Différences Temporelles de Kalman

Matthieu Geist, Olivier Pietquin, Gabriel Fricout

Différences Temporelles de Kalman

Matthieu Geist[1,2,3], Olivier Pietquin[1] et Gabriel Fricout[2]

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

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

[3] INRIA Nancy - Grand Est
Equipe-projet CORIDA, Nancy, France

Résumé : Cette contribution traite de l'approximation de la fonction de valeur ainsi que de la Q-fonction dans des processus décisionnels de Markov déterministes. Un cadre de travail statistique général inspiré du filtrage de Kalman est introduit. Son principe est d'adopter une représentation paramétrique de la fonction de valeur (ou de la Q-fonction), de modéliser le vecteur de paramètres associé comme une variable aléatoire et de minimiser l'erreur quadratique sur les paramètres conditionnée aux récompenses observées depuis l'origine des temps. De ce paradigme général, que nous nommons *Différences Temporelles de Kalman* (KTD pour *Kalman Temporal Differences*), et en utilisant un schéma d'approximation appelé transformation non-parfumée, une famille d'algorithmes du second ordre est dérivée, à savoir KTD-V, KTD-SARSA et KTD-Q, qui ont respectivement comme objectif l'évaluation de la fonction de valeur pour une politique donnée, l'évaluation de la Q-fonction pour une politique donnée, et l'évaluation de la Q-fonction optimale. Cette approche présente un certain nombre d'avantages tels que la capacité à prendre en compte une paramétrisation non-linéaire, l'efficacité de l'apprentissage en terme du nombre d'échantillons observés, la prise en compte d'environnements non-stationnaires ou encore la possibilité d'obtenir une information d'incertitude, que nous utiliserons pour proposer une forme d'apprentissage actif. Ces différents aspects sont discutés et illustrés au travers de plusieurs expériences.

Mots-clés : Apprentissage par renforcement, filtrage de Kalman, approximation de la fonction de valeur, gestion de l'incertitude, problèmes non-stationnaires

1 Introduction

L'apprentissage par renforcement (AR) est un paradigme général dans lequel un agent apprend à contrôler de manière optimale un système dynamique à partir d'interactions avec ce dernier. Un algorithme d'AR devrait présenter certaines caractéristiques importantes : pouvoir prendre en compte des espaces d'états grands voir continus (approximation de la fonction de valeur), être efficace en terme d'échantillons (c'est-à-dire apprendre un bon contrôle avec aussi peu d'interactions avec le système que possible), prendre en compte les non-stationnarités (même si le système est stationnaire, la dualité entre apprentissage et contrôle peut induire des non-stationnarités) et pouvoir maintenir une information d'incertitude (ce qui est une condition quasi-nécessaire à la prise en compte du dilemme entre exploration et exploitation). Tous ces aspects sont adressés par cette contribution qui traite primairement de l'approximation de la fonction de valeur ainsi que de la Q-fonction dans des processus décisionnels de Markov (PDM) déterministes. Les travaux présentés ici ont été précédemment partiellement publiés dans (Geist *et al.*, 2008b, 2009b). Nous nous plaçons ici dans le cadre de PDM déterministes, définis par un tuple $\{S, A, T, R, \gamma\}$ où S est l'espace d'état, A est l'espace d'action, $T : S \times A \to S$ est une fonction de transition déterministe, $R : S \times A \times S \to \mathbb{R}$ une fonction de récompense bornée et γ un coefficient d'actualisation. Une politique π associe à chaque état une action : $\pi : S \to A$. La fonction de valeur d'une politique

donnée est classiquement définie par :

$$V^{\pi}(s) = E[\sum_{i=0}^{\infty} \gamma^i r_i | s_0 = s, \pi] \tag{1}$$

où $r_i = R(s_i, a_i, s_{i+1})$ est la récompense associée à la transition (s_i, a_i, s_{i+1}) observée au temps i. La Q-fonction (ou fonction de qualité) est définie de façon semblable, avec cependant un degré de liberté supplémentaire sur les actions :

$$Q^{\pi}(s, a) = E[\sum_{i=0}^{\infty} \gamma^i r_i | s_0 = s, a_0 = a, \pi] \tag{2}$$

L'objectif de l'apprentissage par renforcement (AR) (Sigaud & Buffet, 2008; Sutton & Barto, 1998; Bertsekas & Tsitsiklis, 1996) est de déterminer (à travers des interactions entre un agent et un environnement) la politique π^* qui maximise la fonction de valeur pour chaque état :

$$\pi^* = \underset{\pi}{\text{argmax}}(V^{\pi}) \tag{3}$$

Deux schémas généraux (parmi d'autres) peuvent mener à la solution. Le premier, nommé *itération de la politique*, consiste à apprendre la fonction de valeur d'une politique donnée, puis à améliorer cette politique, la nouvelle étant gloutonne par rapport à la fonction de valeur apprise. Cela implique de résoudre l'*équation d'évaluation de Bellman*, donnée ici respectivement pour les fonctions de valeur et de qualité :

$$V^{\pi}(s) = R(s, \pi(s), s') + \gamma V^{\pi}(s'), \forall s \tag{4}$$

$$Q^{\pi}(s, a) = R(s, a, s') + \gamma Q^{\pi}(s', \pi(s')), \forall s, a \tag{5}$$

Ici et dans le reste de l'article, s' dénote l'état vers lequel transite le système, c'est-à-dire $s' = T(s, \pi(s))$ ou $s' = T(s, a)$ selon le contexte. Le second schéma, appelé *itération de la valeur*, permet de trouver directement la politique optimale (*via* la fonction de valeur optimale). Il implique de résoudre l'*équation d'optimalité de Bellman*, qui est donnée ici pour la Q-fonction :

$$Q^*(s, a) = R(s, a, s') + \gamma \max_{b \in A} Q^*(s', b), \forall s, a \tag{6}$$

L'objectif de cette contribution est de trouver une solution approchée aux équations de Bellman, pour la fonction de valeur ou de qualité, lorsque l'espace d'état est trop grand pour les approches classiques de programmation dynamique ou d'apprentissage par renforcement. De plus les algorithmes proposés sont en ligne et ne nécessitent pas de connaissances *a priori* du PDM, qui sont deux caractéristiques de l'AR que nous jugeons importantes.

Pour cela, nous considérons les méthodes dites de différences temporelles (ou TD pour *Temporal Differences*). Elles forment une classe d'algorithmes qui consistent à corriger la représentation de la fonction de valeur (ou de qualité) selon l'erreur de différence temporelle (l'erreur TD) faite sur cette dernière. La plupart de ces approches peuvent s'écrire de la façon générique suivante :

$$\theta_i = \theta_{i-1} + K_i \delta_i \tag{7}$$

Dans cette expression, θ_{i-1} est l'ancienne représentation de la fonction de valeur, θ_i est cette même représentation mise à jour selon la dernière transition observée, δ_i est l'erreur de différence temporelle et K_i est un gain indiquant dans quelle direction la représentation de la fonction de valeur doit être corrigée. Chacun de ces termes est à présent discuté.

Si les espaces d'état S et d'action A sont finis de cardinalité suffisamment faible, une représentation exacte de la fonction de valeur est possible, et θ est alors un vecteur avec autant de composantes qu'il y a d'états (ou de couples état-action pour la Q-fonction). C'est une représentation dite tabulaire. Si ces espaces sont trop larges, une approximation $\hat{V}(s)$ est nécessaire. Un choix classique en AR est la paramétrisation linéaire, pour laquelle la fonction de valeur est approchée comme suit :

$$\hat{V}_{\theta}(s) = \sum_{j=1}^{p} w_j \phi_j(s) \tag{8}$$

où $(\phi_j)_{1\leq j\leq p}$ est un ensemble de fonctions de base, qui doivent être définies à l'avance, et les paramètres sont les poids w_j :

$$\theta = [w_1, \ldots, w_p]^T \tag{9}$$

Beaucoup d'algorithmes d'approximation de la fonction de valeur nécessitent une telle représentation pour assurer la convergence (Schoknecht, 2002), ou même pour être applicable (Bradtke & Barto, 1996). D'autres représentations sont possibles, par exemple un réseau de neurones pour lequel θ est composé des poids des connexions synaptiques associées. Le cadre général de KTD permet de considérer n'importe quelle représentation de la fonction de valeur (ou de qualité), du moment qu'elle peut être totalement décrite par un ensemble fini de p paramètres.

Dans l'équation (7), le terme δ_i est l'erreur de différence temporelle. Supposons qu'au temps i la transition $(s_i, a_i, r_i, s_{i+1}, a_{i+1})$ soit observée. Pour les algorithmes d'AR de type TD, c'est-à-dire les algorithmes qui visent l'évaluation de la fonction de valeur pour une politique donnée π, l'erreur TD est :

$$\delta_i = r_i + \gamma \hat{V}_{\theta_{i-1}}(s_{i+1}) - \hat{V}_{\theta_{i-1}}(s_i) \tag{10}$$

Pour les algorithmes de type SARSA, c'est-à-dire les algorithmes qui ont pour but d'évaluer la fonction de qualité d'une politique donnée π, et étant donné l'approximation $\hat{Q}_{\theta_{i-1}}$ de la Q-fonction, l'erreur TD est :

$$\delta_i = r_i + \gamma \hat{Q}_{\theta_{i-1}}(s_{i+1}, a_{i+1}) - \hat{Q}_{\theta_{i-1}}(s_i, a_i) \tag{11}$$

Enfin, pour les algorithmes de type Q-learning, c'est-à-dire les algorithmes dont l'objectif est de calculer la Q-fonction optimale, l'erreur TD est de la forme suivante :

$$\delta_i = r_i + \gamma \max_{b\in A} \hat{Q}_{\theta_{i-1}}(s_{i+1}, b) - \hat{Q}_{\theta_{i-1}}(s_i, a_i) \tag{12}$$

Le type de différence temporelle utilisé est directement lié au type d'équation de Bellman à résoudre (équation d'évaluation pour (10) et (11), équation d'optimalité pour (12)), et donc si l'algorithme associé appartient à la famille de l'itération de la politique ou de la valeur.

Le terme K_i est un gain spécifique à chaque méthode. Certains des plus communs sont passés en revue dans cette section. Pour TD, SARSA et Q-learning (dans leur forme tabulaire, voir Sutton & Barto (1998) par exemple), le gain peut être écrit comme :

$$K_i = \alpha_i e_i \tag{13}$$

où α_i est un taux d'apprentissage classique dans le domaine de l'approximation stochastique, qui devrait vérifier :

$$\sum_{i=0}^{\infty} \alpha_i = \infty \text{ et } \sum_{i=0}^{\infty} \alpha_i^2 < \infty \tag{14}$$

et e_i est un vecteur unitaire, nul partout sauf en la composante correspondant à l'état s_i (ou à la paire état-action (s_i, a_i)) où elle vaut un (fonction de Kronecker). Ces algorithmes ont été étendus au concept de traces d'éligibilité (voir Sutton & Barto (1998) à nouveau), et le gain peut alors être écrit :

$$K_i = \alpha_i \sum_{j=1}^{i} \lambda^{i-j} e_j \tag{15}$$

où λ est le coefficient dit d'éligibilité. Ces algorithmes ont également été étendus à la prise en compte d'une représentation approchée de la fonction de valeur. Nous suivons Baird (1995) et les appelons algorithmes directs. Ils visent à minimiser un coût du type $\|V - \hat{V}\|^2$ avec une approche de type "bootstraping" (Sutton & Barto, 1998, Ch. 8.2.). Sans traces d'éligibilité, le gain s'écrit :

$$K_i = \alpha_i \nabla_{\theta_{i-1}} \hat{V}_{\theta_{i-1}}(s_i) \tag{16}$$

où $\nabla_{\theta_{i-1}} \hat{V}_{\theta_{i-1}}(s_i)$ est le gradient selon le vecteur de paramètres de la fonction de valeur paramétrée évaluée en l'état courant. La fonction de valeur peut être remplacée simplement par la fonction de qualité. Les algorithmes directs peuvent également prendre en compte les traces d'éligibilité :

$$K_i = \alpha_i \sum_{j=1}^{i} \lambda^{i-j} \nabla_{\theta_{i-1}} \hat{V}_{\theta_{i-1}}(s_j) \tag{17}$$

Une autre approche classique est celle des algorithmes résiduels (Baird, 1995), pour lesquels le gain est obtenu par la minimisation de la norme L_2 du résidu de Bellman (qui est la différence entre les deux membres d'une équation de Bellman) :

$$K_i = \alpha_i \nabla_{\theta_{i-1}} \left(\hat{V}_{\theta_{i-1}}(s_i) - \gamma \hat{V}_{\theta_{i-1}}(s_{i+1}) \right) \tag{18}$$

La dernière approche que nous passons en revue est l'algorithme LSTD (*Least Squares Temporal Differences*, Bradtke & Barto (1996)), qui n'est défini que pour une paramétrisation linéaire (8), et pour lequel le gain est défini récursivement :

$$K_i = \frac{C_{i-1}\phi(s_i)}{1 + (\phi(s_i) - \gamma\phi(s_{i+1}))^T C_{i-1}\phi(s_i)} \tag{19}$$

$$C_i = C_{i-1} - \frac{C_{i-1}\phi(s_i)(\phi(s_i) - \gamma\phi(s_{i+1}))^T C_{i-1}}{1 + (\phi(s_i) - \gamma\phi(s_{i+1}))^T C_{i-1}\phi(s_i)} \tag{20}$$

où $\phi(s)$ est défini par l'équation (42). Cet algorithme vise à minimiser le résidu quadratique de Bellman à l'aide d'une approche par moindres carrés et en utilisant des variables instrumentales (Söderström & Stoica, 2002) pour prendre en compte la stochasticité du MDP. Il a également été étendu aux traces d'éligibilité, voir Boyan (1999) pour les détails.

Le problème traité dans cette contribution peut être résumé comme suit : étant donné une représentation de la fonction de valeur (ou de qualité) décrite par le vecteur θ et un schéma d'erreur de différence temporelle (ou de façon équivalente une équation de Bellman à résoudre), quel est le meilleur gain K ? Pour répondre à cette question, un point de vue statistique est adopté et le cadre du filtre de Kalman est suivi (Kalman, 1960). L'approche proposée peut être liée à des travaux sur les processus gaussiens (Engel, 2005), sur les moindres carrés (Bradtke & Barto, 1996) ou sur le filtrage de Kalman (Choi & Roy, 2006; Phua & Fitch, 2007), ce qui sera développé section 6. La section suivante présente la structure générale des différences temporelles de Kalman (KTD). La section 3 la spécialise pour dériver un certain nombre d'algorithmes pour l'évaluation de la fonction de valeur et de qualité (KTD-V et KTD-SARSA) et pour l'optimisation directe de la Q-fonction (KTD-Q). Un schéma d'approximation, la transformation non-parfumée de Julier & Uhlmann (2004), nécessaire pour prendre en compte les paramétrisations non-linéaires et l'équation d'optimalité de Bellman, est également présenté. La section 4 introduit une forme d'apprentissage actif utilisant l'information d'incertitude disponible dans ce cadre de travail. Un certain nombre d'expérimentations sont menées dans la section 5, de façon à illustrer les différents aspects de cette contribution et à la comparer à l'état de l'art. La section 6 discute les avantages et inconvénients de l'approche proposée et la compare à certains travaux similaires. Enfin, la dernière section présente quelques perspectives.

2 Différences temporelles de Kalman : cas général

Dans cette section, un point de vue très général est adopté, les algorithmes spécifiques étant dérivés par la suite. Pour l'instant, une transition est notée de façon générique par :

$$t_i = \begin{cases} (s_i, s_{i+1}) \\ (s_i, a_i, s_{i+1}, a_{i+1}) \\ (s_i, a_i, s_{i+1}) \end{cases} \tag{21}$$

selon que l'objectif soit l'évaluation de la fonction de valeur, de qualité, ou l'optimisation de Q. De façon similaire, pour les mêmes cas, les notations suivantes sont adoptées :

$$g_{t_i}(\theta_i) = \begin{cases} \hat{V}_{\theta_i}(s_i) - \gamma \hat{V}_{\theta_i}(s_{i+1}) \\ \hat{Q}_{\theta_i}(s_i, a_i) - \gamma \hat{Q}_{\theta_i}(s_{i+1}, a_{i+1}) \\ \hat{Q}_{\theta_i}(s_i, a_i) - \gamma \max_{b \in A} \hat{Q}_{\theta_i}(s_{i+1}, b) \end{cases} \tag{22}$$

Ainsi, tous les schémas de différences temporelles peuvent s'écrire de façon générique :

$$\delta_i = r_i - g_{t_i}(\theta_i) \tag{23}$$

Avec ces notations, $g_{t_i}(\theta_i)$ peut être vu comme la prédiction de la récompense au temps i en accord avec la représentation θ_i, et δ_i peut être vu comme un terme d'innovation qui quantifie l'information gagnée en observant la nouvelle récompense r_i. Ces notions sont raffinées plus tard.

Comme annoncé, un point de vue statistique est adopté. Le vecteur de paramètre θ est modélisé comme étant une variable aléatoire suivant une marche aléatoire. Le problème peut ainsi s'exprimer sous une forme dite *espace d'état* :

$$\begin{cases} \theta_i = \theta_{i-1} + v_i & \text{(équation d'évolution)} \\ r_i = g_{t_i}(\theta_i) + n_i & \text{(équation d'observation)} \end{cases} \tag{24}$$

La dénomination espace d'état (*state-space* en anglais) vient du filtrage de Kalman et n'a pas de lien avec la notion d'état d'un PDM. Originellement, le filtrage de Kalman a pour but d'inférer l'état caché d'un système à partir d'observations. Dans cette contribution, cet état caché est le vecteur de paramètres et les observations sont les récompenses et transitions.

Cette formulation est fondamentale pour le paradigme proposé. La première équation est l'équation dite d'évolution, elle spécifie que le vecteur de paramètres suit une marche aléatoire dont la moyenne correspond à l'estimation optimale. Le bruit d'observation v_i est centré, blanc, indépendant et de variance P_{v_i} (choisie par le praticien). Notons d'une part que cette équation n'est pas une mise à jour des paramètres (qui sera abordée plus tard) mais leur évolution naturelle au cours du temps, et d'autre part que cette formulation permet de prendre en compte la non-stationnarité éventuelle du PDM. La seconde équation est l'équation dite d'observation, elle lie la transition observée ainsi que la récompense associée à la fonction de valeur (ou de qualité) à l'aide d'une des équations de Bellman. Le bruit d'observation n_i est supposé blanc, centré, indépendant et de variance P_{n_i} (également choisie par le praticien). Notons que cette hypothèse n'est pas vérifiée pour un PDM ayant des transitions stochastiques, ce qui est la motivation principale de l'hypothèse de transitions déterministes. Ce point sera discuté plus avant dans la section 2.4. Le modèle du bruit d'observation résulte du fait que la solution de l'équation de Bellman considérée n'appartient pas nécessairement à l'espace fonctionnel engendré par l'ensemble des paramètres (la structure d'approximation étant fixée *a priori*).

2.1 Coût minimisé

L'objectif pourrait être d'estimer le vecteur de paramètres qui minimise l'espérance de l'erreur quadratique conditionnée aux récompenses observées depuis l'origine des temps. Le coût associé s'écrit :

$$J_i(\theta) = E\left[\|\theta_i - \theta\|^2 | r_{1:i}\right] \text{ avec } r_{1:i} = r_1, \ldots, r_i \tag{25}$$

De façon générale, l'estimateur minimisant l'erreur quadratique moyenne est l'espérance conditionnelle :

$$\underset{\theta}{\operatorname{argmin}}\, J_i(\theta) = \hat{\theta}_{i|i} = E\left[\theta_i | r_{1:i}\right] \tag{26}$$

Cependant, à part pour des cas spécifiques (notamment le cas où les équations d'évolution et d'observation sont linéaires et les bruits gaussiens), cet estimateur ne peut pas être calculé analytiquement. A la place, l'objectif est ici de trouver le meilleur estimateur *linéaire*. Il peut être écrit sous une forme similaire à l'équation (7) :

$$\hat{\theta}_{i|i} = \hat{\theta}_{i|i-1} + K_i \tilde{r}_i \tag{27}$$

Dans l'équation (27), $\hat{\theta}_{i|i}$ est l'estimation au temps i, $\hat{\theta}_{i|i-1} = E[\theta_i | r_{1:i-1}]$ est la prédiction de cette estimation en accord avec les récompenses observées dans le passé $r_{1:i-1}$. Pour le modèle de marche aléatoire adopté, nous avons $\hat{\theta}_{i|i-1} = \hat{\theta}_{i-1|i-1}$. En effet, en utilisant la définition de la prédiction et l'équation d'évolution nous avons $\hat{\theta}_{i|i-1} = E[\theta_{i-1} + v_i | r_{1:i-1}]$. Or le bruit d'évolution est blanc est centré donc $E[v_i | r_{1:i-1}] = 0$. Ceci mène au résultat : $\hat{\theta}_{i|i-1} = E[\theta_{i-1} | r_{1:i-1}] = \hat{\theta}_{i-1|i-1}$. L'innovation

$$\tilde{r}_i = r_i - \hat{r}_{i|i-1} \tag{28}$$

est la différence entre la récompense observée r_i et sa prédiction basée sur la précédente estimation du vecteur de paramètres, donnée par (rappelons que le bruit d'observation est également blanc et centré) :

$$\hat{r}_{i|i-1} = E\left[r_i|r_{1:i-1}\right] = E\left[g_{t_i}(\theta_i) + n_i|r_{1:i-1}\right] = E\left[g_{t_i}(\theta_i)|r_{1:i-1}\right] \tag{29}$$

Notons que cette innovation n'est pas exactement l'erreur de différence temporelle définie dans l'équation (23), qui est une variable aléatoire en conséquence de sa dépendance au vecteur aléatoire θ_i : c'est son espérance conditionnée aux données précédemment observées. Etant donnée la mise à jour postulée (27), il s'agit de déterminer le gain K_i qui permette la minimisation du coût (25).

2.2 Gain optimal

En utilisant des égalités classiques, la fonction de coût peut se réécrire de la façon suivante (l'opérateur trace associant à une matrice carrée la somme de ses éléments diagonaux) :

$$\begin{aligned} J_i(\theta) &= E\left[\|\theta_i - \theta\|^2|r_{1:i}\right] = E\left[(\theta_i - \theta)^T(\theta_i - \theta)|r_{1:i}\right] \\ &= \text{trace}\left(E\left[(\theta_i - \theta)(\theta_i - \theta)^T|r_{1:i}\right]\right) = \text{trace}\left(\text{cov}\left(\theta_i - \theta|r_{1:i}\right)\right) \end{aligned} \tag{30}$$

Une première étape pour calculer le gain optimal est d'exprimer la covariance de l'erreur sur les paramètres conditionnées aux récompenses comme une fonction du gain K_i. Mais d'abord quelques notations supplémentaires sont introduites (rappelons également la définition de l'innovation (28)) :

$$\begin{cases} \tilde{\theta}_{i|i} = \theta_i - \hat{\theta}_{i|i} & \text{et} \quad \tilde{\theta}_{i|i-1} = \theta_i - \hat{\theta}_{i|i-1} \\ P_{i|i} = \text{cov}\left(\tilde{\theta}_{i|i}|r_{1:i}\right) & \text{et} \quad P_{i|i-1} = \text{cov}\left(\tilde{\theta}_{i|i-1}|r_{1:i-1}\right) \\ P_{r_i} = \text{cov}\left(\tilde{r}_i|r_{i|i-1}\right) & \text{et} \quad P_{\theta r_i} = E\left[\tilde{\theta}_{i|i-1}\tilde{r}_i|r_{1|i-1}\right] \end{cases} \tag{31}$$

En utilisant la mise à jour postulée (27), et les différents estimateurs étant non-biaisés, la covariance peut être expansée :

$$\begin{aligned} P_{i|i} &= \text{cov}\left(\theta_i - \hat{\theta}_{i|i}|r_{1:i}\right) = \text{cov}\left(\theta_i - \left(\hat{\theta}_{i|i-1} + K_i\tilde{r}_i\right)|r_{1:i-1}\right) \\ &= \text{cov}\left(\tilde{\theta}_{i|i-1} - K_i\tilde{r}_i|r_{1:i-1}\right) = P_{i|i-1} - P_{\theta r_i}K_i^T - K_iP_{\theta r_i}^T + K_iP_{r_i}K_i^T \end{aligned} \tag{32}$$

Le gain optimal peut ainsi être obtenu en annulant le gradient de la trace de cette matrice.

Notons tout d'abord que le gradient étant linéaire, pour trois matrices de dimensions *ad hoc* A, B et C, B étant symétrique, nous avons les identités algébriques suivantes :

$$\nabla_A\left(\text{trace}\left(ABA^T\right)\right) = 2AB \text{ et } \nabla_A\left(\text{trace}\left(AC^T\right)\right) = \nabla_A\left(\text{trace}\left(CA^T\right)\right) = C \tag{33}$$

et donc en utilisant l'équation (32) et les identités précédentes nous avons :

$$\nabla_{K_i}\left(\text{trace}\left(P_{i|i}\right)\right) = 0 \Leftrightarrow \quad K_i = P_{\theta r_i}P_{r_i}^{-1} \tag{34}$$

En injectant le gain optimal (34) dans l'expression de la matrice de covariance de l'erreur conditionnée aux récompenses observées (32), nous en obtenons une expressions simplifiée :

$$P_{i|i} = P_{i|i-1} - K_iP_{r_i}K_i^T \tag{35}$$

Il est à noter qu'aucune hypothèse gaussienne n'a été faite pour obtenir ces résultats. Remarquons tout de même que dans le cas linéaire et gaussien, la mise à jour (27) que nous contraignons à la linéarité l'est vraiment.

2.3 Algorithme général

L'algorithme le plus général de différences temporelles de Kalman, qui se subdivise en trois parties, peut maintenant être obtenu. La première étape consiste à calculer les prédictions $\hat{\theta}_{i|i-1}$ et $P_{i|i-1}$. Rappelons que pour un modèle de marche aléatoire la prédiction du vecteur de paramètres

est égale à son estimation précédente : $\hat{\theta}_{i|i-1} = \hat{\theta}_{i-1|i-1}$. La covariance prédite peut également être calculée analytiquement :

$$P_{i|i-1} = \text{cov}\left(\tilde{\theta}_{i|i-1}|r_{1:i-1}\right) = \text{cov}\left(\tilde{\theta}_{i-1|i-1} + v_{i-1}|r_{1:i-1}\right) = P_{i-1|i-1} + P_{v_{i-1}} \tag{36}$$

La seconde étape consiste à calculer quelques statistiques d'intérêt. C'est principalement cette partie qui sera spécialisée dans la section suivante. La première statistique à calculer est la prédiction de la récompense $\hat{r}_{i|i-1}$ (29). La seconde est la covariance entre l'erreur sur les paramètres et l'innovation :

$$P_{\theta r_i} = E\left[(\theta_i - \hat{\theta}_{i|i-1})(r_i - \hat{r}_{i|i-1})|r_{1:i-1}\right] \tag{37}$$

Etant donné la forme de l'équation d'observation ($r_i = g_{t_i}(\theta_i) + n_i$) et l'indépendance du bruit d'observation qui est centré, cette statistique peut se réécrire :

$$P_{\theta r_i} = E\left[(\theta_i - \hat{\theta}_{i|i-1})(g_{t_i}(\theta_i) - \hat{r}_{i|i-1})|r_{1:i-1}\right] \tag{38}$$

Enfin, la dernière statistique à calculer est la variance de l'innovation, qui peut être écrite (en utilisant à nouveau les caractéristiques du bruit d'observation) :

$$\begin{aligned} P_{r_i} &= E\left[(r_i - \hat{r}_{i|i-1})^2|r_{1:i-1}\right] = E\left[(g_{t_i}(\theta_i) - \hat{r}_{i|i-1} + n_i)^2|r_{1:i-1}\right] \\ &= E\left[(g_{t_i}(\theta_i) - \hat{r}_{i|i-1})^2|r_{1:i-1}\right] + P_{n_i} \end{aligned} \tag{39}$$

La dernière étape de l'algorithme est la phase de correction qui consiste à calculer le gain (34) et à mettre à jour le vecteur de paramètres (27) ainsi que la matrice de covariance associée (35). Notons que la méthode proposée étant en ligne, elle se doit d'être initialisée avec un *a priori* sur l'espérance $\hat{\theta}_{0|0}$ et la covariance $P_{0|0}$ des paramètres. L'approche générale proposée est résumée dans l'algorithme 1. Le calcul de la variance $P_{i|i}$ peut sembler à première vue inutile, mais elle est utilisée dans tous les cas (linéaire, linéarisé, non-linéaire) pour calculer les statistiques d'intérêt, comme explicité dans la section 3.

Algorithme 1 : Algorithme KTD général.

Initialisation : *a priori* $\hat{\theta}_{0|0}$ et $P_{0|0}$;

pour $i \leftarrow 1, 2, \ldots$ **faire**

Observer la transition t_i ainsi que la récompense associée r_i ;

Phase de prédiction;

$\hat{\theta}_{i|i-1} = \hat{\theta}_{i-1|i-1}$;

$P_{i|i-1} = P_{i-1|i-1} + P_{v_{i-1}}$;

Calcul des statistiques d'intérêt;

$\hat{r}_{i|i-1} = E[g_{t_i}(\theta_i)|r_{1:i-1}]$;

$P_{\theta r_i} = E\left[(\theta_i - \hat{\theta}_{i|i-1})(g_{t_i}(\theta_i) - \hat{r}_{i|i-1})|r_{1:i-1}\right]$;

$P_{r_i} = E\left[(g_{t_i}(\theta_i) - \hat{r}_{i|i-1})^2|r_{1:i-1}\right] + P_{n_i}$;

Phase de correction;

$K_i = P_{\theta r_i} P_{r_i}^{-1}$;

$\hat{\theta}_{i|i} = \hat{\theta}_{i|i-1} + K_i\left(r_i - \hat{r}_{i|i-1}\right)$;

$P_{i|i} = P_{i|i-1} - K_i P_{r_i} K_i^T$;

2.4 Transitions stochastiques

Le cadre de travail de KTD fait l'hypothèse de transitions déterministes. Si celle-ci n'est pas vérifiée, le bruit d'observation n_i peut ne pas être blanc (dans la mesure où il inclurait la stochasticité du PDM en plus de l'erreur de modélisation). Malheureusement, la blancheur de ce bruit est une

hypothèse nécessaire à l'obtention de KTD. Il est toujours envisageable d'ignorer le problème et de considérer la fonction de coût (25) lié au modèle espace d'état (24) pour un PDM stochastique. Cependant, de façon similaire aux approches minimisant le carré du résidu de Bellman, comme les algorithmes résiduels de Baird (1995), cette fonction de coût est biaisée. Il est possible de montrer que ce biais vaut (donné ici pour l'évaluation de la fonction de valeur, l'extension aux autres cas étant relativement simple) :

$$\text{trace}\left(K_i E\left[\underset{s'|s_i}{\text{cov}}\left(r_i+\gamma V_\theta(s')\right)|r_{1:i-1}\right]K_i^T\right)=\|K_i\|^2 E\left[\underset{s'|s_i}{\text{cov}}\left(r_i+\gamma V_\theta(s')\right)|r_{1:i-1}\right] \tag{40}$$

où K_i est le gain de Kalman, $\|.\|$ est la norme euclidienne usuelle, la covariance dépend des probabilités de transition et l'espérance est sur les paramètres conditionnés aux observations passées.

Ce biais est similaire à celui causé par la minimisation quadratique du résidu de Bellman, la différence résidant en la présence du gain de Kalman et en la stochasticité du vecteur de paramètre. Une idée pourrait être d'introduire un filtre auxiliaire de façon à supprimer le biais, de façon similaire à l'introduction d'une fonction auxiliaire proposée par Antos *et al.* (2008). Cependant l'extension de ces travaux à notre cas est difficile, notamment à cause de l'aspect non-quadratique de la nouvelle fonction de coût. Une autre approche pourrait être d'estimer ce biais en ligne pour le soustraire, comme le font Jo & Kim (2005) dans le cadre du filtrage par moindre-carrés. Une autre idée pourrait être d'étendre le concept des variables instrumentales (Söderström & Stoica, 2002), qui est la base de LSTD dans sa première formulation (Bradtke & Barto, 1996), au filtrage de Kalman. Aucune de ces deux approches n'est évidente. Enfin, une solution pourrait être d'introduire un bruit coloré, comme le font Engel *et al.* (2005) dans un contexte bayésien. Geist *et al.* (2009a) développent cette dernière approche pour KTD. Dans le reste de cette contribution nous continuons à supposer les transitions déterministes.

3 Spécialisations

La principale difficulté de KTD est le calcul des statistiques d'intérêt $\hat{r}_{i|i-1}$, $P_{\theta r_i}$ et P_{r_i} (pour lesquelles les statistiques $\hat{\theta}_{i|i-1}$ et $P_{i|i-1}$ sont nécessaires). Ceci fait l'objet de la présente section.

3.1 KTD-V

Le premier problème d'intérêt est l'évaluation de la fonction de valeur, c'est-à-dire trouver une solution approchée de l'équation d'évaluation de Bellman (4). Dans un premier temps le cas linéaire est considéré, ce qui permet de calculer les statistiques d'intérêt de façon analytique. Un schéma d'approximation, la transformation non-parfumée (*unscented transform* en anglais) de Julier & Uhlmann (2004), est ensuite présenté. Ce dernier se montre utile pour traiter le cas non linéaire.

3.1.1 Paramétrisation linéaire

Dans cette section, la paramétrisation linéaire de l'équation (8) est adoptée, que l'on réécrit :

$$\hat{V}_\theta(s)=\phi(s)^T\theta \tag{41}$$

Le vecteur de paramètres est donné dans l'équation (9) et $\phi(s)$ est un vecteur défini par :

$$\phi(s)=\left[\phi_1(s),\ldots,\phi_p(s)\right]^T \tag{42}$$

La formulation espace d'état (24) peut donc se réécrire :

$$\begin{cases}\theta_i=\theta_{i-1}+v_i\\ r_i=\left(\phi(s_i)-\gamma\phi(s_{i+1})\right)^T\theta_i+n_i\end{cases} \tag{43}$$

Pour raccourcir les équations, nous définissons $H_i=\phi(s_i)-\gamma\phi(s_{i+1})$. Comme l'équation d'observation est linéaire, les statistiques d'intérêt peuvent être calculées analytiquement. La prédiction de la récompense est donnée par :

$$\hat{r}_{i|i-1}=E\left[g_{t_i}(\theta_i)|r_{1:i-1}\right]=E\left[H_i^T\theta_i|r_{1:i-1}\right]=H_i^T E\left[\theta_i|r_{1:i-1}\right]=H_i^T\hat{\theta}_{i|i-1} \tag{44}$$

La covariance entre l'erreur sur les paramètres et l'innovation peut également être calculée analytiquement :

$$\begin{aligned} P_{\theta r_i} &= E\left[\tilde{\theta}_{i|i-1}\left(g_{t_i}(\theta_i) - \hat{r}_{i|i-1}\right)|r_{1:i-1}\right] = E\left[\tilde{\theta}_{i|i-1}H_i^T\tilde{\theta}_{i|i-1}|r_{1:i-1}\right] = E\left[\tilde{\theta}_{i|i-1}\tilde{\theta}_{i|i-1}^T|r_{1:i-1}\right]H_i \\ &= P_{i|i-1}H_i \end{aligned} \tag{45}$$

Enfin, la variance de l'innovation est également calculable :

$$\begin{aligned} P_{r_i} &= E\left[\left(g_{t_i}(\theta_i) - \hat{r}_{i|i-1}\right)^2|r_{1:i-1}\right] + P_{n_i} = E\left[\left(H_i^T\tilde{\theta}_{i|i-1}\right)^2|r_{1:i-1}\right] + P_{n_i} \\ &= H_i^T P_{i|i-1}H_i + P_{n_i} \end{aligned} \tag{46}$$

Le gain peut ainsi être défini de façon algébrique et récursive :

$$K_i = \frac{P_{i|i-1}H_i}{H_i^T P_{i|i-1}H_i + P_{n_i}} \tag{47}$$

Ce gain partage des similarités avec le gain de l'algorithme LSTD de Bradtke & Barto (1996), donné équation (19), ce qui n'est pas une surprise. En effet, LSTD est basé sur une minimisation par moindres carrés (cependant avec l'introduction du concept de variable instrumentale pour prendre en compte le cas des transitions stochastiques), et le filtre de Kalman peut être vu comme une généralisation de cette approche (sans variables instrumentales). Ce gain ressemble également à celui émergeant d'une modélisation paramétrique par des processus gaussiens du problème d'évaluation de la valeur (GPTD paramétrique d'Engel (2005)). L'algorithme KTD-V pour une paramétrisation linéaire est résumé dans l'algorithme 2.

Algorithme 2 : KTD-V : paramétrisation linéaire.

Initialisation : *a priori* $\hat{\theta}_{0|0}$ et $P_{0|0}$;

pour $i \leftarrow 1, 2, \ldots$ **faire**

Observer la transition (s_i, s_{i+1}) ainsi que la récompense associée r_i ;

Phase de prédiction;
$\hat{\theta}_{i|i-1} = \hat{\theta}_{i-1|i-1}$;
$P_{i|i-1} = P_{i-1|i-1} + P_{v_{i-1}}$;

Calcul des statistiques d'intérêt;
$\hat{r}_{i|i-1} = H_i^T\hat{\theta}_{i|i-1}$;
$P_{\theta r_i} = P_{i|i-1}H_i$;
$P_{r_i} = H_i^T P_{i|i-1}H_i + P_{n_i}$;
/* où $H_i = \phi(s_i) - \gamma\phi(s_{i+1})$ */

Phase de correction;
$K_i = P_{\theta r_i}P_{r_i}^{-1}$;
$\hat{\theta}_{i|i} = \hat{\theta}_{i|i-1} + K_i\left(r_i - \hat{r}_{i|i-1}\right)$;
$P_{i|i} = P_{i|i-1} - K_i P_{r_i}K_i^T$;

Le problème suivant est l'évaluation de la fonction de valeur lorsque la paramétrisation est non-linéaire. Basiquement, le problème du calcul des statistiques d'intérêt peut être exprimé de la façon suivante : étant donné la moyenne et la covariance d'une variable aléatoire donnée, comment calculer la moyenne et la covariance d'une transformation non-linéaire (voir éventuellement non dérivable) de cette variable aléatoire ? La prochaine section présente la transformation non-parfumée, qui est un schéma d'approximation permettant de faire face à ce genre de problème.

3.1.2 Transformation non-parfumée

Laissons pour l'instant de côté l'AR et le filtrage de Kalman. Soit X un vecteur aléatoire, et Y une fonction de X. Le problème posé est de calculer la moyenne et la variance de Y en connaissant

celles de X ainsi que la fonctionnelle les liant. Si cette dernière est linéaire, la relation entre X et Y peut s'écrire $Y = AX$ où A est une matrice de dimension *ad hoc*. Dans ce cas, moyenne et covariance peuvent être calculées analytiquement : $E[Y] = AE[X]$ et $E[YY^T] = AE[XX^T]A^T$. Ce résultat a été utilisé pour obtenir l'algorithme KTD-V de la section 3.1.1.

Si la transformation est non-linéaire, elle peut génériquement se mettre sous la forme $Y = f(X)$. Une première solution est d'approximer la fonctionnelle même, c'est-à-dire de la linéariser autour de la moyenne du vecteur aléatoire X. Cela mène aux approximations suivantes pour la moyenne et la covariance de Y : $E[Y] \approx f(E[X])$ et $E[YY^T] \approx (\nabla f(E[X]))\, E[XX^T]\, (\nabla f(E[X]))^T$. Ceci est la base du filtrage de Kalman étendu (EKF pour *Extended Kalman Filtering*, voir Simon (2006) par exemple), qui a été intensivement étudié et utilisé dans les décennies passées. Cependant, cette approche présente quelques sévères limitations. Premièrement, elle ne permet pas de prendre en compte des non-linéarités non dérivables, et ne peut donc pas prendre en compte l'équation d'optimalité de Bellman (6) à cause de l'opérateur max. Cette approche nécessite également d'évaluer le gradient de f, ce qui peut être compliqué voire impossible. Enfin, cela suppose que f est localement linéarisable, ce qui n'est malheureusement pas toujours le cas et peut mener à de mauvaises approximations, comme illustré par Julier & Uhlmann (2004).

L'idée de base de la transformation non-parfumée est qu'il est plus judicieux d'approximer un vecteur aléatoire arbitraire qu'une fonction non-linéaire arbitraire. Son principe est d'échantillonner de façon *déterministe* un ensemble de "sigma-points" à partir de la moyenne et de la covariance de X. Les images de ces sigma-points par l'application f sont ensuite calculées, et elles sont utilisées pour calculer les statistiques d'intérêt. Ce schéma d'approximation ressemble aux méthodes de Monte-Carlo, cependant ici l'échantillonnage est déterministe et nécessite la génération de moins d'échantillons, en permettant cependant une précision donnée (Julier & Uhlmann, 2004).

Nous décrivons à présent la transformation non-parfumée originale. D'autres variantes ont été introduites depuis, mais le principe de base est le même. Soit n la dimension de X. Un ensemble de $2n+1$ sigma-points et poids associés est calculé comme suit :

$$\begin{cases} x^{(0)} = \bar{X} & w_0 = \frac{\kappa}{n+\kappa}, \quad j = 0 \\ x^{(j)} = \bar{X} + \left(\sqrt{(n+\kappa)P_X}\right)_j & w_j = \frac{1}{2(n+\kappa)}, \quad 1 \leq j \leq n \\ x^{(j)} = \bar{X} - \left(\sqrt{(n+\kappa)P_X}\right)_{n-j} & w_j = \frac{1}{2(n+\kappa)}, \quad n+1 \leq j \leq 2n \end{cases} \tag{48}$$

où $\bar{X}$ est la moyenne de X, P_X est sa matrice de variance, κ est un coefficient d'échelle permettant de contrôler la précision de la transformation non-parfumée (Julier & Uhlmann, 2004), et $\left(\sqrt{(n+\kappa)P_X}\right)_j$ est la $j^{\text{ème}}$ colonne de la décomposition de Cholesky de la matrice $(n+\kappa)P_X$. L'image de chacun de ces points par f est ensuite calculée : $y^{(j)} = f(x^{(j)})$, $0 \leq j \leq 2n$. L'ensemble des sigma-points et de leurs images peut alors être utilisé pour calculer les moments d'ordre un et deux de Y, et même P_{XY}, la covariance entre X et Y :

$$\begin{cases} \bar{Y} \approx \bar{y} = \sum_{j=0}^{2n} w_j y^{(j)} \\ P_Y \approx \sum_{j=0}^{2n} w_j \left(y^{(j)} - \bar{y}\right)\left(y^{(j)} - \bar{y}\right)^T \\ P_{XY} \approx \sum_{j=0}^{2n} w_j \left(x^{(j)} - \bar{X}\right)\left(y^{(j)} - \bar{y}\right)^T \end{cases} \tag{49}$$

La transformation non-parfumée ayant été présentée, nous nous intéressons au problème d'évaluation de la fonction de valeur dans le cas d'une paramétrisation non-linéaire.

3.1.3 Paramétrisation non linéaire

Dans cette section, nous considérons une paramétrisation générique de la fonction de valeur $\hat{V}_\theta$: cela peut être un réseau de neurones (Bishop, 1995), une représentation par noyaux semi-paramétrique (Geist *et al.*, 2008a), ou toute autre représentation d'intérêt, du moment qu'elle puisse être décrite par un ensemble fini de p paramètres. La formulation générale espace d'état de l'équation (24) s'écrit dans ce cas :

$$\begin{cases} \theta_i = \theta_{i-1} + v_i \\ r_i = \hat{V}_{\theta_i}(s_i) - \gamma \hat{V}_{\theta_i}(s_{i+1}) + n_i \end{cases} \tag{50}$$

Le problème est toujours de calculer les statistiques d'intérêt, ce qui est fait ici en utilisant la transformation non-parfumée. La première chose à faire est de calculer l'ensemble des sigma-points à partir des statistiques connues $\hat{\theta}_{i|i-1}$ et $P_{i|i-1}$, ainsi que les poids associés :

$$\Theta_{i|i-1} = \left\{ \hat{\theta}^{(j)}_{i|i-1},\ 0 \leq j \leq 2p \right\} \tag{51}$$

$$\mathcal{W} = \{ w_j,\ 0 \leq j \leq 2p \} \tag{52}$$

Ensuite les images de ces sigma-points sont calculées en utilisant l'équation d'observation du modèle espace d'état (50), qui est lié à l'équation d'évaluation de Bellman (4) :

$$\mathcal{R}_{i|i-1} = \left\{ \hat{r}^{(j)}_{i|i-1} = \hat{V}_{\hat{\theta}^{(j)}_{i|i-1}}(s_i) - \gamma \hat{V}_{\hat{\theta}^{(j)}_{i|i-1}}(s_{i+1}),\ 0 \leq j \leq 2p \right\} \tag{53}$$

Enfin, les sigma-points et leurs images étant calculés, les statistiques d'intérêt peuvent être approchées par :

$$\hat{r}_{i|i-1} \approx \sum_{j=0}^{2p} w_j \hat{r}^{(j)}_{i|i-1} \tag{54}$$

$$P_{r_i} \approx \sum_{j=0}^{2p} w_j \left(\hat{r}^{(j)}_{i|i-1} - \hat{r}_{i|i-1} \right)^2 + P_{n_i} \tag{55}$$

$$P_{\theta r_i} \approx \sum_{j=0}^{2p} w_j \left(\hat{\theta}^{(j)}_{i|i-1} - \hat{\theta}_{i|i-1} \right) \left(\hat{r}^{(j)}_{i|i-1} - \hat{r}_{i|i-1} \right) \tag{56}$$

Etant donné que la transformation non-parfumée n'est plus une approximation dans le cas d'une transformation linéaire, cette formulation est toujours valide pour l'évaluation de la fonction de valeur avec une paramétrisation linéaire. L'algorithme 3 résume l'approche.

3.2 KTD-SARSA

Cette section traite du problème de l'évaluation de la fonction de qualité pour une politique donnée. L'algorithme associé est appelé KTD-SARSA, ce qui peut induire en erreur. En effet, de façon générale, le terme SARSA est associé à l'évaluation de la Q-fonction dans le cadre d'une itération optimiste de la politique (politique ϵ-gloutonne par exemple). Ici nous nous focalisons sur l'aspect évaluation de la fonction de qualité, en laissant le contrôle de côté (ce problème étant tout de même considéré dans la section 5). Pour une paramétrisation générale $\hat{Q}_\theta$, en considérant l'équation d'évaluation de Bellman (5), le modèle espace d'état (24) s'écrit :

$$\begin{cases} \theta_i = \theta_{i-1} + v_i \\ r_i = \hat{Q}_{\theta_i}(s_i, a_i) - \gamma \hat{Q}_{\theta_i}(s_{i+1}, a_{i+1}) + n_i \end{cases} \tag{57}$$

Pour une politique fixée, l'évaluation de la Q-fonction sur la chaîne de Markov valuée induite par l'espace état-action est similaire au problème de l'évaluation de la fonction de valeur sur la chaîne de Markov valuée induite par l'espace d'état. Il est alors assez évident d'étendre KTD-V à l'évaluation de la Q-fonction. Nous rappelons que pour une application linéaire, la transformation non-parfumée permet le calcul exact des statistiques d'intérêt, et dans ce cas les algorithmes sous forme algébrique et non-parfumée sont équivalents. C'est pourquoi nous ne donnons que la formulation par sigma-points de KTD-SARSA, également résumée dans l'algorithme 3.

3.3 KTD-Q

Cette section s'intéresse à l'optimisation directe de la Q-fonction, c'est-à-dire à trouver une solution approchée de l'équation d'optimalité de Bellman (6). Une paramétrisation générale $\hat{Q}_\theta$ est adoptée. Le modèle espace d'état (24) est alors spécialisé comme suit :

$$\begin{cases} \theta_i = \theta_{i-1} + v_i \\ r_i = \hat{Q}_{\theta_i}(s_i, a_i) - \gamma \max_{b \in A} \hat{Q}_{\theta_i}(s_{i+1}, b) + n_i \end{cases} \tag{58}$$

Algorithme 3 : KTD-V, KTD-SARSA et KTD-Q.

Initialisation : *a priori* $\hat{\theta}_{0|0}$ et $P_{0|0}$;

pour $i \leftarrow 1, 2, \ldots$ **faire**

Observer la transition $t_i = \begin{cases} (s_i, s_{i+1}) \text{ (KTD-V)} \\ (s_i, a_i, s_{i+1}, a_{i+1}) \text{ (KTD-SARSA)} \\ (s_i, a_i, s_{i+1}) \text{ (KTD-Q)} \end{cases}$ ainsi que la récompense r_i ;

Phase de prédiction;

$\hat{\theta}_{i|i-1} = \hat{\theta}_{i-1|i-1}$;

$P_{i|i-1} = P_{i-1|i-1} + P_{v_{i-1}}$;

Calcul des sigma-points ;

$\Theta_{i|i-1} = \left\{ \hat{\theta}^{(j)}_{i|i-1}, \quad 0 \leq j \leq 2p \right\}$ (en utilisant $\hat{\theta}_{i|i-1}$ et $P_{i|i-1}$);

$\mathcal{W} = \{ w_j, \quad 0 \leq j \leq 2p \quad \}$;

$\mathcal{R}_{i|i-1} = \begin{cases} \left\{ \hat{r}^{(j)}_{i|i-1} = \hat{V}_{\hat{\theta}^{(j)}_{i|i-1}}(s_i) - \gamma \hat{V}_{\hat{\theta}^{(j)}_{i|i-1}}(s_{i+1}),\ 0 \leq j \leq 2p \right\} \text{ (KTD-V)} \\ \left\{ \hat{r}^{(j)}_{i|i-1} = \hat{Q}_{\hat{\theta}^{(j)}_{i|i-1}}(s_i, a_i) - \gamma \hat{Q}_{\hat{\theta}^{(j)}_{i|i-1}}(s_{i+1}, a_{i+1}),\ 0 \leq j \leq 2p \right\} \text{ (KTD-SARSA)} \\ \left\{ \hat{r}^{(j)}_{i|i-1} = \hat{Q}_{\hat{\theta}^{(j)}_{i|i-1}}(s_i, a_i) - \gamma \max_{b \in A} \hat{Q}_{\hat{\theta}^{(j)}_{i|i-1}}(s_{i+1}, b),\ 0 \leq j \leq 2p \right\} \text{ (KTD-Q)} \end{cases}$;

Calcul des statistiques d'intérêt;

$\hat{r}_{i|i-1} = \sum_{j=0}^{2p} w_j \hat{r}^{(j)}_{i|i-1}$;

$P_{\theta r_i} = \sum_{j=0}^{2p} w_j (\hat{\theta}^{(j)}_{i|i-1} - \hat{\theta}_{i|i-1})(\hat{r}^{(j)}_{i|i-1} - \hat{r}_{i|i-1})$;

$P_{r_i} = \sum_{j=0}^{2p} w_j \left(\hat{r}^{(j)}_{i|i-1} - \hat{r}_{i|i-1} \right)^2 + P_{n_i}$;

Phase de correction;

$K_i = P_{\theta r_i} P_{r_i}^{-1}$;

$\hat{\theta}_{i|i} = \hat{\theta}_{i|i-1} + K_i \left(r_i - \hat{r}_{i|i-1} \right)$;

$P_{i|i} = P_{i|i-1} - K_i P_{r_i} K_i^T$;

Ici la distinction entre paramétrisation linéaire ou non n'est pas faite. En effet l'opérateur max, inhérent à l'équation d'optimalité de Bellman, rend l'équation d'observation de (58) non-linéaire. Cet opérateur est difficile à prendre en compte, particulièrement à cause de sa non dérivabilité.

Heureusement, comme elle approxime le vecteur aléatoire plutôt que la fonction, la transformation non-parfumée ne requiert pas de dérivation. Etant donné l'algorithme KTD général présenté section 2 et la transformation non-parfumée décrite dans la section 3.1.2, il est possible d'obtenir KTD-Q, l'algorithme KTD pour l'optimisation directe de la Q-fonction. Une première étape est de calculer les sigma-points associés au vecteur aléatoire de paramètres, comme dans les équations (51-52). Ensuite, l'image de ces sigma-points à travers l'équation d'observation du modèle espace d'état (58), qui contient l'opérateur max, est calculée :

$$\mathcal{R}_{i|i-1} = \left\{ \hat{r}^{(j)}_{i|i-1} = \hat{Q}_{\hat{\theta}^{(j)}_{i|i-1}}(s_i, a_i) - \gamma \max_{b \in A} \hat{Q}_{\hat{\theta}^{(j)}_{i|i-1}}(s_{i+1}, b),\ 0 \leq j \leq 2p \right\} \tag{59}$$

Enfin, les sigma-points et leurs images sont utilisés pour calculer les statistiques d'intérêt, comme dans les équations (54-56). L'approche KTD-Q proposée est aussi résumée dans l'algorithme 3.

3.4 Coût computationnel

Soit p le nombre de paramètres. La transformation non-parfumée implique de calculer une décomposition de Cholesky qui a une complexité computationnelle en $O(p^3)$. Cependant, pour les algorithmes considérés, la structure de mise à jour particulière de la matrice de covariance $P_{i|i-1}$, qui est de rang un, permet de considérer un algorithme spécifique de mise à jour de la décomposition de Cholesky dont la complexité est en $O(p^2)$. Les détails de cette approche "racine carrée" sont

donnés par van der Merwe & Wan (2003). Les différents algorithmes impliquent d'évaluer $2p+1$ fois la fonction g_{t_i} à chaque pas de temps. Pour KTD-V et KTD-SARSA et une paramétrisation générale, chaque évaluation est en $O(p)$. Pour KTD-Q, le maximum selon les actions doit être calculé. Notons $\mathcal{A}$ le nombre d'actions si l'espace correspondant est fini, et la complexité computationnelle de l'algorithme utilisé pour trouver ce maximum sinon (par exemple le nombre d'échantillons tirés multiplié par la complexité de leur évaluation pour Monte-Carlo). Ainsi chaque évaluation est bornée par $O(p\mathcal{A})$. Le reste des opérations est de l'algèbre linéaire basique, de complexité au plus $O(p^2)$. Ainsi, la complexité computationnelle (par itération) de KTD-V et KTD-SARSA est $O(p^2)$, celle de KTD-Q $O(\mathcal{A}p^2)$. L'ensemble des algorithmes requiert de stocker le vecteur de paramètres ainsi que la matrice de covariance associée, leur complexité en mémoire est donc $O(p^2)$.

4 Une forme d'apprentissage actif

Dans cette section, nous présentons le calcul d'incertitude de la valeur qu'il est possible de faire avec le cadre de travail proposé par KTD et introduisons une forme d'apprentissage actif l'utilisant.

Les paramètres étant modélisés par un vecteur aléatoire, et la fonction de valeur paramétrée étant pour un état donné fonction de ces paramètres, c'est une variable aléatoire. Nous montrons dans un premier temps comment calculer son espérance et l'incertitude associée (sa variance) à l'aide de la transformation non-parfumée. Le dilemme entre exploration et exploitation, qui est l'une des grandes problématiques de l'apprentissage par renforcement, peut grandement bénéficier d'une telle information d'incertitude, même si ce n'est pas *a priori* une condition nécessaire pour le traiter. Si peu d'approches dans la littérature permettent à la fois d'approximer la fonction de valeur et une incertitude associée, on peut tout de même citer Engel (2005), qui ne considère pourtant son utilisation qu'en terme de perspectives. Dans un second temps, nous proposons une approche d'apprentissage actif[1] qui est en quelque sorte une forme de politique totalement exploratrice choisissant les actions en fonction de leurs incertitudes, dans le contexte de KTD-Q. Il est montré sur un exemple section 5 que cela permet effectivement d'accélérer l'apprentissage.

4.1 Calcul de l'incertitude

Nous nous intéressons ici à la fonction de valeur, l'extension des résultats présentés à la fonction de qualité étant évidente. Supposons la fonction de valeur $\hat{V}_\theta$ paramétrée par le vecteur aléatoire θ de moyenne $\bar{\theta}$ et de variance P_θ. Notons $\bar{V}_\theta(s)$ son espérance et $\hat{\sigma}^2_{V_\theta}(s)$ sa variance. Si la paramétrisation est linéaire, de la forme $\hat{V}_\theta(s) = \phi(s)^T\theta$ comme dans l'équation (41), alors les quantités d'intérêt peuvent être obtenues analytiquement :

$$\bar{V}_\theta(s) = \phi(s)^T\bar{\theta} \quad \text{et} \quad \hat{\sigma}^2_{V_\theta}(s) = \phi(s)^T P_\theta \phi(s) \tag{60}$$

Si la paramétrisation est non-linéaire, il est toujours possible de propager l'incertitude sur les paramètres à la fonction de valeur en appliquant la transformation non-parfumée. Dans un premier temps, l'ensemble des sigma-points $\Theta = \{\theta^{(j)}, 0 \leq j \leq 2p\}$ et les poids associés $\mathcal{W} = \{w_j, 0 \leq j \leq 2p\}$ sont calculés à partir de $\bar{\theta}$ et P_θ, comme expliqué dans la section 3.1.2. Les images de ces sigma-points par la fonction de valeur, pour un état donné, sont ensuite calculées : $\mathcal{V}(s) = \left\{\hat{V}^{(j)}(s) = \hat{V}_{\theta^{(j)}}(s), \quad 0 \leq j \leq 2p\right\}$. Enfin, connaissant ces images et les poids associés, il est possible d'approcher la valeur moyenne ainsi que la variance associée :

$$\hat{V}(s) = \sum_{j=0}^{2p} w_j \hat{V}^{(j)}(s) \quad \text{et} \quad \hat{\sigma}^2_V(s) = \sum_{j=0}^{2p} w_j \left(\hat{V}^{(j)}(s) - \hat{V}(s)\right)^2 \tag{61}$$

Donc, pour une représentation de la fonction de valeur modélisée par un vecteur aléatoire, il est aisé de calculer l'information d'incertitude associée pour tout état s. La complexité est là encore quadratique. Ainsi, comme à chaque pas de temps une estimation $\hat{\theta}_{i|i}$ et la variance associée $P_{i|i}$ sont disponibles, l'incertitude sur les paramètres peut être propagée à la fonction de valeur (ou de qualité), si nécessaire.

[1] Nous utilisons le terme d'apprentissage actif dans le sens où plutôt que de tirer des actions totalement au hasard, nous les choisissons activement de façon à ce que les transitions résultantes réduisent le plus possible l'incertitude sur le modèle, de façon à accélérer l'apprentissage.

4.2 Apprentissage actif

Nous donnons ici un exemple d'utilisation effective de l'information d'incertitude disponible. Nous nous intéressons à KTD-Q. Cet algorithme est dit *off-policy*, car la politique apprise π (la politique optimale π^* dans ce cas) est différente de la politique suivie ou comportementale (que nous noterons b). Une question naturelle est de savoir quelle politique comportementale permet l'apprentissage le plus rapide de la politique optimale. Un élément de réponse utilisant l'information d'incertitude est proposé ici.

Soit i l'index temporel courant. Le système à contrôler est dans un état s_i, et l'agent doit choisir une action a_i. L'algorithme considéré étant KTD-Q, les estimations $\theta_{i-1|i-1}$ et $P_{i-1|i-1}$ sont disponibles. Elles peuvent être utilisées pour approcher l'incertitude de la Q-fonction paramétrée par θ_{i-1} en l'état s_i pour chaque action a, comme illustré section 4.1. La variance associée est notée $\sigma^2_{Q_{\theta_{i-1}}}(s_i, a)$. L'action a_i est ensuite choisie selon la politique comportementale aléatoire b définie par :

$$b(a_i|s_i) = \frac{\sigma_{Q_{\theta_{i-1}}}(s_i, a_i)}{\sum_{a \in A} \sigma_{Q_{\theta_{i-1}}}(s_i, a)} \tag{62}$$

Une politique totalement exploratrice privilégiant les actions pour lesquelles l'incertitude est la plus grande est ainsi obtenue. Ce n'est qu'une façon parmi d'autres d'utiliser cette information d'incertitude, mais elle permet de montrer que la variance sur les valeurs obtenue dans le contexte de KTD a du sens. Nous l'illustrons section 5 en montrant le gain en vitesse d'apprentissage obtenu par rapport à une politique comportementale totalement aléatoire.

5 Expérimentations

Dans cette section est proposé un ensemble de tests de référence classiques en apprentissage par renforcement de façon à comparer KTD à différents algorithmes de l'état de l'art et à illustrer les différents aspects de cette approche, à savoir le biais causé par les transitions stochastiques (chaîne de Boyan), la robustesse à la non-stationnarité (chaîne de Boyan, *mountain car*), l'incertitude de la valeur utilisée pour une forme d'apprentissage actif (pendule inversé), ainsi que l'efficacité en terme d'échantillons (c'est-à-dire vitesse de convergence, toutes les expériences). Les autres algorithmes considérés sont TD, SARSA, Q-learning et LSTD. Nous ne considérons pas leurs extensions aux traces d'éligibilité, dans la mesure où LSTD produit de meilleurs résultats que TD(λ) et où varier la valeur du facteur d'éligibilité a peu d'incidence sur les performance de LSTD(λ), comme noté par Boyan (1999).

5.1 Chaîne de Boyan

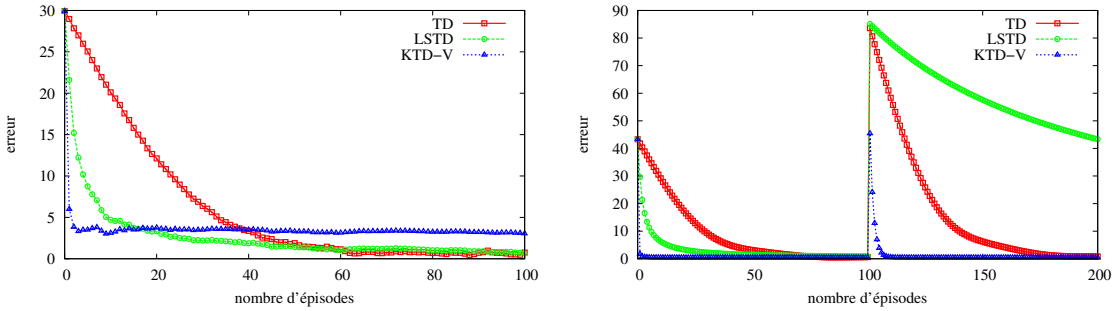
La première expérimentation est la chaîne de Boyan (1999). Le but est d'une part d'illustrer le problème posé par les transitions stochastiques et d'autre part de montrer l'efficacité de KTD en terme d'échantillons et sa capacité à prendre en compte un environnement non-stationnaire sur une version déterministe du problème.

5.1.1 Cas stochastique

La chaîne de Boyan est une chaîne de Markov valuée à 13 états dont l'état s^0 est absorbant, s^1 transite vers s^0 avec une probabilité de 1 et une récompense de -2, et s^i transite vers s^{i-1} ou s^{i-2}, $2 \leq i \leq 12$, pour chacun avec une probabilité de 0.5 et une récompense de -3. Pour cette expérience, KTD-V est comparé à TD ainsi qu'à LSTD. La paramétrisation est linéaire, et les vecteurs de base $\phi(s)$ pour les états s^{12}, s^8, s^4 et s^0 sont respectivement $[1,0,0,0]^T$, $[0,1,0,0]^T$, $[0,0,1,0]^T$ et $[0,0,0,1]^T$. Pour les autres états, les vecteurs de base sont obtenus par interpolation linéaire. L'approximation de la fonction de valeur est donc $\hat{V}_\theta(s) = \theta^T \phi(s)$. La fonction de valeur optimale est linéaire en ces bases, et le vecteur de paramètres optimal correspondant est $\theta^* = [-24, -16, -8, 0]^T$. La performance est mesurée avec la distance euclidienne $\|\theta - \theta^*\|$ entre les vecteurs de paramètres courant et optimal.

Le coefficient d'actualisation γ est fixé à 1 pour cette tâche épisodique. Pour TD, le taux d'apprentissage est fixé à $\alpha_i = 0.1$. Pour LSTD et KTD-V l'*a priori* est fixé à $P_{0|0} = I$, où I est la

matrice identité. Pour KTD-V, le bruit d'observation est fixé à $P_{n_i} = 10^{-3}$ et le bruit de process à $P_{v_i} = 0I$. Choisir ces paramètres requiert un peu de pratique, mais pas forcément plus que le choix d'un taux d'apprentissage pour d'autres algorithmes. Pour toutes les méthodes considérées, le vecteur de paramètres est initialisé à zéro. Les résultats sont présentés figure 1.a.



a. Cas stochastique. b. Cas déterministe et non-stationnaire.

FIG. 1 – Chaîne de Boyan.

LSTD converge plus rapidement que TD, comme attendu, et KTD-V converge encore plus vite. Cependant ce dernier algorithme ne converge pas vers le vecteur de paramètres optimal, ce qui s'explique par le fait que la fonction de coût minimisée est biaisée. Cette expérience a été menée pour montrer le problème causé par les transitions stochastiques, et nous nous concentrons à présent sur des PDM déterministes.

5.1.2 Cas déterministe et non-stationnaire

La chaîne de Boyan est rendue déterministe en posant la probabilité de transiter de s^i à s^{i-1} à 1. KTD-V est à nouveau comparé à LSTD et TD. De plus, pour simuler un changement dans le PDM, et donc la non-stationnarité, le signe de la récompense est inversé à partir du $100^{\text{ème}}$ épisode. La fonction de valeur optimale est toujours linéaire en les fonctions de base, et le vecteur de paramètres optimal est $\theta^*_{(-)} = [-35, -23, -11, 0]^T$ avant le changement de récompense, et $\theta^*_{(+)} = -\theta^*_{(-)}$ après. Les paramètres des différents algorithmes sont les mêmes, sauf le bruit de process qui est maintenant fixé à $P_{v_i} = 10^{-3}I$. Les résultats sont présentés sur la figure 1.b.

A nouveau KTD-V converge plus rapidement que LSTD et TD, cependant maintenant vers le vecteur de paramètres optimal, l'environnement étant déterministe. Après le changement de récompense, LSTD est très lent à converger, à cause de la non-stationnarité induite. TD s'adapte plus rapidement, le taux d'apprentissage étant constant. KTD-V s'adapte quant à lui très rapidement. Ainsi, si KTD-V est biaisé dans le cas stochastique, il converge plus vite et s'adapte plus rapidement que TD et LSTD dans le cas déterministe. Cette capacité à prendre en compte les non-stationnarités est importante pour suivre la dynamique de la fonction de valeur. Même si l'environnement est stationnaire, cela peut être utile dans un contexte de contrôle, comme illustré dans la section 5.3.

5.2 Pendule inversé

La seconde expérience est le pendule inversé tel que décrit par Lagoudakis & Parr (2003). Le but est ici de comparer deux algorithmes de type itération de la valeur, à savoir KTD-Q et Q-learning avec approximation de la Q-fonction, qui ont tous deux pour objectif d'estimer directement la fonction de qualité optimale. Autant que nous le sachions, KTD-Q est le seul algorithme d'ordre deux qui suive un schéma d'itération de la valeur (la difficulté majeur étant l'opérateur max).

Cette tâche nécessite de balancer un pendule de longueur et masse inconnues de façon à ce qu'il reste vertical en appliquant des forces au chariot sur lequel il est fixé. Trois actions sont possibles : pousser à gauche (-1), pousser à droite ($+1$), ou ne rien faire (0). L'état du système est donné par la position angulaire ω et la vitesse angulaire $\dot{\omega}$. Les transitions déterministes sont calculées grâce

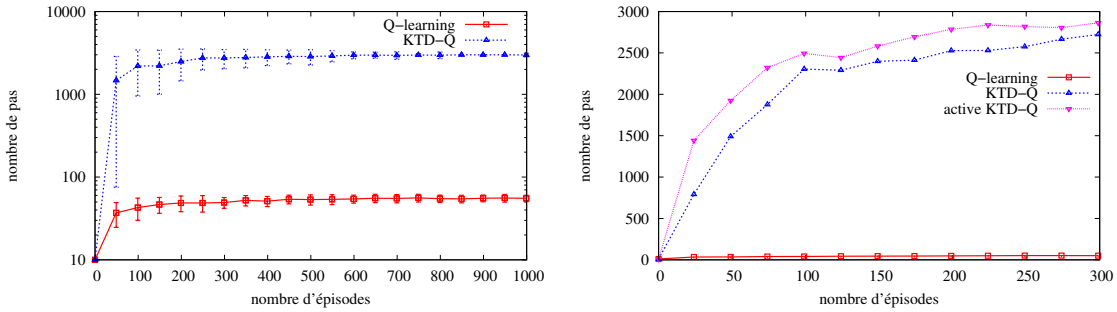
à la dynamique du système physique :

$$\ddot{\omega} = \frac{g\sin(\omega) - \beta ml\dot{\omega}^2\sin(2\omega)/2 - 50\beta\cos(\omega)a}{4l/3 - \beta ml\cos^2(\omega)} \tag{63}$$

où g est la constante de gravitation, m et l sont la masse et la longueur du pendule, M la masse du chariot et $\beta = \frac{1}{m+M}$. Une récompense nulle est donnée tant que la position angulaire appartient à l'intervalle $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Sinon, l'épisode se termine et une récompense de -1 est donnée. La paramétrisation est donnée par un terme constant et un ensemble de 9 noyaux gaussiens équi-répartis (centrés en $\{-\frac{\pi}{4}, 0, \frac{\pi}{4}\} \times \{-1, 0, 1\}$ et d'écart-type 1), cela pour chaque action. Il y a donc 30 fonctions de base. Le facteur d'actualisation γ est fixé à 0.95.

5.2.1 Apprendre la politique optimale

Dans un premier temps nous comparons la capacité des deux algorithmes à apprendre la politique optimale. Pour Q-learning, le taux d'apprentissage est fixé à $\alpha_i = \alpha_0 \frac{n_0+1}{n_0+i}$ où $\alpha_0 = 0.5$ et $n_0 = 200$, en accord avec Lagoudakis & Parr (2003). Pour KTD-Q, les paramètres sont $P_{0|0} = 10I$, $P_{n_i} = 1$ et $P_{v_i} = 0I$. Les vecteurs de paramètres sont initialisés à zéro. La politique suivie par l'agent est aléatoire (équi-probabilité des actions), et les deux algorithmes apprennent à partir des mêmes trajectoires. L'agent est initialisé dans un état aléatoire proche de l'équilibre $(0, 0)$. La longueur moyenne d'un tel épisode aléatoire est d'environ 10 pas. Les résultats sont présentés figure 2.a.



a. Apprendre la politique optimale. b. Apprentissage actif.

FIG. 2 – Pendule inversé.

Pour chaque essai, l'apprentissage est fait sur 1000 épisodes. Tous les 50 épisodes, l'apprentissage est gelé et la politique courante est testée. Pour cela, l'agent est initialisé dans un état aléatoire proche de l'équilibre et la politique gloutonne est suivie. Chaque test est répété 100 fois et moyenné (l'évaluation de la performance de la politique gloutonne courante étant initialisée dans un état aléatoire). La mesure de performance est le nombre de pas d'un épisode. Le nombre maximum de pas autorisé est de 3000, ce qui correspond à maintenir le pendule pendant 5 minutes. Les résultats de la figure 2.1 sont moyennés sur 100 essais (chaque essai correspondant à 1000 épisodes, la politique étant testée tous les 50 épisodes) et présentés en échelle semi-logarithmique.

Asymptotiquement, KTD-Q apprend la politique optimale (c'est-à-dire maintenir le pendule pendant le nombre maximum de pas autorisés) et de bonnes politiques sont apprises après seulement quelques dizaines d'épisodes. Avec le même nombre d'épisodes et la même paramétrisation, Q-learning échoue à apprendre une politique qui permette de maintenir le pendule pendant plus de quelques secondes, ce qui est en accord avec les résultats présentés par Lagoudakis & Parr (2003).

5.2.2 Résultats de l'apprentissage actif

Dans cette section, nous expérimentons la forme d'apprentissage actif présentée section 4, et l'algorithme correspondant est appelé *active* KTD-Q. La politique suivie n'est plus uniformément aléatoire, mais pondérée par l'incertitude associée aux actions. La longueur moyenne d'un tel épisode est de 11 pas, ce qui change peu par rapport à une politique totalement aléatoire. La durée d'un épisode ne peut donc être que faiblement responsable de l'amélioration des résultats. Pour chaque essai, l'apprentissage est fait sur 300 épisodes. Moins d'épisodes sont considérés pour

montrer l'accélération de la convergence (notons qu'asymptotiquement les deux variations de KTD se comportent aussi bien). Tous les 25 épisodes l'apprentissage est gelé et les politiques sont évaluées comme précédemment, avec la même mesure de performance. Les résultats de la figure 2.2 sont moyennés sur 100 essais. Notons que l'échelle n'est plus logarithmique.

Cette expérience compare le KTD-Q actif à KTD-Q et Q-learning. En comparant les deux variantes de KTD, il est clair que choisir les actions en fonction de l'incertitude accélère la convergence. Cette dernière est quasiment doublée sur les 100 premiers épisodes : par exemple, une performance moyenne de 1500 pas est obtenue après seulement 25 épisodes avec le KTD-Q actif, alors qu'elle n'est atteinte qu'après environ 50 épisodes par KTD-Q.

5.3 *Mountain car*

La dernière expérience est le *mountain car* telle que décrite par Sutton & Barto (1998). L'objectif ici est d'illustrer le comportement des algorithmes dans le cadre d'un schéma d'itération de la politique optimiste : apprendre en contrôlant induit des dynamiques de la valeur non-stationnaires. Cette tâche consiste à conduire un véhicule sous-puissant en haut d'une route de montagne abrupte, la gravité étant plus forte que le moteur de la voiture. L'état est donné par la position et la vitesse $(x, \dot{x}) \in [-1.2, 0.5] \times [-0.07, 0.07]$. Les trois actions possibles sont aller à gauche (-1), à droite $(+1)$ ou ne rien faire (0). La dynamique du système est donnée par :

$$\begin{cases} \dot{x}_{i+1} = \text{bound}\left[\dot{x}_i + 10^{-3}(a_i - 2.5\cos(3x_i))\right] \\ x_{i+1} = \text{bound}\left[x_i + \dot{x}_{i+1}\right] \end{cases} \tag{64}$$

où l'opérateur bound force les bornes de la position et de la vitesse. Quand la position atteint la borne inférieure, la vitesse est mise à zéro. Quand elle atteint la borne supérieure, l'épisode se termine avec une récompense nulle. La récompense est de -1 le reste du temps. Le facteur d'actualisation est fixé à 0.1. L'état est normalisé, et la paramétrisation est composée d'un terme constant et d'un ensemble de 9 noyaux gaussiens équi-répartis (centrés en $\{0, 0.5, 1\} \times \{0, 0.5, 1\}$ et d'écart-type 0.1), cela pour chaque action. Il y a donc 30 fonctions de base.

Cette expérience compare SARSA avec approximation de la fonction de qualité, LSTD et KTD-SARSA, dans un contexte d'itération optimiste de la politique. La politique suivie est ϵ-gloutonne, avec $\epsilon = 0.1$. Pour SARSA, le taux d'apprentissage est fixé à $\alpha_i = 0.1$. Pour LSTD l'*a priori* est fixé à $10I$. Pour KTD-SARSA, le même *a priori* est utilisé, et les variances de bruit sont fixées à $P_{n_i} = 1$ et $P_{v_i} = 0.05I$. Pour tous les algorithmes le vecteur de paramètres est initialisé à zéro. Chaque épisode commence dans un état aléatoire (tirage uniforme sur le domaine). Un maximum de 1500 pas est autorisé.

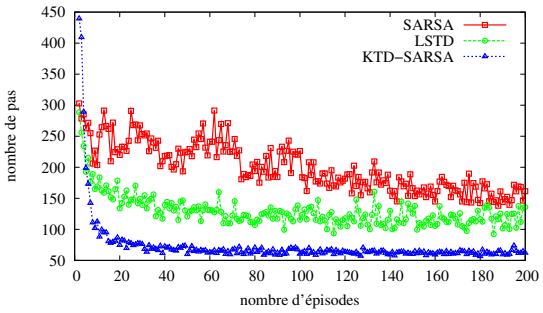


FIG. 3 – *Mountain car.*

Pour chaque essai, l'apprentissage se fait sur 200 épisodes, et la figure 3 montre la longueur de chaque épisode moyennée sur 300 essais. KTD-SARSA converge plus rapidement et vers une meilleure politique que LSTD, qui se comporte mieux que SARSA avec approximation de la Q-fonction. De meilleurs résultats ont peut-être été rapportés dans la littérature pour une paramétrisation de type *tile-coding*, cependant la paramétrisation choisie ici est plus brute et implique bien moins de paramètres. Le fait de suivre une politique ϵ-gloutonne implique la non-stationnarité de la Q-fonction apprise, ce qui peut expliquer que LSTD échoue à trouver une politique quasi-optimale.

LSTD a été étendu par Lagoudakis & Parr (2003) à LSPI (*Least-Squares Policy Iteration*), qui permet de chercher une politique optimale plus efficacement. Cependant cet algorithme est hors-ligne, et n'implique pas d'apprendre tout en contrôlant, c'est pourquoi il n'est pas comparé ici. KTD-SARSA obtient des politiques optimales très rapidement, après seulement quelques dizaines d'épisodes. L'apprentissage est également plus stable avec l'algorithme proposé.

6 Discussion et travaux similaires

Comme annoncé dans la section 1, des approches apparentées ont été proposées précédemment dans la littérature. Engel (2005) modélise le problème de l'approximation de fonction de valeur en apprentissage par renforcement à l'aide d'un modèle génératif basé sur des processus Gaussiens. Le point de vue adopté est donc sensiblement différent de celui de cette contribution, cependant dans le cas de transitions déterministes et de représentation paramétrique de la fonction de valeur, les algorithmes GPTD (*Gaussian Process Temporal Differences*) et KTD-V avec paramétrisation linéaires sont quasiment les mêmes. Les différences principales sont que GPTD permet une représentation non-paramétrique, mais nécessite la linéarité des modèles considérés et ne peut pas prendre en compte de bruit d'évolution, alors que KTD nécessite une représentation paramétrique (et donc un choix de d'architecture d'approximation, par exemple les fonctions de base dans le cas d'une paramétrisation linéaire), mais peut prendre en compte des non-linéarités ainsi qu'un bruit d'évolution. De plus, le principe de représentation non-paramétrique de GPTD peut être adapté à KTD, comme montré par Geist *et al.* (2008a), à condition de connaître à l'avance l'espace d'état et d'action. Ce cadre de travail permet également de dériver une information d'incertitude, qui est illustrée par Engel (2005) mais non utilisée dans le cadre de l'apprentissage. Comme le filtrage de Kalman est fortement lié à la minimisation par moindres carrés (c'en est en fait une généralisation), notre approche partage des similitudes avec LSTD (Bradtke & Barto, 1996), cependant sans prendre en compte le concept de variable instrumentale (Söderström & Stoica, 2002), introduit pour pouvoir traiter les transitions stochastiques. Choi & Roy (2006) proposent une forme de filtre de Kalman conçu pour approcher le point fixe de l'opérateur de Bellman (ou de façon équivalent approcher la solution de l'équation de Bellman) dans le cas d'une paramétrisation linéaire de la valeur. Cette approche peut approximativement être vue comme une version "bootstrapée" de KTD-V (le terme *bootstrap* est utilisé ici avec le même sens que Sutton & Barto (1998), c'est-à-dire considérer que la valeur de l'état vers lequel transite le système est connue et utiliser une valeur approchée à la place). A la place de l'équation d'observation (43), l'équation d'observation suivante est utilisée :

$$r_i + \gamma\phi(s_{i+1})^T\hat{\theta}_{i-1|i-1} = \phi(s_i)^T\theta_i + n_i \tag{65}$$

Autrement dit, la récompense n'est pas considérée comme l'observation, mais une approximation de la fonction de valeur est utilisée pour calculer une pseudo-observation (ou observation "bootstrapée"), et la mise à jour du filtre est faite de façon à corriger l'erreur entre la fonction de valeur de l'état courant et la pseudo-observation. Phua & Fitch (2007) utilisent un banc de filtre de Kalman pour apprendre les paramètres d'une représentation linéaire par morceaux de la fonction de valeur (un filtre de Kalman par morceau linéaire). Cette méthode peut grossièrement être vue comme un cas particulier de KTD, cependant des différences existent : un banc de filtres est utilisé plutôt qu'un seul, et la paramétrisation est linéaire, fait exploité pour développer des spécificités de l'algorithme (notamment concernant la mise à jour des paramètres).

Le cadre de travail proposé présente certains aspects intéressants. Premièrement, il ne suppose pas la stationnarité. Une application immédiate est la prise en compte d'environnements non-stationnaires. Un aspect peut-être encore plus intéressant est le cas du contrôle. Par exemple, l'algorithme LSTD est connu pour mal se comporter lorsqu'il est combiné avec un schéma d'itération de la politique optimiste (politique ϵ-gloutonne par exemple), à cause des non-stationnarités induites par ce schéma spécifique d'apprentissage et de contrôle. Dans la même idée, Bhatnagar *et al.* (2008) préfèrent TD à LSTD comme acteur dans l'approche acteur-critique qu'il proposent, pour le même problème, alors que TD est moins efficace en terme d'échantillons (vitesse de l'apprentissage en fonction du nombre de transitions observées). Le filtrage de Kalman et donc les algorithmes proposés sont robustes aux problèmes de non-stationnarité. Deuxièmement, comme le vecteur de paramètres est modélisé par une variable aléatoire, il est possible de calculer une

information d'incertitude sur la valeur des états, comme explicité dans la section 4. Nous l'utilisons dans une forme d'apprentissage actif section 4, mais cette incertitude pourrait être utile pour traiter du problème plus général du dilemme entre exploration et exploitation, dans l'idée de ce que font Dearden *et al.* (1998) ou encore Strehl *et al.* (2006) par exemple. Enfin, le cadre de KTD devrait assez naturellement pouvoir être étendu au problème de l'observatilité partielle. En effet, le problème d'inférer l'état d'un système à partir d'observations est un problème pouvant profiter du filtrage bayésien dont le formalisme est proche de celui proposé. Il est connu qu'un PDM partiellement observable (PDMPO) peut être exprimé comme un PDM dont les états sont en fait des distributions sur les états du PDMPO. Si ces distributions peuvent être estimées (par exemple en utilisant une méthode de filtrage), elles peuvent être naturellement prises en compte par KTD : la fonction de valeur dépend déjà de la distribution sur les paramètres, l'extension à une distribution sur les états peut se faire de la même façon. Pour l'instant, le problème majeur de KTD est son incapacité (théorique) à prendre en compte des PDM stochastiques. Il est possible de simplement ignorer le problème. Par exemple, KTD-Q a été appliqué avec succès à un problème stochastique dans (Geist *et al.*, 2008a). De plus nous travaillons sur ce problème avec une approche basée sur un bruit d'observation coloré, et les premiers résultats sont encourageants. Une autre difficulté éventuelle peut être le choix des différents paramètres. Cependant, d'une part la nécessité de choisir certains paramètres n'est pas propre à notre approche (taux d'apprentissage pour TD, *a priori* pour LSTD, *et cetera*), et d'autre part KTD peut grandement bénéficier de la vaste littérature sur le filtrage adaptatif, domaine étudié depuis plusieurs décennies.

7 Conclusion et perspectives

Dans cette contribution nous avons proposé un nouveau paradigme pour l'approximation de valeur en apprentissage par renforcement basé sur un cadre de travail statistique inspiré du filtrage de Kalman. Le paradigme général des différences temporelles de Kalman a été présenté section 2. Il a ensuite été spécialisé en un certain nombre d'algorithmes, à savoir KTD-V, KTD-SARSA et KTD-Q, à l'aide de la transformation non-parfumée. Un certain nombre d'autres points ont été abordés : complexité des algorithmes, problème posé par les transitions stochastiques, calcul de l'incertitude de la valeur et son utilisation dans une forme d'apprentissage actif. Les avantages de ce cadre de travail ont été discutés, notamment l'efficacité en terme d'échantillons (*i.e.* une convergence plus rapide pour les algorithmes proposés que pour l'état de l'art), la capacité à prendre en compte le cas non-stationnaire et l'information d'incertitude inhérente au modèle, et ils ont été illustrés section 5, avec des comparaisons à certains algorithmes de l'état de l'art. De plus, les algorithmes proposés sont en ligne et permettent de faire du contrôle. Autant que nous le sachions, nous avons également proposé le premier algorithme de la littérature du type itération de la valeur étant du second ordre (KTD-Q).

Si les résultats obtenus sont satisfaisants, il y a encore des perspectives intéressantes à développer. Premièrement, le cas des PDMs stochastiques est important, et nous développons une méthode basée sur un bruit d'observation coloré pour le prendre en compte, les premiers résultats étant publiés par Geist *et al.* (2009a). Un choix plus automatique des différents paramètres pourrait certainement bénéficier de la vaste littérature sur le filtrage adaptatif, ce point est à creuser. Il pourrait également être intéressant d'étendre le paradigme proposé au cas des PDMPO, ce qui pourrait se faire *a priori* assez naturellement pour les raisons exposées section 6. L'information d'incertitude inhérente au cadre de travail introduit pourrait être utile pour traiter du dilemme entre exploration et exploitation. Enfin, nous souhaitons utiliser KTD dans le cadre d'une architecture acteur-critique. Plus particulièrement, le modèle acteur-critique incrémental basé sur un gradient naturel proposé par Bhatnagar *et al.* (2008) pourrait bénéficier d'un algorithme d'ordre deux pour le critique, cependant TD y est préféré à LSTD en raison de l'aspect non-stationnaire. KTD pourrait s'avérer une alternative intéressante pour le critique.

Remerciements

O. Pietquin souhaite remercier la région Lorraine ainsi que la communauté européenne (projet CLASSiC, FP7/2007-2013, subvention 216594) pour leur support financier.

Références

Antos A., Szepesvári C. & Munos R. (2008). Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, **71**(1), 89–129.

Baird L. C. (1995). Residual Algorithms : Reinforcement Learning with Function Approximation. In *Proceedings of the International Conference on Machine Learning (ICML 95)*, p. 30–37.

Bertsekas D. P. & Tsitsiklis J. N. (1996). *Neuro-Dynamic Programming*. Athena Scientific.

Bhatnagar S., Sutton R. S., Ghavamzadeh M. & Lee M. (2008). Incremental Natural Actor-Critic Algorithms. In *Proceedings of the Twenty-First Annual Conference on Advances in Neural Information Processing Systems (NIPS)*, Vancouver, Canada.

Bishop C. M. (1995). *Neural Networks for Pattern Recognition*. New York, USA : Oxford University Press.

Boyan J. A. (1999). Technical Update : Least-Squares Temporal Difference Learning. *Machine Learning*, **49**(2-3), 233–246.

Bradtke S. J. & Barto A. G. (1996). Linear Least-Squares Algorithms for Temporal Difference Learning. *Machine Learning*, **22**(1-3), 33–57.

Choi D. & Roy B. V. (2006). A Generalized Kalman Filter for Fixed Point Approximation and Efficient Temporal-Difference Learning. *Discrete Event Dynamic Systems*, **16**, 207–239.

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.

Engel Y., Mannor S. & Meir R. (2005). Reinforcement Learning with Gaussian Processes. In *Proceedings of International Conference on Machine Learning (ICML-05)*.

Geist M., Pietquin O. & Fricout G. (2008a). Bayesian Reward Filtering. In S. G. et al., Ed., *Proceedings of the European Workshop on Reinforcement Learning (EWRL 2008)*, volume 5323 of *Lecture Notes in Artificial Intelligence*, p. 96–109. Lille (France) : Springer Verlag.

Geist M., Pietquin O. & Fricout G. (2008b). Kalman Temporal Differences : Uncertainty and Value Function Approximation. In *NIPS Workshop on Model Uncertainty and Risk in Reinforcement Learning*, Vancouver (Canada).

Geist M., Pietquin O. & Fricout G. (2009a). Différences temporelles de kalman : le cas stochastique. In *actes des Journées Francophones de Planification, Décision et Apprentissage pour la conduite de systèmes (JFPDA 2009)*, Paris, France.

Geist M., Pietquin O. & Fricout G. (2009b). Kalman Temporal Differences : the deterministic case. In *Proceedings of the IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL 2009)*, Nashville, TN, USA.

Jo S. & Kim S. W. (2005). Consistent Normalized Least Mean Square Filtering with Noisy Data Matrix. *IEEE Transactions on Signal Processing*, **53**(6), 2112–2123.

Julier S. J. & Uhlmann J. K. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, **92**(3), 401–422.

Kalman R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering*, **82**(Series D), 35–45.

Lagoudakis M. G. & Parr R. (2003). Least-Squares Policy Iteration. *Journal of Machine Learning Research*, **4**, 1107–1149.

Phua C. W. & Fitch R. (2007). Tracking Value Function Dynamics to Improve Reinforcement Learning with Piecewise Linear Function Approximation. In *International Conference on Machine Learning (ICML 07)*.

Schoknecht R. (2002). Optimality of Reinforcement Learning Algorithms with Linear Function Approximation. In *Conference on Neural Information Processing Systems (NIPS 15)*.

Sigaud O. & Buffet O. (2008). *Processus décisionnels de Markov en intelligence artificielle*. Hermes Science Publications / Lavoisier.

Simon D. (2006). *Optimal State Estimation : Kalman, H Infinity, and Nonlinear Approaches*. Wiley & Sons, 1. auflage edition.

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*. The MIT Press, 3rd edition.

Söderström T. & Stoica P. (2002). Instrumental variable methods for system identification. *Circuits, Systems, and Signal Processing*, **21**, 1–9.

van der Merwe R. & Wan E. (2003). Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models. In *Proceedings of the Workshop on Advances in Machine Learning*, Montreal, Canada.