Optimisation de contrôleurs par essaim particulaire

Jérémy Fix, Matthieu Geist

Optimisation de contrôleurs par essaim particulaire

Jeremy Fix et Matthieu Geist

Supélec, Équipe IMS
2 rue edouard Belin, 57070 Metz (France)
prenom.nom@supelec.fr

Résumé : Trouver des contrôleurs optimaux pour des systèmes stochastiques est un problème particulièrement difficile abordé dans les communautés d'apprentissage par renforcement et de contrôle optimal. Le paradigme classique employé pour résoudre ces problèmes est celui des processus décisionnel de Markov. Néanmoins, le problème d'optimisation qui en découle peut être difficile à résoudre. Dans ce papier, nous explorons l'utilisation de l'optimisation par essaim particulaire pour apprendre des contrôleurs optimaux. Nous l'appliquons en particulier à trois problèmes classiques : le pendule inversé, le *mountain car* et le double pendule.

Mots-clés : optimisation par essaim particulaire, contrôle optimal, recherche de politique.

1. Introduction

L'apprentissage par renforcement (AR) (Sutton & Barto, 1998) formalise le problème du contrôle optimal. Dans ce paradigme, à chaque pas de temps (discret), le système à contrôler est dans un état (ou configuration) donné. L'agent doit alors choisir l'action à exécuter. Le système change alors d'état avec une certaine distribution de probabilité (supposée ici Markovienne) et un oracle fourni une récompense associée à cette transition. La récompense n'est qu'une indication locale de la qualité du contrôleur, le but de l'agent étant de choisir une séquence d'actions maximisant une certaine fonction du cumul des récompenses. L'un des avantages de ce paradigme est que la récompense n'est qu'une indication grossière de la qualité du contrôleur puisqu'elle ne précise pas l'action que l'agent aurait dû exécuter (par exemple, pour un

joueur d'échec, la récompense pourrait être délivrée s'il gagne la partie sans préciser qu'il aurait dû prendre la reine). La fonction associant les configurations aux actions est appelée politique (ou contrôle) ; sa qualité est quantifiée par la fonction de valeur qui associe à chaque état l'espérance du cumul de récompenses partant de cet état et suivant la politique. La politique optimale est celle qui maximise la fonction de valeur. Il existe différentes approches pour apprendre une politique optimale. L'une d'elle consiste à paramétrer la politique et à chercher dans l'espace des politiques celle qui maximise la fonction de valeur en suivant le gradient de la fonction de valeur par rapport aux paramètres de la politique (*e.g.*, Baxter & Bartlett (1999)). Cette approche conduit néanmoins à certaines difficultés. Même pour des politiques simples, le calcul du gradient de la fonction d'utilité associée n'est pas aisé.

Dans la communauté de l'optimisation numérique, il existe un certain nombre d'algorithmes qui ne nécessitent pas de calculer le gradient de la fonction de coût mais simplement de savoir l'évaluer (*e.g.* les algorithmes génétiques, les essaims particulaires, etc.). Ces approches impliquent une population d'individus (chacun représentant un jeu de paramètres) qui sont combinés afin d'atteindre un optimum global de la fonction de coût. L'algorithme d'optimisation par essaim particulaire (*Particle Swarm Optimization*, PSO), introduit par Kennedy & Eberhart (1995), fait partie de cette classe d'algorithmes. Plusieurs variations de l'algorithme de Kennedy ont été proposées et on trouvera une revue de ces variantes dans Engelbrecht (2005). Il a été montré expérimentalement que l'algorithme PSO (la version originale ou ses variantes) est très efficace pour résoudre des problèmes d'optimisation uni- ou multimodaux avec une fonction de coût statique ou dynamique et ce, même dans des espaces de recherche de grande dimension (Engelbrecht, 2010).

Dans cet article, nous introduisons un algorithme simple de recherche de politique reposant sur l'optimisation par essaim particulaire et nous montrons son efficacité à résoudre trois problèmes standards d'optimisation en apprentissage par renforcement : le pendule inversé, le *mountain car* et le double pendule. Les deux premiers problèmes sont stochastiques puisque du bruit est présent dans l'évolution du système et dans la sélection de l'action. La *fitness* associée à un contrôleur étant évaluée par Monte-Carlo, cette stochasticité se retrouve dans le paysage de la fonction d'utilité. Pour le dernier problème, nous nous intéressons aux capacités de l'algorithme à travailler dans des espaces de recherche de grande dimension. Le double pendule est un problème très difficile et la plupart des approches éprouvent beaucoup de difficulté à trouver un contrôleur optimal.

2. Monte Carlo Swarm Policy Search (MCSPS)

Un processus décisionnel de Markov (PDM) est défini par la donnée de l'ensemble $\{S, A, P, R\}$ où S est l'espace d'états, A l'espace d'actions, P un ensemble de probabilités de transitions Markoviennes et R une fonction de récompense. Une politique est une fonction de l'espace d'état vers les distributions de probabilités sur les actions : $\pi : S \rightarrow \mathcal{P}(A)$. A chaque pas de temps i, le système à contrôler est dans un état s_i, l'agent choisit une action a_i selon la politique π, $a_i \sim \pi(.|s_i)$. L'action est appliquée au système qui évolue de manière probabiliste vers un état s_{i+1} selon les probabilités de transition Markoviennes $P(s_{i+1}|s_i, a_i)$. L'agent reçoit alors une récompense $r_i = R(s_i, a_i, s_{i+1})$. Le but de l'agent est d'apprendre une politique qui maximise la fonction de valeur, fonction du cumul des récompenses. Il a plusieurs façons de définir cette fonction de valeur. Il est habituel de considérer l'espérance du cumul actualisé des récompenses (espérance calculée selon la stochasticité des transitions et de la politique) : $V^\pi(s) = E[\sum_{i=0}^{\infty} \gamma^i r_i | s_0 = s, \pi]$, le terme $\gamma \in [0, 1[$ étant le facteur d'actualisation qui pondère l'influence des récompenses à long terme par rapport aux récompenses à plus court terme. Dans le cas où le problème est à horizon fini T, il est également possible de considérer la fonction de valeur suivante : $V^\pi(s) = E[\sum_{i=0}^{T} r_i | s_0 = s, \pi]$. Enfin, un autre critère moins utilisé parce que moins pratique d'une point de vue mathématique est la récompense moyenne : $V^\pi(s) = \lim_{n\to\infty} \frac{1}{n} E[\sum_{i=0}^{n} r_i | s_0 = s, \pi]$. Quelle que soit la fonction de valeur choisie, la fonction d'utilité à maximiser est l'espérance de la fonction de valeur selon la distribution p_0 des états initiaux :

$$\rho^\pi = E[V^\pi(s_0) | s_0 \sim p_0]. \tag{1}$$

La politique optimale π^* est celle qui maximise ce critère :

$$\pi^* = \underset{\pi: S \rightarrow \mathcal{P}(A)}{\operatorname{argmax}} \rho^\pi. \tag{2}$$

Dans le cadre de la recherche directe de politique, nous introduisons quelques hypothèses. Tout d'abord, nous supposons que le modèle du système à contrôler (les probabilités de transition et la fonction de récompense) est inconnu mais qu'on dispose d'un simulateur permettant d'échantillonner des trajectoires selon n'importe quelle politique (cette hypothèse n'est pas forcément très forte). Par ailleurs, nous supposons que la politique est paramétrée, *i.e.*

chaque politique π_θ est paramétrée par un vecteur de paramètres de dimension $p\,\theta \in \mathbb{R}^p$ (par example, la politique peut être un échantillonneur de Gibbs sur un réseau à fonctions de base radiales, les paramètres étant les poids des noyaux associés). Le problème d'optimisation à résoudre se formule alors ainsi :

$$\theta^* = \underset{\theta \in \mathbb{R}^p}{\operatorname{argmax}}\, \rho^{\pi_\theta}. \tag{3}$$

Ce problème d'optimisation n'est pas simple à résoudre. Certains auteurs ont proposé de le résoudre par montée de gradient (Baxter & Bartlett, 1999) ou par des méthodes d'entropie croisée (Mannor *et al.*, 2003). Le modèle étant supposé inconnu, si l'optimisation repose sur le gradient de la fonction de valeur, il doit être estimé par simulation ; mais cette estimation peut avoir une grande variance.

Dans ce papier, nous introduisons une idée simple : utiliser l'optimisation par essaim particulaire pour résoudre le problème d'optimisation précédent. Dans ce cadre, chaque particule a une position dans l'espace des paramètres, cette position déterminant une politique dont la *fitness* associée est ρ^{π_θ}. Le modèle étant supposé inconnu, la *fitness* ne peut pas être calculée analytiquement mais elle peut être estimée par une méthode de Monte Carlo en simulant plusieurs trajectoires d'une même politique. On génère ainsi M trajectoires, partant d'un état initial aléatoire s_0 distribué selon p_0. Chaque trajectoire est obtenue en appliquant la même politique π_θ. A partir de ces trajectoires $\left\{(s_0^m, r_0^m, s_1^m, r_1^m \ldots s_T^m, r_{T-1}^m)_{1 \leq m \leq M}\right\}$, avec T la longueur de l'épisode (on considère ici des problèmes à horizon fini), on peut calculer un estimateur non biaisé de la *fitness* ρ^{π_θ} par :

$$\hat{\rho}^{\pi_\theta} = \frac{1}{M} \sum_{m=1}^{M} \sum_{i=0}^{T-1} r_i^m, \tag{4}$$

Pour l'algorithme d'optimisation par essaim, nous utilisons une grille de 5×5 particules avec une topologie de von Neumann. Il existe plusieurs règles de mise à jour des positions et vitesses des particules, dont on trouvera une revue dans (Engelbrecht, 2005), et nous utilisons l'algorithme de base de Kennedy & Eberhart (1995) avec un facteur de constriction (Clerc & Kennedy, 2002). En particulier, les vitesses $\mathbf{v}_i$ et positions $\mathbf{p}_i$ d'une particule i sont mises à jour par :

$$\begin{aligned} \forall j \in [|1, p|], \mathbf{v}_{ij} &= w\mathbf{v}_{ij} + c_1 r_1.(\mathbf{b}_{ij} - \mathbf{p}_{ij}) + c_2 r_2.(\mathbf{l}_{ij} - \mathbf{p}_{ij}) \\ \mathbf{p}_i &= \mathbf{p}_i + \mathbf{v}_i \end{aligned} \tag{5}$$

avec les méta-paramètres $w = 0.729844, c_1 = c_2 = 1.496180$, r_1, r_2 étant des nombres aléatoires tirés uniformément dans $[0, 1]$, $\mathbf{b}_i$ étant la meilleure position que la particule a occupée dans son existence, $\mathbf{l}_i$ la meilleure position qu'une particule dans la voisinage de la particule i a occupée dans son existence et p le nombre de paramètres à estimer. Les positions des particules sont initialisées aléatoirement dans l'espace des paramètres et les vitesses sont initialement nulles. Les positions et vitesses des particules sont mises à jour de manière asynchrone. L'algorithme est détaillé ci-dessous. Notez que, les problèmes étant stochastiques, on réévalue à chaque itération la *fitness* de la meilleure particule du voisinage. Les scripts pour les simulations présentées par la suite sont disponible en ligne (Fix & Geist, 2011).

Algorithme 1: Algorithme Monte Carlo Swarm Policy Search

Initialization;
pour $t = 1$ **à** N **faire**
  particles $\rightarrow$ shuffle();
  pour $i = 1$ **à** P **faire**
    // Mise à jour des positions, vitesses et *fitness* de la particule i
    $\mathbf{v}_i \leftarrow w\mathbf{v}_i + c_1\mathbf{R}_1.(\mathbf{b}_i - \mathbf{p}_i) + c_2\mathbf{R}_2.(\mathbf{l}_i - \mathbf{p}_i)$;
    BoundVelocity($\mathbf{v}_i$);
    $\mathbf{p}_i \leftarrow \mathbf{p}_i + \mathbf{v}_i$;
    BoundPosition($\mathbf{p}_i$);
    $f_i \leftarrow$ MonteCarlo($\mathbf{p}_i$);
    $f_i^b \leftarrow$ MonteCarlo($\mathbf{b}_i$);
    si $f_i > f_i^b$ **alors**
      $\mathbf{b}_i \leftarrow \mathbf{p}_i$;
      $f_i^b \leftarrow f_i$;
    fin
    // Mise à jour des meilleurs positions dans le voisinage
    pour $j = 1$ **à** P **faire**
      si $i \in \mathcal{N}_j$ *and* $f_i^b > f_j^l$ **alors**
        $\mathbf{l}_j \leftarrow \mathbf{b}_i$;
        $f_j^l \leftarrow f_i^b$;
      fin
    fin
  fin
fin

3. Résultats

3.1. Pendule inversé

3.1.1. Problème

Le problème du pendule inversé est un problème classique en apprentissage par renforcement et a déjà été résolu grâce à diverses méthodes (voir par exemple Lagoudakis & Parr (2003)). Nous utilisons la même définition du problème que Lagoudakis & Parr (2003). Ce problème consiste à trouver la force à appliquer à un mobile, sur lequel est accroché un pendule, afin de maintenir ce pendule dans la position verticale instable. L'état du système est formé par l'angle que forme le pendule avec la verticale et sa vitesse angulaire $(\theta, \dot{\theta})$. L'évolution de l'état est régie par les équations suivantes :

$$\left\{ \begin{array}{lcl} \theta_{t+1} & = & \theta_t + \tau\dot{\theta}_t \\ \dot{\theta}_{t+1} & = & \dot{\theta}_t + \tau\frac{g\sin(\theta_t) - \alpha ml\sin(2\theta_t)\dot{\theta}_t^2/2 - \alpha\cos(\theta_t)(f_t+\eta)}{4l/3 - \alpha ml\cos^2(\theta_t)} \end{array} \right. \quad (6)$$

avec g la constante de gravité ($g = 9.8m/s^2$), m et l les masses et longueurs du pendule ($m = 2.0$ kg, $l = 0.5$ m), M la masse du mobile ($M = 8.0$ kg) et $\alpha = \frac{1}{m+M}$. Le pas de discrétisation τ est fixé à 0.1s. Le pendule doit être maintenu dans le domaine $D = [-\frac{\pi}{2}; \frac{\pi}{2}]$. Une politique optimale peut maintenir le pendule indéfiniment. On limite la durée d'un épisode à 3000 interactions. A chaque interaction, une récompense nulle $r = 0$ est obtenue sauf si le pendule quitte le domaine D qui conduit à une récompense négative $r = -1$ et termine l'épisode. Il faut remarquer que ce signal de récompense est très grossier puisqu'il indique uniquement que le pendule ne doit pas tomber et ne précise pas que la position optimale est à la verticale (ce qui peut être induit par une récompense en cosinus). Le pendule est initialisé aléatoirement proche de l'équilibre instable ($\theta_0 \in [-0.1; 0.1], \dot{\theta}_0 \in [-0.1; 0.1]$). Le système est contrôlé en appliquant une force $f_t \in \{-50, 0, 50\}$ N. perturbée par un bruit uniforme $\eta \in [-5; 5]$ N. .

Le contrôleur est défini par 9 fonctions radiales (RBF) et un terme constant par action. Les 9 gaussiennes ont des moyennes uniformément réparties dans $[-\pi/4, \pi/4] \times [-1.0; 1.0]$, leur variance étant fixée à $\sigma = 1,0$. Optimiser ce contrôleur consiste à trouver 30 paramètres (les amplitudes des 27 gaussiennes et les 3 termes constants). Le RBF associé à chaque action défini, à

un facteur de normalisation près, la probabilité de sélectionner cette action :

$$\forall i \in [1,3], P_i = \exp(c_i + \sum_{j=1}^{9} a_{i,j} \exp(-\frac{(\theta - \theta_j)^2 + (\dot{\theta} - \dot{\theta}_j)^2}{2\sigma^2}))$$

3.1.2. Résultats expérimentaux

Les expériences sont répétées 1000 fois avec 200 itérations de l'essaim pour chaque expérience. Pour chaque itération de l'essaim, la *fitness* est évaluée à partir d'une seule trajectoire ce qui rend l'itération plus rapide mais également la *fitness* plus sujette au bruit (d'évolution, de sélection de l'action ou de définition de l'état initial). Nous ne l'illustrons pas dans cet article mais la *fitness* reste effectivement stochastique. Même si la *fitness* à maximiser est calculée sur une seule trajectoire, la qualité d'un contrôleur est évaluée *a posteriori* sur 500 trajectoires pour en avoir une bonne estimation.

La figure 1 montre la durée moyenne d'un épisode pour la meilleure particule à chaque itération de l'essaim ainsi que sa variance. D'après ces résultats, l'essaim converge vers un contrôleur optimal qui maintient le pendule dans le domaine but pendant tout l'épisode. Il faut en moyenne 50 itérations pour converger vers ce contrôleur optimal, ce qui correspond à 1250 trajectoires avec un essaim composé de 25 particules.

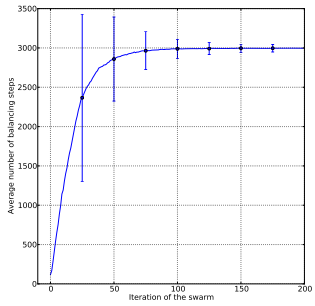


FIGURE 1: Durée moyenne d'un épisode ainsi que sa variance pour le meilleur contrôleur de l'essaim. Cette moyenne est calculée sur 1000 essais avec 500 trajectoires à chaque itération de l'essaim. Il est à noter que cette mesure de performance n'est pas directement la *fitness* mais se comporte qualitativement de la même façon et est plus intuitive.

3.2. Mountain car

3.2.1. Problème

Le second problème que nous considérons est le problème du *mountain car* décrit dans (Sutton & Barto, 1998, Chap 8). Le but est de contrôler un véhicule pour qu'il sorte d'une vallée. Le couple du véhicule est limité, ce qui l'oblige à osciller d'avant en arrière pour gagner de la vitesse afin de s'extraire de la vallée. L'état est défini par la position $x \in [-1.2, 0.5]$ et la vitesse $\dot{x} \in [-0.07, 0.07]$ du véhicule. Trois actions discrètes sont autorisées $a \in \{-1, 0, 1\}$ pour accélérer vers la gauche, ne rien faire et accélérer vers la droite. L'évolution de l'état est régie par les équations suivantes :

$$\left\{ \begin{array}{rcl} \dot{x}_{t+1} & = & \text{bound}(\dot{x}_t + 10^{-3}(a_t - 2.5\cos(3x_t))) \\ x_{t+1} & = & \text{bound}(x_t + \dot{x}_{t+1}) \end{array} \right. \tag{7}$$

La position du véhicule est bornée dans $[-1.2, 0.5]$. L'état du véhicule est initialisé aléatoirement proche du pire cas, c'est à dire au fond de la vallée avec une vitesse proche de zéro $x_0 \in [-0.75, -0.25], \dot{x}_0 \in [-0.02, 0.02]$. Lorsque le véhicule atteint la borne à gauche $x_{t+1} \leq -1.2$, il reste sur place et sa vitesse est annulée. Lorsque le véhicule atteint la position but $x_{t+1} \geq 0.5$, l'épisode se termine avec une récompense nulle $r = 0$. Une récompense négative $r = -1$ est délivrée à chaque itération. L'optimisation consiste à maximiser les récompenses espérées ce qui est équivalent ici à minimiser le temps mis pour atteindre la sortie.

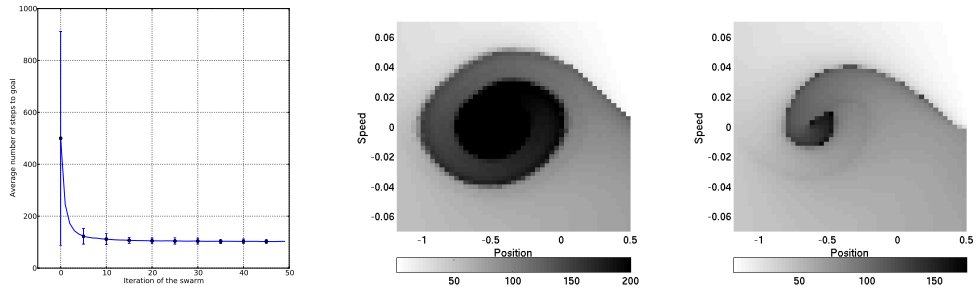
Le contrôleur est défini par 9 fonctions de base radiales (gaussiennes) et un terme constant pour chaque action, ce qui conduit à un problème d'optimisation en dimension 30. En appliquant une mise à l'échelle de l'état (position et vitesse) pour que celui-ci soit dans le domaine $[0, 1] \times [0, 1]$, les moyennes des gaussiennes sont uniformément réparties dans $[0, 1] \times [0, 1]$ et la variance fixée à $\sigma = 0.3$. Comme pour le pendule inversé, le résultat de ces fonctions radiales définit, à un facteur de normalisation près, la probabilité de sélectionner chacune des actions.

3.2.2. Résultats expérimentaux

L'expérience est répétée 1000 fois et chaque expérience consiste en 500 itérations de l'essaim (qui est largement suffisant pour obtenir un contrôleur optimal). La *fitness* d'une particule est évaluée sur 30 trajectoires ce qui ne supprime pas complètement sa stochasticité tout en limitant le temps pour son

calcul. La qualité d'un contrôleur est évaluée *a posteriori* sur 1000 trajectoires pour en avoir une bonne estimation. Le nombre moyen de pas pour atteindre l'état but et sa variance sont représentés sur la figure 2a).

Le nombre moyen de pas pour atteindre la sortie en fonction de l'état de départ pour une politique initiale et la politique optimale obtenue sur une expérience est représenté sur les figures 2b,c.



a) b) c)

FIGURE 2: a) Nombre moyen de pas pour atteindre l'état but et sa variance. Cette moyenne est calculée sur 1000 répétitions de l'expérience avec 1000 trajectoires pour chaque itération. Le temps moyen pour atteindre l'état but pour différents états de départ pour une politique initiale b) et optimale c) d'une expérience. Cette moyenne est saturée à 200 mais peut atteindre 1500 pour la politique initiale (fig. b.).

3.3. Double pendule

3.3.1. Problème

Ce dernier problème consiste à amener et maintenir un double pendule à la position verticale pointant vers le haut (instable) partant de la position pointant vers le bas (stable). L'action est le couple appliqué à l'articulation entre les deux bras du pendule. La mécanique du système fait que le seul moyen d'atteindre l'état but est de balancer le pendule. L'état du système est défini par les angles θ_1, θ_2 et vitesses angulaires $\dot{\theta}_1, \dot{\theta}_2$ des deux bras (voir fig. 3). Ce double pendule est contrôlé par le couple τ appliqué à l'articulation entre les deux bras. L'évolution de l'état du système est régie par les équations

à temps discrets 8 (Spong, 1995) avec $\tau \in \{-1, 0, 1\}$, $\Delta t = 0.01s.$, $\dot{\theta}_{1,t} \in [-9\pi, 9\pi]$, $\dot{\theta}_{2,t} \in [-4\pi, 4\pi]$, $m_1 = m_2 = 1$, $l_1 = l_2 = 1$, $l_{c_1} = l_{c_2} = 0.5$, $I_1 = I_2 = ml^2/12$, $g = 9.8$. Le pendule est initialisé dans la position verticale pointant vers le bas avec des vitesses angulaires nulles $\theta_{1,0} = 3\pi/2, \theta_{2,0} = 0, \dot{\theta}_{1,0} = \dot{\theta}_{2,0} = 0$. Nous ne considérons donc pas de bruit dans l'état initial ni dans les équations d'évolution. Les temps de simulation ne permettant pas de considérer un problème stochastique, nous nous intéressons ici plutôt à la capacité de l'optimisation par essaim à travailler dans des espaces de grande dimension (ici 256, voir ci-après).

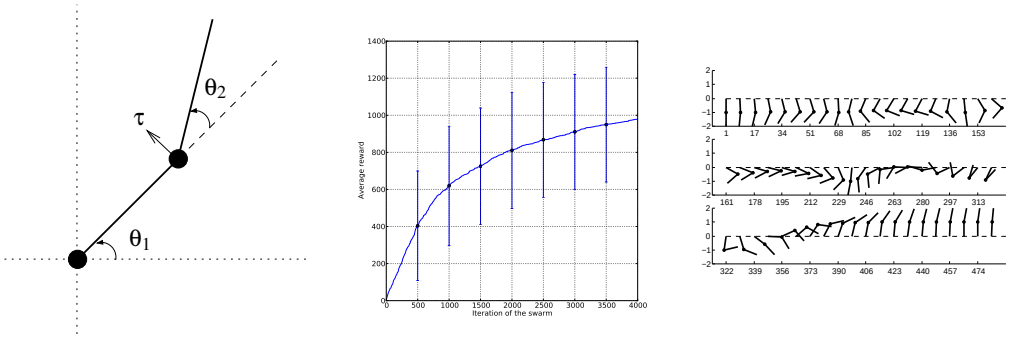
$$\left\{\begin{array}{rcl}
\theta_{1,t+1} & = & \theta_{1,t} + \Delta t.\dot{\theta}_{1,t} \\
\theta_{2,t+1} & = & \theta_{2,t} + \Delta t.\dot{\theta}_{2,t} \\
\dot{\theta}_{1,t+1} & = & \dot{\theta}_{1,t} + \Delta t.\ddot{\theta}_{1,t} \\
\dot{\theta}_{2,t+1} & = & \dot{\theta}_{2,t} + \Delta t.\ddot{\theta}_{2,t} \\
\ddot{\theta}_{1,t+1} & = & -\frac{d_2\ddot{\theta}_{2,t}+\phi_1}{d_1} \\
\ddot{\theta}_{2,t+1} & = & \frac{\tau+\frac{d_2}{d_1}\phi_1-m_2l_1l_{c_2}\dot{\theta}_{1,t}^2\sin(\theta_{2,t})-\phi_2}{m_2l_{c_2}^2+I_2-\frac{d_2^2}{d_1}} \\
d_1 & = & m_1l_{c_1}^2 + m_2(l_1^2 + l_{c_2}^2 + 2l_1l_{c_2}\cos(\theta_{2,t})) + I_1 + I_2 \\
d_2 & = & m_2(l_{c_2}^2 + l_1l_{c_2}\cos(\theta_{2,t})) + I_2 \\
\phi_1 & = & m_2l_1l_{c_2}(-\dot{\theta}_{2,t}^2\sin(\theta_{2,t}) + 2\dot{\theta}_{2,t}\dot{\theta}_{1,t}\sin(\theta_{2,t})) \\
 & & +(m_1l_{c_1} + m_2l_1)g\cos(\theta_{1,t} - \pi/2) + \phi_2 \\
\phi_2 & = & m_2l_{c_2}g\cos(\theta_{1,t} + \theta_{2,t} - \pi/2)
\end{array}\right. \quad (8)$$

Ce problème de contrôle est particulièrement difficile (Munos & Moore, 1999). Pour rendre le problème plus simple, nous considérons un contrôleur combinant des fonctions radiales de bases (RBF) et un régulateur quadratique linéaire optimal (LQR) (Spong, 1995). Le contrôleur LQR peut maintenir le pendule en position verticale instable mais est incapable de le balancer pour l'y amener. Par ailleurs, le contrôleur LQR fonctionne parfaitement dans un domaine très restreint de l'espace d'état ; pour des vitesses angulaires nulles $\dot{\theta}_1 = \dot{\theta}_2 = 0$, ce contrôleur stabilise correctement le pendule dans le domaine $\theta_2 = 0, \theta_1 = \pi/2 \pm \pi/24$. La tâche du second contrôleur, le contrôleur RBF, est alors d'amener le pendule dans ce domaine pour que le contrôleur LQR puisse le stabiliser. Le contrôleur RBF consiste en la tangente hyperbolique de 256 gaussiennes (4 gaussiennes par dimension, pour 4 dimensions). Lorsque le pendule est proche de l'état but ($\theta_1 = \pi/2 \pm \pi/4, \theta_2 = 0 \pm \pi/4, \dot{\theta}_1 = 0 \pm \pi/2, \dot{\theta}_2 = 0 \pm \pi/2$, ce que l'on note $\mathbf{D}_\theta$), c'est le contrôleur LQR qui prend la main. Il doit être noté que bien que cela soit possible, nous n'opti-

misons pas le contrôleur LQR mais celui-ci est calculé *a priori* (voir Spong (1995)). De meilleurs contrôleurs pourraient certainement être définis mais nous souhaitons par cet exemple illustrer les capacités de l'algorithme par essaim à résoudre des problèmes d'optimisation dans des espaces de grande dimension. Le contrôleur est ainsi définis par les équations (9).

$$\tau = \begin{cases} -\mathbf{K}^T.\theta, \theta = (\theta_1 - \pi/2; \theta_2; \dot{\theta_1}; \dot{\theta_2}) & \text{si } \theta \in \mathbf{D}_\theta \\ 2\tanh(\sum_{j=1}^{256} a_j e^{-\frac{\sin^2(\theta_1-\theta_1^j)}{2\sigma_1^2} - \frac{\sin^2(\theta_2-\theta_2^j)}{2\sigma_2^2} - \frac{(\dot{\theta_1}-\dot{\theta_1}^j)^2}{2\sigma_3^2} - \frac{(\dot{\theta_2}-\dot{\theta}_2^j)^2}{2\sigma_4^2}}) & \text{sinon} \end{cases} \tag{9}$$

avec $\sigma_1 = \sigma_2 = 0.25, \sigma_3 = \sigma_4 = 4.5$, les moyennes des gaussiennes étant uniformément réparties dans $[-\pi/4, 5\pi/4] \times [-\pi/4, 5\pi/4] \times [-9, 9] \times [-9, 9]$.



a) b) c)

FIGURE 3: a) Description physique du problème de double pendule. Partant de l'état à la position verticale pointant vers le bas, le contrôleur doit amener le pendule en position verticale pointant vers le haut instable en appliquant un couple τ à l'articulation entre les deux segments. b) Nombre moyen de pas passés dans le domaine but en fonction de l'itération de l'essaim. Cette moyenne est calculée sur une trajectoire avec 300 répétitions de l'expérience. c) Comportement d'un contrôleur optimal trouvé par l'algorithme. Sa *fitness* est de l'ordre de 1600.

3.3.2. Résultats expérimentaux

L'expérience est répétée 300 fois. Une simulation est limitée à $20s.$ soit 2000 itérations. L'essaim évolue pendant 4000 itérations. Une récompense $r = 1$ est fournie chaque fois que le pendule est le domaine but $\mathbf{D}_\theta$ et $r = 0$ sinon. La moyenne des *fitness* est illustrée sur la figure 3b. Comme on peut le voir, l'essaim ne converge pas tout le temps vers un contrôleur optimal ce qui est probablement dû à l'architecture du contrôleur considérée. Par ailleurs, l'essaim a tendance à rester sur des plateaux. Il y a néanmoins des politiques proches de l'optimum comme celle illustrée sur la figure 3c. Cette dernière application permet d'illustrer que l'optimisation par essaim reste une solution viable pour l'optimisation dans des espaces de recherche de grande dimension.

4. Discussion

L'optimisation par essaim particulaire est un algorithme d'optimisation particulièrement efficace. En plus des différents résultats empiriques dans la littérature qui illustrent les performances de ces algorithmes, principalement sur des problèmes dont la fonction de coût est déterministe, nous avons montré que ces algorithmes sont performants sur des problèmes d'optimisation stochastiques. Ces algorithmes sont par ailleurs très simples à mettre en oeuvre puisqu'ils n'impliquent pas de dérivées de la fonction de coût et ne dépendent que de très peu de méta-paramètres. Pour certains problèmes d'apprentissage par renforcement, l'évaluation d'une trajectoire peut être coûteuse (*e.g.* le double pendule). Utiliser un grand nombre de trajectoires permet de réduire la stochasticité de la fonction de coût. L'optimisation par essaim particulaire (et cela reste valable pour d'autres algorithmes d'optimisation par population) a besoin de pouvoir comparer des individus et cette comparaison peut être réalisée par des tests statistiques dont on adapte le nombre de trajectoires considérées, comme le suggère Stagge (1998) (voir aussi Heidrich-Meisner & Igel (2009) pour la mise en oeuvre de cette idée dans la recherche de politique avec CMA-ES). Par ailleurs, il est prévu dans des futurs travaux de comparer l'approche par essaim particulaire à d'autres approches proposées dans la littérature reposant sur l'entropie croisée (Mannor *et al.*, 2003), l'algorithme évolutionnaire CMA-ES (Heidrich-Meisner & Igel, 2008), la montée de gradient proposée par (Baxter & Bartlett, 1999) ou d'autres algorithmes impliquant de calculer le gradient de la politique (Peters & Schaal, 2006).

Cette comparaison entre les algorithmes impliquera nécessairement une phase de méta-optimisation ou les méta-paramètres optimaux devront être trouvés. Une des faiblesses de l'approche que nous présentons réside dans le fait qu'il faut disposer d'un simulateur pour évaluer les qualités d'un contrôleur. Nous travaillons actuellement sur une adaptation de l'algorithme Fitted-Q qui permettrait de relâcher cette contrainte. Pour ce faire, on remplace l'estimation Monte-Carlo pour une approximation de la fonction de valeur (Geist & Pietquin, 2011). On peut également imaginer développer des versions en ligne de cet algorithme avec un contrôleur optimisant sa politique au fur et à mesure de ses interactions avec l'environnement.

Références

BAXTER J. & BARTLETT P. (1999). Direct gradient-based reinforcement learning. *Proceedings of the International Symposium on Circuits and Systems*, **3**, 271–274.

CLERC M. & KENNEDY J. (2002). The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comp.*, **6**(1), 58–73.

ENGELBRECHT A. (2005). *Fundamentals of Computational Swarm Intelligence*. Wiley.

ENGELBRECHT A. (2010). Heterogeneous Particle Swarm Optimization. In *Swarm Intelligence*, volume 6234 of *Lecture Notes in Computer Science*, p. 191–202. Springer.

FIX J. & GEIST M. (2011). `http://jeremy.fix.free.fr/spip.php?article33`.

GEIST M. & PIETQUIN O. (2011). Parametric Value Function Approximation : a Unified View. In *ADPRL 2011*, p. 9–16.

HEIDRICH-MEISNER V. & IGEL C. (2008). Evolution Strategies for Direct Policy Search. In G. RUDOLPH, Ed., *Parallel Problem Solving from Nature (PPSN X)*, LNCS 5199, p. 428–437. Berlin Heidelberg : Springer-Verlag.

HEIDRICH-MEISNER V. & IGEL C. (2009). Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search. In A. P. DANYLUK, L. BOTTOU & M. L. LITTMAN, Eds., *ICML*, volume 382 of *ACM International Conference Proceeding Series*, p.51 : ACM.

KENNEDY J. & EBERHART R. (1995). Particle swarm optimization. In *Proceedings IEEE International Joint Conference on Neural Networks*, p. 1942–1948.

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

MANNOR S., RUBINSTEIN R. & GAT Y. (2003). The cross entropy method for fast policy search. *International Conference on Machine Learning*, **20**(2), 512.

MUNOS R. & MOORE A. (1999). Variable Resolution Discretization for High-Accuracy Solutions of Optimal Control Problems. In *IJCAI*, p. 1348–1355.

PETERS J. & SCHAAL S. (2006). Policy Gradient Methods for Robotics. In *IROS*, p. 2219–2225 : IEEE.

SPONG M. W. (1995). The swing up control problem for the Acrobot. *IEEE Control Systems*, **15**, 49–55.

STAGGE P. (1998). Averaging Efficiently in the Presence of Noise. In A. E. EIBEN, T. BÄCK, M. SCHOENAUER & H.-P. SCHWEFEL, Eds., *PPSN*, volume 1498 of *Lecture Notes in Computer Science*, p. 188–200 : Springer.

SUTTON R. & BARTO A. (1998). *Reinforcement Learning : An Introduction*. MIT Press.