Analyse non asymptotique d'un test séquentiel de détection de ruptures et application aux bandits non stationnaires

Conférence GRETSI @ Lille, Août 2019

Lilian Besson

Doctorant

Équipe SCEE, labo IETR, CentraleSupélec à Rennes
& Équipe SequeL, labo CRIStAL, Inria à Lille

Jeudi 29 Août 2019

1. Problèmes de bandits multi-bras (stationnaires)

1. **Problèmes de bandits multi-bras (stationnaires)**
2. Problèmes de bandits multi-bras stationnaires par morceaux
3. Le test BGLR et ses propriétés à horizon fini
4. L'algorithme B-GLRT + klUCB
5. Simulations numériques

Bandits manchots ? (= un bras)

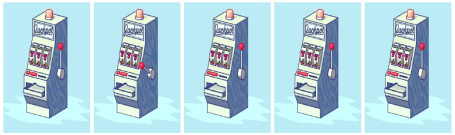
Un autre nom pour une machine à sous :

↪ Source Lucky Luke tome 18, © Dargaud.

Bandits multi-bras

= Prise de décisions séquentielles face à des environnements incertains :

Total Reward	Total Plays
14	24



1

	Arm 1	Arm 2	Arm 3	Arm 4	Arm 5
Rewards:	6	2	2	2	2
Pulls:	8	4	4	4	4
Estimated Probs:	0.750	0.500	0.500	0.500	0.500
UCBs:	1.641	1.761	1.761	1.761	1.761

↪ Démo interactive perso.crans.org/besson/phd/MAB_interactive_demo/

Modèle mathématique

- Étapes de temps discret $t = 1, \dots, T$
  L'*horizon* T est fixé et généralement inconnu

Modèle mathématique

- Étapes de temps discret $t = 1, \dots, T$
  L'*horizon* T est fixé et généralement inconnu
- Au temps t, une joueuse *choisit le bras* $A(t) \in \{1, \dots, K\}$,
  et elle observe *une récompense aléatoire iid* : $r(t) \sim \nu_k$, $r(t) \in \mathbb{R}$

Modèle mathématique

- Étapes de temps discret $t = 1, \dots, T$
  L'*horizon* T est fixé et généralement inconnu
- Au temps t, une joueuse *choisit le bras* $A(t) \in \{1, \dots, K\}$,
  et elle observe *une récompense aléatoire iid* : $r(t) \sim \nu_k$, $r(t) \in \mathbb{R}$
- On se restreint souvent à des bras de Bernoulli $\nu_k = \mathcal{B}(\mu_k)$, de moyenne $\mu_k \in [0, 1]$, qui donnent des récompenses binaires $r(t) \in \{0, 1\}$.

Modèle mathématique

- Étapes de temps discret $t = 1, \dots, T$
  L'*horizon* T est fixé et généralement inconnu
- Au temps t, une joueuse *choisit le bras* $A(t) \in \{1, \dots, K\}$,
  et elle observe *une récompense aléatoire iid* : $r(t) \sim \nu_k$, $r(t) \in \mathbb{R}$
- On se restreint souvent à des bras de Bernoulli $\nu_k = \mathcal{B}(\mu_k)$, de moyenne $\mu_k \in [0, 1]$, qui donnent des récompenses binaires $r(t) \in \{0, 1\}$.
- **But** : maximiser la somme des récompenses $\sum_{t=1}^{T} r(t)$

Modèle mathématique

- Étapes de temps discret $t = 1, \dots, T$
  L'*horizon* T est fixé et généralement inconnu
- Au temps t, une joueuse *choisit le bras* $A(t) \in \{1, \dots, K\}$,
  et elle observe *une récompense aléatoire iid* : $r(t) \sim \nu_k$, $r(t) \in \mathbb{R}$
- On se restreint souvent à des bras de Bernoulli $\nu_k = \mathcal{B}(\mu_k)$, de moyenne $\mu_k \in [0, 1]$, qui donnent des récompenses binaires $r(t) \in \{0, 1\}$.
- **But** : maximiser la somme des récompenses $\sum_{t=1}^{T} r(t)$
- ou maximiser la somme des récompenses *moyennes* $\mathbb{E}\left[\sum_{t=1}^{T} r(t)\right]$

Modèle mathématique

- Étapes de temps discret $t = 1, \dots, T$
  L'*horizon* T est fixé et généralement inconnu
- Au temps t, une joueuse *choisit le bras* $A(t) \in \{1, \dots, K\}$,
  et elle observe *une récompense aléatoire iid* : $r(t) \sim \nu_k$, $r(t) \in \mathbb{R}$
- On se restreint souvent à des bras de Bernoulli $\nu_k = \mathcal{B}(\mu_k)$, de moyenne $\mu_k \in [0, 1]$, qui donnent des récompenses binaires $r(t) \in \{0, 1\}$.
- **But** : maximiser la somme des récompenses $\sum_{t=1}^{T} r(t)$
- ou maximiser la somme des récompenses *moyennes* $\mathbb{E}\left[\sum_{t=1}^{T} r(t)\right]$
- Une stratégie efficace doit résoudre le compromis entre exploration et exploitation : explorer tous les bras pour découvrir le meilleur, tout en exploitant le meilleur bras courant.

Mesurer la performance d'un algorithme $\mathcal{A}$ par son regret moyen $R_{\mathcal{A}}(T)$

- Différence entre les récompenses accumulées par un "oracle" et $\mathcal{A}$

Mesurer la performance d'un algorithme $\mathcal{A}$ par son regret moyen $R_{\mathcal{A}}(T)$

- Différence entre les récompenses accumulées par un "oracle" et $\mathcal{A}$
- L'algorithme "oracle" joue toujours le meilleur bras (inconnu) $k^* = \arg\max_k \mu_k$ (la meilleure moyenne est notée $\mu_{k^*} = \mu^*$)

Mesurer la performance d'un algorithme $\mathcal{A}$ par son regret moyen $R_{\mathcal{A}}(T)$

- Différence entre les récompenses accumulées par un "oracle" et $\mathcal{A}$
- L'algorithme "oracle" joue toujours le meilleur bras (inconnu) $k^* = \arg\max_k \mu_k$ (la meilleure moyenne est notée $\mu_{k^*} = \mu^*$)
- But équivalent : maximiser la somme des récompenses moyennes $\Longleftrightarrow$ minimiser le regret moyen

$$R_{\mathcal{A}}(T) = \mathbb{E}\left[\sum_{t=1}^{T} r_{k^*}(t)\right] - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right] = T\mu^* - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right].$$

Mesurer la performance d'un algorithme $\mathcal{A}$ par son regret moyen $R_{\mathcal{A}}(T)$

- Différence entre les récompenses accumulées par un "oracle" et $\mathcal{A}$
- L'algorithme "oracle" joue toujours le meilleur bras (inconnu) $k^* = \arg\max_k \mu_k$ (la meilleure moyenne est notée $\mu_{k^*} = \mu^*$)
- But équivalent : maximiser la somme des récompenses moyennes $\Longleftrightarrow$ minimiser le regret moyen

$$R_{\mathcal{A}}(T) = \mathbb{E}\left[\sum_{t=1}^{T} r_{k^*}(t)\right] - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right] = T\mu^* - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right].$$

Mesurer la performance d'un algorithme $\mathcal{A}$ par son regret moyen $R_{\mathcal{A}}(T)$

- Différence entre les récompenses accumulées par un "oracle" et $\mathcal{A}$
- L'algorithme "oracle" joue toujours le meilleur bras (inconnu) $k^* = \arg\max_k \mu_k$ (la meilleure moyenne est notée $\mu_{k^*} = \mu^*$)
- But équivalent : maximiser la somme des récompenses moyennes $\Longleftrightarrow$ minimiser le regret moyen

$$R_{\mathcal{A}}(T) = \mathbb{E}\left[\sum_{t=1}^{T} r_{k^*}(t)\right] - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right] = T\mu^* - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right].$$

Régime typique pour des bandits stationnaires (bornes inf & sup)

- Aucun algorithme $\mathcal{A}$ ne peut atteindre un regret meilleur que $R_{\mathcal{A}}(T) \geq \Omega(\log(T))$
- Et un algorithme efficace $\mathcal{A}$ obtient $R_{\mathcal{A}}(T) \leq \mathcal{O}(\log(T))$

2. Bandits stationnaires par morceaux

1. Problèmes de bandits multi-bras (stationnaires)
2. **Problèmes de bandits multi-bras stationnaires par morceaux**
3. Le test BGLR et ses propriétés à horizon fini
4. L'algorithme B-GLRT + klUCB
5. Simulations numériques

Problèmes de bandits non stationnaires

Problèmes de bandits stationnaires

Le bras k tire une récompense selon la même distribution pour chaque instant :
$\forall t, r_k(t) \overset{\text{iid}}{\sim} \nu_k = \mathcal{B}(\mu_k)$.

Problèmes de bandits non stationnaires

Problèmes de bandits stationnaires

Le bras k tire une récompense selon la même distribution pour chaque instant :
$\forall t, r_k(t) \overset{\text{iid}}{\sim} \nu_k = \mathcal{B}(\mu_k)$.

Problèmes de bandits *non* stationnaires

Le bras k tire une récompense selon une distribution différente à chaque instant :
$\forall t, r_k(t) \overset{\text{iid}}{\sim} \nu_k(t) = \mathcal{B}(\mu_k(t))$.

$\Longrightarrow$ 😕 problème plus difficile ! Et très difficile si $\mu_k(t)$ peut changer n'importe quand !

Problèmes de bandits non stationnaires

Problèmes de bandits stationnaires

Le bras k tire une récompense selon la même distribution pour chaque instant :
$\forall t, r_k(t) \overset{\text{iid}}{\sim} \nu_k = \mathcal{B}(\mu_k)$.

Problèmes de bandits *non* stationnaires

Le bras k tire une récompense selon une distribution différente à chaque instant :
$\forall t, r_k(t) \overset{\text{iid}}{\sim} \nu_k(t) = \mathcal{B}(\mu_k(t))$.

$\Longrightarrow$ 😕 problème plus difficile ! Et très difficile si $\mu_k(t)$ peut changer n'importe quand !

Problèmes de bandits **stationnaires par morceaux**

$\hookrightarrow$ on se concentre sur le cas où il y a au plus $o(\sqrt{T})$ intervalles sur lesquels les moyennes sont toutes stationnaires (= **séquences**)

Ruptures et séquences stationnaires

On définit

- Le nombre de rupture $\Upsilon_T = \sum_{t=1}^{T-1} \mathbb{1}(\exists k \in \{1, \dots, K\} : \mu_k(t) \neq \mu_k(t+1))$
- Le $i$ème point d'arrêt $\tau^i = \inf\{t > \tau^{i-1} : \exists k : \mu_k(t) \neq \mu_k(t+1)\}$ (avec $\tau^0 = 0$)

Ruptures et séquences stationnaires

On définit

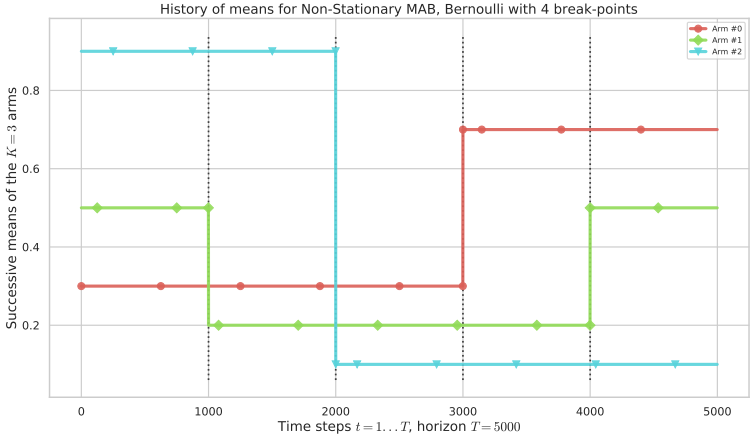
- Le nombre de rupture $\Upsilon_T = \sum_{t=1}^{T-1} \mathbb{1}(\exists k \in \{1, \dots, K\} : \mu_k(t) \neq \mu_k(t+1))$
- Le $i$ème point d'arrêt $\tau^i = \inf\{t > \tau^{i-1} : \exists k : \mu_k(t) \neq \mu_k(t+1)\}$ (avec $\tau^0 = 0$)

Hypothèses sur les problèmes stationnaires par morceaux (p.m.)

- Les récompenses $r_k(t)$ générées par chaque bras k sont *iid* sur chaque intervale $[\tau^i + 1, \tau^{i+1}]$ (la $i$ème séquence)
- Il y a $\Upsilon_T = o(\sqrt{T})$ ruptures
- Et Υ_T peut être connu à l'avance
- Les séquences sont toutes "assez longues"

Exemple d'un problème de bandits stationnaire p.m.

On affiche les moyennes $\mu_1(t)$, $\mu_2(t)$, $\mu_3(t)$ des $K = 3$ bras.
Il y a $\Upsilon_T = 4$ ruptures, et 5 séquences entre $t = 1$ et $T = 5000$:



Regret pour les bandits stationnaires par morceaux ?

L'algorithme "oracle" joue le meilleur bras (inconnu) $k^*(t) = \arg\max \mu_k(t)$ (qui change entre chaque séquence stationnaire)

$$R_{\mathcal{A}}(T) = \mathbb{E}\left[\sum_{t=1}^{T} r_{k^*(t)}(t)\right] - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right] = \left(\sum_{t=1}^{T} \max_k \mu_k(t)\right) - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right].$$

Regret pour les bandits stationnaires par morceaux ?

L'algorithme "oracle" joue le meilleur bras (inconnu) $k^*(t) = \arg\max \mu_k(t)$ (qui change entre chaque séquence stationnaire)

$$R_{\mathcal{A}}(T) = \mathbb{E}\left[\sum_{t=1}^{T} r_{k^*(t)}(t)\right] - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right] = \left(\sum_{t=1}^{T} \max_k \mu_k(t)\right) - \sum_{t=1}^{T} \mathbb{E}\left[r(t)\right].$$

Régimes typiques pour les bandits stationnaires par morceaux

- La borne inférieure est $R_{\mathcal{A}}(T) \geq \Omega(\sqrt{KT\Upsilon_T})$
- Actuellement, les meilleurs algorithmes $\mathcal{A}$ obtiennent
  - $R_{\mathcal{A}}(T) \leq \mathcal{O}(K\sqrt{T\Upsilon_T \log(T)})$ si T et Υ_T sont connus
  - $R_{\mathcal{A}}(T) = \mathcal{O}(K\Upsilon_T\sqrt{T\log(T)})$ si T et Υ_T sont inconnus

3. Le test BGLR et ses propriétés à horizon fini

1. Problèmes de bandits multi-bras (stationnaires)
2. Problèmes de bandits multi-bras stationnaires par morceaux
3. **Le test BGLR et ses propriétés à horizon fini**
4. L'algorithme B-GLRT + klUCB
5. Simulations numériques

Le problème de détection de rupture

Imaginez le jeu suivant...

- Vous observez des données $X_1, X_2, \cdots, X_t, \cdots \in [0, 1]$...
- Vous savez que X_t est généré par une certaine distribution inconnue...

Le problème de détection de rupture

Imaginez le jeu suivant...

- Vous observez des données $X_1, X_2, \cdots, X_t, \cdots \in [0,1]$...
- Vous savez que X_t est généré par une certaine distribution inconnue...
- Votre but est de distinguer entre deux hypothèses :
  - $\mathcal{H}_0$ Les distributions **ont toutes la même moyenne** ("pas de rupture")
    $\exists \mu_0, \mathbb{E}[X_1] = \mathbb{E}[X_2] = \cdots = \mathbb{E}[X_t] = \mu_0$
  - $\mathcal{H}_1$ Les distributions ont **changé de moyennes au temps** τ
    $\exists \mu_0, \mu_1, \tau, \mathbb{E}[X_1] = \cdots = \mathbb{E}[X_\tau] = \mu_0,\ \mu_0 \neq \mu_1,\ \mathbb{E}[X_{\tau+1}] = \mathbb{E}[X_{\tau+2}] = \cdots = \mu_1$
- Vous arrêtez au temps $\widehat{\tau}$, dès que vous détectez une rupture

Le problème de détection de rupture

Imaginez le jeu suivant...

- Vous observez des données $X_1, X_2, \cdots, X_t, \cdots \in [0,1]$...
- Vous savez que X_t est généré par une certaine distribution inconnue...
- Votre but est de distinguer entre deux hypothèses :
  - $\mathcal{H}_0$ Les distributions **ont toutes la même moyenne** ("pas de rupture")
    $\exists \mu_0, \mathbb{E}[X_1] = \mathbb{E}[X_2] = \cdots = \mathbb{E}[X_t] = \mu_0$
  - $\mathcal{H}_1$ Les distributions ont **changé de moyennes au temps** τ
    $\exists \mu_0, \mu_1, \tau, \mathbb{E}[X_1] = \cdots = \mathbb{E}[X_\tau] = \mu_0,\ \mu_0 \neq \mu_1,\ \mathbb{E}[X_{\tau+1}] = \mathbb{E}[X_{\tau+2}] = \cdots = \mu_1$
- Vous arrêtez au temps $\widehat{\tau}$, dès que vous détectez une rupture

Un détecteur de rupture séquentiel est un **temps d'arrêt** $\widehat{\tau}$, mesurable selon $\mathcal{F}_t = \sigma(X_1, \cdots, X_t)$, qui rejette l'hypothèse $\mathcal{H}_0$ lorsque $\widehat{\tau} < \infty$.

Test de rapport de vraisemblances de Bernoulli

Hypothèses : toutes les distributions sont Bernoulli ($\nu_k = \mathcal{B}(\mu_k)$)

Le problème se résume à distinguer

$\mathcal{H}_0$: $(\exists \mu_0 : \forall i \in \mathbb{N}^*, X_i \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_0))$, contre l'alternative

$\mathcal{H}_1$: $(\exists \mu_0 \neq \mu_1, \tau > 1 : X_1, \cdots, X_\tau \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_0)$ et $X_{\tau+1}, \cdots \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_1))$.

Test de rapport de vraisemblances de Bernoulli

Hypothèses : toutes les distributions sont Bernoulli ($\nu_k = \mathcal{B}(\mu_k)$)

Le problème se résume à distinguer

$\mathcal{H}_0$: $(\exists \mu_0 : \forall i \in \mathbb{N}^*, X_i \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_0))$, contre l'alternative

$\mathcal{H}_1$: $(\exists \mu_0 \neq \mu_1, \tau > 1 : X_1, \cdots, X_\tau \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_0)$ et $X_{\tau+1}, \cdots \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_1))$.

Après avoir observé $X_1, \cdots, X_n$, la statistique du test de rapport de vraisemblances pour cette hypothèse est

$$\mathcal{L}(n) = \frac{\sup\limits_{\mu_0, \mu_1, \tau < n} \ell(X_1, \cdots, X_n; \mu_0, \mu_1, \tau)}{\sup\limits_{\mu_0} \ell(X_1, \cdots, X_n; \mu_0)},$$

où $\ell(X_1, \cdots, X_n; \mu_0)$ et $\ell(X_1, \cdots, X_n; \mu_0, \mu_1, \tau)$ sont les vraisemblances des observations selon les modèles $\mathcal{H}_0$ et $\mathcal{H}_1$.

Test de rapport de vraisemblances de Bernoulli

Hypothèses : toutes les distributions sont Bernoulli ($\nu_k = \mathcal{B}(\mu_k)$)

Le problème se résume à distinguer

$\mathcal{H}_0$: $(\exists \mu_0 : \forall i \in \mathbb{N}^*, X_i \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_0))$, contre l'alternative

$\mathcal{H}_1$: $(\exists \mu_0 \neq \mu_1, \tau > 1 : X_1, \cdots, X_\tau \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_0)$ et $X_{\tau+1}, \cdots \overset{\text{i.i.d.}}{\sim} \mathcal{B}(\mu_1))$.

Après avoir observé $X_1, \cdots, X_n$, la statistique du test de rapport de vraisemblances pour cette hypothèse est

$$\mathcal{L}(n) = \frac{\sup\limits_{\mu_0, \mu_1, \tau < n} \ell(X_1, \cdots, X_n; \mu_0, \mu_1, \tau)}{\sup\limits_{\mu_0} \ell(X_1, \cdots, X_n; \mu_0)},$$

où $\ell(X_1, \cdots, X_n; \mu_0)$ et $\ell(X_1, \cdots, X_n; \mu_0, \mu_1, \tau)$ sont les vraisemblances des observations selon les modèles $\mathcal{H}_0$ et $\mathcal{H}_1$.

$\hookrightarrow$ De grandes valeurs de $\mathcal{L}(n)$ tendent à rejeter $\mathcal{H}_0$ en faveur de $\mathcal{H}_1$.

(log) Rapport de vraisemblances de Bernoulli

On peut réécrire cette statistique $\mathcal{L}(n) = \frac{\sup\limits_{\mu_0,\mu_1,\tau<n} \ell(X_1,\cdots,X_n;\mu_0,\mu_1,\tau)}{\sup\limits_{\mu_0} \ell(X_1,\cdots,X_n;\mu_0)}$, en utilisant les vraisemblances de Bernoulli, et les moyennes glissantes $\widehat{\mu}_{k:k'} = \frac{1}{k'-k+1}\sum\limits_{s=k}^{k'} X_s$:

$$\log \mathcal{L}(n) = \max_{s\in\{1,\cdots,n-1\}} \big[s \times \mathrm{kl}(\underbrace{\widehat{\mu}_{1:s}}_{\text{avant rupture}}, \underbrace{\widehat{\mu}_{1:n}}_{\text{tout}}) + (n-s) \times \mathrm{kl}(\underbrace{\widehat{\mu}_{s+1:n}}_{\text{après rupture}}, \underbrace{\widehat{\mu}_{1:n}}_{\text{tout}}) \big].$$

Où $\mathrm{kl}(x,y) = x\ln\left(\frac{x}{y}\right) + (1-x)\ln\left(\frac{1-x}{1-y}\right)$ est l'entropie relative binaire

Le test généralisé de rapport de vraisemblances de Bernoulli (B-GLRT)

- On peut étendre ce test si les observations sont sous-Bernoulli.
  (fonction génératrice ϕ des moments bornée par celle d'une loi de Bernoulli $\phi_{\mathcal{B}}$)
- Et toute distribution à support borné dans $[0, 1]$ est sous-Bernoulli !
- $\Longrightarrow$ le test B-GLR peut être appliqué à n'importe quelles observations bornées 

Le test généralisé de rapport de vraisemblances de Bernoulli (B-GLRT)

- On peut étendre ce test si les observations sont sous-Bernoulli.
  (fonction génératrice ϕ des moments bornée par celle d'une loi de Bernoulli $\phi_{\mathcal{B}}$)
- Et toute distribution à support borné dans $[0, 1]$ est sous-Bernoulli !
- $\implies$ le test B-GLR peut être appliqué à n'importe quelles observations bornées

Le test de détection séquentielle de rupture B-GRLT

Le B-GLRT est le temps d'arrêt définit par

$$\widehat{\tau}_\delta = \inf\left\{ n \in \mathbb{N}^* : \max_{s \in \{1, \cdots, n-1\}} \left[s \,\mathrm{kl}\left(\widehat{\mu}_{1:s}, \widehat{\mu}_{1:n}\right) + (n-s)\,\mathrm{kl}\left(\widehat{\mu}_{s+1:n}, \widehat{\mu}_{1:n}\right) \right] \geq \beta(n, \delta) \right\}$$

- avec une function seuil $\beta(n, \delta)$ spécifiée plus tard,
- n est le nombre d'observations $(X_1, \ldots, X_n)$,
- δ est le niveau de confiance ($\delta \in (0, 1)$).

Probabilité de fausse alarme

Un bon test ne doit pas détecter de rupture s'il n'y a pas de rupture à détecter…

Probabilité de fausse alarme

Un bon test ne doit pas détecter de rupture s'il n'y a pas de rupture à détecter...

Définition : fausse alarme

Le temps d'arrêt est $\widehat{\tau}_\delta$, et une rupture est détectée si $\widehat{\tau}_\delta < \infty$.
Soit $\mathbb{P}_{\mu_0}$ un modèle de probabilité selon lequel les observations sont $\forall t, X_t \in [0,1]$ et $\forall t, \mathbb{E}[X_t] = \mu_0$.
La probabilité de fausse alarme est $\mathbb{P}_{\mu_0}(\widehat{\tau}_\delta < \infty)$.

$\Longrightarrow$ But : contrôler l'événement de fausse alarme ! (en forte proba)

Premier résultat pour le test BGLR

Contrôler la probabilité de fausse alarme

Pour n'importe quel niveau de confiance $0 < \delta < 1$, le test BGLR satisfait

$$\mathbb{P}_{\mu_0}(\widehat{\tau}_\delta < \infty) \leq \delta$$

avec la fonction seuil

$$\beta(n, \delta) = 2\,\mathcal{T}\left(\frac{\ln(3n\sqrt{n}/\delta)}{2}\right) + 6\ln(1 + \ln(n)) \simeq \ln\left(\frac{3n\sqrt{n}}{\delta}\right) = \mathcal{O}\left(\log\left(\frac{n}{\delta}\right)\right).$$

Où $\mathcal{T}(x)$ vérifie $\mathcal{T}(x) \simeq x + \ln(x)$ pour x assez grand

Premier résultat pour le test BGLR

Contrôler la probabilité de fausse alarme

Pour n'importe quel niveau de confiance $0 < \delta < 1$, le test BGLR satisfait

$$\mathbb{P}_{\mu_0}(\widehat{\tau}_\delta < \infty) \leq \delta$$

avec la fonction seuil

$$\beta(n, \delta) = 2\,\mathcal{T}\left(\frac{\ln(3n\sqrt{n}/\delta)}{2}\right) + 6\ln(1 + \ln(n)) \simeq \ln\left(\frac{3n\sqrt{n}}{\delta}\right) = \mathcal{O}\left(\log\left(\frac{n}{\delta}\right)\right).$$

Où $\mathcal{T}(x)$ vérifie $\mathcal{T}(x) \simeq x + \ln(x)$ pour x assez grand

Preuve ?

Voir la version longue de notre article HAL-02006471 et arXiv:1902.01575

Délai de détection

Un bon test devrait détecter une rupture “assez vite” s’il y a une rupture à détecter, avec assez d’échantillons avant la rupture…

Délai de détection

Un bon test devrait détecter une rupture "assez vite" s'il y a une rupture à détecter, avec assez d'échantillons avant la rupture...

Definition : délai de détection

Soit $\mathbb{P}_{\mu_0,\mu_1,\tau}$ un modèle de probabilité selon lequel $\forall t, X_t \in [0,1]$ et $\forall t \leq \tau, \mathbb{E}[X_t] = \mu_0$ et $\forall t \geq \tau+1, \mathbb{E}[X_t] = \mu_1$, avec $\mu_0 \neq \mu_1$.
L'écart de cette rupture est $\Delta = |\mu_0 - \mu_1|$.
Le délai de détection est définit par $u = \widehat{\tau}_\delta - \tau \in \mathbb{N}$.

$\Longrightarrow$ But : contrôler le délai de détection! (en forte probabilité)

Second résultat pour le test BGLR

Contrôler le délai de détection

Pour une rupture d'amplitude $\Delta = |\mu_1 - \mu_0|$, le test BGLR satisfait

$$\mathbb{P}_{\mu_0,\mu_1,\tau}(\widehat{\tau}_\delta \geq \tau + u) \leq \exp\left(-\frac{2\tau u}{\tau + u}\left(\max\left[0, \Delta - \sqrt{\frac{\tau+u}{2\tau u}\beta(\tau+u,\delta)}\right]\right)^2\right)$$
$$= \mathcal{O}(\text{décroissance exponentielle en } u) = \mathcal{O}(\exp \searrow (u)).$$

avec la même fonction de seuil $\beta(n,\delta) \simeq \ln(3n\sqrt{n}/\delta)$.

Conséquence

En forte probabilité, le délai $\widehat{\tau}_\delta$ du BGLR est borné par $\mathcal{O}(\Delta^{-2}\ln(1/\delta))$ si assez d'échantillons sont observés avant la rupture au temps τ.

BGLR est un test de détection de rupture efficace 😎 !

- On a vu qu'en choisissant
  - un niveau de confiance δ,
  - et une bonne fonction de seuil $\beta(n, \delta) \simeq \ln(3n\sqrt{n}/\delta) = \mathcal{O}(\log(n/\delta))$,

BGLR est un test de détection de rupture efficace 😎 !

- On a vu qu'en choisissant
  - un niveau de confiance δ,
  - et une bonne fonction de seuil $\beta(n, \delta) \simeq \ln(3n\sqrt{n}/\delta) = \mathcal{O}(\log(n/\delta))$,
- on peut contrôler les deux propriétés du test BGLR :
  - sa probabilité de fausse alarme : $\mathbb{P}_{\mu_0}(\widehat{\tau}_\delta < \infty) \leq \delta$,
  - son délai de détection : $\mathbb{P}_{\mu_0,\mu_1,\tau}(\widehat{\tau}_\delta \geq \tau + u)$ décroît exponentiellement rapidement en u (s'il y a assez d'observations avant et après la rupture).
- $\Longrightarrow$ Le test BGLR détecte les ruptures efficacement 😎

BGLR est un test de détection de rupture efficace 😎 !

- On a vu qu'en choisissant
  - un niveau de confiance δ,
  - et une bonne fonction de seuil $\beta(n, \delta) \simeq \ln(3n\sqrt{n}/\delta) = \mathcal{O}(\log(n/\delta))$,
- on peut contrôler les deux propriétés du test BGLR :
  - sa probabilité de fausse alarme : $\mathbb{P}_{\mu_0}(\widehat{\tau}_\delta < \infty) \leq \delta$,
  - son délai de détection : $\mathbb{P}_{\mu_0,\mu_1,\tau}(\widehat{\tau}_\delta \geq \tau + u)$ décroît exponentiellement rapidement en u (s'il y a assez d'observations avant et après la rupture).
- $\implies$ Le test BGLR détecte les ruptures efficacement 😎

Garanties non asymptotiques 😎 [Maillard, ALT, 2019] [Lai & Xing, Sequential Analysis, 2010]

De telles garanties non asymptotiques sont des résultats assez récents !

4. L'algorithme B-GLRT + klUCB

1. Problèmes de bandits multi-bras (stationnaires)
2. Problèmes de bandits multi-bras stationnaires par morceaux
3. Le test BGLR et ses propriétés à horizon fini
4. **L'algorithme B-GLRT + klUCB**
5. Simulations numériques

On combine le test BGRL + les indices kl-UCB

Idées principales

- Calculer un indice UCB sur chaque bras k (Borne Sup. Confiance)
- Presque tout le temps, jouer le bras $A(t) = \arg \max_{k \in \{1,\dots,K\}} \text{kl-UCB}_k(t)$
- Utiliser le test BGLR pour détecter les ruptures sur le bras joué
- Si une rupture est détectée, effacer la mémoire des observations de *tous les* bras

On combine le test BGRL + les indices kl-UCB

Idées principales

- Calculer un indice UCB sur chaque bras k (Borne Sup. Confiance)
- Presque tout le temps, jouer le bras $A(t) = \arg \max_{k \in \{1,\dots,K\}} \text{kl-UCB}_k(t)$
- Utiliser le test BGLR pour détecter les ruptures sur le bras joué
- Si une rupture est détectée, effacer la mémoire des observations de *tous les* bras

Les indices kl-UCB

- $\tau_k(t)$ est le temps depuis le dernier redémarrage du bras k,
- $n_k(t)$ compte les sélections et $\widehat{\mu}_k(t)$ est la moyenne empirique des observations du bras k depuis $\tau_k(t)$,
- Soit $\text{kl-UCB}_k(t) = \max\left\{q \in [0,1] : n_k(t) \times \mathrm{kl}\left(\widehat{\mu}_k(t), q\right) \leq f(t - \tau_k(t))\right\}$
- $f(t) = \ln(t) + 3\ln(\ln(t))$ contrôle l'amplitude de la Borne Sup. de Confiance (UCB).

Deux détails sur notre algorithme

i) Comment est utilisé le test BGLR ? (paramètre δ)

Avec les observations $Z_1, \cdots, Z_n$ on utilise le test BGLR pour détecter une rupture avec un niveau de confiance δ quand

$$\sup_{1 \leq s \leq n-1} \left[s \times \mathrm{kl}\left(\widehat{Z}_{1:s}, \widehat{Z}_{1:n} \right) + (n-s) \times \mathrm{kl}\left(\widehat{Z}_{s+1:n}, \widehat{Z}_{1:n} \right) \right] \geq \beta(n, \delta)$$

Deux détails sur notre algorithme

i) Comment est utilisé le test BGLR ? (paramètre δ)

Avec les observations $Z_1, \cdots, Z_n$ on utilise le test BGLR pour détecter une rupture avec un niveau de confiance δ quand

$$\sup_{1 \leq s \leq n-1} \left[s \times \mathrm{kl}\left(\widehat{Z}_{1:s}, \widehat{Z}_{1:n} \right) + (n-s) \times \mathrm{kl}\left(\widehat{Z}_{s+1:n}, \widehat{Z}_{1:n} \right) \right] \geq \beta(n, \delta)$$

ii) Exploration forcée (paramètre α)

- On utilise une exploration forcée sur tous les bras...
  ie, en moyenne, le bras k est forcé à être joué au moins $T \times \alpha / K$ fois
- $\Longrightarrow$ dans le but de pouvoir détecter les ruptures sur tous les bras
- et pas seulement sur le bras joué par les indices kl-UCB

L'algorithme BGLR + kl-UCB

Data : *Paramètre du problème* : $T \in \mathbb{N}^*$, $K \in \mathbb{N}^*$

Data : *Paramètre* : $\alpha \in (0,1)$, $\delta > 0$ `// peut utiliser` T `et` Υ_T

Initialisation : $\forall k \in \{1, \dots, K\}$, $\tau_k = 0$ et $n_k = 0$

for $t = 1, 2, \dots, T$ **do**

L'algorithme BGLR + kl-UCB

Data : *Paramètre du problème* : $T \in \mathbb{N}^*$, $K \in \mathbb{N}^*$

Data : *Paramètre* : $\alpha \in (0,1)$, $\delta > 0$ `// peut utiliser` T `et` Υ_T

Initialisation : $\forall k \in \{1,\dots,K\}$, $\tau_k = 0$ et $n_k = 0$

for $t = 1, 2, \dots, T$ **do**

  if $t \mod \left\lfloor \frac{K}{\alpha} \right\rfloor \in \{1,\dots,K\}$ **then**

    $A(t) = t \mod \left\lfloor \frac{K}{\alpha} \right\rfloor$ `// exploration forcée`

  else

    $A(t) = \arg \max_{k \in \{1,\dots,K\}} \text{kl-UCB}_k(t)$ `// plus grand indice UCB`

L'algorithme BGLR + kl-UCB

Data : *Paramètre du problème* : $T \in \mathbb{N}^*$, $K \in \mathbb{N}^*$

Data : *Paramètre* : $\alpha \in (0,1)$, $\delta > 0$ `// peut utiliser` T `et` Υ_T

Initialisation : $\forall k \in \{1,\dots,K\}$, $\tau_k = 0$ et $n_k = 0$

for $t = 1, 2, \dots, T$ **do**

  if $t \mod \lfloor \frac{K}{\alpha} \rfloor \in \{1,\dots,K\}$ **then**

    $A(t) = t \mod \lfloor \frac{K}{\alpha} \rfloor$ `// exploration forcée`

  else

    $A(t) = \arg \max_{k \in \{1,\dots,K\}} \text{kl-UCB}_k(t)$ `// plus grand indice UCB`

  Joue le bras $k = A(t)$, et incrémente le compteur $n_{A(t)} = n_{A(t)} + 1$

  Observe la récompense $X_{A(t),t}$, et la stocker $Z_{A(t),n_{A(t)}} = X_{A(t),t}$

L'algorithme BGLR + kl-UCB

Data : *Paramètre du problème* : $T \in \mathbb{N}^*$, $K \in \mathbb{N}^*$
Data : *Paramètre* : $\alpha \in (0,1)$, $\delta > 0$ `// peut utiliser` T `et` Υ_T
Initialisation : $\forall k \in \{1,\dots,K\}$, $\tau_k = 0$ et $n_k = 0$
for $t = 1, 2, \dots, T$ **do**
  if $t \mod \lfloor \frac{K}{\alpha} \rfloor \in \{1,\dots,K\}$ **then**
    $A(t) = t \mod \lfloor \frac{K}{\alpha} \rfloor$ `// exploration forcée`
  else
    $A(t) = \arg\max_{k \in \{1,\dots,K\}} \text{kl-UCB}_k(t)$ `// plus grand indice UCB`
  Joue le bras $k = A(t)$, et incrémente le compteur $n_{A(t)} = n_{A(t)} + 1$
  Observe la récompense $X_{A(t),t}$, et la stocker $Z_{A(t),n_{A(t)}} = X_{A(t),t}$
  if $\text{BGLRT}_\delta(Z_{A(t),1}, \cdots, Z_{A(t),n_{A(t)}}) =$ *Vrai* **then**
    $\forall k, \tau_k = t$ **et** $n_k = 0$ `// efface la mémoire de tous les bras`
end

5. Simulations numériques

1. Problèmes de bandits multi-bras (stationnaires)
2. Problèmes de bandits multi-bras stationnaires par morceaux
3. Le test BGLR et ses propriétés à horizon fini
4. L'algorithme B-GLRT + klUCB
5. **Simulations numériques**

Simulations numériques

On considère trois problèmes avec

- $K = 3$ bras de Bernoulli
- $T = 5000$ étapes de temps (horizon fini)
- $\Upsilon_T = 4$ ruptures ($= 5$ séquences stationnaires)
  Les algorithmes peuvent utiliser cette connaissance de T et Υ_T
- 1000 simulations indépendantes, on affiche le regret moyen

Simulations numériques

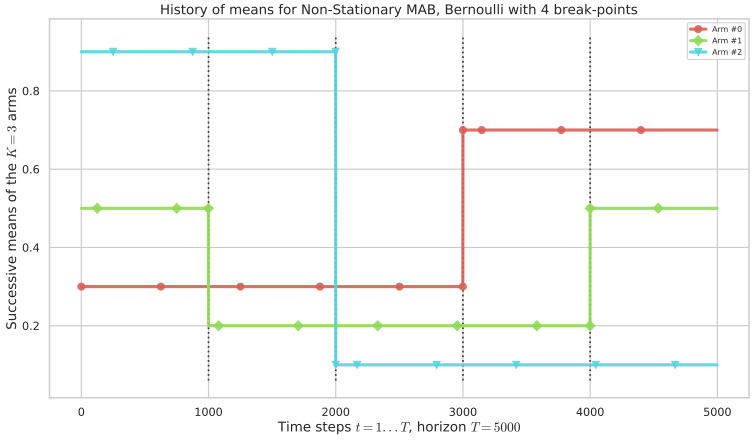
On considère trois problèmes avec

- $K = 3$ bras de Bernoulli
- $T = 5000$ étapes de temps (horizon fini)
- $\Upsilon_T = 4$ ruptures ($= 5$ séquences stationnaires)
  Les algorithmes peuvent utiliser cette connaissance de T et Υ_T
- 1000 simulations indépendantes, on affiche le regret moyen

Référence

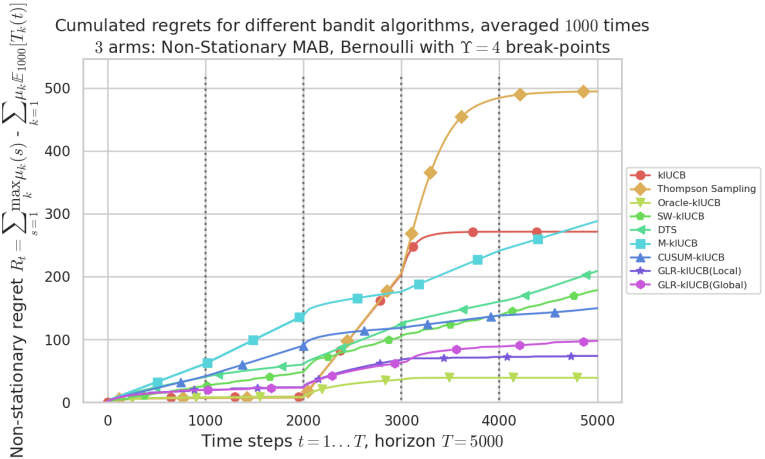
- On utilise ma bibliothèque open-source en Python pour la simulation de problèmes de bandits, **SMPyBandits**
  $\hookrightarrow$ Publiée en ligne à `SMPyBandits.GitHub.io`
- Plus d'expériences dans la version longue de l'article !
  $\hookrightarrow$ sur HAL-02006471 et arXiv:1902.01575

Problème 1 : seulement des changements locaux



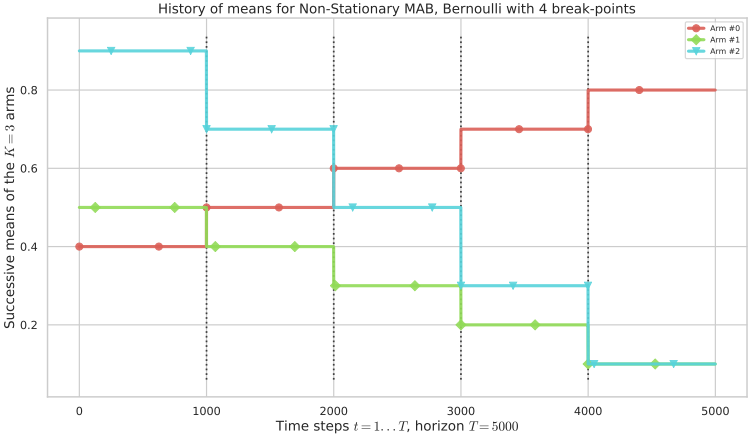
On affiche les moyennes : $\mu_1(t)$, $\mu_2(t)$, $\mu_3(t)$.

Résultats pour le problème 1

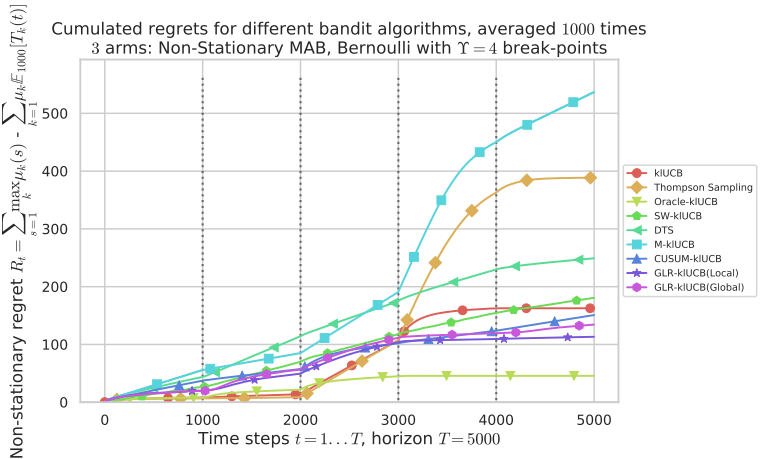


$\Longrightarrow$ BGLR atteint les meilleures performances parmi les algorithmes non-oracles 😎 !

Problème 2 : seulement des changements globaux

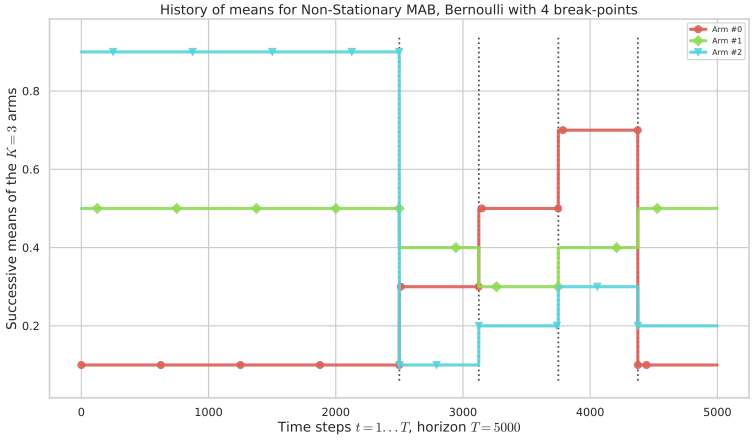


Résultats pour le problème 2

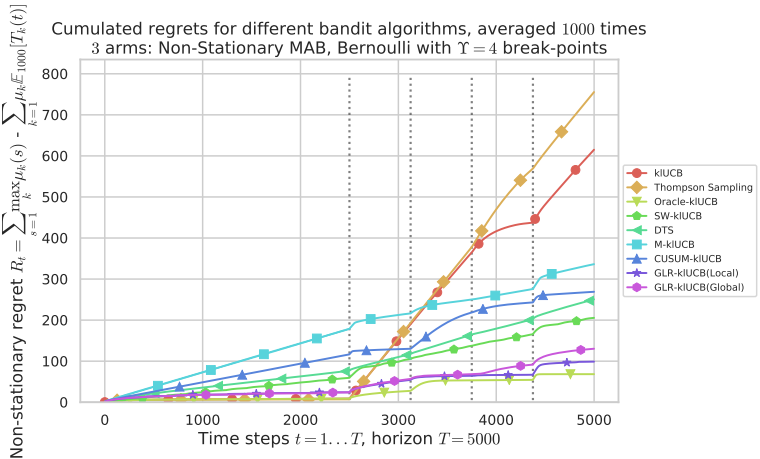


$\Longrightarrow$ BGLR atteint encore les meilleures performances 😎 !

Pb 3 : longueurs non uniformes des séquences



Résultats pour le problème 3



$\Longrightarrow$ BGLR atteint les meilleures performances parmi les algorithmes non-oracles 😎 !

Interprétation des simulations

Conclusions pour le regret

- En pratique, on vérifie que le test BGLR est efficace 😎 :
  - il a une faible probabilité de fausse alarme,
  - il a un faible délai si les séquences stationnaires sont assez longues.

  Et cela est aussi le cas hors des hypothèses de notre analyse
- Utiliser les indices kl-UCB donnent une bonne performance 😎

$\Longrightarrow$ Notre algorithme (test BGLR + kl-UCB) est efficace
$\Longrightarrow$ On vérifie qu'il obtient des bonnes performances.

Résumé

On a présenté…

- Les problèmes de bandits multi-bras stationnaires ou stationnaires par morceaux
- Le très efficace test généralisé de rapport de vraisemblances de Bernoulli (T-BGLR) 😎
  - pour détecter les ruptures sans fausse alarme et faible délai
  - pour des données de Bernoulli, et aussi des données sous-Bernoulli (n'importe quelle distribution bornée !),
  - et n'a *pas* besoin de connaître l'amplitude des ruptures
- On peut le combiner avec une stratégie de bandit efficace : BGLR + kl-UCB 😎
- Sa borne de regret est $R_T = \mathcal{O}(K\sqrt{T\Upsilon_T \log(T)})$ 😎 (état de l'art)
- Notre algorithme est compétitif avec le reste de l'état de l'art sur nos simulations numériques😎

Conclusion

Merci de votre attention. 

Questions & Discussion ?

Si besoin : `Lilian.Besson@CentraleSupelec.fr`