Multi-Agent Systems

Homework Assignment 4

MSc AI, VU

E.J. Pauwels

Version: November 26, 2021 — Deadline: Friday, December 3, 2021 (23h59)

4 Monte Carlo simulation

4.1 MC sampling

Recall that Monte Carlo sampling allows us to estimate the expectation of a random function by sampling from the corresponding probability distribution. More precisely, if $f(x)$ is a 1-dim (continuous) probability density, and $X \sim f$ is a stochastic variable distributed according to this density f, then the expected value of some function φ can be estimated using Monte Carlo sampling by:

$$E_f(\varphi(X)) \equiv \int \varphi(x)f(x)\,dx \approx \frac{1}{n}\sum_{i=1}^{n}\varphi(X_i) \qquad \text{for sample of independent } X_1, X_2, \ldots, X_n \sim f.$$

1. Assume that $X \sim N(0,1)$ is standard normal. Estimate the mean value $E(\cos^2(X))$. Quantify the uncertainty on your result.

2. Suppose you're designing a deep neural network that needs to maximize some score function S. The actual design of the network depends on some hyperparameter A. Training the networks is computationally very demanding and time consuming, and as a consequence you have only been able to perform ten experiments to date. Based on these ten data points you observe a slight positive correlation of 0.3 between the value of the hyperparameter A and the score S. If this result is genuine, it suggest to increase A in the next experiment in order to improve the score. But if the correlation is not significant, increasing A could lead you astray. How would you use MC to decide whether the correlation is significant?
Hint: Compute the empirical p-value of the observed result, under the assumption of independence.

4.2 Importance Sampling

Importance sampling extends the basic MC approach to cases where it is difficult to sample from f but (relatively) easy to sample from a (somewhat) similar distribution g. More precisely:

$$
\begin{aligned}
E_f(\varphi(X)) &= \int \varphi(x) f(x)\, dx \\
&= \int \varphi(x) \frac{f(x)}{g(x)}\, g(x)\, dx \equiv E_g \left[\varphi(X) \frac{f(X)}{g(X)} \right] \\
&\approx \frac{1}{n} \sum_{i=1}^{n} \varphi(X_i) \frac{f(X_i)}{g(X_i)} \qquad \text{for sample of independent } X_1, \ldots, X_n \sim g.
\end{aligned}
$$

1. Let $X \sim N(0, 1)$ be a standard normal stochastic variable. Use importance sampling to estimate $E(X^2)$ by sampling from a uniform distribution $q \sim U(-5, 5)$ on the interval $[-5, 5]$. What value do you expect (based on your knowledge of the normal distribution)? How accurate is your estimate based on importance sampling?

2. Suppose some random process produces output ($-1 \leq X \leq 1$) that is distributed according to the following continuous density:

$$
f(x) = \frac{1 + \cos(\pi x)}{2} \qquad (\text{for } -1 \leq x \leq 1).
$$

   Again we are interested in estimation $E(X^2)$. However, as this is not a standard distribution it makes sense to use importance sampling to estimate this value. Quantify the uncertainty on your result.

4.3 Kullback-Leibler divergence

The Kullback-Leibler (KL) divergence quantifies the similarity (or dissimilarity) of two probability densities. More specifically, given two continuous (1-dim) probability densities f, g, the KL-divergence is defined as:

$$
KL(f||g) = \int_{-\infty}^{\infty} f(x) \log\left(\frac{f(x)}{g(x)}\right) dx \quad \equiv \quad E_f \left[\log\left(\frac{f(X)}{g(X)}\right) \right] \tag{1}
$$

1. Let $f \sim N(\mu, \sigma^2)$ and $g \sim N(\nu, \tau^2)$ both be normal distributions. Express $KL(f||g)$ as a function of the means and variances of f and g. We mention in passing that the KL expression in eq.1 is called a **divergence** rather than a **distance** because it's not symmetric. Use the expression obtained above to convince yourself of this fact.

2. Check your theoretical result in (1) by computing a sample-based estimate of the KL-divergence (Monte Carlo simulation). Pick an appropriate sample size. Compare the MC estimate to the theoretical result.

5 Exploitation versus Exploration

5.1 UCB versus ϵ-greedy for k-bandit problem

Write a programme to experiment with the exploration/exploitation for the k-bandit problem (pick some value $5 \leq k \leq 20$). Assume that the arms (a) generate normally distributed rewards with unit standard deviation, but different means $q(a)$ (e.g. randomly generated). Assume that in every single experiment the agent can take a total of $T = 1000$ actions (i.e. arm pulls). Let $L(t)$ be the expected total regret at time t, defined as:

$$\ell(t) = E\left(\sum_{i=1}^{t}(q^* - q(a_i))\right)$$

- Compute the experimental $L(t)$ curves for different strategies (ϵ-greedy for different values of ϵ, UCB). Compare to the theoretical lower bound found by Lai-Robbins:

$$\ell(t) \geq A\log(t) \qquad \text{where} \qquad A = \sum_{a:\Delta_a \neq 0} \frac{\Delta_a}{KL(f_a||f_a^*)} \quad \text{and} \quad \Delta_a = q^* - q(a).$$

- Compute and compare the percentage correct decisions (selection of correct arm) under the different strategies (i.e. ϵ-greedy for different values of ϵ, UCB). What is the influence of the UCB hyper-parameter c?

PS: *No need to submit code, only the results.*

SOLUTIONS

Preliminary remark: How to compute uncertainty on MC estimate?

Suppose we want to use Monte Carlo (MC) to estimate the expectation $E(\phi(X))$ where X has a known density function $f(x)$ and ϕ is a known function. To compute the MC estimate we draw a large sample $X_1, X_2, \ldots, X_n$ from the distribution f and compute the corresponding function values $\varphi_i = \phi(X_i)$ for $i = 1, \ldots, n$. The mean of these function values is the MC estimate for the expectation:

$$E\phi(X) \approx \frac{1}{n}(\varphi_1 + \varphi_2 + \ldots + \varphi_n).$$

To compute the uncertainty on the estimate in the RHS we notice that this RHS is a sample mean $\overline{\varphi}$ of the φ_i observations which means that variance is equal to the variance of the φ_i observations divided by sample size:

$$Var(\overline{\varphi}) = \frac{Var(\varphi_i)}{n}$$

or equivalently (in terms of the standard deviation):

$$std(\overline{\varphi}) = \frac{std(\varphi_i)}{\sqrt{n}} \qquad \text{a.k.a. } \textit{standard error (s.e.)} \text{ on MC estimate.}$$

The variance (or standard deviation) in the RHS can be estimated by computing the variance (or standard deviation) of the sample $\varphi_1, \varphi_2, \ldots, \varphi_n$. It's standard practice to use two or three times the standard error as a measure for the uncertainty on the MC estimate.

4.1 Mont Carlo Sampling

Estimate $E(\cos^2(X))$ for $X \sim N(0,1)$: Matlab code

```
sample_size =  10000;    %  sample size for MC estimate

X = randn(sample_size,1);  %  random sample from N(0,1) population
F = cos(X).^2;     %  compute function value at each sample point

%  The MC estimate for m = E((cos(X))^2) is obtained by computing the
%  sample average:

m_mc = mean(F);

%  Since each sample point in F is and independent sample from cos(X).^2
%  (where X ~ N(0,1), the standard deviation  std(F) is an estimate of the
%  corresponding population standard deviation.  The corresponding standard
%  deviation for the sample mean is therefore equal to std(F)/sqrt(sample_size)

m_mc_std = std(F)/sqrt(sample_size);
```

Conclusion Based on the above matlab code we conclude that $E(\cos^2(X)) \approx 0.5665$ (based on 10000 MC samples). Since the standard error (standard deviation for sample mean) equals 0.003 (approximately), we estimate the accuracy on the result as $3 \times 0.003 \approx 0.01$. Hence we conclude:

$$E(\cos^2(X)) \approx 0.57 \pm 0.01.$$

Correlation between score and hyperparameter

- Assume that there is no correlation;
- Use this assumption to draw random samples (of size 10) from this distribution and compute the correlation coefficient.
- Compare the observed result $r_{obs} = 0.3$ to the correlations for the simulated samples. Compute how "extreme" the observed result is (i.e. compute its p-value). If the p-value is small (e.g. $p < 0.05$) the observed trend is likely to be genuine.
- BONUS: Since we have no guarantee that the data points are distributed according to a normal distribution, one could try some additional simulations in which one uses other likely distributions, to investigate how this impacts on the conclusion. However, for small samples most histograms would be compatible with the normal distribution.

```
MATLAB code:

%  We have 10 data points for which the observed correlation equals r_obs = 0.3.

r_obs = 0.3;
n = 10;   % number of experimental data points

%  Assume that there is no correlation between the two parameters, then the
%  observed correlation is a random fluctuation.  To test how likely this
%  size of fluctation is, we generate independent variables and tally how
%  often a correlation of r_obs (or larger) is observed.

nr_samples = 1000;
Rho_MC = zeros(nr_samples,1);

for i = 1:nr_samples
    %  Generated randomly distributed but independent samples for S and A
    S = randn(n,1);
    A = randn(n,1);
    % Compute and store the observed correlation coef for each sample
    Rho  = corrcoef(A,S);   % full correlaton matrix
    Rho_mc(i) = Rho(1,2) ;   % correlation btw. variables 1 and 2 in corr matrix

end

%  Compute the p-value of the observed value
```

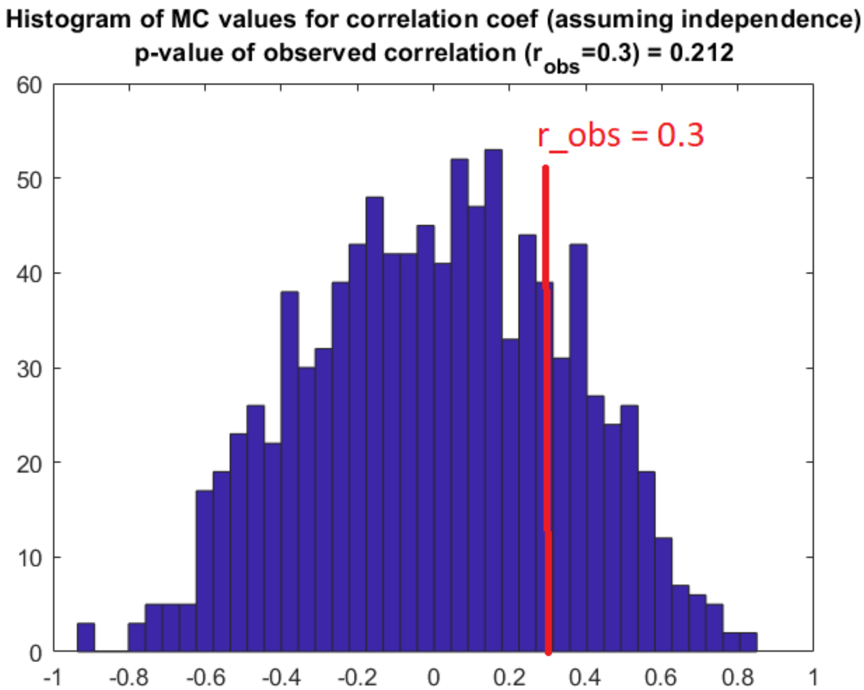

Figure 1: Mistogram of MC-based correlation values

```
pval = length(find(Rho_mc > r_obs))/nr_samples;
```

Conclusion The MC simulation assumes that there is no correlation between the hyperparameter and the score. In that case it turns out that approximately 20% of the simulated correlation coefficients exceed the observed value $r_{obs} = 0.3$ (i.e. p-value equals 0.2). Hence, the observed correlation is not significant, and does therefore not provide proof for a positive trend.

4.2 Importance Sampling

Matlab code for estimating EX^2 using samples from uniform

```
%  X ~ N(0,1), hence density  = f(x) = 1/sqrt{2\pi}  exp(-x^2/2)
%     Density for uniform U(-5,5) :  g(x) = 1/10;
%
%  We need to estimate  EX^2 = Var(X) = 1 by sampling from the uniform;
%
%  This means that we need to sample say U ~ U(-5,5) and compute the sample
%  value :
%      F = phi(U) (f(U)/g(U))   where phi(u) = u^2
```

```
sample_size = 10000

U = 10*rand(sample_size,1)-5;

F = (U.^2) .* (10*normpdf(U));

mc_estimate = mean(F);
mc_population_std = std(F);
mc_estimate_std = std(F)/sqrt(sample_size)
```

Conclusion The matlab code above produces the following result (uncertainty = 3 s.e.):

$$\text{MC estimate} = 0.99 \pm 0.03$$

where the error margins are based on the fact that the standard error (standard deviation of the sample mean) equals $se = 0.0105$. We take three times this value to quantify the uncertainty. Notice that the theoretical value is given by $EX^2 = VarX = 1$ (since $EX = 0$).

Matlab code for estimating from unusual distribution

```
%  Question 2:
%--------------

%  X is distributed according to density   f(x) = (1+cos(pi*x))/2;
f = @(x) (1+cos(pi*x))/2;  % definition of density

sample_size = 1000   % for MC sample

%  Sample from uniform
%==========================

U = 2*rand(sample_size,1)-1;    %  Uniform on -1, 1;  density = 1/2

%  Compute the function value (weighted with correction factor)

F = (U.^2) .* (f(U)/(1/2));  % compute the pointwise result;

% Compute mean, std and se.

mc_estimate = mean(F);
mc_population_std = std(F);
mc_estimate_std = std(F)/sqrt(sample_size);

%  Sample from triangular
%=============================
```

```
%  Define triangular density g(x) = x+1 (if x<=0) and
%   g(x) = 1-x (if x>0):

g = @(x) (x<=0).*(x+1) + (x>0).*(-x+1);    % density of the triangular density

%  To create an observation from the triangular distribution,
%  add two independent unir
Z = rand(sample_size,1) + rand(sample_size,1) - 1;    %

% Compute function value (weighted with correction factor)
F = (Z.^2) .* (f(Z)./g(Z));  % compute the pointwise result;

% Compute mean, std and se
mc_estimate = mean(F);
mc_population_std = std(F);
mc_estimate_std = std(F)/sqrt(sample_size);


+++++++++++++++++++++++++++++++++++++++++
MATLAB code yields following result:
+++++++++++++++++++++++++++++++++++++++++

===========  Sampling from uniform on [-1,1] ============
MC estimate = 0.13251 with s.e. = 0.0028724
===========  Sampling from triangular density on [-1,1] ============
MC estimate = 0.13431 with s.e. = 0.0037221
```

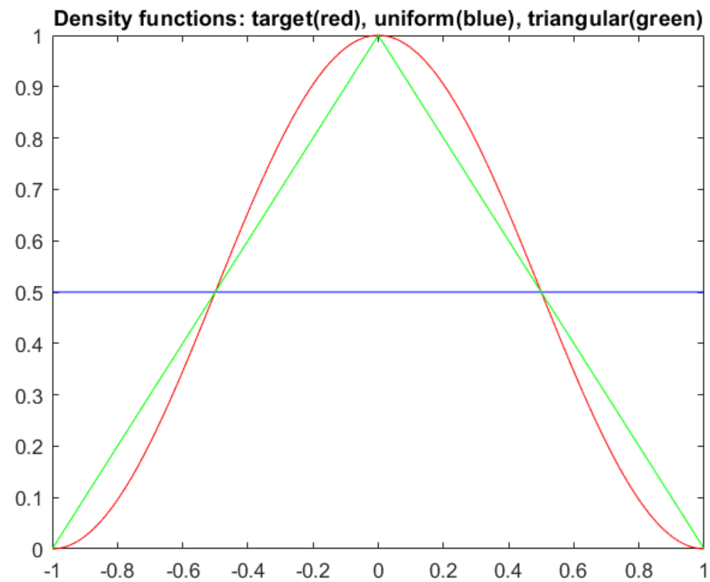

Figure 2: Target density (red) from which we need to sample. Two approximations used for importance sampling: uniform (blue) and triangular(green)

4.3 Kullback-Leibler divergence

KL for two gaussians Assuming normal densities $f \sim N(\mu_1, \sigma_1^2)$ and $g \sim N(\mu_2, \sigma_2^2)$, a straightforward computation yields:

$$KL(f||g) = \int f(x) \log\left(\frac{f(x)}{g(x)}\right) dx$$

$$\begin{aligned}
\log\left(\frac{f(x)}{g(x)}\right) &= \log\left(\frac{e^{-(x-\mu_1)^2/2\sigma_1^2}}{\sqrt{\cancel{2\pi}\sigma_1^2}} \frac{\sqrt{\cancel{2\pi}\sigma_2^2}}{e^{-(x-\mu_2)^2/2\sigma_2^2}}\right) \\
&= \log\left(\frac{\sigma_2}{\sigma_1} e^{-(x-\mu_1)^2/2\sigma_1^2 + (x-\mu_2)^2/2\sigma_2^2}\right) \\
&= \log\left(\frac{\sigma_2}{\sigma_1}\right) + \log\left(e^{-(x-\mu_1)^2/2\sigma_1^2 + (x-\mu_2)^2/2\sigma_2^2}\right) \\
&= \log\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{-(x-\mu_1)^2}{2\sigma_1^2} + \frac{(x-\mu_2)^2}{2\sigma_2^2}
\end{aligned}$$

$$
\begin{aligned}
KL(f||g) &= \int f(x)\left(\log\left(\frac{\sigma_2}{\sigma_1}\right) - \frac{(x-\mu_1)^2}{2\sigma_1^2} + \frac{(x-\mu_2)^2}{2\sigma_2^2}\right) dx \\
&= \int f(x)\log\left(\frac{\sigma_2}{\sigma_1}\right) dx - \int f(x)\frac{(x-\mu_1)^2}{2\sigma_1^2} dx + \int f(x)\frac{(x-\mu_2)^2}{2\sigma_2^2} dx \\
&= \log\left(\frac{\sigma_2}{\sigma_1}\right) - \frac{\sigma_1^2}{2\sigma_1^2} + \int f(x)\frac{(x-\mu_1+\mu_1-\mu_2)^2}{2\sigma_2^2} dx \\
&= \log\left(\frac{\sigma_2}{\sigma_1}\right) - \frac{1}{2} + \int f(x)\frac{(x-\mu_1)^2 - 2(x-\mu_1)(\mu_1-\mu_2) + (\mu_1-\mu_2)^2}{2\sigma_2^2} dx \\
&= \log\left(\frac{\sigma_2}{\sigma_1}\right) - \frac{1}{2} + \frac{\sigma_1^2 - 0 + (\mu_1-\mu_2)^2}{2\sigma_2^2} \\
&= \log\left(\frac{\sigma_2}{\sigma_1}\right) - \frac{1}{2} + \frac{\sigma_1^2 + (\mu_1-\mu_2)^2}{2\sigma_2^2}
\end{aligned}
$$

$$
KL(f||g) = \int f(x)\log\left(\frac{f(x)}{g(x)}\right) dx = \log\frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1-\mu_2)^2}{2\sigma_2^2} - \frac{1}{2}.
$$

Notice the asymmetric role of both densities. Although not obvious from the above, the KL distribution is always non-negative.

MC for KL estimation

```
% f ~ N(mu1,sigma1^2)
mu1 = 0;    sigma1 = 2;

% g ~ N(mu2,sigma2^2)
mu2 = 2;    sigma2 = 3;

sample_size = 1000

%  Sample from f and compute the KL value at each sample point

X = mu1 + sigma1*randn(sample_size,1);
KL = log(normpdf(X,mu1,sigma1)./normpdf(X,mu2,sigma2));

KL_div = mean(KL);   %  KL divergence
KL_div_std = std(KL)/sqrt(sample_size);

% KL divergence based on theoretical expression:

KL_div_theory = log(sigma2/sigma1) + (sigma1^2+(mu1-mu2)^2)/(2*sigma2^2) - 1/2;

++++++++++++++++++   MATLAB OUTPUT  ++++++++++++++++++++++++++++++++
Monte Carlo estimate of KL-divergence: 0.33517 with s.e.= 0.019843
Theoretical (exact) value = 0.34991
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

5.1 UCB versus ϵ-greedy for k-bandit problem

Lai-Robbins bound for two Gaussian For clarity's sake, we restrict our attention to two Gaussian densities f_1 and f_2, both with unit variance, but different means. Let's denote the difference in the means as $\Delta = |\mu_1 - \mu_2|$. The KL divergence (see previous problem) is therefore given by:

$$KL(f_1||f_2) = 0 + \frac{1+\Delta^2}{2} - \frac{1}{2} = \frac{\Delta^2}{2},$$

from which it follows that the Lai-Robbins coefficient is given by:

$$A = \frac{\Delta}{\Delta^2/2} = \frac{2}{\Delta}.$$

Question 1 Consider two Gaussians ($k = 2$) with randomly generated means -1.0430 and -0.3677 and unit variance. Hence, $\Delta = 0.6753$ and Lai-Robbins bound: $A_{LR} = 2.9615$. (see Fig 3). We use UCB hyperparameter $c = 2$

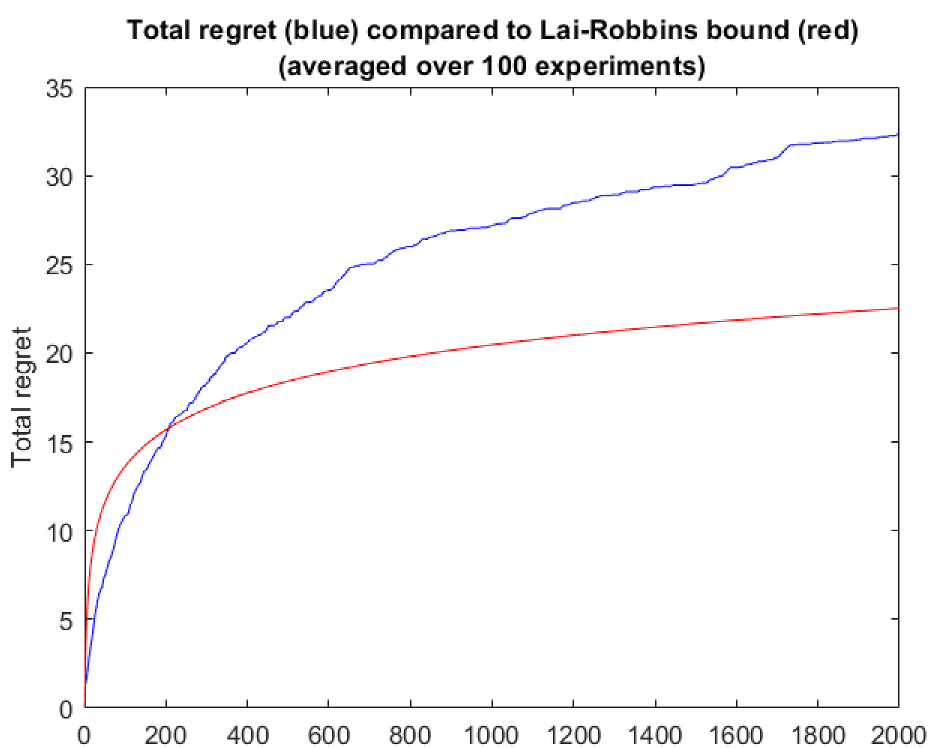

Figure 3: Total regret (blue) compared to Lai-Robbins (red)

Question 2 Comparing percentage of correct action choices for different algorithms (see Fig 4). Notice that by its very definition ϵ-greedy (with $\epsilon = 0.1$) cannot improve beyond the 90% level. The UCB performance clearly depends on the value of hyper-parameter c.

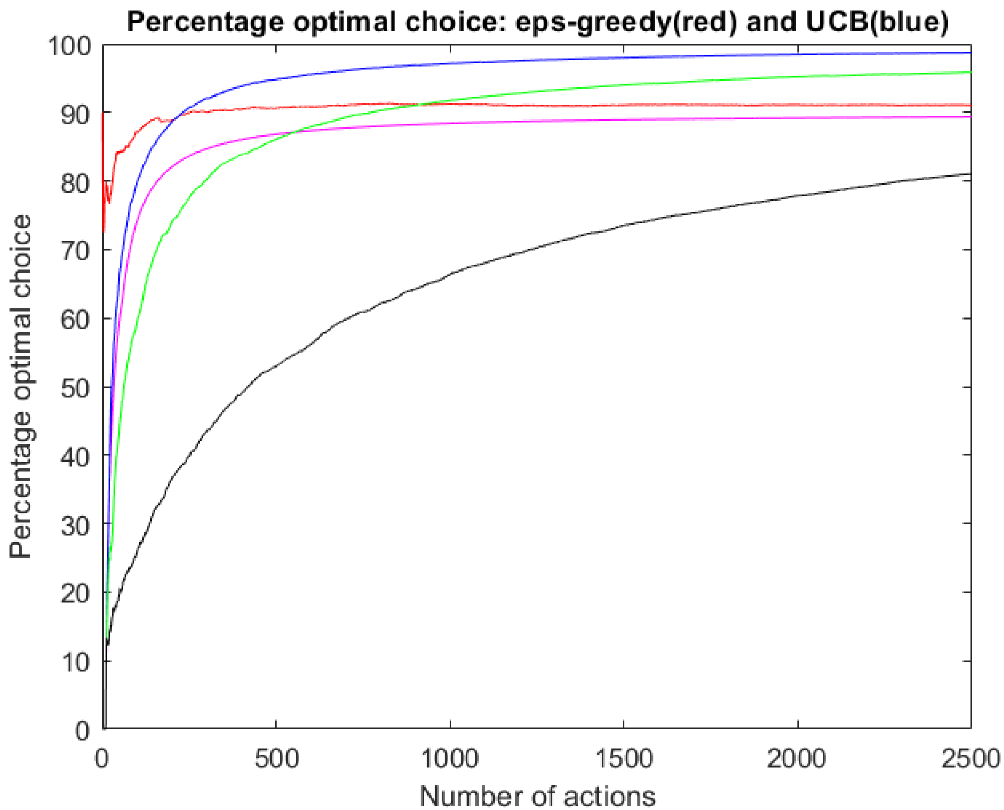

Figure 4: Comparing the percentage of correct choices for different exploration-exploitation strategies. The shown graphs are averaged over 10 experiments. Strategies: ϵ-greedy ($\epsilon = 0.1$): red, UCB with $c = 0.25$ (magenta), $c = 1$ (blue), $c = 2$ (green), $c = 5$ (black).