

# Apprentissage actif par modification de la politique de décision courante

Raphael Fonteneau<sup>1</sup>, Susan Murphy<sup>2</sup>, Louis Wehenkel<sup>1</sup>, Damien Ernst<sup>1</sup>

<sup>1</sup> Département d'Electricité, Electronique et Informatique, Université de Liège  
Liège, BELGIQUE  
raphael.fonteneau@ulg.ac.be, l.wehenkel@ulg.ac.be et dernst@ulg.ac.be

<sup>2</sup> Département de Statistiques, Université du Michigan  
Ann Arbor, MI, USA  
samurphy@umich.edu

**Résumé** : Cet article propose une stratégie de génération d'information dans le contexte de l'apprentissage par renforcement en mode "batch". Cette stratégie repose sur l'idée que les expériences susceptibles de mener à une modification de la politique de décision courante sont particulièrement informatives. Etant donné *a priori* un algorithme d'inférence de politiques de décision ainsi qu'un modèle prédictif du système, une expérience est réalisée si, étant donné le modèle prédictif, cette expérience mène à l'apprentissage d'une politique de décision différente. La stratégie est testée sur un problème-jouet pour lequel des résultats prometteurs sont obtenus.

**Mots-clés** : Apprentissage par renforcement, Apprentissage actif, Contrôle Optimal

## 1 Introduction

De nombreux problèmes de décision dans les domaines de l'ingénierie (Riedmiller (2005)), de la finance (Ingersoll (1987)), de la médecine (Murphy (2003, 2005)) ou de l'intelligence artificielle (Sutton & Barto (1998)) peuvent être formalisés comme des problèmes de contrôle optimal, dont l'objectif est de déterminer une politique de décision menant à l'optimisation d'un critère numérique. Souvent, ces problèmes sont abordés avec peu de connaissances sur la dynamique du système et la fonction de récompense qui les définissent.

Différentes approches ont déjà été proposées pour calculer des solutions approchées à ces problèmes dans le cas où les informations disponibles sont données sous forme d'un ensemble de transitions du système. Chacune de ces transitions est constituée d'un état, d'une décision prise dans cet état, de la valeur de la fonction de récompense et de la dynamique associées à ce couple état-décision. En particulier, une branche issue de l'apprentissage par renforcement (RL, de l'anglais Reinforcement Learning) - dont le but initial était la mise au point d'agents intelligents autonomes - aborde spécifiquement ce problème, que l'on désigne par BMRL par la suite (de l'anglais Batch Mode RL).

Etant donné un algorithme de type BMRL (c'est-à-dire capable d'apprendre une politique de décision dans un contexte BMRL), on s'intéresse dans cet article au problème de la génération d'ensembles de transitions à partir desquels l'algorithme BMRL puisse apprendre des politiques de décision performantes. La stratégie proposée fait appel à un modèle prédictif (PM, de l'anglais Predictive Model) permettant d'estimer, à partir des transitions déjà disponibles, la dynamique du système et la fonction de récompense en tout point de l'espace état-décision. Le choix d'échantillonner une transition du système en un couple état-décision se fait si, considérant la valeur prédite par le modèle PM en ce couple, on observe une modification de la politique de décision calculée par l'algorithme BMRL. En pratique, cette stratégie consiste donc à chercher un couple état-décision pour lequel on prédit une modification de la politique de décision courante.

Cette approche est motivée par deux constatations : d'une part, si l'ajout d'une nouvelle transition dans l'ensemble des transitions disponibles provoque une modification du résultat calculé par l'algorithme BMRL, alors cette transition est très probablement informative ; d'autre part, la mise

au point d'un modèle PM à partir des données disponibles peut se faire simplement pour un grand nombre de problèmes. Partant de ces deux constatations, la stratégie développée dans cet article (i) explore de manière itérative un ensemble de couples état-décision, (ii) calcule pour chacun de ces couples la valeur prédite de la transition en utilisant le modèle PM, et (iii) analyse l'influence de la transition prédite ajoutée aux transitions déjà disponibles sur la solution calculée par l'algorithme BMRL. Le résultat de cette analyse est utilisé afin de (iv) sélectionner un couple état-décision pour lequel une modification de la politique de décision calculée par l'algorithme BMRL est prédite.

Cette approche, ainsi que le contexte dans lequel elle s'inscrit, sont détaillées dans les sections 2, 3 et 4. Des résultats de simulations obtenus sur le problème-jouet "car-on-the-hill" sont présentés en section 5. La section 6 propose une discussion de travaux connexes puis la section 7 conclut et suggère quelques perspectives d'amélioration.

## 2 Formalisation du problème

On considère un système déterministe à temps discret dont la dynamique stationnaire est donnée par l'équation

$$x_{t+1} = f(x_t, u_t) \quad t = 0, 1, \dots, T-1,$$

où, pour tout  $t \in \{0, \dots, T-1\}$ , l'état  $x_t$  est un élément d'un espace d'état normé  $(\mathcal{X}, \|\cdot\|_{\mathcal{X}})$  et  $u_t$  est un élément d'un espace de décision fini  $\mathcal{U} = \{d^1, \dots, d^m\}$  avec  $m \in \mathbb{N}_0$ .  $T \in \mathbb{N}_0$  désigne l'horizon d'optimisation supposé fini. Une récompense instantanée

$$r_t = \rho(x_t, u_t) \in \mathbb{R}$$

est associée à une décision  $u_t \in \mathcal{U}$  prise dans un état  $x_t \in \mathcal{X}$ . L'état initial du système  $x_0 \in \mathcal{X}$  est supposé connu. Etant donnée une séquence de décisions  $\mathbf{u} = (u_0, \dots, u_{T-1}) \in \mathcal{U}^T$ , on définit le retour  $J^{\mathbf{u}}(x_0)$  de la séquence  $\mathbf{u}$  à partir de  $x_0$  :

$$\forall \mathbf{u} \in \mathcal{U}^T, J^{\mathbf{u}}(x_0) = \sum_{t=0}^{T-1} \rho(x_t, u_t)$$

avec  $x_{t+1} = f(x_t, u_t), \forall t \in \{0, \dots, T-1\}$ . On note  $J^*(x_0)$  la valeur maximale de  $J^{\mathbf{u}}(x_0)$  :

$$J^*(x_0) = \max_{\mathbf{u} \in \mathcal{U}^T} J^{\mathbf{u}}(x_0).$$

Une séquence de décisions  $\mathbf{u}^*$  est optimale si

$$J^{\mathbf{u}^*}(x_0) = J^*(x_0).$$

On appelle "transition du système" un quadruplet

$$(x, u, \rho(x, u), f(x, u)) \in \mathcal{X} \times \mathcal{U} \times \mathbb{R} \times \mathcal{X}$$

qui rassemble les valeurs des fonctions  $f$  et  $\rho$  en un couple  $(x, u)$  de l'espace conjoint  $\mathcal{X} \times \mathcal{U}$ . Les algorithmes BMRL (Ormoneit & Sen (2002); Ernst *et al.* (2005); Riedmiller (2005)) ont été introduits afin d'inférer des lois de contrôle quasi-optimales à partir d'un ensemble de transitions du système

$$\mathcal{F}_n = \{(x^l, u^l, r^l, y^l)\}_{l=1}^n$$

où  $r^l = \rho(x^l, u^l)$  et  $y^l = f(x^l, u^l)$ . Dans la suite de cet article, on désigne par *BMRL* un algorithme générique de type BMRL et on note  $BMRL(\mathcal{F}_n, x_0)$  la politique de décision calculée par cet algorithme.

Cet article propose une stratégie d'échantillonnage dont l'objectif est d'acquérir un ensemble de transitions  $\mathcal{F}_n$  de taille maximale  $N_{\max} \in \mathbb{N}$  (c'est à dire  $n \leq N_{\max}$ ), à partir duquel une politique de décision de qualité  $\tilde{\mathbf{u}}_{\mathcal{F}_n}^* \in \mathcal{U}^T$  puisse être apprise par *BMRL*, c'est à dire telle que  $J^{\tilde{\mathbf{u}}_{\mathcal{F}_n}^*}(x_0)$  soit aussi proche que possible de  $J^*(x_0)$ .

### 3 Stratégie d'échantillonnage

Cette section propose une première implémentation de l'approche basée sur la modification de la politique de décision courante présentée en section 1 afin de traiter le problème formalisé en section 2.

Etant donné un algorithme *BMRL*, un modèle prédictif *PM*, et une suite d'entiers  $(L_n)_n$ , on procède itérativement pour chaque  $n < N_{\max}$  :

- A partir de l'ensemble  $\mathcal{F}_n = \{(x^l, u^l, r^l, y^l)\}_{l=1}^n$  des transitions préalablement générées, on calcule une politique de décision

$$\tilde{\mathbf{u}}_{\mathcal{F}_n}^* = \text{BMRL}(\mathcal{F}_n, x_0) \quad ;$$

- On tire au hasard un couple  $(x, u) \in \mathcal{X} \times \mathcal{U}$  selon une distribution uniforme  $p_{\mathcal{X} \times \mathcal{U}}(\cdot)$  sur l'espace  $\mathcal{X} \times \mathcal{U}$  ;
- A partir de  $\mathcal{F}_n$  et du modèle *PM*, on calcule une “transition prédite” :

$$(x, u, \hat{r}_{\mathcal{F}_n}(x, u), \hat{y}_{\mathcal{F}_n}(x, u)) = \text{PM}(\mathcal{F}_n, x, u)$$

et on construit l'“ensemble prédit” :

$$\hat{\mathcal{F}}_{n+1}(x, u) = \mathcal{F}_n \cup \{(x, u, \hat{r}_{\mathcal{F}_n}(x, u), \hat{y}_{\mathcal{F}_n}(x, u))\} \quad ,$$

que l'on utilise pour calculer une “politique de décision prédite” :

$$\hat{\mathbf{u}}_{\hat{\mathcal{F}}_{n+1}(\mathbf{x}, \mathbf{u})}^* = \text{BMRL}(\hat{\mathcal{F}}_{n+1}(x, u), x_0) \quad ;$$

- Si  $\hat{\mathbf{u}}_{\hat{\mathcal{F}}_{n+1}(\mathbf{x}, \mathbf{u})}^* \neq \tilde{\mathbf{u}}_{\mathcal{F}_n}^*$ , on considère que  $f$  et  $\rho$  méritent d'être échantillonnées en  $(x, u)$ , ce que l'on réalise ; on obtient  $(x^{n+1}, u^{n+1}, r^{n+1}, y^{n+1})$  avec  $x^{n+1} = x$ ,  $u^{n+1} = u$ ,  $r^{n+1} = \rho(x, u)$  et  $y^{n+1} = f(x, u)$ , et on ajoute cette nouvelle transition à l'ensemble courant :

$$\mathcal{F}_{n+1} = \mathcal{F}_n \cup \{(x^{n+1}, u^{n+1}, r^{n+1}, y^{n+1})\} \quad ;$$

- Si  $\hat{\mathbf{u}}_{\hat{\mathcal{F}}_{n+1}(\mathbf{x}, \mathbf{u})}^* = \tilde{\mathbf{u}}_{\mathcal{F}_n}^*$ , on tire un autre couple  $(x', u')$  selon  $p_{\mathcal{X} \times \mathcal{U}}(\cdot)$  et on itère le processus ;
- Si  $L_n$  couples ont été testés sans mener à une modification de la politique de décision courante, on tire un couple  $(x^{n+1}, u^{n+1})$  au hasard selon  $p_{\mathcal{X} \times \mathcal{U}}(\cdot)$ , et on ajoute la transition  $(x^{n+1}, u^{n+1}, \rho(x^{n+1}, u^{n+1}), f(x^{n+1}, u^{n+1}))$  à  $\mathcal{F}_n$ .

**Influence de l'algorithme *BMRL* et du modèle *PM*.** Pour obtenir de bons résultats, il est nécessaire que les capacités d'inférence de *BMRL* soient les meilleures possibles. En général, les algorithmes *BMRL* utilisent des approximateurs de fonctions (Busoniu *et al.* (2010)) dont le but est de décrire soit le système lui-même ( $f$  et  $\rho$ ), soit des fonctions de valeur état-décision, soit des politiques de décision. Etant donné qu'ici, pour chaque itération de l'algorithme, la seule connaissance disponible sur le problème est un ensemble de transitions du système, on suggère d'utiliser un algorithme *BMRL* faisant appel à des approximateurs non-paramétriques, comme des méthodes du type “plus proche(s) voisin(s)” ou à base d'arbres.

Le meilleur modèle prédictif *PM* envisageable est un algorithme qui, pour chaque couple état-décision  $(x, u)$ , renvoie une prédiction égale à  $(x, u, \rho(x, u), f(x, u))$ . Prédire avec précision  $\rho(x, u)$  et  $f(x, u)$  peut s'avérer difficile. On peut dès lors envisager de travailler avec des ensembles de prédictions, ce qui d'un côté augmente la probabilité de détecter un couple susceptible de mener à une modification de la politique de décision courante, mais de l'autre, augmente également la probabilité que le couple choisi ne mène à aucune modification réelle. Si des connaissances a priori sur les fonctions  $f$  et  $\rho$  sont disponibles, on peut construire des ensembles de transitions “compatibles” avec ces connaissances ainsi qu'avec les transitions précédemment acquises (voir par exemple (Fonteneau *et al.* (2011)) où des connaissances de continuité Lipschitzienne sont exploitées). Ces connaissances peuvent ainsi être utilisées pour augmenter la précision de *PM*.

**Influence de la suite  $(L_n)_n$ .** Chaque terme de la suite  $(L_n)_n$  définit le nombre maximal d'essais autorisés pour identifier une transition pour laquelle on prédit une modification de la politique de

décision lorsque  $n$  transitions ont déjà été collectées. La valeur du terme  $L_n$  devrait être choisie de manière à assurer que, à la  $n$ -ième itération, s'il existe un couple état-décision pour lequel la transition correspondante mène à une modification de la politique courante, alors ce couple devrait être identifié avec une grande probabilité. Il peut cependant arriver que, pour certaines itérations  $n$ , il n'existe pas de transition (prédite) menant à une modification de la politique de décision (prédite). Dans un tel cas,  $L_n$  essais seront tout de même réalisés, ce qui peut être gênant en termes de temps de calcul si  $L_n$  est grand. Le choix des valeurs des termes de la suite  $(L_n)_n$  résulte donc d'un compromis entre la volonté d'identifier avec grande probabilité les transitions menant à une modification de la politique de décision, et le besoin de limiter les temps de calcul lorsqu'il n'y a rien à identifier.

## 4 Implementation avec une méthode du plus proche voisin

Dans cette section, on introduit les algorithmes *BMRL* et *PM* utilisés pour illustrer notre stratégie d'échantillonnage dans les expériences détaillées en section 5. L'algorithme *BMRL* fonctionne en approximant les fonctions  $f$  et  $\rho$  à partir des transitions disponibles puis en résolvant de manière exacte le problème de contrôle optimal défini par ces fonctions approchées. Cet algorithme est détaillé en section 4.1. En section 4.2, on détaille l'algorithme *PM*, qui fonde ses prédictions sur les approximations de  $f$  et  $\rho$  utilisées par *BMRL*.

### 4.1 Choix de l'algorithme *BMRL*

**RL basé sur l'apprentissage d'un modèle.** Le RL basé sur l'apprentissage d'un modèle consiste à résoudre de manière approchée un problème de contrôle optimal en approximant les fonctions inconnues  $f$  et  $\rho$  et en résolvant le problème de contrôle optimal "approché" défini par les approximations de  $f$  et  $\rho$ . Les valeurs  $y^l$  (resp.  $r^l$ ) de la fonction  $f$  (resp.  $\rho$ ) en  $(x^l, u^l)$ ,  $l = 1 \dots n$  sont utilisées pour apprendre une fonction  $\tilde{f}_{\mathcal{F}_n}$  (resp.  $\tilde{\rho}_{\mathcal{F}_n}$ ) définie sur l'espace  $\mathcal{X} \times \mathcal{U}$ . Le problème de contrôle optimal approché défini par les fonctions  $\tilde{f}_{\mathcal{F}_n}$  et  $\tilde{\rho}_{\mathcal{F}_n}$  est résolu et sa solution est utilisée comme solution approchée du problème de contrôle optimal défini par les "vraies" fonctions  $f$  et  $\rho$ .

Etant donnée une séquence de décisions  $\mathbf{u} \in \mathcal{U}^T$  et un algorithme de type BMRL par apprentissage de modèle, on note  $\tilde{J}_{\mathcal{F}_n}^{\mathbf{u}}(x_0)$  le retour approché de la séquence de décisions  $\mathbf{u}$ , c'est à dire le retour obtenu en considérant les approximations  $\tilde{f}_{\mathcal{F}_n}$  et  $\tilde{\rho}_{\mathcal{F}_n}$  :

$$\forall \mathbf{u} \in \mathcal{U}^T, \tilde{J}_{\mathcal{F}_n}^{\mathbf{u}}(x_0) = \sum_{t=0}^{T-1} \tilde{\rho}_{\mathcal{F}_n}(\tilde{x}_t, u_t)$$

avec

$$\tilde{x}_{t+1} = \tilde{f}_{\mathcal{F}_n}(\tilde{x}_t, u_t), \quad \forall t \in \{0, \dots, T-1\}$$

et

$$\tilde{x}_0 = x_0.$$

On note  $\tilde{J}_{\mathcal{F}_n}^*(x_0)$  le retour approché maximal au départ de l'état initial  $x_0 \in \mathcal{X}$  et selon les approximations  $\tilde{f}_{\mathcal{F}_n}$  et  $\tilde{\rho}_{\mathcal{F}_n}$  :

$$\tilde{J}_{\mathcal{F}_n}^*(x_0) = \max_{\mathbf{u} \in \mathcal{U}^T} \tilde{J}_{\mathcal{F}_n}^{\mathbf{u}}(x_0).$$

En utilisant ces notations, les algorithmes BMRL par apprentissage de modèle calculent une séquence de décisions  $\tilde{\mathbf{u}}_{\mathcal{F}_n}^* \in \mathcal{U}^T$  telle que  $\tilde{J}_{\mathcal{F}_n}^{\tilde{\mathbf{u}}_{\mathcal{F}_n}^*}(x_0)$  soit le plus proche possible de (idéalement, égal à)  $\tilde{J}_{\mathcal{F}_n}^*(x_0)$ . Ces algorithmes supposent implicitement qu'une politique de décision destinée au modèle appris mène aussi à un retour élevé pour le vrai modèle.

**Partition de Voronoi.** On spécifie dans cette section l'algorithme BMRL par apprentissage de modèle utilisé par la suite dans les simulations. Cet algorithme approxime  $f$  et  $\rho$  en utilisant des fonctions constantes par morceaux sur une partition de type Voronoi (Aurenhammer (1991)) de

l'espace  $\mathcal{X} \times \mathcal{U}$  (ce qui correspond à une approximation du plus proche voisin). L'algorithme est noté VRL (pour Voronoi RL) dans la suite. Etant donné un état initial  $x_0 \in \mathcal{X}$ , l'algorithme VRL retourne une séquence de décisions en boucle ouverte correspondant à un "déplacement optimal" parmi les cellules de Voronoi.

Tout d'abord, on fait l'hypothèse que les couples de l'ensemble  $\{(x^l, u^l)\}_{l=1}^n$  donnés par  $\mathcal{F}_n$  sont distincts deux à deux :

$$\forall l, l' \in \{1, \dots, n\}, (x^l, u^l) = (x^{l'}, u^{l'}) \implies l = l' .$$

On fait également l'hypothèse que chaque décision de l'espace  $\mathcal{U}$  a été prise au moins une fois lors de la génération des transitions incluses dans  $\mathcal{F}_n$  :

$$\forall u \in \mathcal{U}, \exists l \in \{1, \dots, n\}, u^l = u .$$

Le modèle appris se base sur  $n$  cellules de Voronoi  $\{V^l\}_{l=1}^n$  définissant une partition de taille  $n$  de l'espace  $\mathcal{X} \times \mathcal{U}$ . La cellule  $V^l$  associée au couple  $(x^l, u^l)$  est définie comme l'ensemble des couples  $(x, u) \in \mathcal{X} \times \mathcal{U}$  tels que

$$(i) \quad u = u^l , \tag{1}$$

$$(ii) \quad l \in \arg \min_{l': u^{l'} = u} \left\{ \|x - x^{l'}\|_{\mathcal{X}} \right\} , \tag{2}$$

$$(iii) \quad l = \min_{l'} \left\{ l' \in \arg \min_{l': u^{l'} = u} \left\{ \|x - x^{l'}\|_{\mathcal{X}} \right\} \right\} . \tag{3}$$

$\{V^l\}_{l=1}^n$  forme bien une partition de  $\mathcal{X} \times \mathcal{U}$  puisque l'on peut aisément vérifier que chaque couple  $(x, u) \in \mathcal{X} \times \mathcal{U}$  appartient à une et à une seule cellule de Voronoi. La fonction  $f$  (resp.  $\rho$ ) est approximée par une fonction constante par morceaux  $\tilde{f}_{\mathcal{F}_n}$  (resp.  $\tilde{\rho}_{\mathcal{F}_n}$ ) de la manière suivante :

$$\begin{aligned} \forall l \in \{1, \dots, n\}, \forall (x, u) \in V^l, \quad \tilde{f}_{\mathcal{F}_n}(x, u) &= y^l, \\ \tilde{\rho}_{\mathcal{F}_n}(x, u) &= r^l . \end{aligned}$$

A partir de  $\tilde{f}_{\mathcal{F}_n}$  et  $\tilde{\rho}_{\mathcal{F}_n}$ , on définit une suite finie de fonctions de valeur approchées  $(\tilde{Q}_{T-t}^*)_{t=0}^{T-1}$  comme suit :  $\forall t \in \{0, \dots, T-1\}, \forall (x, u) \in \mathcal{X} \times \mathcal{U}$ ,

$$\tilde{Q}_{T-t}^*(x, u) = \tilde{\rho}_{\mathcal{F}_n}(x, u) + \arg \max_{u' \in \mathcal{U}} \tilde{Q}_{T-t-1}^*(\tilde{f}_{\mathcal{F}_n}(x, u), u') ,$$

avec

$$Q_1^*(x, u) = \tilde{\rho}_{\mathcal{F}_n}(x, u), \quad \forall (x, u) \in \mathcal{X} \times \mathcal{U} .$$

A partir de la suite de fonctions  $(\tilde{Q}_{T-t}^*)_{t=0}^{T-1}$ , on calcule une politique de décision en boucle ouverte

$$\tilde{\mathbf{u}}_{\mathcal{F}_n}^* = (\tilde{u}_{\mathcal{F}_n,0}^*, \dots, \tilde{u}_{\mathcal{F}_n,T-1}^*) \in \mathcal{U}^T$$

solution du problème de contrôle optimal approché, c'est à dire telle que

$$\tilde{J}_{\mathcal{F}_n}^{\tilde{\mathbf{u}}_{\mathcal{F}_n}^*}(x_0) = \tilde{J}_{\mathcal{F}_n}^*(x_0) ,$$

de la manière suivante :

$$\tilde{u}_{\mathcal{F}_n,0}^* \in \arg \max_{u' \in \mathcal{U}} \tilde{Q}_T^*(\tilde{x}_0^*, u') ,$$

et,  $\forall t \in \{0, \dots, T-2\}$ ,

$$\tilde{u}_{\mathcal{F}_n,t+1}^* \in \arg \max_{u' \in \mathcal{U}} \tilde{Q}_{T-(t+1)}^*(\tilde{f}_{\mathcal{F}_n}(\tilde{x}_t^*, \tilde{u}_{\mathcal{F}_n,t}^*), u')$$

où

$$\tilde{x}_0^* = x_0$$

**Entrées :** un état initial  $x_0 \in \mathcal{X}$ , un ensemble de transitions  $\mathcal{F}_n = \{(x^l, u^l, r^l, y^l)\}_{l=1}^n$  ;

**Sorties :** une séquence de décisions  $\tilde{\mathbf{u}}_{\mathcal{F}_n}^*$  et  $\tilde{J}_{\mathcal{F}_n}^*(x_0)$  ;

**Initialisation :**  
 Créer une matrice  $n \times m$   $V$  telle que  $V(i, j)$  contient l'indice de la cellule de Voronoi (CV) où  $(\tilde{f}_{\mathcal{F}_n}(x^i, u^i), d^j)$  se trouve ;  
**for**  $i = 1$  **to**  $n$  **do**  
    $Q_{1,i} \leftarrow r^i$  ;  
**end for**

**Algorithme :**  
**for**  $t = T - 2$  **to**  $0$  **do**  
   **for**  $i = 1$  **to**  $n$  **do**  
    $l \leftarrow \arg \max_{l' \in \{1, \dots, m\}} \{Q_{T-t-1, V(i, l')}\}$  ;  
    $Q_{T-t, i} \leftarrow r^i + Q_{T-t-1, V(i, l)}$  ;  
   **end for**  
**end for**  
 $l \leftarrow \arg \max_{l' \in \{1, \dots, m\}} Q_{T, i'}$  où  $i'$  désigne l'indice de la CV où  $(x_0, d^{l'})$  se trouve ;  
 $l_0^* \leftarrow$  indice de la CV où  $(x_0, d^l)$  se trouve ;  
 $\tilde{J}_{\mathcal{F}_n}^*(x_0) \leftarrow Q_{T, l_0^*}$  ;  
 $i \leftarrow l_0^*$  ;  
 $\tilde{u}_{\mathcal{F}_n, 0}^* \leftarrow u^{l_0^*}$  ;  
**for**  $t = 0$  **to**  $T - 2$  **do**  
    $l_{t+1}^* \leftarrow \arg \max_{l' \in \{1, \dots, m\}} \{Q_{T-t-1, V(i, l')}\}$  ;  
    $\tilde{u}_{\mathcal{F}_n, t+1}^* \leftarrow d^{l_{t+1}^*}$  ;  
    $i \leftarrow V(i, l_{t+1}^*)$  ;  
**end for**  
**return**  $\tilde{\mathbf{u}}_{\mathcal{F}_n}^* = (\tilde{u}_{\mathcal{F}_n, 0}^*, \dots, \tilde{u}_{\mathcal{F}_n, T-1}^*)$  ,  $\tilde{J}_{\mathcal{F}_n}^*(x_0)$ .

FIGURE 1 – L'algorithme VRL (de l'anglais Voronoi Reinforcement Learning).  $Q_{T-t, l}$  est la valeur prise par la fonction  $\tilde{Q}_{T-t}^*$  dans la cellule de Voronoi  $V^l$ .

et

$$\tilde{x}_{t+1}^* = \tilde{f}_{\mathcal{F}_n}(\tilde{x}_t^*, \tilde{u}_{\mathcal{F}_n, t}^*), \forall t \in \{0, \dots, T-1\}.$$

Toutes les fonctions de la suite  $(\tilde{Q}_{T-t}^*)_{t=0}^{T-1}$  sont constantes dans chaque cellule, ce qui permet d'extraire facilement la politique de décision  $\tilde{\mathbf{u}}_{\mathcal{F}_n}^*$  en utilisant un algorithme de type Viterbi dont la complexité est linéaire en fonction de  $n$ ,  $T$  et la cardinalité  $m$  de l'espace  $\mathcal{U}$ . Une version tabulaire de l'algorithme VRL est donnée en figure 1. D'autre part, l'algorithme VRL possède des propriétés de consistance lorsque les fonctions  $f$  et  $\rho$  sont Lipschitziennes et que la dispersion de l'ensemble de transitions  $\mathcal{F}_n$  converge vers 0 (Fonteneau & Ernst (2010)).

## 4.2 Choix de l'algorithme PM

L'algorithme *PM* utilisé dans les simulations fait appel aux fonctions  $\tilde{f}_{\mathcal{F}_n}$  et  $\tilde{\rho}_{\mathcal{F}_n}$  calculées par l'algorithme VRL. Etant donné un ensemble de transitions  $\mathcal{F}_n$  et un couple  $(x, u) \in \mathcal{X} \times \mathcal{U}$ , l'algorithme *PM* renvoie

$$(x, u, \hat{r}_{\mathcal{F}_n}(x, u), \hat{y}_{\mathcal{F}_n}(x, u)) = PM(\mathcal{F}_n, x, u)$$

tel que

$$\hat{r}_{\mathcal{F}_n}(x, u) = \tilde{\rho}_{\mathcal{F}_n}(x, u)$$

et

$$\hat{y}_{\mathcal{F}_n}(x, u) = \tilde{f}_{\mathcal{F}_n}(x, u) .$$

## 5 Résultats expérimentaux

Cette section illustre la stratégie d'échantillonnage décrite ci-dessus sur le problème-jouet “car-on-the-hill” (Ernst (2005)), un problème classique souvent utilisé pour tester les algorithmes d'apprentissage par renforcement.

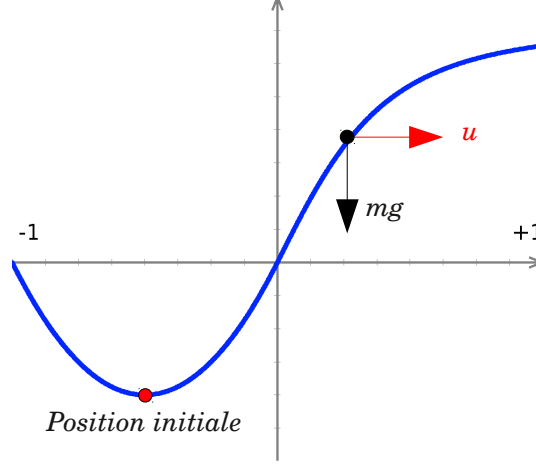


FIGURE 2 – Illustration du problème-jouet “car-on-the-hill”.

### 5.1 Présentation du problème jouet “car-on-the-hill”

Un point de masse unitaire - représentant un véhicule - doit être conduit au sommet d’une colline située à droite sur la figure 2 par application d’une force horizontale. Pour certains états initiaux du système, la puissance maximale du véhicule ne permet pas d’atteindre le sommet de la colline. Le véhicule doit donc grimper sur le flanc d’une autre colline située à gauche, puis la redescendre afin de prendre de la vitesse et atteindre le sommet de la colline de droite.

La dynamique du véhicule est donnée par l’équation différentielle :

$$\ddot{z} = \frac{1}{1 + \left(\frac{dH(z)}{dz}\right)^2} \left( \frac{u}{m_c} - g \frac{dH(z)}{dz} - \dot{z}^2 \frac{dH(z)}{dz} \frac{d^2H(z)}{dz^2} \right)$$

où  $z \in [-1, 1]$  est la position horizontale du véhicule (donnée en  $m$ ),  $\dot{z} \in [-3, 3]$  est la vitesse du véhicule (donnée en  $m/s$ ),  $u \in \{-4, 4\}$  est la force horizontale (donnée en  $N$ ),  $g = 9.81 m/s^2$  est la constante de gravité et  $H$  est le profil du terrain :

$$H(z) = \begin{cases} z^2 + z & \text{if } z < 0, \\ \frac{z}{\sqrt{1+5z^2}} & \text{if } z \geq 0. \end{cases}$$

La masse du véhicule vaut  $m_c = 1 kg$ . La durée d’un pas de temps est  $T_s = 0.1s$  et la dynamique à temps discret du véhicule  $f$  est obtenue par intégration de la dynamique à temps continu entre chaque pas de temps. L’espace de décision  $\mathcal{U}$  contient deux éléments :  $-4$  et  $4$ . Lorsque la position  $z$  ou la vitesse  $\dot{z}$  dépasse les bornes, le véhicule atteint un état absorbant dans lequel il reste indépendamment de la décision prise. Si  $z_{t+1} < -1$  ou si  $|\dot{z}_{t+1}| > 3$ , alors le véhicule atteint un état absorbant “perdant”  $s_{-1}$  et reçoit une récompense de  $-1$  à chaque pas de temps jusqu’à  $t = T - 1$ . Si  $z_{t+1} \geq 1$  et  $|\dot{z}_{t+1}| \leq 3$ , alors le véhicule atteint un état absorbant “gagnant”  $s_1$ , et reçoit une récompense de  $+1$  à chaque pas de temps jusqu’à  $t = T - 1$ . Les états absorbants  $s_{-1}$  et  $s_1$  sont supposés connus. L’espace d’état est donc égal à

$$\mathcal{X} = [-1, 1] \times [-3, 3] \cup \{s_1, s_{-1}\}.$$

L'objectif est de déterminer une séquence de décisions maximisant la somme des récompenses obtenues sur un horizon  $T = 20$  lorsque le véhicule démarre au creux de la vallée en  $x_0 = [-0.5, 0]$ . Une telle séquence permettra aussi de mener le véhicule au sommet de la colline en une durée minimale.

L'algorithme VRL détaillé en section 4.1 ne donne pas d'information sur la manière de gérer les états absorbants. Cela peut être fait en ajoutant à l'ensemble de transitions  $m \times n_{abs}$  "transitions artificielles", où  $n_{abs}$  désigne le nombre d'états absorbants du problème. Dans le cadre du problème "car-on-the-hill", cela se traduit par l'ajout de 4 transitions artificielles :

$$\{(s_1, 4, 1, s_1), (s_1, -4, 1, s_1), (s_{-1}, 4, -1, s_{-1}), (s_{-1}, -4, -1, s_{-1})\}.$$

La définition des cellules de Voronoi reste identique à celle donnée par les équations (1), (2) et (3) lorsque  $x^l$  n'est pas un état absorbant. Dans tous les autres cas, la norme  $\|\cdot\|_{\mathcal{X}}$  peut être (abusivement) étendue aux états absorbants de la manière suivante :

$$\|x - x^l\|_{\mathcal{X}} = \begin{cases} 0 & \text{si } x = x^l, \\ +\infty & \text{si } x \neq x^l. \end{cases}$$

## 5.2 Protocole expérimental

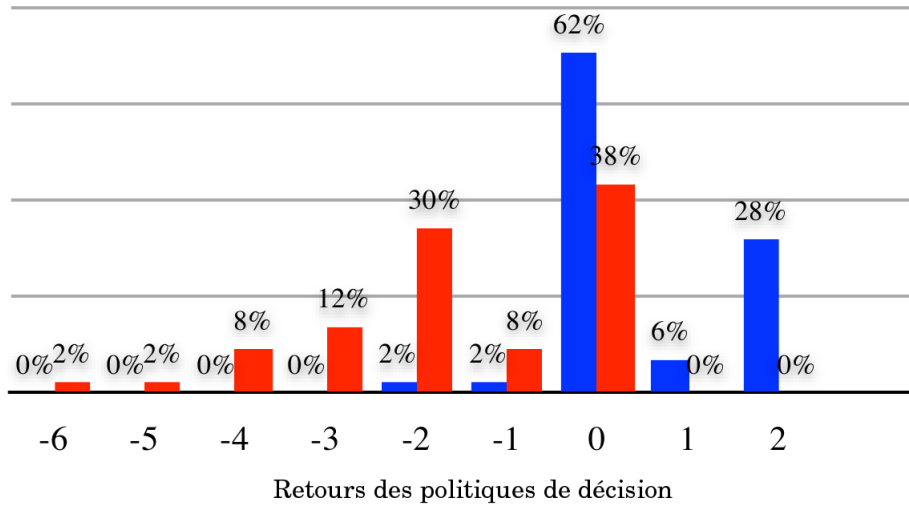


FIGURE 3 – Distribution des retours des politiques de décision calculées à partir de  $\mathcal{F}_{N_{\max}}^k$ ,  $k = 1 \dots q$  (en bleu, à gauche) et  $\mathcal{G}_{N_{\max}}^k$ ,  $k = 1 \dots q$  (en rouge, à droite).

Les performances de notre stratégie d'échantillonnage sont comparées avec les performances d'une stratégie d'échantillonnage uniforme. Pour ce faire, on teste  $q = 50$  fois notre stratégie d'échantillonnage, où chaque test  $k = 1 \dots q$  est initialisé avec un ensemble de transitions  $\mathcal{F}_m^k$  contenant  $m = 2$  transitions (une transition pour chaque décision) :

$$\forall k \in \{1, \dots, q\}, \mathcal{F}_m^k = \{(x_0, -4, \rho(x_0, -4), f(x_0, -4)), (x_0, +4, \rho(x_0, +4), f(x_0, +4))\}.$$

Notre stratégie d'échantillonnage est mise en oeuvre sur chacun des ensembles de transitions  $\mathcal{F}_m^k$ ,  $k = 1 \dots q$  jusqu'à ce que chacun contienne  $N_{\max} = 1000$  transitions. On obtient ainsi  $q$  suites finies d'ensembles de transitions, chaque suite contenant  $(N_{\max} - m + 1)$  termes :

$$\mathcal{F}_m^1, \mathcal{F}_{m+1}^1, \dots, \mathcal{F}_{N_{\max}}^1, \dots, \mathcal{F}_m^q, \mathcal{F}_{m+1}^q, \dots, \mathcal{F}_{N_{\max}}^q.$$



On génère également  $q$  suites finies d'ensembles de transitions contenant chacune  $(N_{\max} - m + 1)$  termes

$$\mathcal{G}_m^1, \mathcal{G}_{m+1}^1, \dots, \mathcal{G}_{N_{\max}}^1, \dots, \mathcal{G}_m^q, \mathcal{G}_{m+1}^q, \dots, \mathcal{G}_{N_{\max}}^q$$

où, pour chaque  $k = 1 \dots q$ , et pour chaque  $n = m \dots N_{\max} - 1$ , chaque ensemble  $\mathcal{G}_{n+1}^k$  est obtenu en ajoutant à  $\mathcal{G}_n^k$  une transition  $(x, u, \rho(x, u), f(x, u))$  telle que  $(x, u)$  est tiré selon  $p_{\mathcal{X} \times \mathcal{U}}(\cdot)$ . Les termes de la suite  $(L_n)_n$  utilisée pour ces simulations sont définis de la manière suivante :

$$\forall n \in \{m, \dots, N_{\max}\}, L_n = mn .$$

La distribution de probabilités  $p_{\mathcal{X} \times \mathcal{U}}(\cdot)$  est telle que la probabilité de tirer un couple état-décision  $(x, u)$  avec  $x = s_1$  ou  $x = s_{-1}$  est nulle, et uniforme ailleurs.

### 5.3 Analyse des résultats

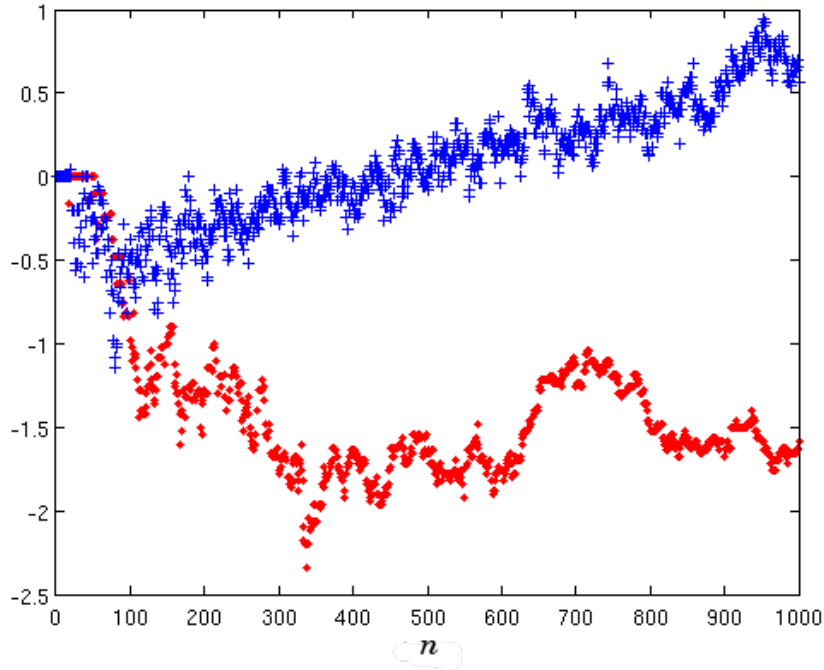


FIGURE 4 – Evolution des performances moyennes de notre stratégie d'échantillonnage  $\mathcal{M}(n)$  (croix bleues) comparée à l'évolution des performances moyennes d'un échantillonnage uniforme  $M_{unif}(n)$  (points rouges).

**Performances des politiques de décision calculées à partir des ensembles de  $N_{\max}$  transitions.** On calcule les retours des  $2q$  politiques de décision calculées par l'algorithme VRL à partir des ensembles finaux contenant  $N_{\max}$  transitions  $\mathcal{F}_{N_{\max}}^k$  et  $\mathcal{G}_{N_{\max}}^k$ ,  $k = 1 \dots q$ . Les résultats, exprimés en termes de distribution des retours des politiques de décision apprises, sont donnés en figure 3.

On observe que l'algorithme VRL parvient à calculer, pour 28% des ensembles de transitions obtenus à partir de notre stratégie d'échantillonnage, une politique de décision pour laquelle le retour vaut 2, alors qu'aucune politique de décision menant à un retour strictement positif n'est calculée à partir des bases de données générées par tirage uniforme. A titre informatif, il est nécessaire de générer des ensembles de 10000 transitions si l'on souhaite obtenir, avec un tirage uniforme, des performances équivalentes.

**Performances moyennes et distribution des retours des politiques de décision apprises.** Pour une cardinalité donnée  $n$  ( $m \leq n \leq N_{\max}$ ), on calcule la performance moyenne  $\mathcal{M}(n)$  des  $q$

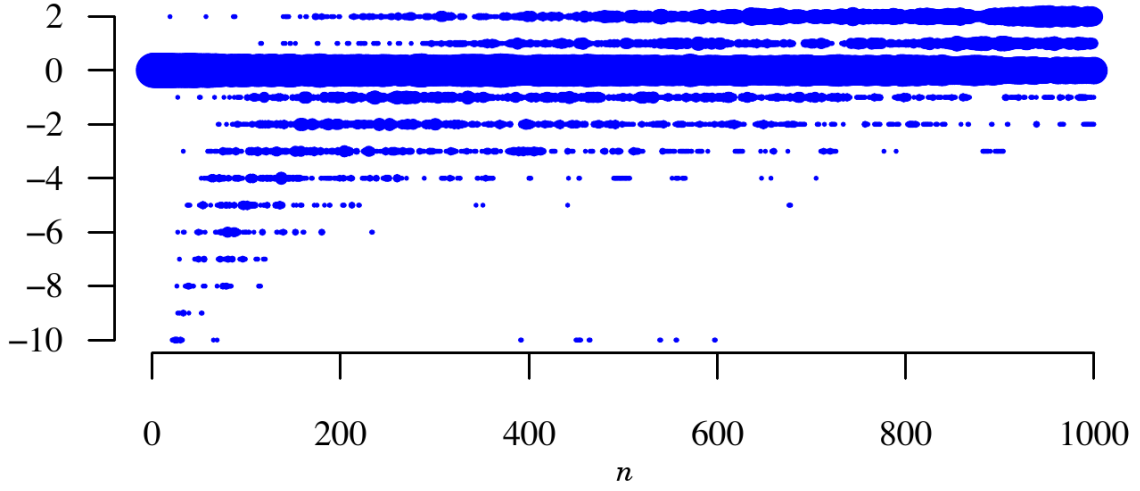


FIGURE 5 – Distribution des retours des politiques de décision  $\tilde{\mathbf{u}}_{\mathcal{F}_n^k}^*$ ,  $k=1 \dots q$ ,  $n = m \dots N_{\max}$ . Pour chaque valeur de  $n$ , la surface d'un disque correspondant à un retour  $r = -10 \dots 2$  est proportionnelle au nombre de politiques de l'ensemble  $\{\tilde{\mathbf{u}}_{\mathcal{F}_n^k}^*\}_{k=1}^q$  pour lesquelles le retour vaut  $r$ .

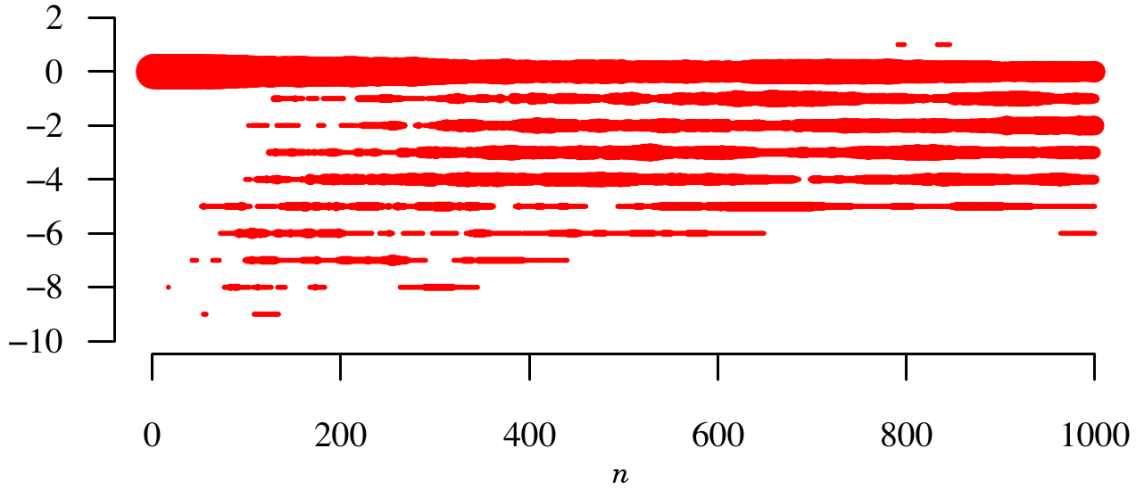


FIGURE 6 – Distribution des retours des politiques de décision  $\tilde{\mathbf{u}}_{\mathcal{G}_n^k}^*$ ,  $k=1 \dots q$ ,  $n = m \dots N_{\max}$ . Pour chaque valeur de  $n$ , la surface d'un disque correspondant à un retour  $r = -10 \dots 2$  est proportionnelle au nombre de politiques de l'ensemble  $\{\tilde{\mathbf{u}}_{\mathcal{G}_n^k}^*\}_{k=1}^q$  pour lesquelles le retour vaut  $r$ .

séquences de décisions  $\tilde{\mathbf{u}}_{\mathcal{F}_n^k}^*$ ,  $k=1 \dots q$  calculées par l'algorithme VRL à partir des ensembles de transitions  $\mathcal{F}_n^k$ ,  $k = 1 \dots q$  :

$$\mathcal{M}(n) = \frac{1}{q} \sum_{k=1}^q J^{\tilde{\mathbf{u}}_{\mathcal{F}_n^k}^*}(x_0) .$$

On calcule également la performance moyenne  $\mathcal{M}_{unif}(n)$  des  $q$  séquences de décisions  $\tilde{\mathbf{u}}_{\mathcal{G}_n^k}^*$ ,  $k = 1 \dots q$  calculées par l'algorithme VRL à partir des ensembles  $\mathcal{G}_n^k$ ,  $k = 1 \dots q$  obtenus par échan-

tillonnage uniforme :

$$\mathcal{M}_{unif}(n) = \frac{1}{q} \sum_{k=1}^q J^{\tilde{\mathbf{u}}_{\mathcal{G}_n^k}^*}(x_0) .$$

Les valeurs de  $\mathcal{M}(n)$  et  $\mathcal{M}_{unif}(n)$  pour  $n = m \dots N_{\max}$  sont comparées en figure 4. On donne également en figure 5 (resp. 6) la distribution des retours des politiques  $\tilde{\mathbf{u}}_{\mathcal{F}_n^k}^*, k = 1 \dots q, n = m \dots N_{\max}$  (resp.  $\tilde{\mathbf{u}}_{\mathcal{G}_n^k}^*, k = 1 \dots q, n = m \dots N_{\max}$ ).

On observe que, à partir de notre stratégie d'échantillonnage, des politiques menant à un retour de 2 sont apprises à partir d'ensembles contenant moins de 200 transitions. On remarque également qu'aucune politique menant à un retour de 2 n'a pu être apprise à partir des ensembles de transitions tirées uniformément  $\mathcal{G}_n^k, k = 1 \dots q, n = m \dots N_{\max}$ .

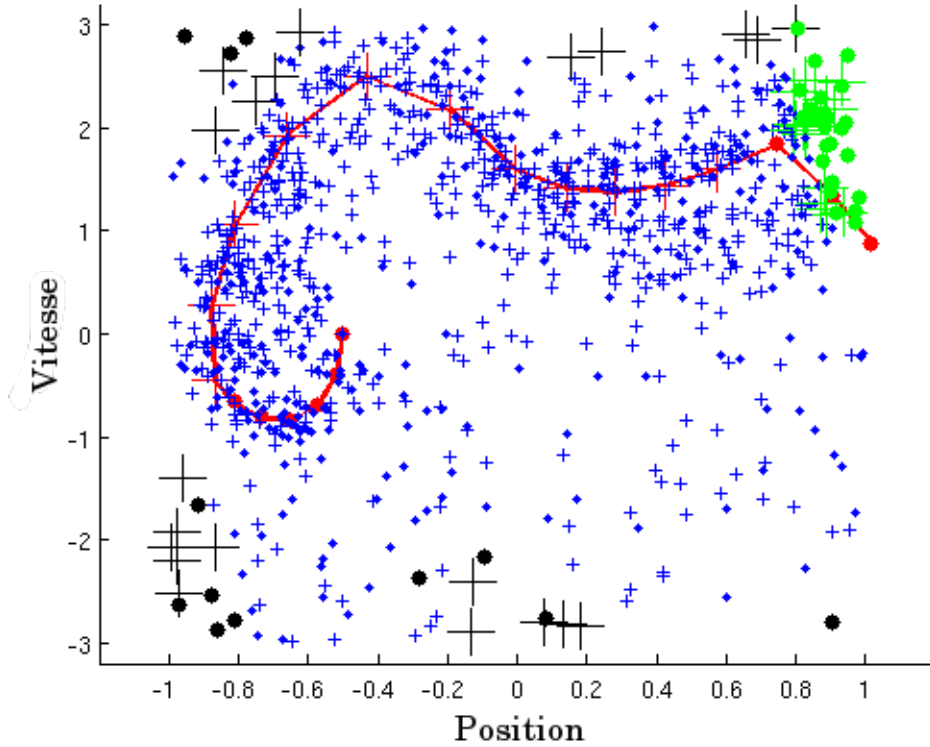


FIGURE 7 – Représentation de l'ensemble de transitions  $\mathcal{F}_{N_{\max}}^1$  (obtenu avec notre stratégie d'échantillonnage).

**Représentations de  $\mathcal{F}_{N_{\max}}^1$  et  $\mathcal{G}_{N_{\max}}^1$ .** On représente graphiquement les transitions contenues dans l'ensemble  $\mathcal{F}_{N_{\max}}^1$  (resp.  $\mathcal{G}_{N_{\max}}^1$ ) en figure 7 (resp. 8). Chaque transition  $(x^l, u^l, r^l, y^l)$  est représentée par un symbole situé en  $x^l = [z, \dot{z}]$ . Un signe '+' indique que  $u^l = +4$ , tandis qu'un signe '•' indique que  $u^l = -4$ . Le symbole est bleu si  $r^l = 0$ . Des symboles plus grands et coloriés en noir (vert) sont utilisés si  $r^l = -1$  ( $r^l = 1$ ). La courbe rouge représente la trajectoire du véhicule conduit selon la politique de décision  $\tilde{\mathbf{u}}_{\mathcal{F}_{N_{\max}}^1}^*$  (resp.  $\tilde{\mathbf{u}}_{\mathcal{G}_{N_{\max}}^1}^*$ ). On peut observer à la figure 7 que notre stratégie tend à échantillonner des transitions situées au voisinage de trajectoires ayant de bonnes performances.

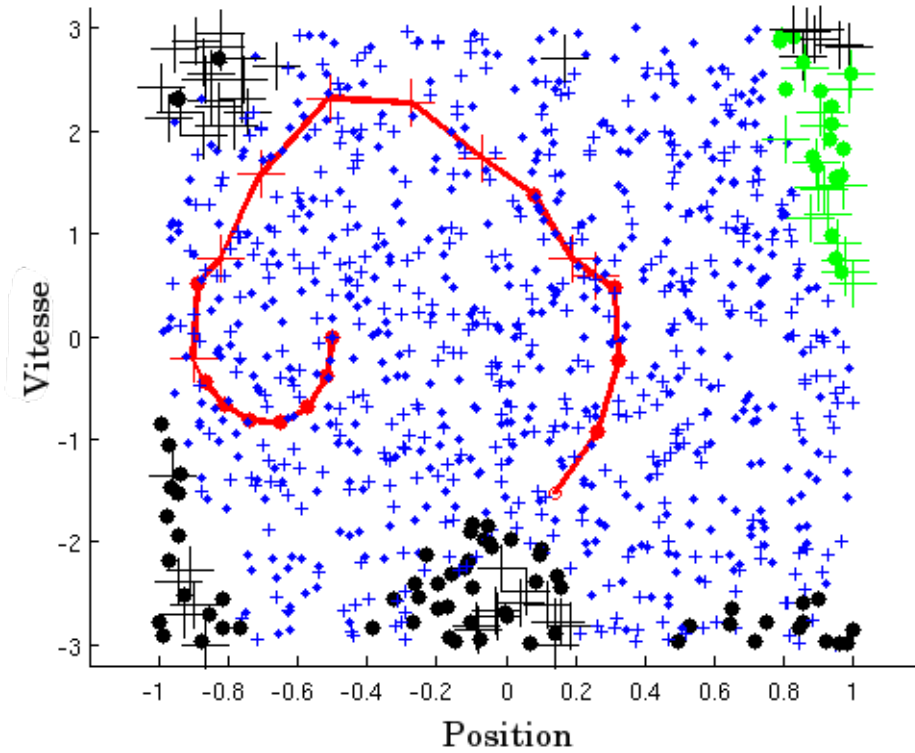


FIGURE 8 – Représentation de l'ensemble de transitions  $\mathcal{G}_{N_{\max}}^1$  (obtenu à partir de tirages uniformes).

## 6 Travaux connexes

Echantillonner de manière adéquate la dynamique et la fonction de récompense d'un système inconnu est un problème qui a déjà été abordé par de nombreux auteurs. L'approche développée dans (Ephsteyn *et al.* (2008)) est probablement celle qui se rapproche le plus de notre stratégie d'échantillonnage. Dans (Ephsteyn *et al.* (2008)), les auteurs proposent une stratégie itérative favorisant les zones de l'espace supposées influencer la politique de décision. Ces travaux sont menés dans un contexte stochastique, stationnaire et pour un espace d'état fini, alors que nous considérons ici des problèmes déterministes dans des espaces d'état continus.

Dans (Fonteneau *et al.* (2010)), une autre stratégie d'échantillonnage est proposée. Son principe est de calculer, pour un ensemble de politiques de décision, des bornes sur les retours des politiques afin de déterminer des zones d'échantillonnage supposées améliorer la précision de ces bornes. Cette approche est fondée sur des hypothèses de continuité Lipschitzienne de la dynamique et de la fonction de récompense, et sa mise en oeuvre implique la résolution d'un problème d'optimisation complexe.

La plupart des travaux issus du RL et abordant le problème de la génération d'échantillons informatifs ont préféré des solutions visant à contrôler un système de manière à générer des informations pouvant être utilisées pour augmenter les performances des politiques de décision, tout en gardant potentiellement un comportement générateur de bonnes performances. Une approche classique pour aborder ce dilemme entre exploration et exploitation (Auer (2003); Cohen *et al.* (2007)) est d'adopter une politique de type  $\epsilon$ -greedy qui prend l'initiative, avec une probabilité donnée, de prendre une décision différente de celle suggérée par la politique supposée optimale (Thrun (1992); Kaelbling (1993); Sutton & Barto (1998)). Ce problème a été particulièrement bien étudié dans le cas de problèmes ayant un état unique (Bubeck *et al.* (2009)).

Dans le domaine de la discrétisation adaptative pour la programmation dynamique, on peut

également trouver des travaux qui proposent des stratégies se rapprochant de notre approche. Dans ces travaux, l'espace état-décision est itérativement discrétisé de sorte à mener rapidement à une politique de décision optimale (voir par exemple Munos & Moore (2002)). Cependant, la complexité - en termes de temps de calcul - de notre stratégie ne lui permet pas d'être une stratégie d'échantillonnage adaptative performante.

Enfin, on peut également mentionner le fait qu'identifier un sous-ensemble de petite taille de transitions à partir duquel on puisse apprendre une bonne politique de décision est un problème qui a déjà été traité dans des contextes différents du nôtre. Par exemple, (Ernst (2005)) propose une approche pour extraire un sous-ensemble particulièrement informatif de transitions à partir de l'estimation d'erreurs d'approximation dans des équations de Bellman. Dans (Rachelson *et al.* (2011)), où aucune contrainte sur le nombre total de transitions générées n'est fixée, les auteurs se concentrent sur l'identification d'un petit sous-ensemble de transitions et parviennent à apprendre via un algorithme BMRL une politique de décision optimale basée sur moins de 20 transitions, mais au prix de centaines de milliers de transitions générées.

## 7 Conclusions

Cet article présente une stratégie d'échantillonnage itérative dont la finalité est de générer des ensembles de transitions informatifs dans le cas de la résolution de problèmes de contrôle optimaux déterministes à espace d'état continu. Fondée sur un mécanisme de prédiction, cette stratégie permet d'identifier des transitions susceptibles de mener à une modification de la politique de décision courante. Une implémentation de cette stratégie est réalisée en utilisant une méthode du plus proche voisin.

Des expériences réalisées sur le problème-jouet "car-on-the-hill" ont donné des résultats prometteurs. En particulier, la stratégie proposée se montre nettement plus efficace qu'une stratégie d'échantillonnage uniforme. Ces premiers résultats encouragent à étendre l'analyse de cette approche. En particulier, il serait intéressant d'étudier sous quelles conditions une modification de la politique causée par l'ajout d'une nouvelle transition correspond également à une amélioration réelle de la politique de décision. Il serait tout aussi intéressant de caractériser l'erreur de prédiction et son influence sur les détections de transitions qui n'apportent finalement pas de modifications de la politique courante. L'objectif de ces travaux serait d'identifier sous quelles hypothèses les ensembles de transitions ainsi générés pourraient converger vers des ensembles de transitions à partir desquels des politiques de décision (quasi-)optimales pourraient être apprises.

Enfin, la stratégie d'échantillonnage introduite dans cet article a été spécifiée et expérimentée dans un contexte déterministe, avec un espace de décision discret et fini. Il serait intéressant d'étudier comment mettre en oeuvre cette stratégie dans des contextes différents.

## Remerciements

Raphael Fonteneau est financé par le FRIA (Fonds pour la Formation à la Recherche dans l'Industrie et dans l'Agriculture). Damien Ernst est chercheur qualifié du FRS-FNRS (Fonds de la Recherche Scientifique). Les auteurs remercient également le réseau européen d'excellence PASCAL2, ainsi que les pôles d'attraction interuniversitaire BIOMAGNET et DYSCO, à l'initiative de la Politique Scientifique Fédérale de l'Etat Belge. Les auteurs remercient les financements NIH P50 DA10075 et R01 MH080015. Merci également à Bertrand Cornélusse. Le matériel scientifique reste sous la responsabilité des auteurs.

## Références

- AUER P. (2003). Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, **3**, 397 – 422.
- AURENHAMMER F. (1991). Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, **23**(3), 345–405.
- BUBECK S., MUNOS R., STOLTZ G. & SZEPESVÁRI C. (2009). Online optimization in X-armed bandits. In *Advances in Neural Information Processing Systems 21*, p. 201–208 : MIT Press.

- BUSONI L., BABUSKA R., DE SCHUTTER B. & ERNST D. (2010). *Reinforcement Learning and Dynamic Programming using Function Approximators*. Taylor & Francis CRC Press.
- COHEN J., MCCLURE S. & YU A. (2007). Should I stay or should I go? How the human brain manages the trade-off between exploitation and exploration. *Philosophical Transactions of the Royal Society B* 29, **362**(1481), 933–942.
- EPHSTEYN A., VOGEL A. & DEJONG G. (2008). Active reinforcement learning. In *Proceedings of the 25th international conference on Machine learning (ICML 2008)*, volume 307.
- ERNST D. (2005). Selecting concise sets of samples for a reinforcement learning agent. In *Proceedings of the Third International Conference on Computational Intelligence, Robotics and Autonomous Systems (CIRAS 2005)*, Singapore.
- ERNST D., GEURTS P. & WEHENKEL L. (2005). Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, **6**, 503–556.
- FONTENEAU R. & ERNST D. (2010). Voronoi model learning for batch mode reinforcement learning. *Technical Report, University of Liège*.
- FONTENEAU R., MURPHY S., WEHENKEL L. & ERNST D. (2010). Generating informative trajectories by using bounds on the return of control policies. In *Proceedings of the Workshop on Active Learning and Experimental Design 2010 (in conjunction with AISTATS 2010)*.
- FONTENEAU R., MURPHY S. A., WEHENKEL L. & ERNST D. (2011). Towards min max generalization in reinforcement learning. In *Agents and Artificial Intelligence : International Conference, ICAART 2010, Valencia, Spain, January 2010, Revised Selected Papers. Series : Communications in Computer and Information Science (CCIS)*, volume 129, p. 61–77 : Springer, Heidelberg.
- INGERSOLL J. (1987). *Theory of Financial Decision Making*. Rowman and Littlefield Publishers, Inc.
- KAEHLING L. (1993). *Learning in Embedded Systems*. MIT Press.
- MUNOS R. & MOORE A. (2002). Variable resolution discretization in optimal control. *Machine Learning*, **49**, 291–323.
- MURPHY S. (2003). Optimal dynamic treatment regimes. *Journal of the Royal Statistical Society, Series B*, **65**(2), 331–366.
- MURPHY S. (2005). An experimental design for the development of adaptive treatment strategies. *Statistics in Medicine*, **24**, 1455–1481.
- ORMONEIT D. & SEN S. (2002). Kernel-based reinforcement learning. *Machine Learning*, **49**(2-3), 161–178.
- RACHELSON E., SCHNITZLER F., WEHENKEL L. & ERNST D. (2011). Optimal sample selection for batch-mode reinforcement learning. In *3rd International Conference on Agents and Artificial Intelligence (ICAART 2011)*, Roma, Italy.
- RIEDMILLER M. (2005). Neural fitted Q iteration - first experiences with a data efficient neural reinforcement learning method. In *Proceedings of the Sixteenth European Conference on Machine Learning (ECML 2005)*, p. 317–328, Porto, Portugal.
- SUTTON R. & BARTO A. (1998). *Reinforcement Learning*. MIT Press.
- THRUN S. (1992). The role of exploration in learning control. In D. WHITE & D. SOFGE, Eds., *Handbook for Intelligent Control : Neural, Fuzzy and Adaptive Approaches* : Van Nostrand Reinhold.