Filtrage bayésien de la récompense

Matthieu Geist[1,2], Olivier Pietquin[1] et Gabriel Fricout[2]

[1] Supélec
Groupe de recherche IMS, Metz, France
{matthieu.geist,olivier.pietquin}@supelec.fr

[2] ArcelorMittal Research
Département MCE, Maizières-lès-Metz, France
gabriel.fricout@arcelormittal.com

Résumé :
Une large variété de schémas d'approximation de la fonction de valeur a été appliquée à l'apprentissage par renforcement. Cependant, les approches par filtrage bayésien, qui se sont pourtant montrées efficaces dans d'autres domaines comme l'apprentissage de paramètres pour les réseaux neuronaux, ont été peu étudiées jusqu'à présent. Cette contribution introduit un cadre de travail général pour l'apprentissage par renforcement basé sur le filtrage bayésien, ainsi qu'une implémentation spécifique basée sur un filtre de Kalman à sigma-points et une paramétrisation par noyaux. Cela nous permet de proposer un algorithme de différences temporelles pour des espaces d'état et/ou d'action continus qui soit *model-free* et *off-policy*. Il sera illustré sur deux problèmes simples.

Mots-clés : Apprentissage par renforcement, filtrage bayésien, méthodes à noyaux.

1 Introduction

Le contrôle optimal d'un système dynamique et stochastique peut être un problème très complexe. Même avec une connaissance parfaite du système physique, il peut être impossible de déterminer la politique de contrôle optimale de façon analytique. La réponse habituelle de l'industrie à ce type de problème est de faire appel à des heuristiques, qui reposent sur de fortes connaissances humaines *a priori* et pour lesquelles il est difficile de garantir l'optimalité. D'un autre côté, dans le domaine de l'apprentissage numérique, ce problème est traité par l'apprentissage par renforcement (AR) (Sutton & Barto, 1998; Bertsekas, 1995). Le problème de contrôle est alors décrit en terme d'états, d'actions et de récompenses. Dans ce cadre de travail, un agent artificiel essaye d'apprendre une politique de contrôle optimale à partir d'interactions avec son environnement. Il observe l'état du système et choisit une action à lui appliquer, en accord avec sa politique interne qui lie les états aux actions. En retour, l'agent reçoit un signal numérique de récompense, qui est une indication locale et instantanée de la qualité du contrôle. Cette information de récompense est utilisée par l'agent pour apprendre de façon incrémentale la politique de contrôle optimale qui maximisera une fonction du cumul futur des récompenses. Classiquement, la connaissance qu'a l'agent de l'environnement est modélisée par une Q-fonction qui associe à un couple état-action donné une estimation du cumul espéré de récompenses associé. La Q-fonction optimale (notée Q^*) lie chaque paire état-action au maximum possible de cumul de récompenses. L'action optimale est alors celle qui maximise cette fonction pour l'état courant, ce qui rend la connaissance de Q^* suffisante pour réaliser la tâche de contrôle. Le rôle de l'agent peut donc se résumer en l'apprentissage de cette fonction à partir d'interactions avec l'environnement. Cela sera décrit plus avant dans la section 2.1.

L'AR est basé sur un principe d'apprentissage par essai/erreur qui fait sa force mais également sa faiblesse. En fait, l'AR ne nécessite aucune connaissance *a priori* du système, mais suppose d'acquérir des informations sur ce dernier à travers des essais qui pourraient l'endommager. Cela est particulièrement vrai lorsque l'espace d'état est large, étant donné qu'il reste alors majoritairement inconnu au début de l'apprentissage. En pratique, il arrive souvent que l'espace d'état soit trop large (par exemple continu), ce qui l'empêche d'être exploré de façon exhaustive par l'agent. Proposer des algorithmes capables de prendre en compte de tels espaces tout en conservant un apprentissage incrémental à partir d'interactions est ainsi devenu un défi pour la communauté de l'apprentissage numérique. D'autre part, l'AR consiste également à

apprendre à agir dans le doute. L'incertitude provient de la nature stochastique du système à contrôler, mais également de la connaissance partielle qu'a l'agent de son environnement. Cette connaissance partielle a au moins deux origines distinctes : soit l'espace d'état n'a pas été suffisamment exploré par l'agent, soit l'état lui même n'est pas directement observable (observabilité partielle). Cela rend le processus d'apprentissage encore plus difficile, et l'agent doit constamment choisir entre deux types d'action : les actions exploitatives, qui sont optimales respectivement à la connaissance qu'a l'agent du système, et les actions exploratoires, qui ont pour but d'améliorer cette connaissance. Ce problème est connu sous le nom de dilemme entre exploration et exploitation. La généralisation, l'observabilité partielle et le dilemme entre exploration et exploitation sont actuellement les domaines de recherches les plus importants en apprentissage par renforcement.

Classiquement, dans le domaine de l'intelligence artificielle, l'incertitude est prise en compte à l'aide de méthodes bayésiennes. Ces dernières sont adaptées à un apprentissage en ligne étant donné qu'elles sont généralement incrémentales. Le filtrage bayésien peut par exemple être utilisé pour apprendre les paramètres d'une fonction approximant une fonction d'intérêt tout en maintenant une incertitude liée à la régression (voir par exemple van der Merwe, 2004). Nous présentons le filtrage bayésien section 2.2. Bien que ces méthodes de filtrage aient déjà été appliquées à l'apprentissage par renforcement, elles ont jusqu'à présent été étonnamment peu étudiées. Dans cette contribution, nous proposons les prémisses d'une méthode basée sur le filtrage Bayésien et pouvant traiter les problèmes majeurs de l'apprentissage par renforcement, et particulièrement celui de la généralisation qui sera le seul traité ici. Le reste de cet article est organisé comme suit. Dans un premier temps quelques notions de bases sont présentées. Puis la formulation générale de l'apprentissage par renforcement comme un problème de filtrage bayésien ainsi qu'une implémentation spécifique sont proposées. Enfin nous montrons les premiers résultats concernant deux problèmes simples, le *wet-chicken* et le *mountain-car*, avant de conclure et d'esquisser nos travaux futurs.

2 Contexte

Dans cette section, nous présentons le formalisme mathématique classiquement associé à l'apprentissage par renforcement et l'algorithme du Q-learning, ainsi que le paradigme du filtrage bayésien et ses solutions spécifiques. Par la suite, une variable x servira à désigner un vecteur colonne ou un scalaire, ce qui devrait apparaître clairement d'après le contexte.

2.1 Apprentissage par renforcement

Un processus décisionnel de Markov (PDM) consiste en un espace d'état S, un espace d'action A, une probabilité de transition markovienne $p : S \times A \rightarrow \mathcal{P}(S)$ et une fonction de récompense bornée $r : S \times A \times S \rightarrow \mathbb{R}$. Une politique est une fonction qui à un état associe une action : $\pi : S \rightarrow A$. Au temps k, le système est dans un état s_k, l'agent choisit une action $a_k = \pi(s_k)$, et le système passe alors dans un nouvel état s_{k+1} selon la distribution conditionnelle markovienne $p(.|s_k, a_k)$. L'agent reçoit alors la récompense associée $r_k = r(s_k, a_k, s_{k+1})$. Son objectif est de trouver la politique qui maximise l'espérance du cumul des récompenses pondérées, c'est-à-dire la quantité $E_\pi[\sum_{k\in\mathbb{N}} \gamma^k r(S_k, A_k, S_{k+1})|S_0 = s_0]$, pour tout état initial possible s_0, l'espérance étant calculée sur les transitions d'états, la politique π étant suivie, et $\gamma \in [0, 1[$ étant un facteur d'oubli permettant de tenir plus ou moins compte de la valeur des états futurs.

Une approche classique pour résoudre ce problème est d'introduire une fonction Q définie par :

$$Q^\pi(s, a) = \int_S p(z|s, a)\Big(r(s, a, z) + \gamma Q^\pi(z, \pi(z))\Big)dz$$

C'est le cumul espéré de récompenses en choisissant l'action a dans l'état s puis en suivant la politique π par la suite. Le critère d'optimalité est de trouver la politique π^* (et la fonction Q^* associée) telle que pour tout état s et pour toute politique π, on ait $\max_{a\in A} Q^*(s, a) \geq \max_{a\in A} Q^\pi(s, a)$. La Q-fonction optimale Q^* est donnée par l'équation de Bellman :

$$Q^*(s, a) = \int_S p(z|s, a)\left(r(s, a, z) + \gamma \max_{b\in A} Q^*(z, b)\right)dz$$

La politique optimale s'en déduit aisément, c'est la politique gloutonne associée : $\forall s \in S$, $\pi^*(s) = \text{argmax}_{a\in A} Q^*(s, a)$. Dans le cas d'espaces d'état et d'action discrets et finis, l'algorithme du Q-learning

fournit une solution à ce problème. Son principe est de mettre à jour une approximation tabulaire de la Q-fonction optimale après chaque transition (s, a, r, s') :

$$\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha \left(r + \gamma \max_{b \in A} \hat{Q}(s',b) - \hat{Q}(s,a) \right)$$

où α est un taux d'apprentissage. Un fait intéressant est que le Q-learning est un algorithme *off-policy*, c'est-à-dire qu'il permet d'apprendre la politique optimale (via la Q-fonction associée) en en suivant une sous-optimale, si cette dernière est suffisamment exploratoire. La présente contribution peut être vue comme une extension de cet algorithme dans un cadre bayésien (avec d'autres avantages). Nous renvoyons le lecteur à Sutton & Barto (1998) pour une introduction plus complète à l'apprentissage par renforcement.

2.2 Filtrage Bayésien

Le problème du filtrage bayésien peut s'exprimer sous sa forme espace d'état :

$$\begin{aligned} x_{k+1} &= f_k(x_k, v_k) \\ y_k &= g_k(x_k, n_k). \end{aligned}$$

L'objectif est d'inférer séquentiellement l'état caché x_k sachant les observations $y_{1:k} = \{y_1, y_2, \ldots, y_k\}$. L'évolution de l'état se fait selon la (potentiellement non-linéaire et non-stationnaire) fonction f_k est le bruit d'évolution v_k. L'observation y_k est une fonction (également potentiellement non-linéaire et non-stationnaire) de l'état x_k, corrompue par une bruit d'observation n_k. Le principe général est de prédire le nouvel état x_k étant donné les précédentes observations $y_{1:k-1}$, et de corriger cette prédiction en utilisant la nouvelle observation y_k, en accord avec les équations dites de prédiction et de correction (utilisant la loi de Bayes) :

$$\begin{aligned} p(X_k|Y_{1:k-1}) &= \int_{\mathcal{X}} p(X_k|X_{k-1}) p(X_{k-1}|Y_{1:k-1}) dX_{k-1} \text{ (prédiction),} \\ p(X_k|Y_{1:k}) &= \frac{p(Y_k|X_k) p(X_k|Y_{1:k-1})}{\int_{\mathcal{X}} p(Y_k|X_k) p(X_k|Y_{1:k-1}) dX_k} \text{ (correction).} \end{aligned}$$

Ces équations ne sont généralement pas résolubles. Si les fonctions sont linéaires et les bruits gaussiens, la solution optimale est donnée par le filtre de Kalman : les quantités d'intérêt sont des variables aléatoires, et l'inférence (c'est-à-dire la prédiction de ces quantités et leur correction étant donné une nouvelle observation) est faite en ligne par propagation des statistiques suffisantes à travers des transformations linéaires. Si les fonctions sont non-linéaires (et si les bruits sont toujours gaussiens), une solution est de les linéariser autour de l'état : c'est le principe du filtrage de Kalman étendu (ou FKE), les statistiques d'intérêt sont toujours propagées à travers des transformations linéaires. Une autre approche est l'ensemble des filtres de Kalman à sigma-points, ou FKSP (van der Merwe, 2004). L'idée sous-jacente à ce modèle est qu'il est plus facile d'approximer une distribution de probabilité qu'une fonction non-linéaire arbitraire. Un ensemble de points (appelés sigma-points) est calculé de façon déterministe à partir des statistiques de premier et deuxième ordre de l'état caché. Ils sont représentatifs de la distribution d'intérêt. Les images de ces points par les fonctions d'évolution et d'observation sont calculées, et elles sont utilisées pour déterminer les statistiques nécessaires pour les équations de prédiction et de correction. L'algorithme 1 esquisse une mise à jour d'un FKSP dans le cas de bruits additifs, où des notations standards sont utilisées : $x_{k|k-1}$ désigne une prédiction, $x_{k|k}$ une estimée (ou une correction), $P_{x,y}$ la matrice de covariance des vecteurs aléatoires x et y, $\bar{n}_k$ une moyenne et k est l'index temporel. Le lecteur peut se référer à van der Merwe (2004) pour plus de détails. C'est le type de filtre que nous utilisons dans l'implémentation spécifique proposée. Une dernière méthode est le filtrage particulaire, ou Monte Carlo séquentiel. C'est une approche numérique adaptée au cas où les fonctions sont non-linéaires et les bruits non-gaussiens. Nous renvoyons le lecteur à Chen (2003) pour une vue d'ensemble complète des différentes méthodes de filtrage bayésien.

3 Formulation générale

Maintenant que l'apprentissage par renforcement et le filtrage bayésien ont été présentés, nous pouvons introduire le canevas du filtrage bayésien de la récompense. Nous commençons par présenter l'idée générale ainsi que des travaux similaires, avant de présenter plus formellement le cadre de travail général proposé.

Algorithme 1 Mise à jour d'un FKSP

Entrées : $x_{k-1|k-1}$, $P_{k-1|k-1}$
Sorties : $x_{k|k}$, $P_{k|k}$
Calcul des sigma-points :
Calculer de façon déterministe l'ensemble des sigma-points $X_{k-1|k-1}$ à partir de $x_{k-1|k-1}$ et $P_{k-1|k-1}$;
Etape de prédiction :
Calculer $X_{k|k-1}$ à partir de $f_k(X_{k-1|k-1}, \bar{v}_k)$ et de la covariance du bruit d'évolution ;
Calculer $x_{k|k-1}$ et $P_{k|k-1}$ à partir de $X_{k|k-1}$;
Etape de Correction :
Observer y_k ;
$Y_{k|k-1} = g_k(X_{k|k-1}, \bar{n}_k)$;
Calculer $y_{k|k-1}$, $P_{y_{k|k-1}}$ et $P_{x_{k|k-1},y_{k|k-1}}$ à partir de $X_{k|k-1}$, $Y_{k|k-1}$ et de le covariance du bruit d'observation ;
$K_k = P_{x_{k|k-1},y_{k|k-1}} P^{-1}_{y_{k|k-1}}$; {*gain de Kalman*}
$x_{k|k} = x_{k|k-1} + K_k(y_k - y_{k|k-1})$;
$P_{x_{k|k}} = P_{x_{k|k-1}} - K_k P_{y_{k|k-1}} K_k^T$;

3.1 Idée générale et travaux similaires

L'idée générale est de paramétrer la Q-fonction, et de considérer le vecteur de paramètres associé comme étant l'état caché que devra inférer un filtre bayésien. L'équation d'observation associée lie les récompenses aux paramètres grâce à l'équation de Bellman (1957). L'idée d'utiliser le filtrage bayésien pour résoudre des problèmes d'apprentissage par renforcement est déjà apparue dans quelques publications. Engel (2005) utilise des processus Gaussiens pour l'apprentissage par renforcement. Son approche peut être vue comme une extension du filtre de Kalman à un vecteur d'état de dimension infinie (le processus Gaussien), cependant l'équation d'observation est nécessairement linéaire, et contrairement à cette contribution seuls des algorithmes *on-policy* sont possibles, ce qui impose d'adopter des schémas du type itération de la politique. Dans (Phua & Fitch, 2007) un banc de filtres de Kalman est utilisé pour déterminer une paramétrisation linéaire par morceau de la fonction de valeur. Cette approche peut être vue comme un cas spécifique (bien que non trivial) du canevas général proposé ici.

3.2 Filtrage bayésien de la récompense

L'équation de Bellman peut s'écrire :

$$Q^*(s,a) = r(s,a,s') + \gamma \max_{b \in A} Q^*(s',b) - n_{s,a}(s')$$

$$\text{avec } n_{s,a}(s') = \int_S p(z|s,a) \left\{ r(s,a,s') - r(s,a,z) + \gamma \left(\max_{b \in A} Q^*(s',b) - \max_{b \in A} Q^*(z,b) \right) \right\} dz.$$

Le terme de bruit $n_{s,a}$ est la transformation non linéaire de la variable aléatoire $(S'|S = s, A = a)$ de loi $p(.|s,a)$, c'est donc une variable aléatoire. Il peut être montré qu'elle est centrée et de variance finie (d'après l'inégalité de Minkowsky), c'est-à-dire, r_{max} étant la borne sur la fonction de récompense :

$$E[n_{s,a}] = \int_S n_{s,a}(z) p(z|s,a) dz = 0 \quad \text{et} \quad E[n_{s,a}^2] = \int_S n_{s,a}^2(z) p(z|s,a) dz \leq \left(\frac{r_{max}}{1-\gamma} \right)^2$$

Pour une transition (s,a,r,s') observée, l'équation de Bellman peut donc être réécrite :

$$r(s,a,s') = Q^*(s,a) - \gamma \max_{b \in A} Q^*(s',b) + n_{s,a}(s')$$

Cette forme est similaire à la mise à jour de l'algorithme du Q-learning, et est de première importance pour le cadre de travail proposé.

Supposons que la Q-fonction est paramétrée par un vecteur θ, linéairement ou non. L'objectif est de déterminer une bonne approximation $\hat{Q}_\theta$ de la Q-fonction optimale Q^* à partir de l'observation de transitions

(s, a, r, s'). Ce problème de régression par la récompense est exprimé sous une forme espace d'état. Pour une transition (s_k, a_k, r_k, s'_k), il s'écrit :

$$\begin{aligned}\theta_{k+1} &= \theta_k + v_k \\ r_k &= \hat{Q}_{\theta_k}(s_k, a_k) - \gamma \max_{a \in A} \hat{Q}_{\theta_k}(s'_k, a) + n_k.\end{aligned}$$

Ici v_k est un bruit d'évolution artificiel et n_k est un bruit d'observation centré incluant toute la stochasticité du PDM (ainsi que l'erreur de modélisation). La formulation générale est posée, mais elle est encore loin d'être résolue. L'équation d'observation est non-linéaire (à cause de l'opérateur $\max$), c'est pourquoi les filtres classiques comme celui de Kalman ne peuvent être utilisés. Formellement, le bruit d'évolution est nul, mais introduire un bruit artificiel d'évolution peut permettre d'améliorer la stabilité et la convergence du filtre (par exemple en aidant à éviter les optima locaux). Un bruit d'observation, potentiellement adaptatif, doit être choisi, ainsi qu'une paramétrisation pour la Q-fonction. Enfin, comme pour toute approche bayésienne, un *a priori* sur les paramètres doit être fixé pour initialiser l'algorithme itératif. Une implémentation spécifique d'un filtre bayésien de la récompense est proposée dans la section suivante.

4 Une solution pratique

Les deux plus grands degrés de liberté sont le choix de la paramétrisation et du filtre. Nous choisissons une paramétrisation par noyaux pour la Q-fonction, que nous rendons parcimonieuse à l'aide d'une méthode de dictionnaire. L'équation d'observation contenant l'opérateur $\max$, le filtre doit être non-linéaire et ne pas nécessiter de calcul de dérivées. Nous choisissons d'utiliser un filtre à sigma-points.

4.1 Paramétrisation

Une paramétrisation par noyaux est choisie pour la Q-fonction, en raison de son expressivité donnée par le théorème de Mercer (Vapnik, 1998). Un noyaux est une fonction continue, symétrique et définie positive. Chaque noyau correspond à un produit scalaire dans un espace de (généralement) plus grande dimension. Plus précisément, pour chaque noyau K, il existe une fonction φ de l'espace de départ $\mathcal{X}$ dans l'espace dit des caractéristiques $\mathcal{F}$ tel que $\forall x, y \in \mathcal{X}$, $K(x, y) = \langle \varphi(x), \varphi(y) \rangle$. Cette propriété fondamentale est importante pour l'initialisation de notre algorithme. Plus précisément, des noyaux gaussiens sont choisis, et leurs centres, leurs variances ainsi que les poids associés sont paramétrés :

$$\hat{Q}_\theta(s, a) = \sum_{i=1}^{p} \alpha_i K_{\sigma_i^s}(s, s_i) K_{\sigma_i^a}(a, a_i) \text{ avec } K_{\sigma_i^x}(x, x_i) = \exp\left(-(x - x_i)^T (\Sigma_i^x)^{-1} (x - x_i)\right),$$
$$\text{où } x = s, a,\ \Sigma_i^x = \mathrm{diag}(\sigma_i^x)^2, \text{ et } \theta = [(\alpha_i)_{i=1}^p, (s_i^T)_{i=1}^p, (a_i^T)_{i=1}^p, ((\sigma_i^s)^T)_{i=1}^p, ((\sigma_i^a)^T)_{i=1}^p]^T$$

L'opérateur diag appliqué à un vecteur colonne produit une matrice diagonale. Notons que le noyau utilisé $K_{(\sigma_i^s, \sigma_i^a)}([s^T, a^T]^T, [s_i^T, a_i^T]^T) = K_{\sigma_i^s}(s, s_i) K_{\sigma_i^a}(a, a_i)$ est un produit de noyaux gaussiens, c'est donc un noyau gaussien.

4.2 Dictionnaire

Un premier problème est de choisir le nombre p de fonctions noyaux et l'*a priori* associé. Pour cela, nous posons un *a priori* sur la largeur des gaussiennes : $\sigma_0^T = [(\sigma_0^s)^T, (\sigma_0^a)^T]$. Une méthode de dictionnaire proposée notamment dans (Engel, 2005) est ensuite utilisée pour déterminer le nombre de noyaux et l'*a priori* sur leurs centres. Considérons un noyau K, la fonction associée φ et un ensemble de points $X = \{x_1, x_2, \ldots\}$. L'objectif de cette méthode est de calculer un dictionnaire de p points $\mathcal{D} = \{\tilde{x}_1, \ldots, \tilde{x}_p\}$ tel que $\varphi(\mathcal{D})$ soit une base approximative de $\varphi(X)$ qui est inclus dans l'espace des caractéristiques associé au noyau K.

Cette procédure est itérative. Supposons que les échantillons $x_1, x_2, \ldots$ sont séquentiellement observés (dans notre cas particulier, les échantillons sont des couples état-action). Au temps k, un dictionnaire $\mathcal{D}_{k-1} = (\tilde{x}_j)_{j=1}^{m_{k-1}} \subset (x_j)_{j=1}^{k-1}$ est disponible où par construction les vecteurs caractéristiques $\varphi(\tilde{x}_j)$ sont approximativement linéairement indépendants dans $\mathcal{F}$. L'exemple x_k est ensuite observé, et est ajouté au

dictionnaire si $\varphi(x_k)$ est approximativement linéairement indépendant de $\mathcal{D}_{k-1}$. Pour tester cela, des poids $w = (w_1, \ldots, w_{m_{k-1}})^T$ doivent être calculés de façon à vérifier

$$\delta_k = \min_w \left\| \sum_{j=1}^{m_{k-1}} w_j \varphi(\tilde{x}_j) - \varphi(x_k) \right\|^2$$

Un seuil prédéfini ν, qui sert à déterminer la qualité de l'approximation (et en conséquence la parcimonie du dictionnaire), est utilisé. Si $\delta_k > \nu$, $x_k = \tilde{x}_{m_k}$ est ajouté au dictionnaire, sinon $\varphi(x_k)$ s'écrit :

$$\varphi(x_k) = \sum_{i=1}^{m_{k-1}} w_i \varphi(\tilde{x}_i) + \varphi_{res} \text{ avec } \|\varphi_{res}\|^2 \leq \nu$$

Nous définissons la matrice $\tilde{K}_{k-1}$ de taille $m_{k-1} \times m_{k-1}$ et le vecteur $\tilde{k}_{k-1}(x)$ de taille $m_{k-1} \times 1$ par :

$$\left(\tilde{K}_{k-1}\right)_{i,j} = K(\tilde{x}_i, \tilde{x}_j) \text{ et } \left(\tilde{k}_{k-1}(x)\right)_i = K(x, \tilde{x}_i)$$

En utilisant la bilinéarité du produit scalaire et en remplaçant ces derniers par les noyaux, δ_k peut s'exprimer de la façon suivante :

$$\delta_k = \min_w \left\{ w^T \tilde{K}_{k-1} w - 2 w^T \tilde{k}_{k-1}^T(x_k) + K(x_k, x_k) \right\}$$

La solution de ce problème est $\delta_k = K(x_k, x_k) - \tilde{k}_{k-1}(x_k) w_k$ avec $w_k = \tilde{K}_{k-1}^{-1} \tilde{k}_{k-1}(x_k)$. De plus il existe une version efficace de cet algorithme utilisant le lemme d'inversion matricielle (qui permet de calculer directement l'inverse, de façon incrémentale). Le lecteur peut se référer à Engel (2005) pour plus de détails. Pratiquement nous supposerons que S et A sont des ensembles compacts et que leur bornes sont connues. Le dictionnaire associé peut alors être calculé dans une phase de prétraitement à partir de N échantillons générés uniformément à partir de $S \times A$.

4.3 *A priori* gaussien

Comme pour chaque approche bayésienne, un *a priori* doit être choisi pour la distribution des paramètres. Nous posons $\theta_0 \sim \mathcal{N}(\bar{\theta}_0, \Sigma_{\theta_0})$ où

$$\bar{\theta}_0 = [\alpha_0, \ldots, \mathcal{D}_s, \mathcal{D}_a, (\sigma_0^s)^T, \ldots, (\sigma_0^a)^T, \ldots]^T$$
$$\Sigma_{\theta_0} = \text{diag}(\sigma^2_{\alpha_0}, \ldots, \sigma^2_{\mu_0^s}, \ldots, \sigma^2_{\mu_0^a}, \ldots, \sigma^2_{\sigma_0^s}, \ldots, \sigma^2_{\sigma_0^a}, \ldots)$$

Dans ces expressions, α_0 est la moyenne *a priori* sur les poids des noyaux, $\mathcal{D} = \mathcal{D}_s \times \mathcal{D}_a$ est l'ensemble des moyennes *a priori* des centres des noyaux, calculé en prétraitement par la méthode de dictionnaire à partir de la moyenne *a priori* de l'écart-type des noyaux gaussiens $\sigma_0^T = [(\sigma_0^s)^T, (\sigma_0^a)^T]$, et $\sigma^2_{\alpha_0}$, $\sigma^2_{\mu_0^x}$, $\sigma^2_{\sigma_0^x}$ sont respectivement les variances *a priori* sur les poids, les centres et les écarts-type des noyaux, pour $x = s, a$. Tous ces paramètres (excepté le dictionnaire) doivent être définis à l'avance, en utilisant les connaissances que l'on peut avoir du problème d'intérêt. Soit $q = (2(n_a + n_s) + 1)\, p$, où n_s (respectivement n_a) est la dimension de l'espace d'état (respectivement de l'espace d'action). Notons que $\bar{\theta}_0 \in \mathbb{R}^q$ et $\Sigma_{\theta_0} \in \mathbb{R}^{q \times q}$. Nous posons également l'*a priori* sur les bruits : $v_0 \sim \mathcal{N}(0, R_{v_0})$ et $n_0 \sim \mathcal{N}(0, R_{n_0})$, où $R_{n_0} = \sigma^2_{n_0}$.

4.4 Mise à jour des paramètres

Une fois que les paramètres sont initialisés, il reste toujours à les mettre à jour quand de nouvelles observations (s_k, a_k, r_k, s'_k) sont disponibles. Pour cela, nous utilisons un FKSP spécifique, un SR-CDKF (*Square-Root Central Difference Kalman Filter*). Le lecteur peut se référer à van der Merwe (2004) pour une description plus précise. Un dernier problème est de choisir un bruit artificiel d'évolution. Formellement, comme la fonction cible est stationnaire, il est nul. Cependant, son introduction peut améliorer la stabilité et la convergence du filtre.

La covariance du bruit d'évolution est choisie comme étant une fraction de la covariance des paramètres :

$$R_{v_k} = (\lambda^{-1} - 1)\Sigma_{\theta_k}$$

où $\lambda \in]0, 1]$ $(1 - \lambda \ll 1)$ est un facteur d'oubli similaire à celui utilisé dans l'algorithme des moindres carrés récursifs. Dans ce papier, nous choisissons un bruit d'observation constant, c'est-à-dire que $R_{n_k} = R_{n_{k-1}}$. Le filtre bayésien de la récompense proposé est résumé dans l'algorithme 2.

Algorithme 2 Un filtre bayésien de la récompense

entrées : ν, N, α_0, σ_0, σ_{α_0}, $\sigma_{\mu_0^x}$, $\sigma_{\sigma_0^x}$, σ_{v_0}, σ_{n_0}
sorties : $\bar{\theta}$, Σ_θ
Calcul du dictionnaire :
$\forall i \in \{1 \dots N\},\ [s_i^T, a_i^T]^T \sim \mathcal{U}_{\mathcal{S}\times\mathcal{A}}$;
Poser $X = \{[s_1^T, a_1^T]^T, \dots, [s_N^T, a_N^T]^T\}$;
$\mathcal{D}$ =`Compute-Dictionary`(X,ν,σ_0)
Initialisation :
Initialiser $\bar{\theta}_0$, Σ_{θ_0}, R_{n_0}, R_{v_0} ;
pour $k = 1, 2, \dots$ **faire**
  Observer $t_k = (s_k, a_k, r_k, s'_k)$;
  Mise à jour du SR-CDKF :
  $[\bar{\theta}_k, \Sigma_{\theta_k}, K_k]$ = `SR-CDKF-Update`($\bar{\theta}_{k-1}$,$\Sigma_{\theta_{k-1}}$,t_k,$R_{v_{k-1}}$,$R_{n_{k-1}}$) ;
  Mise à jour du bruit artificiel d'évolution :
  $R_{v_k} = (\lambda^{-1} - 1)\Sigma_{\theta_k}$;
fin pour

4.5 Maximum sur l'espace d'action

Une difficulté technique est le calcul du maximum de la Q-fonction paramétrée sur l'espace d'action. Ce calcul est nécessaire pour la mise à jour du filtre. Avec la paramétrisation adoptée, cela n'est pas résoluble analytiquement. Une première solution serait d'échantillonner l'espace d'état et de calculer le maximum des échantillons obtenus. Cependant ce n'est pas très efficace. La méthode utilisée est proche de celle proposée par Carreira-Perpinan (2000).

Le maximum sur les centres des noyaux actions est d'abord calculé : $\mu^a = \mathrm{argmax}_{a_i}\, \hat{Q}_\theta(s, a_i)$. Il sert d'initialisation pour la méthode de Newton-Raphson utilisée pour trouver le maximum :

$$a_m \leftarrow a_m - \Big((\nabla_a \nabla_a^T)\hat{Q}_\theta(s,a)\Big)^{-1}_{a=a_m} \nabla_a \hat{Q}_\theta(s,a)\Big|_{a=a_m}$$

Lorsque la hessienne est singulière, une méthode de type descente de gradient/point fixe est utilisée :

$$a_m \leftarrow a_m + \nabla_a \hat{Q}_\theta(s,a)\Big|_{a=a_m}$$

L'action a_m obtenue est considérée comme l'action qui maximise la Q-fonction paramétrée.

5 Résultats préliminaires

L'approche proposée est testée sur deux problèmes. Le premier, le *wet-chicken*, est une tâche stochastique à espaces d'état et d'action continus. Le second, le *mountain car* est une tâche déterministe, à espace d'action discret et à espace d'état continu. Ces choix simples sont fait afin de démontrer que la méthode peut s'adapter à différentes structures de problèmes. Nous adopterons une paramétisation hybride pour le second. Mais nous discutons dans un premier temps le choix des paramètres.

5.1 Choix des paramètres

Pour les deux problèmes le facteur γ est fixé à 0.9. Le facteur de parcimonie du dictionnaire a été fixé à une valeur assez importante, $\nu = 0.9$. Le facteur d'oubli associé au bruit artificiel d'évolution a également été fixé à une haute valeur : $\lambda^{-1} - 1 \simeq 10^{-6}$.

Le choix initial des écarts-type des noyaux est dépendant du problème. Cependant, en pratique, un bon choix semble être de prendre une fraction de la quantité $x(j)_{max} - x(j)_{min}$ pour l'écart-type associé à $x(j)$, la $j^{\text{ème}}$ composante du vecteur x, $x(j)_{max}$ et $x(j)_{min}$ étant les bornes sur les valeurs prises par $x(j)$. Nous supposons l'*a priori* sur les poids des noyaux centrés, et nous posons l'écart-type associé comme étant une faible fraction de la borne théorique de la Q-fonction $\frac{r_{max}}{1-\gamma}$. En raison de la géométrie des noyaux gaussiens, nous supposons que les centres fournis par le dictionnaire sont approximativement uniformément

distribués, et nous posons l'écart-type de la $j^{\text{ème}}$ composante du vecteur x comme étant une faible fraction de $(x(j)_{max} - x(j)_{min})p^{-\frac{1}{ns+na}}$, avec la convention que pour les espaces discrets $n = 0$. Finalement, nous posons l'écart-type de l'*a priori* associé à l'écart-type des noyaux comme étant une faible fraction de ces derniers. Autrement dit, nous posons $\sigma_{\sigma_0^{x(j)}}$ comme étant une petite fraction de $\sigma_0^{x(j)}$, pour la $j^{\text{ème}}$ composante de x.

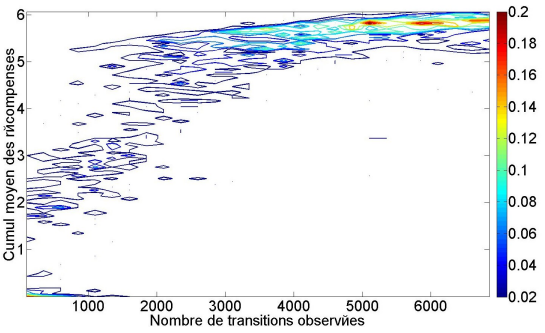

FIG. 1 – Histogramme 2d du cumul moyen de récompenses pour le *wet-chicken* (100 *runs*).

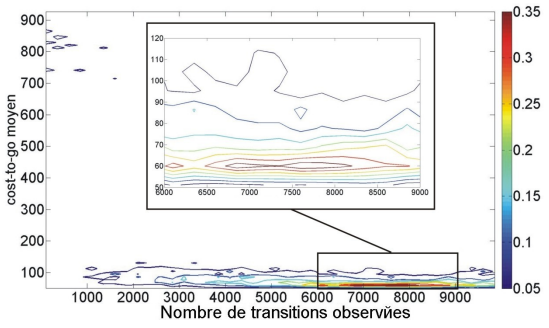

FIG. 2 – Histogramme 2d du *cost-to-go* moyen pour le problème du *mountain-car* (100 *runs*).

5.2 Problème du *wet-chicken*

Dans le problème du *wet-chicken* (inspiré de Schneegass *et al.*, 2006), un canoéiste doit pagayer sur une rivière jusqu'à atteindre une chute d'eau. La récompense augmente linéairement avec la proximité de la chute. Le canoéiste reçoit une punition s'il tombe, et il doit alors recommencer. Des turbulences simulées rendent le problème stochastique. Plus formellement, l'espace d'état est $S = [0, 10]$ (10 étant la position de la chute), l'espace d'action est $A = [-1, 1]$ (continûment de pagayer complètement vers l'arrière à pagayer complètement vers l'avant), la transition est $s' = s + a + c$ avec $c \sim \mathcal{N}(0, \sigma_c)$, $\sigma_c = 0.3$, et la récompense associée est $r = \frac{s'}{10}$. Si $s' \geq 10$ le canoéiste tombe, la récompense est $r = -1$ et l'épisode est terminé.

Pour tester l'algorithme proposé, des transitions aléatoires sont générées uniformément et présentées à l'entrée du filtre : à chaque pas de temps k, un état s_k et une action a_k sont générées uniformément à partir de $S \times A$, et utilisés pour tirer une transition (aléatoire) vers s'_k, avec la récompense associée r_k, et la transition (s_k, a_k, s'_k, r_k) est présentée en entrée de l'algorithme. Les résultats sont présentés figure 1. Pour chaque *run* de l'algorithme et toutes les 250 transitions observées, le cumul espéré de récompenses pour la politique courante a été calculé comme le cumul moyen de récompenses sur 1000 épisodes, chacun étant initialisé aléatoirement (distribution uniforme). La moyenne est donc faite sur la stochasticité des transitions et de l'état de départ. Notons que la durée de vie de l'agent (la durée maximum d'un épisode) est bornée à 100 interactions avec l'environnement. Nous avons ensuite calculé l'histogramme en deux dimensions de ces cumul moyens de récompenses sur 100 *runs* différents de l'algorithme. Autrement dit, nous présentons la distribution du cumul de récompenses sur les différents *runs* de l'algorithme comme une fonction du nombre de transitions observées. La barre sur la droite indique les pourcentages associés à l'histogramme.

La politique optimale a un cumul moyen de récompenses de 6 (calculé avec l'algorithme d'itération de la valeur sur un espace d'état-action très finement échantillonné). On peut voir sur la figure 1 que l'algorithme proposé permet d'apprendre des politiques quasi-optimales. Après 1000 transitions certaines politiques ont un score de 5 (85% de la politique optimale), ce qui est atteint par la majorité des politiques après 3000 transitions. Après 7000 transitions, des politiques très proches de l'optimale ont été trouvées pour quasiment tous les *runs* (le mode de la distribution associée est à 5.85, soit 98% de la politique optimale). Pour représenter la Q-fonction approximée, seulement 7.7 ± 0.7 fonctions noyaux ont été utilisées.

Deux remarques peuvent être faites sur ce test. Premièrement, le bruit d'observation est dépendant des entrées, étant donné qu'il modélise la stochasticité du PDM. Rappelons que nous avons choisi ici un bruit d'observation constant. Deuxièmement, le bruit peut ne pas être gaussien. Par exemple, à proximité de la chute d'eau, il aura tendance à être bimodal en raison de la discontinuité de la fonction de récompense. Rappelons que le filtre utilisé suppose que les bruits sont gaussiens. Ainsi l'algorithme proposé est relativement robuste, et il permet d'atteindre d'assez bons résultats si l'on considère que les observations étaient totalement aléatoires (aspect *off-policy*).

5.3 Problème du *mountain car*

Le second problème que nous testons est celui du *mountain car*. Un voiture sous-motorisée doit gravir une route à fort dénivelé. L'état est bi-dimensionel (position et vélocité) et continu, et il y a 3 actions possibles (avancer, reculer et neutre). La description du problème est donnée par Sutton & Barto (1998). Une récompense nulle est donnée à chaque pas de temps, et $r = 1$ est obtenu lorsque la voiture atteint le but.

Un premier problème est de trouver une paramétrisation pour cette tâche. Celle que nous avons discutée est adaptée aux problèmes continus, et non hybrides. Cependant elle peut aisément être étendue au problèmes à états continus et actions discrètes. Une solution simple consiste à avoir une paramétrisation pour chaque action discrète, c'est-à-dire de la forme $\theta = [\theta^{a_1}, \theta^{a_2}, \theta^{a_3}]$, la Q-fonction associée étant $Q_\theta(s, a) = Q_{\theta^a}(s)$. Toutefois, on peut noter que pour un état fixé et différentes actions les Q-valeurs vont être très proches, ou autrement dit $Q^*(s, a_1)$, $Q^*(s, a_2)$ et $Q^*(s, a_3)$ vont avoir des formes similaires, en tant que fonctions sur l'espace d'état. Nous considérons donc que les poids vont être spécifiques à chaque action, mais les centres et les largeurs des noyaux seront partagés. Plus formellement le vecteur de paramètres est

$$\theta = [(\alpha_i^{a_1})_{i=1}^p, (\alpha_i^{a_2})_{i=1}^p, (\alpha_i^{a_3})_{i=1}^p, (s_i^T)_{i=1}^p, ((\sigma_i^s)^T)_{i=1}^p]^T$$

les notations étant les mêmes que dans les sections précédentes.

Comme pour le problème du *wet-chicken*, des transitions aléatoires ont été présentées à l'entrée du filtre. Les résultats sont présentés figure 2, qui est un histogramme bidimensionel similaire à celui introduit précédemment. La légère différence est que la mesure de performance est maintenant le *cost-to-go*, c'est-à-dire le nombre de pas de temps nécessaires pour atteindre le but. Ce coût peut être directement lié au cumul moyen de récompenses, nous l'utilisons car nous pensons que c'est une mesure plus significative pour ce problème. Pour chaque *run* de l'algorithme et toutes les 250 transitions, le *cost-to-go* moyen pour la politique courante a été calculé comme étant une moyenne sur 1000 épisodes aléatoirement initialisés (la moyenne est donc faite uniquement sur les états de départ, les transitions étant déterministes). La durée de vie de l'agent a été limitée à 1000 interactions avec l'environnement. L'histogramme est calculé sur 100 *runs*.

La politique optimale a un *cost-to-go* moyen de 55 (calculé avec une itération de la valeur sur un espace d'état très finement échantillonné). Il apparaît sur la figure 2 que l'algorithme proposé permet de trouver des politiques quasi-optimales. Après 1500 transitions, la plupart des politiques on un *cost-to-go* de moins de 120. Après 6000 transitions, des politiques très proches de l'optimale ont été trouvées pour quasiment tous les *runs* de l'algorithme (le mode de la distribution associée est à 60). Pour représenter la Q-fonction approchée, seulement 7.5 ± 0.8 fonctions noyaux ont été utilisées.

Ce problème n'est pas stochastique, mais les récompenses informatives sont très rares (ce qui pourrait causer une convergence trop rapide du gain de Kalman), les transitions sont non-linéaires et les récompenses binaires. Malgré cela, le filtre proposé montre un bon comportement. Une fois encore, nous considérons que l'algorithme proposé atteint d'assez bons résultats étant donné la tâche à traiter.

5.4 Comparaison aux autres méthodes

Nous soutenons que la qualité des politiques apprises est comparable aux méthodes de l'état de l'art. Mesurer cette qualité dépend de la configuration du problème et des mesures de performance, cependant le filtrage bayésien de la récompense permet d'obtenir des politiques quasi-optimales. Le lecteur peut se référer à Schneegass *et al.* (2006) pour quelques éléments de comparaison.

Dans la plupart des approches, le système est contrôlé pendant l'apprentissage, ou pour les méthodes hors-ligne, les exemples observés sont générés par une politique sous-optimale. Dans les expérimentations proposées, des transitions totalement aléatoires sont observées, et la façon dont est abordé le problème est plus proche de la régression pure. Cependant, pour la tâche du *mountain car*, il est souvent rapporté la nécessité de quelques centaines d'épisodes pour apprendre un politique quasi-optimale (voir par exemple Sutton & Barto, 1998), et chaque épisode devrait contenir de quelques dizaines à plusieurs centaines d'interactions avec l'environnement (cela dépend de la qualité du contrôle courant). Dans l'approche proposée quelques milliers de transitions doivent être observées pour atteindre une politique quasi-optimale. C'est très grossièrement le même ordre de grandeur en terme de vitesse de convergence.

Pour l'instant, nous traitons les différentes tâches de contrôle comme des problèmes de régression, l'apprentissage se faisant à partir de transitions aléatoires. Notre approche est donc pour l'instant difficilement comparable à l'état de l'art, pour lequel l'apprentissage se fait à partir de trajectoires. Il est cependant prévu

de conduire des comparaisons plus poussées à d'autres approches quand un organe de contrôle sera ajouté au paradigme proposé.

6 Conclusion et perspectives

Nous avons introduit une formulation de l'apprentissage par renforcement sous forme de filtrage bayésien. En observant les récompenses (et les transitions associées), le filtre est capable d'inférer une politique quasi-optimale via la Q-fonction paramétrée. Une implémentation spécifique, basée sur un FKSP, une paramétrisation non-linéaire par noyaux et une méthode de dictionnaire permettant une représentation parcimonieuse de la fonction de valeur a été décrite. Elle a été testée sur deux problèmes, chacun exhibant des difficultés spécifiques pour l'algorithme. Notre filtre *off-policy* s'est avéré efficace sur ces tâches continues.

Cependant, ce papier n'a fait qu'effleurer certaines des possibilités du canevas proposé. L'approche par filtrage bayésien permet de dériver une information d'incertitude sur la Q-fonction estimée, elle pourrait être utilisée pour le dilemme entre exploration et exploitation, dans l'idée de Dearden *et al.* (1998) ou de Strehl *et al.* (2006). Cela pourrait permettre d'accélérer et d'améliorer l'apprentissage. De plus, nous pensons que l'observabilité partielle devrait assez naturellement être incluse dans ce cadre de travail, la Q-fonction pouvant être considérée ici comme une fonction sur des densités de probabilité.

Nous avons l'intention pour la suite de traiter les deux problèmes mentionnés précédemment, notre objectif principal étant de regrouper dans un seul modèle ce que nous considérons comme les trois principaux problèmes de l'apprentissage par renforcement, à savoir la généralisation, l'observabilité partielle et le compromis entre exploration et exploitation. Cependant il existe d'autres points d'intérêt. Un bruit artificiel d'évolution plus efficace pourrait améliorer la stabilité et les qualités de convergence du filtre. Un bruit adaptatif d'observation pourrait s'avérer nécessaire pour des tâches plus complexes, étant donné qu'il est formellement dépendant des entrées. De plus, il y a encore quelques soucis techniques, comme la recherche du maximum sur les actions. Enfin, une étude théorique peut s'avérer nécessaire, notamment en ce qui concerne la convergence, qui peut être principalement compromise par l'opérateur $\max$.

Références

BELLMAN R. (1957). *Dynamic Programming*. Dover Publications, sixth edition.

BERTSEKAS D. P. (1995). *Dynamic Programming and Optimal Control*. Athena Scientific, 3rd edition.

CARREIRA-PERPINAN M. A. (2000). Mode-finding for mixtures of gaussian distributions. *IEEE Transactions on Pattern Analalysis and Machine Intelligence*, **22**(11), 1318–1323.

CHEN Z. (2003). *Bayesian Filtering : From Kalman Filters to Particle Filters, and Beyond*. Rapport interne, Adaptive Systems Lab, McMaster University.

DEARDEN R., FRIEDMAN N. & RUSSELL S. J. (1998). Bayesian Q-learning. In *AAAI/IAAI*, p. 761–768.

ENGEL Y. (2005). *Algorithms and Representations for Reinforcement Learning*. PhD thesis, Hebrew University.

PHUA C. W. & FITCH R. (2007). Tracking value function dynamics to improve reinforcement learning with piecewise linear function approximation. In *ICML 07*.

SCHNEEGASS D., UDLUFT S. & MARTINETZ T. (2006). Kernel rewards regression : an information efficient batch policy iteration approach. In *AIA'06 : Proceedings of the 24th IASTED international conference on Artificial intelligence and applications*, p. 428–433, Anaheim, CA, USA : ACTA Press.

STREHL A. L., LI L., WIEWIORA E., LANGFORD J. & LITTMAN M. L. (2006). Pac model-free reinforcement learning. In *23rd International Conference on Machine Learning (ICML 2006)*, p. 881–888, Pittsburgh, PA, USA.

SUTTON R. S. & BARTO A. G. (1998). *Reinforcement Learning : An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press, 3rd edition.

VAN DER MERWE R. (2004). *Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models*. PhD thesis, OGI School of Science & Engineering, Oregon Health & Science University, Portland, OR, USA.

VAPNIK V. N. (1998). *Statisical Learning Theory*. John Wiley & Sons, Inc.