nu}{2\tau^2}\right)xf(x) + \left(-\frac{1}{2\sigma^2} + \frac{1}{2\tau^2}\right)x^2 f(x)\right\}
\end{aligned}$$

Given that

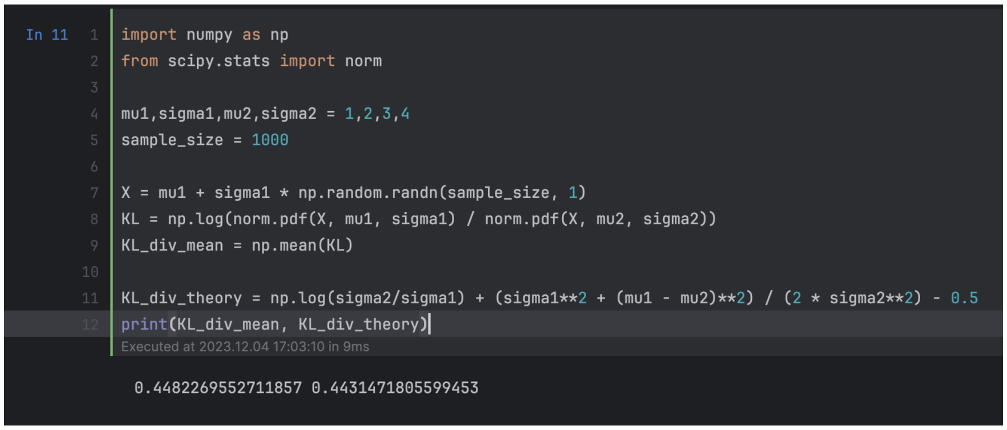
$$\begin{aligned}
&\int_{-\infty}^{\infty} f(x) = 1, \\
&\int_{-\infty}^{\infty} xf(x) = \mu, \\
&\int_{-\infty}^{\infty} x^2 f(x) = \mu^2 + \sigma^2
\end{aligned}$$

$$\begin{aligned}
KF(f \parallel g) &= \left(\log\frac{\tau}{\sigma} - \frac{\mu^2}{2\sigma^2} + \frac{\nu^2}{2\tau^2}\right) + \left(\frac{2\mu}{2\sigma^2} - \frac{2\nu}{2\tau^2}\right)\mu + \left(-\frac{1}{2\sigma^2} + \frac{1}{2\tau^2}\right)(\mu^2+\sigma^2)) \\
&= \log\frac{\tau}{\sigma} + \frac{-\mu^2 + 2\mu^2 - \mu^2 - \sigma^2}{2\sigma^2} + \frac{\nu^2 - 2\mu\nu + \mu^2 + \sigma^2}{2\tau^2} \\
&= log(\frac{\tau}{\sigma}) - \frac{1}{2} + \frac{(\mu-\nu)^2 + \sigma^2}{2\tau^2}
\end{aligned}$$

From the equation above, we can see that f and g play asymmetric roles, indicating that KL is a divergence rather than a distance.

Question 4

We picked a sample size as 1000, and set μ, σ, ν, τ as 1,2,3,4 respectively. Run the following python code:



```
In 11
import numpy as np
from scipy.stats import norm

mu1,sigma1,mu2,sigma2 = 1,2,3,4
sample_size = 1000

X = mu1 + sigma1 * np.random.randn(sample_size, 1)
KL = np.log(norm.pdf(X, mu1, sigma1) / norm.pdf(X, mu2, sigma2))
KL_div_mean = np.mean(KL)

KL_div_theory = np.log(sigma2/sigma1) + (sigma1**2 + (mu1 - mu2)**2) / (2 * sigma2**2) - 0.5
print(KL_div_mean, KL_div_theory)
Executed at 2023.12.04 17:03:10 in 9ms

  0.4482269552711857 0.4431471805599453
```

Figure 1: Monte Carlo Simulations in Python and Results

And the Monte Carlo simulated result is 0.4482, while the theoretical result should be 0.4431. We can see that the results of the two are quite similar.

6 Exploitation versus Exploration

UCB versus ϵ-greedy for k-bandit problem

Question 1

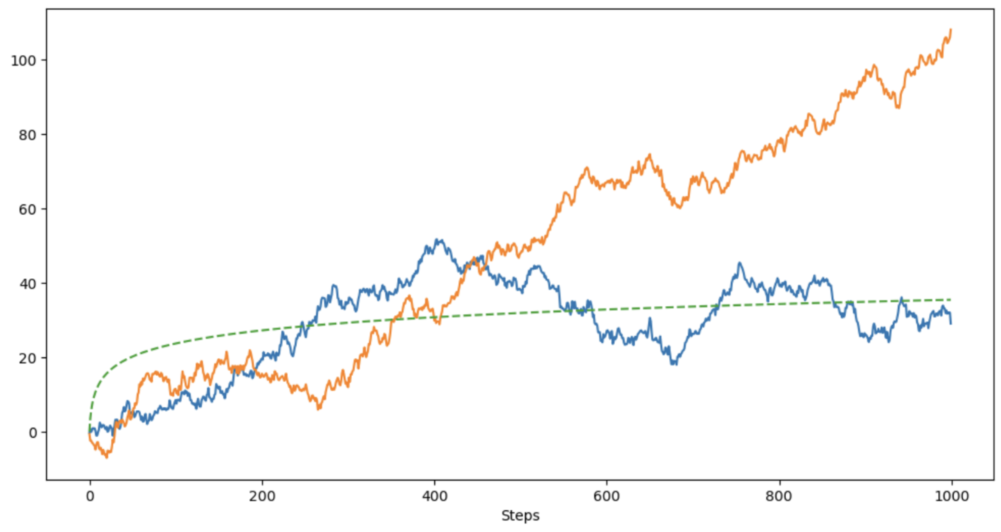
For k=2 we have created two Gaussians with:

- Arm 1: Mean = 0.2775
- Arm 2: Mean = 0.4723

Calculating the Δ values for each arm:

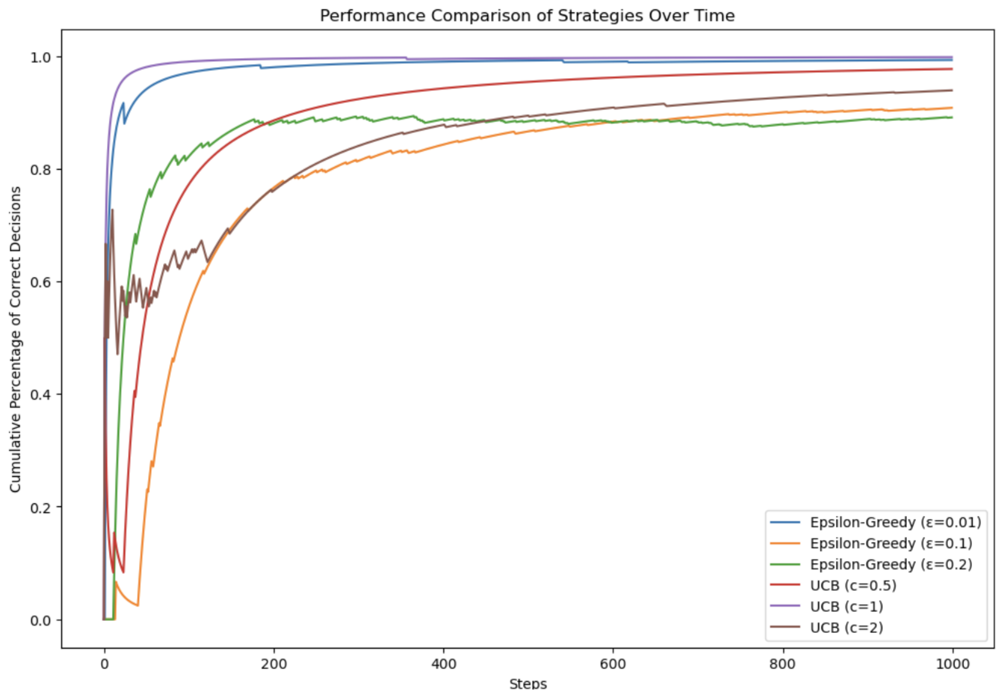
- Arm 1: Δ = 0.1948
- Arm 2: Δ = 0.0000

UCB value we used was 2 and ϵ-greedy value was set to 0.1 for our experiment. And the Lai-Robbins lower bound for cumulative regret after 1000 steps: 35.4581. The plot looks like:



Question 2

We tried different values for ϵ-greedy (0.001,0.1 and 0.2). Also doe UCB (0.5,0.1,0.2). The plot is shown:



Apparently, for UCB setting c = 1 gave us the best value, with second best c=0.5. Unfortunately 2 was lagging compared to the other 2.

Appendix

Chapter 5, Question 1

```python
import numpy as np

n = 1_000_000

samples = np.random.normal(0, 1, n)

cos2_samples = np.cos(samples)**2

mean_estimate = np.mean(cos2_samples)

sem = np.std(cos2_samples) / np.sqrt(n)

mean_estimate, sem
```

Chapter 5, Question 3

```python
import numpy as np
from scipy.stats import pearsonr

num_simulations = 1_000_000

observed_correlation = 0.3

A_values = np.random.rand(num_simulations)
S_values = np.random.rand(num_simulations)

# compute correlation for each pair
simulated_correlations = [pearsonr(A_values[i:i+10], S_values[i:i+10])[0] for i in range(0,
    num_simulations, 10)]

# compute p-value: proportion of correlations as extreme as or more extreme than observed
p_value = np.mean([abs(corr) >= observed_correlation for corr in simulated_correlations])

p_value
```