論　文

オンラインEMアルゴリズムによる強化学習法のacrobot制御への応用

吉本潤一郎[†]　　石井　信[†,††]　　佐藤　雅昭[††]

Application of Reinforcement Learning Based on On-Line EM Algorithm to Balancing of Acrobot

Junichiro YOSHIMOTO[†], Shin ISHII[†,††], and Masa-aki SATO[††]

あらまし　acrobot は 2 リンク 2 関節からなるロボットで，第 2 関節のみにアクチュエータが存在する．acrobot は非線形なダイナミクスを持ち，状態変数および制御変数の空間がともに連続であるために，強化学習によってこの制御を獲得することは難しい課題の一つである．本論文では，acrobot をバランスする制御に強化学習を応用する．我々の強化学習法は actor-critic アーキテクチャを用いて学習が行われる．actor は現在の状態に対して制御信号を出力し，critic は将来を通して得られる報酬の累積 (期待報酬) を予測する．actor と critic はともに正規化ガウス関数ネットワークによって近似され，オンライン EM アルゴリズムを用いて学習が行われる．また，critic の学習を促進させるための新たな手法を導入する．本手法が少ない試行回数から良い制御を獲得できることを計算機シミュレーションの結果により示す．

キーワード　acrobot，強化学習，actor-critic モデル，正規化ガウス関数ネットワーク，EM アルゴリズム

1. まえがき

人間は身体のダイナミクスに関する詳細な知識がなくても，試行錯誤を通して多くの複雑な運動制御を獲得することができる．強化学習はこのような実際の経験に基づいた機械学習の手法である．

強化学習はゲームの戦略獲得 [1], [2] のように有限個の状態と有限個の行動を持つマルコフ決定問題に広く応用され，成功を収めてきた．一方で，人体やロボットの運動制御の問題に強化学習を応用することは，対象システムの状態変数および制御変数の空間がともに連続であるため前者に比べてはるかに難しい．このような場合，近似精度と汎化能力に優れた関数近似器と高速な学習アルゴリズムが必須である．

以前に，我々はオンライン EM アルゴリズム [3], [4] に基づく強化学習法を提案し，状態変数および制御変数の空間がともに連続である 2 つのタスクに応用した [5]．本論文では，この手法を acrobot [6], [7] をバランスさせる制御へ応用することについて述べる．

我々の強化学習法は Barto らによって提案された actor-critic モデル [8] に基づくアーキテクチャを用いて学習が行われる．actor は現在の状態に対して制御信号を出力し，critic は将来を通して得られる報酬の累積 (期待報酬) を予測する．元の actor-critic モデルでは，TD 誤差と呼ばれる，期待報酬の予測値に対する時間的な誤差を用いて，actor は可能な各行動に対する選択確率を，critic は評価関数を，それぞれ近似していた [8]．この学習法は制御変数の空間が離散的で，actor の出力する可能な制御信号の数が少数であることを前提としたものである．したがって，制御変数の空間が連続である acrobot の制御への応用が困難である．

この問題点を解決するために，我々は actor および critic の近似する対象を元のモデルとは異なるものとし，かつ，学習法も異なるものを用いる．我々の学習法では，actor には現在の critic の出力を増大させるような制御信号を学習信号として用い，critic は現在の actor を用いて制御を行った場合の期待報酬 (Q 関数値) を Bellman 方程式 [9] に基づいて学習する．actor と critic の関数近似器にはともに正規化ガウ

† 奈良先端科学技術大学院大学情報科学研究科，生駒市
Graduate School of Information Science, Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara 630-0101, Japan
†† ATR 人間情報通信研究所，京都府
ATR Human Information Processing Research Laboratories, 2-2-2 Hikaridai, Seika, Soraku, Kyoto 619-0237, Japan

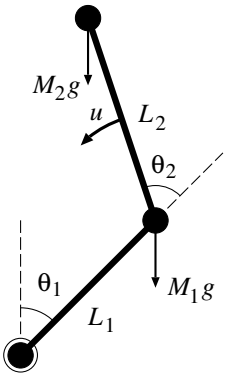


図 1　acrobot
Fig. 1　The acrobot

ス関数ネットワーク (normalized Gaussian network, NGnet) [10] を用いる．NGnet は正規化されたガウス関数により入力空間を滑らかに領域分割し，局所的に線形近似を行うモデルである．

TD 学習の枠組みの中で actor の制御信号を連続値を取り得るように拡張した先行研究として Morimoto ら [11] がある．ここでも NGnet による actor-critic アーキテクチャを用いているが，我々の手法と大きく異なる点がある．まず，actor と critic に TD 法に基づく学習法を用いている．もう一つは，NGnet の学習に勾配法を用いていることである．一方で，本手法では NGnet の学習にオンライン EM アルゴリズム [3] を用いる．以前の論文 [3] で，オンライン EM アルゴリズムは勾配法よりも高速に学習でき，時間とともに入出力分布が変化するような動的な環境においても有効であることが示されている．actor や critic の関数近似もこのような動的な環境における問題の一つであると考えられるので，オンライン EM アルゴリズムが有効に働くと期待できる．また，critic の学習を促進させるための新たな手法を導入する．

本手法が少ない試行回数から良い制御を獲得できることを計算機シミュレーションの結果により示す．

2. Acrobot

本論文で制御対象として取り上げる acrobot [6], [7] について概説する．acrobot は，図 1 で示されるような 2 リンク 2 関節からなるアクチュエータロボットであり，鉄棒運動のダイナミクスと類似している．ただし，腰の部分に対応する第 2 関節にトルクをかけることができるが，鉄棒を持つ手の部分に対応する第 1 関節にトルクをかけることはできない．図 1 のようにモデル化された acrobot のダイナミクスは以下の 2 階の微分方程式で与えられる．

$$d_{11}\ddot{\theta}_1 + d_{12}\ddot{\theta}_2 + c_1 + \phi_1 = -\mu_1\dot{\theta}_1 \qquad (1a)$$
$$d_{12}\ddot{\theta}_1 + d_{22}\ddot{\theta}_2 + c_2 + \phi_2 = u - \mu_2\dot{\theta}_2 \qquad (1b)$$

$$\begin{aligned}
d_{11} \equiv & M_1L_1^2 + M_2(L_1^2 + L_2^2 + 2L_1L_2\cos(\theta_2)) \\
d_{22} \equiv & M_2L_2^2 \\
d_{12} \equiv & M_2(L_2^2 + L_1L_2\cos(\theta_2)) \\
c_1 \equiv & -M_2L_1L_2\dot{\theta}_2\left(2\dot{\theta}_1 + \dot{\theta}_2\right)\sin(\theta_2) \\
c_2 \equiv & M_2L_1L_2\dot{\theta}_1^2\sin(\theta_2) \\
\phi_1 \equiv & -(M_1L_1 + M_2L_1)g\sin(\theta_1) \\
& - M_2L_2g\sin(\theta_1 + \theta_2) \\
\phi_2 \equiv & -M_2L_2g\sin(\theta_1 + \theta_2)
\end{aligned}$$

ここで，M_1 (M_2)，L_1 (L_2)，μ_1 (μ_2) はそれぞれ第 1 リンク (第 2 リンク) の質量，長さ，摩擦係数であり，acrobot の物理パラメータである．g は重力加速度である．θ_1 は頂点方向から計った第 1 リンクの角度であり，θ_2 は第 1 リンク方向からの第 2 リンクの角度である．また，$\dot{\theta}_1$ と $\dot{\theta}_2$ はそれぞれ第 1 リンクと第 2 リンクの角速度である．システムは 4 次元の連続な状態変数を持っており，これを $x_c \equiv \left(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2\right)$ と表記する．制御器はすべての状態変数を観測できるが，システムのダイナミクスや物理パラメータに関しては全く分からないものと仮定する．制御器によって発生されるトルク u は連続であり，$|u| \leqq u_{max}$ の範囲で制限されているものとする．

acrobot を制御するための課題として，最短時間でリンクを振り上げるもの [7] や，頂点位置付近でバランスさせるもの [6] などが考えられる．本論文では，後者の課題を取り上げる．この課題は，システムが強い非線形性を持っており，状態変数および制御変数の空間がともに連続であるため，強化学習の問題の中でも非常に難しいものの一つである．

3. NGnet とオンライン EM アルゴリズム

本節では，後に述べる制御関数と Q 関数の近似に用いる NGnet [10] とオンライン EM アルゴリズム [3] について説明する．

NGnet は N 次元の入力ベクトル x から D 次元の出力ベクトル y へ変換するモデルで，以下の式で定義される．

$$y = \sum_{i=1}^{M} \left(\frac{G_i(x)}{\sum_{j=1}^{M} G_j(x)} \right) (W_i x + b_i) \tag{2a}$$

$$G_i(x) \equiv (2\pi)^{-N/2} |\Sigma_i|^{-1/2} \times \exp\left[-\frac{1}{2}(x-\mu_i)' \Sigma_i^{-1} (x-\mu_i) \right] \tag{2b}$$

ここで，M は NGnet を構成するユニットの数，$i \in \{1, ..., M\}$ はユニット番号，プライム記号 $(')$ は転置を表している．$G_i(x)$ は，N 次元中心ベクトル μ_i と $(N \times N)$ 次共分散行列 Σ_i を持つ N 次元ガウス関数である．W_i と b_i はそれぞれ $(D \times N)$ 次線形回帰行列と D 次元バイアスベクトルである．以下では，$\tilde{W}_i \equiv (W_i, b_i)$ と $\tilde{x}' \equiv (x', 1)$ の表記法を用いる．

NGnet は入力と出力の組 (x, y) を確率事象 (不完全事象) とする確率モデルの出力期待値として定式化できる．各不完全事象に対しては必ず 1 つのユニットが選ばれるものと仮定し，選ばれたユニットの番号 i を隠れ変数とみなす．このとき，確率モデルは完全事象と呼ばれる (x, y, i) の組に対する確率分布 $P(x, y, i)$ を以下のように与えることによって定義される．

$$P(x, y, i|\theta) = M^{-1} G_i(x) (2\pi)^{-D/2} \sigma_i^{-D} \times \exp\left[-\frac{1}{2\sigma_i^2} (y - \tilde{W}_i \tilde{x})^2 \right] \tag{3}$$

ここで，$\theta \equiv \left\{ \mu_i, \Sigma_i, \sigma_i, \tilde{W}_i \;\middle|\; i = 1, ..., M \right\}$ はモデルパラメータの集合である．この確率分布から入力 x が与えられた時の出力 y の期待値 $E[y|x] \equiv \int y P(y|x) dy$ が求められ，これは NGnet の出力と一致する．すなわち，確率分布 (3) は NGnet の確率モデルを定義しているといえる．

T 個の不完全事象 (観測データ) $(X, Y) \equiv \{ (x(t), y(t)) \mid t = 1, ..., T \}$ が与えられると，確率モデル (3) のパラメータ θ は最尤推定法を用いて決定することができる．特に，(3) のように隠れ変数を持つモデルに対しては EM アルゴリズム [14] を用いることができる．EM アルゴリズムでは，以下の E ステップと M ステップを繰り返すことによってモデルパラメータの最尤推定量が漸近的に求まる．

• E(Expectation) ステップ : 現在のモデルパラメータの値を $\bar{\theta}$ とする．$\bar{\theta}$ を用いて各観測データ $(x(t), y(t))$ に対して i 番目のユニットが選ばれる事後確率をベイズ則を用いて以下の式で求める．

$$P(i|x(t), y(t), \bar{\theta}) = \frac{P(x(t), y(t), i|\bar{\theta})}{\sum_{j=1}^{M} P(x(t), y(t), j|\bar{\theta})} \tag{4}$$

• M(Maximization) ステップ : 事後確率 (4) を用いて，完全事象に対する期待対数尤度 $L(\theta|\bar{\theta}, X, Y)$ は以下の式で定義される．

$$L(\theta|\bar{\theta}, X, Y) = \sum_{t=1}^{T} \sum_{i=1}^{M} P(i|x(t), y(t), \bar{\theta}) \times \log P(x(t), y(t), i|\theta) \tag{5}$$

$L(\theta|\bar{\theta}, X, Y)$ が増加することは観測データ (X, Y) に対する対数尤度が増加することを意味している [14]．そこで，$L(\theta|\bar{\theta}, X, Y)$ を θ について最大化を行う．最大化の必要条件である $\partial L / \partial \theta = 0$ の解は以下で与えられる [15]．

$$\mu_i = \langle x \rangle_i(T) / \langle 1 \rangle_i(T) \tag{6a}$$

$$\Sigma_i^{-1} = \left[\frac{\langle x x' \rangle_i(T)}{\langle 1 \rangle_i(T)} - \mu_i(T) \mu_i'(T) \right]^{-1} \tag{6b}$$

$$\tilde{W}_i = \langle y \tilde{x}' \rangle_i(T) \left[\langle \tilde{x} \tilde{x}' \rangle_i(T) \right]^{-1} \tag{6c}$$

$$\sigma_i^2 = \frac{\langle |y|^2 \rangle_i(T) - \mathrm{Tr}\left(\tilde{W}_i \langle \tilde{x} y' \rangle_i(T) \right)}{D \cdot \langle 1 \rangle_i(T)} \tag{6d}$$

ここで，$\mathrm{Tr}(\cdot)$ は行列の対角和である．$\langle \cdot \rangle_i$ は事後確率 (4) に関する重み付き平均であり，以下の式で定義される．

$$\langle f(x, y) \rangle_i(T) \equiv \frac{1}{T} \sum_{t=1}^{T} f(x(t), y(t)) \times P(i|x(t), y(t), \bar{\theta}) \tag{7}$$

上記の EM アルゴリズムはバッチ学習であり，モデルパラメータはすべての観測データが与えられた後で更新される．

ここでは，データが逐次的に与えられ，その度ごとにモデルパラメータを更新できるオンライン学習法 [3] を示す．なお，Neal らは EM アルゴリズムのインクリメンタルな学習法 [12] を提案しているが，この手法は学習データを全て保存してから学習を行う必要があ

るため，データがオンライン的に多数与えられる強化学習などへの応用には不向きであると考えられる．

t 番目の観測データが与えられた後のモデルパラメータの推定値を $\theta(t)$ とする．オンライン EM アルゴリズムにおいて，重み付き平均 (7) は以下の式で置き換えられる．

$$\langle\!\langle f(x,y)\rangle\!\rangle_i(t) \equiv \eta(t)\sum_{\tau=1}^{t}\left(\prod_{s=\tau+1}^{t}\lambda(s)\right) \times f(x(\tau),y(\tau))P_i(\tau) \quad (8)$$

ここで，$P_i(\tau) \equiv P(i|x(\tau),y(\tau),\theta(\tau-1))$ である．パラメータ $\lambda(s)$ $(0 \leqq \lambda(s) \leqq 1)$ は過去の良くない推定値による効果を徐々に忘却するための忘却係数である．$\eta(t) \equiv \left[\sum_{\tau=1}^{t}\left(\prod_{s=\tau+1}^{t}\lambda(s)\right)\right]^{-1}$ は正規化係数であり，学習係数のような役割を果たしている．(8) 式より、

$$\langle\!\langle f(x,y)\rangle\!\rangle_i(t) = \frac{\eta(t)\lambda(t)}{\eta(t-1)}\langle\!\langle f(x,y)\rangle\!\rangle_i(t-1) + \eta(t)f(x(t),y(t))P_i(t) \quad (9)$$

という関係式が得られる。さらに，$1/\eta(t) = 1 + \lambda(t)/\eta(t-1)$ という関係が成り立つので，これらより，新しい重み付き平均 $\langle\!\langle\cdot\rangle\!\rangle_i(t)$ は以下の逐次計算式を用いて求めることができる．

$$\langle\!\langle f(x,y)\rangle\!\rangle_i(t) = (1-\eta(t))\langle\!\langle f(x,y)\rangle\!\rangle_i(t-1) + \eta(t)f(x(t),y(t))P_i(t) \quad (10)$$

新しい重み付き平均を用いて，新しいモデルパラメータは以下の逐次計算式によって求められる．

$$\mu_i(t) = \langle\!\langle x\rangle\!\rangle_i(t)/\langle\!\langle 1\rangle\!\rangle_i(t) \quad (11a)$$

$$\tilde{W}_i(t) = \tilde{W}_i(t-1) + \eta(t)P_i(t) \times \left(y(t) - \tilde{W}_i(t-1)\tilde{x}(t)\right)\tilde{x}'(t)\tilde{\Lambda}_i(t) \quad (11b)$$

$$\sigma_i^2(t) = \frac{\langle\!\langle |y|^2\rangle\!\rangle_i(t) - \mathrm{Tr}\left(\tilde{W}_i(t)\langle\!\langle \tilde{x}y'\rangle\!\rangle_i(t)\right)}{D\langle\!\langle 1\rangle\!\rangle_i(t)} \quad (11c)$$

$$\tilde{\Lambda}_i(t) = \frac{1}{1-\eta(t)}\left[\tilde{\Lambda}_i(t-1) - \Phi_i(t)\right] \quad (11d)$$

$$\Phi_i(t) \equiv \frac{P_i(t)\tilde{\Lambda}_i(t-1)\tilde{x}(t)\tilde{x}'(t)\tilde{\Lambda}_i(t-1)}{(1/\eta(t)-1)+P_i(t)\tilde{x}'(t)\tilde{\Lambda}_i(t-1)\tilde{x}(t)} \quad (11e)$$

ここで，$\tilde{\Lambda}_i(t) \equiv [\langle\!\langle \tilde{x}\tilde{x}'\rangle\!\rangle_i(t)]^{-1}$ は $\Sigma_i^{-1}(t)$ を求めるための補助変数である．$\Sigma_i^{-1}(t)$ は以下の $\tilde{\Lambda}_i(t)$ との関係式から求められる．

$$\tilde{\Lambda}_i(t)\langle\!\langle 1\rangle\!\rangle_i(t) = \begin{pmatrix} \Sigma_i^{-1}(t) & -\Sigma_i^{-1}(t)\mu_i(t) \\ -\mu_i'(t)\Sigma_i^{-1}(t) & 1+\mu_i'(t)\Sigma_i^{-1}(t)\mu_i(t) \end{pmatrix} \quad (12)$$

忘却係数 $\lambda(t)$ を以下のようにスケジューリングするとオンライン EM アルゴリズムは最尤推定量を求めるための確率近似法になっていることを示すことができる [3].

$$\lambda(t) \stackrel{t\to\infty}{\longrightarrow} 1 - (1-a)/(at+b) \quad (13)$$

ここで，a $(1 > a > 0)$ と b $(b > 0)$ は定数である．

また，データの入出力分布に応じてユニットを効率良く配置するために，ユニットの動的操作の機構を導入する [3]．$P(x(t),y(t)|\theta(t-1))$ は現在のパラメータ $\theta(t-1)$ の下で確率モデルがデータ $(x(t),y(t))$ を生成する確率を表している．もし，この確率がある閾値より小さいならば，現在の確率モデルを使ってそのデータを説明するのは困難である．この場合，新しいユニットをそのデータの説明のために生成する．重み付き平均 $\langle\!\langle 1\rangle\!\rangle_i(t)$ は，i 番目のユニットが t 番目までのデータを説明するのにどの程度使われたかを表している．もし，この値がある閾値より小さいならば，そのユニットはほとんど使われていないことを意味するので削除する．

ユニット i の領域において入力分布の次元が入力空間の次元から縮退している場合，$\tilde{\Lambda}_i(t)$ は t とともに指数的に発散する．このような入力分布に対処するために Σ_i^{-1} に対する正則化を以下のように行う．

$$\Sigma_i^{-1} = \left[\frac{\Xi_i(t) + \alpha\langle\!\langle \Delta_i^2\rangle\!\rangle_i(t) I_N}{\langle\!\langle 1\rangle\!\rangle_i(t)}\right]^{-1} \quad (14a)$$

$$\Xi_i(t) \equiv \langle\!\langle xx'\rangle\!\rangle_i(t) - \mu_i(t)\mu_i'(t)\langle\!\langle 1\rangle\!\rangle_i(t) \quad (14b)$$

$$\langle\!\langle \Delta_i^2\rangle\!\rangle_i(t) = \frac{\langle\!\langle |x|^2\rangle\!\rangle_i(t) - |\mu_i(t)|^2\langle\!\langle 1\rangle\!\rangle_i(t)}{N} \quad (14c)$$

ここで，I_N は $N \times N$ 次単位行列であり，α は小さな正定数である．(14) に対応する正則化された $\tilde{\Lambda}_i(t)$ は (11d) と同様にオンライン的に計算することができる [3].

4. 強化学習

本節では，オンライン EM アルゴリズムを用いた強化学習法 [5] について述べる．以下では，学習システムは各時刻において acrobot の状態を観測でき，各状態と各制御信号の組に対して直接報酬と呼ばれるスカラー値が与えられるものと仮定する．

4.1 Actor-critic モデル

我々が提案する学習法は actor-critic モデル [8] に基づくアーキテクチャを用いて学習が行われる．actor は現在の状態に対して制御信号を出力する制御器の役割を果たし，critic は将来にわたって得られる直接報酬の累積値 (期待報酬) を予測する．元の actor-critic モデルでは，TD 誤差と呼ばれる，期待報酬の予測値に対する時間的な誤差を用いて，actor は可能な各行動に対する選択確率を，critic は評価関数を，それぞれ近似していた [8]．しかしながら，我々の方法は以下で説明するように元の actor-critic モデルにおける学習法とは大きく異なる．

actor は現在の状態 $x_c(t)$ を観測すると制御関数 $\Omega(\cdot)$ に従って制御信号 u を生成する．すなわち，$u(t) = \Omega(x_c(t))$ である．その後，現在の状態 $x_c(t)$ は $u(t)$ とシステムのダイナミクスに従って $x_c(t+1)$ へと変化する．この時，学習システムには直接報酬 $r\left(x_c(t), u(t)\right)$ が与えられる．学習システムの目的は以下で定義される期待報酬を最大化する制御関数 $\Omega(\cdot)$ を求めることである．

$$V(x_c) \equiv \sum_{t=0}^{\infty} \gamma^t r\left(x_c(t), \Omega\left(x_c(t)\right)\right)\Big|_{x_c(0)=x_c} \tag{15}$$

ここで，$\gamma(0 < \gamma < 1)$ は減衰係数であり，(15) 式が発散するのを防ぐ．$V(x_c)$ は評価関数と呼ばれ，現在の制御関数 $\Omega(\cdot)$ に依存している．また，Q 関数は以下の式で定義される．

$$Q(x_c, u) = r(x_c, u) + \gamma V(x_c(t+1)) \tag{16}$$

ここで，$x_c(t) = x_c$ と $u(t) = u$ を仮定している．$Q(x_c, u)$ は，制御器が現在の状態 x_c に対しては制御信号 u を出力し，以後の状態に対しては常に制御関数 $\Omega(\cdot)$ に従って制御信号を出力した場合の期待報酬を表している．(15) 式と (16) 式より，評価関数 $V(\cdot)$ と Q 関数の間には以下の関係が成り立つ．

$$V(x_c) = Q(x_c, \Omega(x_c)) \tag{17}$$

また，(15)，(16)，(17) 式より，Q 関数は以下の条件式を満たさなければならない．

$$\begin{aligned} Q\left(x_c(t), u(t)\right) =& r\left(x_c(t), u(t)\right) \\ &+ \gamma Q\left(x_c(t+1), \Omega\left(x_c(t+1)\right)\right) \end{aligned} \tag{18}$$

これは Bellman 方程式 [9] と呼ばれている．我々の actor-critic モデルでは，critic は Bellman 方程式 (18) を用いて Q 関数の近似を行う．一方，actor は critic の出力を最大化する制御関数を近似するように学習更新される．actor と critic の関数近似器としてはともに NGnet を用いて，それぞれ，actor ネットワーク，critic ネットワークと呼ぶ．

学習過程は以下のように行われる．現在の状態 $x_c(t)$ に対して，現在の actor ネットワークが制御信号 $u(t)$ を出力する．次に，学習システムは次の状態 $x_c(t+1)$ と直接報酬 $r\left(x_c(t), u(t)\right)$ を観測する．

critic ネットワークはオンライン EM アルゴリズムを用いて学習を行う．critic ネットワークへの入力は状態信号と制御信号の組，すなわち，$(x_c(t), u(t))$ である．ターゲットとなる出力は (18) 式の右辺である．これは時刻 $t+1$ の状態 $x_c(t+1)$ に対する現在の actor ネットワークと現在の critic ネットワークの出力を用いて計算する．

学習システムは，ある一定時間 (t_{max}) の間，固定された actor ネットワークを用いて上記の制御を繰り返す．この間に，critic ネットワークは固定された actor ネットワークに対する Q 関数を近似するようにオンライン学習を行う．また，状態の軌道 $\{x_c(t) \mid t = 1, 2, ..., t_{max}\}$ はセーブされる．

この制御過程が終了すると，セーブされた状態の軌道に沿って actor の学習が行われる．actor ネットワークの学習にもオンライン EM アルゴリズムを用いる．actor ネットワークへの入力は $x_c(t)$ であり，ターゲットとなる出力は現在の critic ネットワークの勾配を用いて与えられる以下の u_{target} である [16]．

$$u_{target} = \Omega(x_c(t)) + \epsilon \frac{\partial Q}{\partial u}\left(x_c(t), \Omega\left(x_c(t)\right)\right) \tag{19}$$

ここで，$Q(\cdot)$ と $\Omega(\cdot)$ はそれぞれ現在の critic ネットワークと現在の actor ネットワークの出力である．ϵ は小さな正定数を表している．(19) 式の u_{target} は現在の actor の出力である制御信号 $\Omega\left(x_c(t)\right)$ よりも現

在の critic ネットワークの出力を増加させるより良い制御信号となるように決められている．以上の手続きを学習ステージにおける 1 エピソードと定義する．

4.2 Critic の学習促進

ここでは，critic の学習を促進するための新たな手法を導入する．この手法は TD 学習における TD(λ) 法に相当するものである [13].

(15)，(16)，(17) 式より，もし制御信号の系列が固定された制御関数に従って決定的に与えられているならば，Q 関数は任意の正整数 n に対して以下の条件式を満たさなければならない．

$$\begin{aligned} & Q\left(x_c(t), u(t)\right) \\ =& \sum_{s=1}^{n} \gamma^{s-1} r\left(x_c(t+s-1), u(t+s-1)\right) \\ & + \gamma^n Q\left(x_c(t+n), u(t+n)\right) \\ \equiv & R_n(t) \end{aligned} \tag{20}$$

$R_n(t)$ は $x_c(t)$ と $u(t)$ に対する n ステップ報酬と呼ばれる．もし n ステップ報酬が求められるならば，Q 関数の近似のために有効に使うことができる．また，もし n ステップ報酬のすべての組 $\{R_1(t), R_2(t), ..., R_{t_{max}-t}(t)\}$ が求められるならば，以下で定義される n ステップ報酬の重み付き平均も Q 関数の近似に有効に使うことができる．

$$R(t) \equiv \eta_\lambda(t) \sum_{n=1}^{t_{max}-t} \lambda_R^{n-1} R_n(t) \tag{21}$$

ここで，λ_R $(0 \leqq \lambda_R \leqq 1)$ は減衰係数であり，各時刻で与えられる報酬による効果を過去に観測された状態へ伝搬する割合を示す．$\eta_\lambda(t) \equiv \left[\sum_{n=1}^{t_{max}-t} \lambda_R^{n-1}\right]^{-1}$ は正規化係数である．セーブされた各状態に対する関数 $R(t)$ は，観測された報酬と critic ネットワークの出力の系列を用いて求めることができる．critic ネットワークの近似がまだ十分に行われていない場合，次の時刻で与えられる報酬しか用いない 1 ステップ報酬はターゲットとしてあまり信頼できない．このような状況は，特に学習ステージの初期で起こる．このような場合，将来に与えられる多くの報酬を用いる $R(t)$ の方がターゲットとしてふさわしい．

しかし，系列の最後まで待たなければ $R(t)$ を求めることはできない．そのために，学習過程を制御過程と同時に行っていく場合には，ターゲットとして $R(t)$ を用いることは適していない．したがって，新しい学習スキームでは，1 ステップ報酬を用いた critic の学習を制御過程と並行してオンライン的に行い，制御過程が終了した後に，$R(t)$ を用いて再び critic の学習をバッチ的に行う．

Initialize the critic network and the actor network
Repeat (for each episode) :
1. Initialize $x_c(0)$, $u(0)$
2. Repeat (for t = 1, 2, ... , t_{max}) :
   (a) Observe $x_c(t)$ and $r(t)$
   (b) $u(t) \leftarrow \Omega(x_c(t))$
   (c) Train the critic network
      input : $(x_c(t\text{-}1), u(t\text{-}1))$
      target : $r(t) + \gamma Q(x_c(t), u(t))$
   (d) $e \leftarrow 1$; $\Gamma \leftarrow 1$;
      $q(t\text{-}1) \leftarrow 0$; $R(t\text{-}1) \leftarrow 0$;
   (e) Repeat (for s = 1, 2, ... , t) :
      $q(t\text{-}s) \leftarrow q(t\text{-}s) + \Gamma r(t)$
      $\Gamma \leftarrow \Gamma\gamma$
      $R(t\text{-}s) \leftarrow R(t\text{-}s) + e\,[q(t\text{-}s) + \Gamma Q(x_c(t), u(t)) - R(t\text{-}s)]$
      $e \leftarrow \lambda e / [1 + \lambda e]$
3. Train the critic network for t = 0, 1, ... , t_{max} - 1
      input : $(x_c(t), u(t))$
      target : $R(t)$
4. Train the actor network for t = 0, 1, ... , t_{max} - 1
      input : $x_c(t)$
      target : $u(t) + \varepsilon \frac{\partial Q}{\partial u}(x_c(t), u(t))$

図 2　学習スキーム
Fig. 2　Learning scheme

この新しい学習スキームは図 2 のようにまとめられる．ここで，簡潔さのために $r\left(x_c(t-1), u(t-1)\right)$ を $r(t)$ で表現した．ステップ 2 (e) において $e = \lambda_R^{s-1}(1-\lambda_R)/(1-\lambda_R^s)$ は学習係数の役割を果たす．また，補助変数 $q(t)$ は直接報酬の重み付き和を表す．

過去の状態 $x_c(t)$ と制御信号 $u(t)$ の組に対する $R(t)$ は，e と $q(t)$，Γ を用いてステップ 2 (e) の逐次計算式で求められる．この逐次計算式の第 2 項は直接報酬 $r(t)$ の代りに重み付き直接報酬和 $q(t)$ ，減数係数 γ の代りに $\Gamma = \gamma^s$ を用いた時の TD 誤差と同じ形をしていることに注意する．t_{max} が大きい場合には，ステップ 2 (e) における反復計算は計算量が大きくなってしまう．しかしながら，ステップ 2 (e) で e が非常に小さくなった後では，$R(s)$ はもう変化しないことが分かる．この性質によってこのステップは簡略化できる．すなわち，e は指数的に減衰するので，T_0 を $\lambda_R^{T_0} \approx 0$ を満足する整数として，2 (e) の繰り返し計算は $s = 1, ..., T$ $(T = \min(T_0, t))$ に対してのみ行えばよい．したがって，新しい学習スキームを用いても計算量はそれほど増加しない．

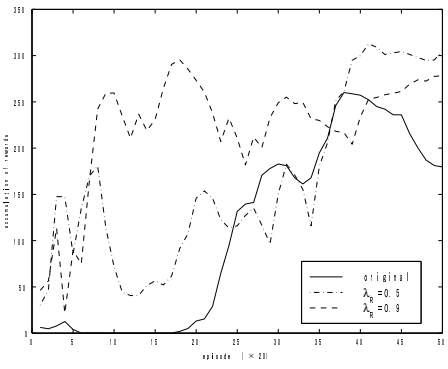


図 3　学習スキームの比較
Fig. 3　Comparison between learning schemes

5. 実　　験

5.1　λ_R の効果

困難なタスクである acrobot の制御について述べる前に，λ_R の導入による効果を簡単な比較実験によって示す．実験に用いたタスクは不十分な最大トルクしか出ない制御器によって倒立振子の振り上げを行うもの [18] である．各エピソードは垂れ下がりの位置で静止している状態を初期状態として 7 秒間制御と学習を行う．報酬は振子の先端の高さに比例して 0 から 1 までの値が与えられる．比較する学習方法は，1 ステップ報酬だけを用いる元の学習法と図 2 で示される新しい学習法である．λ_R 以外の学習パラメータはすべて同じものを用いるものとする．

図 3 はそれぞれの学習方法における獲得された報酬の累積の変化を示したものである．実線は 1 ステップ報酬だけを用いる元の学習法によるもの，破線 (一点鎖線) は $\lambda_R = 0.9$ ($\lambda_R = 0.5$) の時の新しい学習法によるものであり，各データ点は 1 エピソード間に獲得された報酬の累積を 20 エピソードごとに平均化したものである．いずれの場合でも学習の進行にともなって得られる報酬の累積は増加するが，1 ステップ報酬だけを用いる手法に比べて，λ_R を導入したものは学習の初期においても多くの報酬を獲得できており，学習が速く進行することが分かる．

5.2　acrobot の制御

acrobot の物理パラメータとして $M_1 = M_2 = 1.0$, $L_1 = L_2 = 1.0$, $\mu_1 = \mu_2 = 0.01$, $g = 9.8$, $u_{max} = 30.0$ を用いた．システムのダイナミクスは (1) 式に従って決定的である．学習システムは $\Delta\tau = 0.01$ 秒ごとにシステムの状態を観測し，制御信号を出力する．ダイナミクスは時間間隔 $\Delta\tau$ の Runge-Kutta 法を用いて積分される．直接報酬 $r\left(x_c(t), u(t)\right)$ は $\tilde{r}\left(x_c(t+1)\right)$ で与えられるものとし，$\tilde{r}\left(x_c(t+1)\right)$ は以下で定義した．

$$\tilde{r}(x_c) = \exp\left(\frac{-(1-h)^2}{2\nu_1^2} - \frac{\dot{\theta}_1^2}{2\nu_2^2} - \frac{\dot{\theta}_2^2}{2\nu_3^2}\right) - 1 \tag{22a}$$

$$h \equiv \frac{L_1\cos\theta_1 + L_2\cos(\theta_1+\theta_2) + L_1 + L_2}{2(L_1+L_2)} \tag{22b}$$

ここで，ν_1，ν_2，ν_3 は正定数である．$h \in [0,1]$ は第 2 リンクの先端の高さを正規化したものである．報酬 $\tilde{r}(x_c) \in [-1,0]$ は頂点位置で静止している時に最大となり，acrobot が高い位置に留まりやすくなるように設計されている．

頂点位置付近から acrobot を放し，actor を用いた制御過程を 7 秒間行う．すなわち，1 エピソードは $t_{max} = 700$ ステップである．状態変数の初期値，すなわち，$\theta_1(0)$，$\theta_2(0)$，$\dot{\theta}_1(0)$，$\dot{\theta}_2(0)$ は 0 を中心とした一様分布からランダムに生成され，その分布の範囲はエピソードの回数にともなって線形的に広げられる．学習は図 2 で示される過程で行われる．強化学習ではより良い制御関数を探索するために確率的制御器が必要となることが多いが，この問題においては，acrobot のシステムがカオス的な性質を持っているために決定的制御器の方がよいと思われる．確率的制御器を用いる代わりに，すべての x_c と u に対して $Q(x_c, u) = 0$ となるように critic を初期化することによって制御空間の探索が行われやすいようにした [13]．ここで，すべての直接報酬が 0 以下であるために，真の Q 関数では任意の x_c と u に対して $Q(x_c, u) \leqq 0$ となることに注意する．すなわち，この初期化では任意の状態に対する任意の制御信号が最善のものであると仮定している．実際に試行された制御信号に対しては Q 関数値が負になるので，より良い制御を探索する時に最善の Q 関数値 (=0) を持つまだ試行されていない制御信号が選ばれやすくなり，結果として制御空間の探索が行われることになる．

以上の条件のもとで実験を行った結果として，37 エピソード後に制御器は初期状態として頂点位置に近い場合に acrobot を直立させることができる．図 4 と図 5 は学習後の典型的な制御過程を示している．図 4 では acrobot の動きをストロボ的に表している．図 5 は

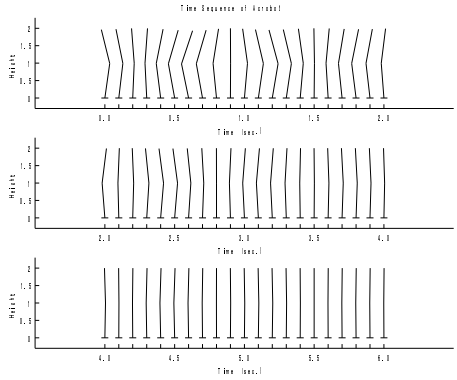


図 4　学習後の制御の様子
Fig. 4　A typical control process after learning

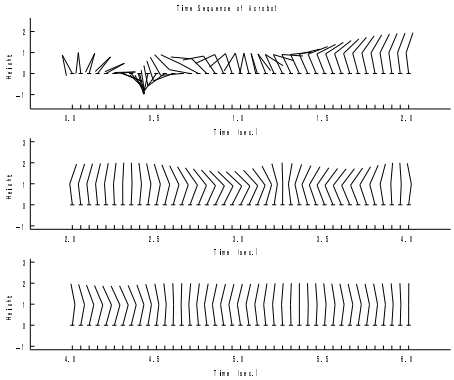


図 6　難しい初期状態からの成功例
Fig. 6　Successful example from a difficult initial state

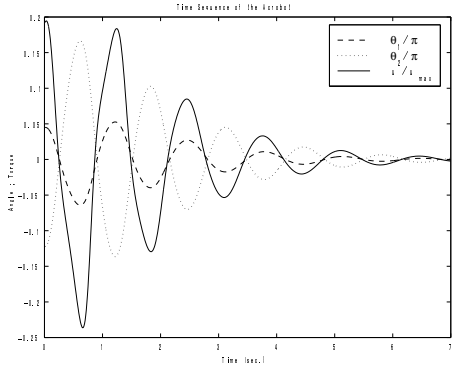


図 5　図 4 に対応する状態と制御信号の時系列
Fig. 5　State/control time-series corresponding to Figure 4

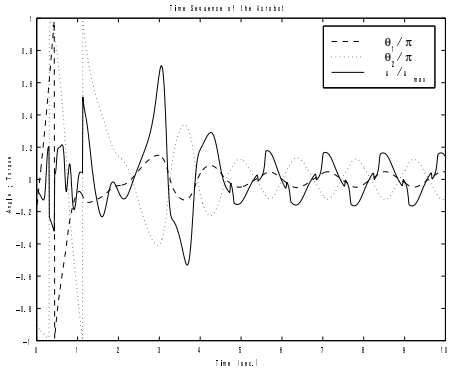


図 7　図 6 に対応する状態と制御信号の時系列
Fig. 7　State/control time-series corresponding to Figure 6

システムの状態と制御信号の時系列を示している．ここで，点線と破線は，それぞれ，θ_1/π，θ_2/π を，実線は actor によって生成された制御信号 u/u_{max} を表している．制御器は第 2 リンクの振動を徐々に減衰させ最後には頂点位置で静止させる．また，学習システムの初期状態によっては，図 6 と図 7 に示すような異なった制御を学習する．図 6 の制御過程の初期では，第 1 リンクは大きな初期速度のために回転して倒れてしまう．しかし，actor は 2 つのリンクを振り上げるような制御信号を出力し，最後には頂点位置付近でリミットサイクル的に振動させながらバランスを保つ．

この実験で，学習後の actor ネットワークと critic ネットワークが必要とするユニットの数は，それぞれ，38 個と 54 個である．オンラインでの強化学習において学習を安定して行うためには，NGnet のような局所モデルの方が多層パーセプトロンのような大域モデルよりも適している [17]．一方で，局所モデルで必要となるユニットの数は，一般に，入力次元が増加するにつれて指数的に増加する．にもかかわらず，本手法はユニットの動的操作の機構によってユニットを効率良く配置できるために，システムの次元に比べて必要となるユニットの数はあまり増加しない．したがって，本手法は高次元のシステムを制御するような問題に対しても有効であると期待できる．本手法では，37 エピソードの学習を行うのに，ワークステーション DEC Alpha Station 600 au を用いて 135 秒かかった．実際のシステムの物理的な時間は 260 秒である．この学習時間は十分に短いものであり，実問題に対しても応用可能であると考えられる．

6. むすび

本論文では，オンライン EM アルゴリズムを用いた

actor-critic アーキテクチャを acrobot をバランスさせる制御の問題に応用した．また，critic ネットワークの学習を促進させるための手法を新しく導入した．計算機シミュレーションの結果によれば，本手法は少ない試行回数から acrobot に対する良い制御を獲得できる．強化学習法を現実のシステムの自動制御の問題に応用するためには，近似精度と汎化能力に優れた関数近似器と高速な学習アルゴリズムが必須である．本論文の実験結果は我々の手法がこれらの特徴を持っていることを示すものであると考えられる．

謝辞　本研究は、科学技術庁の平成 11 年度科学技術振興調整費による「ヒトを含む霊長類のコミュニケーションの研究」の一環として行なわれた。本研究に関して、国際電気通信基礎技術研究所 (ATR) の銅谷賢治博士、および論文査読委員のコメントに感謝する。

文　献

[1] Tesauro, G., “Practical issues in temporal difference learning,” *Machine Learning*, **8**, pp. 257-277, 1992

[2] Yoshioka, T., Ishii, S., & Ito, M., “Strategy acquisition for game othello based on min-max reinforcement learning,” *IEICE Transactions on Information and Systems*, to appear

[3] Sato, M., & Ishii, S., “On-line EM algorithm for the normalized Gaussian network,” *Neural Computation*, **12**(2), 2000

[4] 石井信，佐藤雅昭，“ 正規化ガウス関数ネットワーク，Mixture of experts と EM アルゴリズム，”日本神経回路学会誌，**6**(1)，pp. 30-40，1999

[5] Sato, M., & Ishii, S., “Reinforcement learning based on on-line EM algorithm,” in *Advances in Neural Information Processing Systems 11*, Kearns, M. S., Solla S. A., & Cohn, D. A., eds., pp. 1052-1058, MIT Press, Cambridge, MA, 1999

[6] Hauser, J., & Murray, R. M., “Nonlinear controllers for non-integrable systems: The acrobot example,” in *Proceedings of the 1990 American Control Conference*, **1**, pp. 669-671, San Diego, CA, USA, 1990

[7] Sutton, R. S., “Generalization in reinforcement learning: Successful examples using sparse coarse coding,” in *Advances in Neural Information Processing Systems 8*, Touretzky, D. S., Mozer, M. C., & Hasselmo M. E., eds., pp. 1038-1044, MIT Press, Cambridge, MA, 1996

[8] Barto, A. G., Sutton, R. S., & Anderson, C. W., “Neuronlike adaptive elements that can solve difficult learning control problems,” in *IEEE Trans. Systems, Man, and Cybernetics*, **SMC-13**, pp. 834-846, 1983

[9] Bellman, R. E., *Dynamic Programming*, Princeton University Press, Princeton, 1957

[10] Moody, J., & Darken, C. J., “Fast learning in networks of locally-tuned processing units,” *Neural Computation*, **1**, pp. 281-294, 1989

[11] Morimoto J. & Doya, K., “Reinforcement learning of dynamic motor sequence: Learning to stand up”, in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, **3**, pp. 1721-1726, 1998

[12] Neal, R. M. & Hinton, G. E., “A view of the EM algorithm that justifies incremental, sparse, and other variants,” in *Learning in Graphical Models*, Jordan M. I., eds., pp. 355-368, Kluwer Academic Publishers, Dordrecht, 1998

[13] Sutton, R. S., & Barto, A. G., *Reinforcement Learning*, MIT Press, Cambridge, MA, 1998

[14] Dempster, A. P., Laird, N. M., & Rubin, D. B., “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of Royal Statistical Society B*, **39**, pp. 1-22, 1977

[15] Xu, L., Jordan, M. I., & Hinton, G. E., “An alternative model for mixtures of experts,” in *Advances in Neural Information Processing Systems 7*, Tesauro, G., Touretzky, D. S., & Leen, T. K., eds., pp. 633-640, MIT Press, Cambridge, MA, 1995

[16] Sofge, D. A., & White, D. A., “Applied learning - optimal control for manufacturing,” in *Handbook of Intelligent Control*, White D. A., & Sofge D. A., eds., pp. 259-282, Van Nostrand Reinhold, New York, 1992

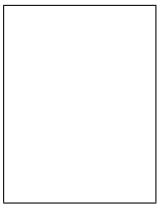
[17] Boyan, J. A. & Moore, A. W., “Generalization in reinforcement learning: Safely approximation the value function,” in *Advances in Neural Information Processing Systems 7*, Tesauro, G., Touretzky, D. S., & Leen, T. K., eds., pp. 369-376, MIT Press, Cambridge, MA, 1995

[18] Doya, K., “Temporal difference learning in continuous time and space” in *Advances in Neural Information Processing Systems 8*, Touretzky, D. S., Mozer, M. C., & Hasselmo M. E., eds., pp. 1073-1079, MIT Press, Cambridge, MA, 1996

（平成 11 年 7 月 1 日受付，10 月 1 日再受付）

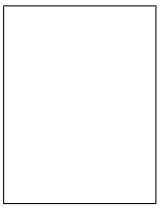
吉本潤一郎

1998 年関西大学総合情報学部卒．同年奈良先端科学技術大学院大学情報科学研究科博士前期課程に入学．現在に至る．主に，ニューラルネットワークと強化学習に興味を持つ．



石井　　信　（正員）

昭 61 東大・工卒．昭 63 同大学院修士課程 (情報工学) 了．工博．(株) リコー中央研究所研究員，ATR 人間情報通信研究所研究員を経て，現在，奈良先端科学技術大学院大学情報科学研究科 助教授



佐藤　雅昭　（正員）

昭 50 阪大・理・物理学科卒．昭 55 阪大・理・博士課程・物理学専攻了．ニューヨーク大学助手，フロリダ大学助手，ATR 視聴覚機構研究所主任研究員を経て，平 5 より ATR 人間情報通信研究所主任研究員．非線形力学系，カオス，ニューラルネットワーク，強化学習，ロボット制御，統計的学習理論の研究に従事．