Bandits stochastiques

emilie.kaufmann@telecom-paristech.fr

19 mai 2014

1 Construire un problème de bandit

Dans le problème de bandit stochastique, tirer un bras revient à observer une récompense qui est une réalisation i.i.d d'une distribution propre au bras. Dans ce TP, on se concentrera sur des distributions à valeurs dans $[0, 1]$. Plusieurs exemples de telles distributions sont proposés dans les classes suivantes :

`armBernoulli.m` `armBeta.m` `armFinite.m` `armExp.m`

Pour chaque objet Arm d'une de ces classes, on dispose des commandes suivantes :
– *Arm.mean* retourne la moyenne du bras
– *Arm.play* retourne un tirage du bras
Un problème de bandit est alors un objet du type suivant :

```
Problem = {Arm1, Arm2, ...,ArmK}
```

Vous pouvez compléter le début du fichier *'mainBandits.m'* avec un premier problème de bandit de votre choix sur lequel vous pourrez faire des expériences.

2 UCB versus ϵ-Greedy

La première approche vue en cours est l'algorithme ϵ-greedy, qui avec probabilité ϵ_t choisit un bras au hasard et avec probabilité $1-\epsilon_t$ choisit le bras ayant la meilleure moyenne empirique. On choisira pour nos expériences un taux d'exploration dépendant de deux paramètres C et α :

$$\epsilon_t = \frac{C}{t^\alpha}$$

Pour des variables aléatoires à valeurs dans $[0, 1]$, l'algorithme UCB-1 [Auer et al.02] choisit au temps t (après une phase d'initialisation où chaque bras est tiré, comme d'ailleurs dans l'algorithme précédent)

$$I_t = \underset{i=1\dots K}{\operatorname{argmax}} \ \hat{\mu}_{i,T_i(t-1)} + \sqrt{\frac{\alpha \log t}{2T_i(t-1)}}$$

où $\hat{\mu}_{i,s}$ est la moyenne empirique de s observations du bras i et $T_i(t)$ est le nombre de tirages du bras i entre les instants 1 et t.

1. La classe *armExp* correspond à des lois exponentielles tronquées. On en obtient une réalisation en prenant $\min(X, 1)$ où $X \sim \mathcal{E}(\lambda)$. Retrouver la valeur de l'espérance pour cette classe.
2. Justification de l'algorithme UCB1. L'inégalité de Hoeffding pour des distributions à support dans $[0, 1]$ donne

$$\mathbb{P}(\mu_i - \hat{\mu}_{i,s} > \epsilon) \leq e^{-2s\epsilon^2}.$$

   Justifier que l'indice calculé pour chaque bras par l'algorithme UCB1 est bien une borne de confiance supérieure pour la moyenne inconnue μ_i du bras i. Avec quel niveau de confiance ?
3. On donne l'implémentation de la stratégie naive (*naive.m*) qui choisit à chaque instant le bras ayant la meilleure moyenne empirique (et qui correspond donc à ϵ-greedy avec $C = 0$). Implémenter les fonctions

```
[rec,tir]=Greedy(n,C,alpha,Problem)
[rec,tir]=UCB(n,alpha,Problem)
```

   qui simulent un jeu de bandit de n coups avec la stratégie ϵ-Greedy et UCB : rec et tir désignent respectivement la suite des n récompenses obtenues et celle des bras tirés.

   Sur une trajectoire, afficher le regret cumulé des stratégies naives, Greedy et UCB. Essayer sur plusieurs trajectoires. Que peut-on dire de la variabilité des résultats pour les différents algorithmes ?
4. La performance des algorithmes de bandits est généralement mesurée à l'aide du *regret moyen* (l'espérance du regret)

$$\mathbb{E}[R_n] = \mathbb{E}\left[\sum_{i=1}^{n}(\mu^* - \mu_{I_t})\right],$$

   que l'on estime par Monte-Carlo à l'aide d'un grand nombre de répétitions de l'algorithme. Pour deux problèmes de bandits de votre choix, l'un facile, l'autre difficile, afficher sur un même graphique les courbes de regret moyen cumulé pour différentes stratégies et plusieurs valeurs de paramètre, afin d'illustrer quels sont les meilleurs paramètres sur chacun des problèmes. Pour ϵ-greedy, on essayera en particulier le paramètre pour lequel le cours donne une garantie théorique.

3 KL-UCB pour des récompenses binaires

Dans cette partie on considère un problème de bandit à 10 bras, dont le bras i est une loi de Bernoulli $\mathcal{B}(\mu_i)$ où le vecteur des moyennes est donné par

$$\mu = [0.1 \quad 0.05 \quad 0.05 \quad 0.05 \quad 0.02 \quad 0.02 \quad 0.02 \quad 0.01 \quad 0.01 \quad 0.01].$$

Dans ce problème, chaque bras a une moyenne très faible, ce qui est aussi le cas dans des contextes de recommandation où les probabilités de clic sont très faibles. Dans de tels contextes, on va voir dans cette partie qu'il est crucial d'utiliser des intervalles de confiances adaptés aux distributions (et construits à l'aide de la divergence de Kullback-Leibler) à la place de l'algorithme UCB1 ci-dessus. On introduit la notation suivante pour le divergence de Kullback-Leibler entre deux distributions de Bernoulli :

$$d(x,y) = D\left(\mathcal{B}(x)||\mathcal{B}(y)\right) = x\log\frac{x}{y} + (1-x)\log\frac{1-x}{1-y}.$$

L'inégalité de Chernoff se réécrit, pour $x > \mu_i$ et pour $y < \mu_i$

$$\mathbb{P}(\hat{\mu}_{i,s} > x) \leq e^{-s\times d(x,\mu_i)} \quad \text{et} \quad \mathbb{P}(\hat{\mu}_{i,s} < y) \leq e^{-s\times d(y,\mu_i)}$$

En introduisant la quantité

$$u_{i,s}(t) = \max\left\{q > \hat{\mu}_{i,s} : sd\left(\hat{\mu}_{i,s}, q\right) \leq \alpha\log(t)\right\}$$

on retrouve la la propriété de borne de confiance supérieure : $\mathbb{P}(u_{i,s}(t) < \mu_i) \leq \frac{1}{t^\alpha}$. On définit alors l'algorithme KL-UCB [Cappé et al. 13] comme l'algorithme choisissant à l'instant t (après une phase d'initialisation) le bras

$$I_t = \underset{i=1,\ldots,K}{\operatorname{argmax}}\ u_{i,T_i(t-1)}(t).$$

1. On donne la fonction *klIC.m*, qui permet de calculer le sommet d'un intervalle de confiance de type KL présenté ci-dessus (voir les commentaires dans le code). Implémenter des fonctions

   ```
   [rec,tir]=KLUCB(n,C,alpha,mu)
   [rec,tir]=UCB(n,alpha,mu)
   ```

   qui dépendent du vecteur de moyennes μ d'un problème de bandits à récompenses binaires et retournent comme précédemment la suite des récompenses obtenues et des bras tirés par ces algorithmes jusqu'à un horizon n.

2. Pour $\alpha = 1$ (une valeur pour laquelle on peut prouver que les algorithmes sont efficaces sur tous les problèmes), afficher des courbes de regret moyen cumulé jusqu'à un horizon $n = 10000$ pour le problème à 10 bras ci-dessus. Ajouter la borne inférieure de Lai et Robbins, qui dit que pour tout bon algorithme, le regret moyen à l'horizon n, $\mathbb{E}[R_n]$, est minoré, lorsque n est suffisamment grand, de la manière suivante :

   $$\mathbb{E}[R_n] \geq \sum_{i:\mu_i<\mu^*} \frac{(\mu^* - \mu_i)}{d(\mu_i,\mu^*)}\log(n).$$

   Conclure.