強化学習の連続値への適用

柴田 聡志*　　神谷 昭基**

Reinforcement Learning Applied to Continuous Problems

Satoshi SHIBATA　　Akimoto KAMIYA

Abstract- The reinforcement learning is drawing attention as a technique that can solve problems automatically and adaptably. With the reinforcement learning, a series of actions is learned by maximizing a reward signal through trial and error. In general, the reinforcement learning is used for discrete problems. In order to solve real-world problems that tend to include continuous numbers, a couple of methods which is capable to deal with continuous problems has been proposed so far. In this paper, we discuss how the learning performance will be affected by changing the number of features within a linear architecture method, a type of method applied for solving continuous problems.

Key Words: reinforcement learning, function approximation

1　はじめに

近年，コンピュータに対する環境への自律性及び適応性が期待され，環境への自律性及び適応性に対処できる手法である強化学習が注目されている．強化学習では，試行錯誤を通じて，数値化された報酬信号を最大にするための一連の行動を学習する．報酬は，目的に達した行動に与えられる．しかし，報酬を最大にするためには，報酬が与えられない行動も重要な場合がある．そこで強化学習では，試行錯誤を通じて，初期に実行した行動に対しても報酬を伝播し，目的を達成するための一連の行動を学習する．初期に実行した行動とっては，報酬が遅れて発生することになる．これを遅延報酬と呼ぶ．これらの試行錯誤的な探索と遅延報酬が，強化学習の特徴である．

また，強化学習は一般的に離散値問題を対象とする．しかし，現実の問題では連続値を取り扱う必要がある場合が少なくない．そこで，強化学習で連続値問題を取り扱うために関数近似をよく用いる．関数近似手法には，線形の関数近似と非線形の関数近似がある．線形の関数近似は，複数の関数を線形的に結合し近似する手法であり，非線形の関数近似は複数の関数を非線形的に結合する手法である．強化学習では，線形の関数近似がよく利用される．その理由として，非線形の関数近似を用いると，大域最適解への収束は保証されず，学習が発散するからである [1]．線形の関数近似には，線形アーキテクチャと呼ばれる手法がある．線形アーキテクチャとは，連続値空間を一定の範囲ごとに分割し，分割した部分に関数を使用して連続的に補間を行う手法である．

本論文では，線形アーキテクチャの特徴数を変化させて実験を行い，その実験結果の考察を行う．実験では，線形アーキテクチャを用いることで近似した価値関数の大域最適解を獲得できることが保証される，強化学習アルゴリズムの一種である Q-learning を使用する [1]．以上より，線形アーキテクチャの特徴数を変化させることで，学習性能にどのような影響を与えるかを検証する．

2　強化学習

強化学習とは，試行錯誤を通じて未知の環境に適応する学習制御の枠組みである [2]．一般的な教師付き学習では学習機構の外部から理想的な出力の教師信号が与えられる．しかし，現実的には理想的な出力の例を

* 釧路高専専攻科 電子情報システム工学専攻
** 釧路高専情報工学科

得ることが困難である問題が多く存在する．これに対し，強化学習では状態入力に対する正しい行動出力を明示的に示す教師を必要とせず，代わりに報酬というスカラーの情報を手がかりに学習を行う．環境との相互作用の繰り返しを通じて，最適な政策を学習することが強化学習の目的である．最適な政策とは，一連の行動決定において，各状態で常に最適な行動を選択することである．最適な行動とは，将来にわたって最も多くの報酬を得られる行動である．ここで，強化学習の枠組みを図 1 に示す．

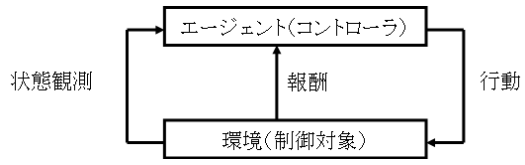


図 1: 強化学習の枠組み

図 1 のように，行動決定は「エージェント」によって行われる．エージェントは「環境」の状態を観測し，その状態に対応した行動を行う．エージェントの行動によって環境の状態が遷移し，エージェントはその遷移によって目的を達した場合に報酬を得る．つまり，強化学習では，状態観測 → 行動 → 状態遷移 → 報酬 (目的に達した場合) を繰り返す．

強化学習で扱う問題は，エージェントが何回かの状態遷移を繰り返した後に初めて報酬を得ることが多い．例えば，迷路問題でゴールまでの最短経路を導出する問題の場合には，ゴールに辿り着いた行動に対して報酬が与えられる．強化学習では，この報酬によって，エージェントの実行した一連の行動が正しかったどうかを判断する必要がある．

そこで，強化学習は一連の行動によって得た報酬を，ゴールに辿り着いた最後の行動以外の行動にも伝播する．この報酬の伝播を試行錯誤により繰り返すことで，ゴールに辿り着く一連の行動を強化する．そして，伝播された報酬を用いて，各状態でどの行動を取ればいいか (政策) を学習する．すなわち，エージェントは環境に対して行動を起こして報酬を得るまでのサイクルを繰り返し，試行錯誤しながら学習することによって，政策を学習する．この報酬の遅れにも対応できることが，強化学習の最大の特長と言える．

3 Q-learning

Q-learning とは，離散有限マルコフ決定過程下において各状態で取れる全ての行動を十分に実行することで，最適な政策を学習できる強化学習アルゴリズムの 1 種である．Q-learning では，各状態でどの行動が良いかを評価するために，各状態と行動の全ての組合せに評価値というものを持たせる．Q-learning では，この評価値を Q 値と呼ぶ．Q-learning のアルゴリズムを以下に示す [3].

Q-learning アルゴリズム

1. エージェントは環境の状態 s_t を観測する．
2. エージェントは与えられる行動選択法に従って行動 a_t を実行する．
3. 環境から報酬 r_{t+1} を受け取る．
4. 状態遷移後の状態 s_{t+1} を観測する．
5. 以下の更新式により Q 値を更新する．

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)] \quad (1)$$

   α は学習率 ($0 < \alpha \leq 1$)，γ は割引率 ($0 \leq \gamma < 1$) をそれぞれ表している．
6. 状態 s_{t+1} が目的状態ならば終了する．
7. 時間ステップ t を $t+1$ へ進めて 1 へ戻る．

上記アルゴリズム内の t は現在の時刻，s_t は時刻 t における状態，a_t は時刻 t において実行した行動，r_{t+1} は状態 s_t における行動 a_t を取り状態 s_{t+1} に遷移したことによって得られる報酬，$Q(s_t,a_t)$ は状態 s_t における行動 a_t の Q 値をそれぞれ表している．

α と γ は，強化学習の学習パラメータである．学習率 α の値が小さい方が更新量が小さく，逆に大きい方が更新量が大きくなる．一般的に $\alpha = 0.1$ ぐらいの値を設定することが多い [4]．α の値が小さい方が評価値が少しずつ更新され，Q-learning のアルゴリズムを繰り返すことで，漸近的に最適な評価値に近づけることができる．また，割引率 γ は，将来の報酬が現在においてどれだけ価値があるかを決定するパラメータで，一般的に $\gamma = 0.9$〜0.99 ぐらいの値を設定することが多い [4].

4 連続値への適用 [5]

現実の問題ではエージェントに与えられる状態やエージェントが行う行動が連続値である場合が少なくない．

そして，連続値空間では，距離的に近い状態や行動ではQ値も近い値を持つと仮定すると，Q値の表現に関数近似を用いることができる．関数近似を用いることで，状態の汎化による状態数の減少によって学習を高速化することができる．また，今までに経験したことのない状態に遭遇しても，似た状態での経験を生かして適切な行動選択ができる．代表的な関数近似法として，三角分布や動径基底関数 (RBF) を使用して関数近似を行う線形アーキテクチャが提案されている．以下の説明では，簡単化のために1次元の状態を対象として説明する．また，行動に対しても同様な近似が行うことができる．ここでは，行動に対する近似の説明を省略する．

4.1 線形アーキテクチャによる近似

線形アーキテクチャによる近似を行う真の価値関数の一例を図2に示す．

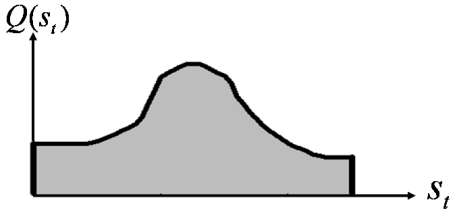


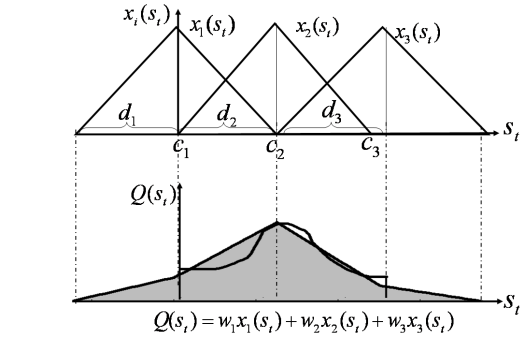
図 2: 真の価値関数

線形アーキテクチャは，複数の関数 (特徴ベクトル) にそれぞれ重み (重みベクトル) を掛け合わせることで，真の価値関数を近似する手法である．線形アーキテクチャを数式化したものを以下に示す．

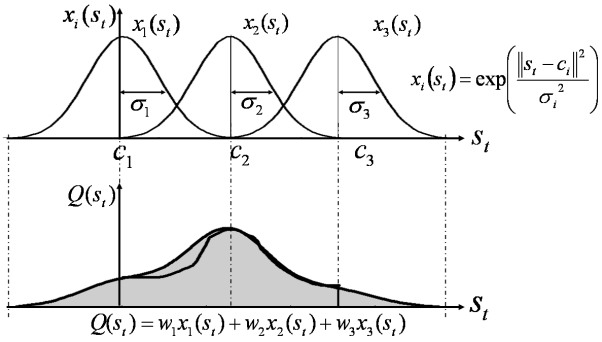
$$\begin{aligned} Q(s_t) &= \sum_{i=1}^{n} w_i x_i(s_t) \\ &= \vec{w}\vec{x}^T \\ &= [w_1 \quad w_2 \quad \dots \quad w_n] \\ &\quad [x_1(s_t) \quad x_2(s_t) \quad \dots \quad x_n(s_t)]^T \end{aligned} \tag{2}$$

式 (2) における $Q(s_t)$ は状態 s_t におけるQ値，$\vec{w}$ は重みベクトル，$\vec{x}$ は特徴ベクトル，n は特徴数をそれぞれ表している．そして，線形アーキテクチャでは，$\vec{x}$ に対して三角分布やRBFを使用する方法が提案されている．特徴数 $n = 3$ として，三角分布やRBFなどの分布関数を使用して近似する方法を説明する．

まず，三角分布及びRBFによる真の価値関数の近似例を図3を示す．



(a) 三角分布を用いた近似



(b)RBF を用いた近似

図 3: 線形アーキテクチャによる近似

図3の $c_i(i = 1, 2, 3)$ は各特徴の中心位置，三角分布の $d_i(i = 1, 2, 3)$ とRBFの $\sigma_i(i = 1, 2, 3)$ はそれぞれ補間を行う範囲を表している．特徴ベクトル $\vec{x}$ に対して重みベクトル $\vec{w}$ を掛け合わせることで，真の価値関数の近似を行っている．

三角分布による近似の特徴は，特徴の中心で微分不可能であり，線形的に周囲を補間することである．また，RBFによる近似の特徴は，全ての領域において微分可能であり，指数的に周囲を滑らかに補間することである．しかし，どちらの近似が優れているかは一概に言えず，真の価値関数の形状によって三角分布が良いか，RBFが良いかは変わると考えられる．

線形アーキテクチャに関数近似では，特徴数 n を増やすことで近似性能を向上させることが可能である．

4.2 線形アーキテクチャにおける Q-learning の更新処理

以下に線形アーキテクチャを用いた場合のQ-learningの更新式を示す．また，更新式では，行動も含めている．

線形アーキテクチャにおける更新式

・線形アーキテクチャにおける Q 値の表現

$$Q(s_t, a_t) = \sum_{i=1}^{n} w_i x_i(s_t, a_t) \quad (3)$$

・重みベクトルの更新処理

$$w_i \leftarrow w_i + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] x_i(s_t, a_t) \quad (4)$$

線形アーキテクチャを用いた場合，式 (3) のように Q 値は，特徴ベクトル $\vec{x}$ と重みベクトル $\vec{w}$ を掛け合わせることで表現される．この時，重みベクトル $\vec{w}$ を更新することで，Q 値が更新されることになる．この重みベクトル $\vec{w}$ の更新には，式 (4) を使用する．式 (4) では，現在の Q 値である $Q(s_t, a_t)$ を目標の Q 値である $r_{t+1} + \gamma \max_a Q(s_{t+1}, a_t)$ に近づけるように重み w_i を更新する．ここで，式 (4) において現在の Q 値を目標の Q 値に最も近づける方向の勾配

$$\frac{\partial Q(s_t, a_t)}{\partial w_i} = x_i(s_t, a_t) \quad (5)$$

を掛けることで，勾配を降るように真の価値関数との誤差を小さくすることができる．

5 実験

5.1 実験の目的

強化学習を連続値に適用させる手法として，線形の関数近似である線形アーキテクチャを使用する．ここでは，学習率 α と線形アーキテクチャの特徴数 n によって，どのような影響が出るか実験を行い，考察することが本実験の目的である．

本実験では，強化学習アルゴリズムとして Q-learning を使用する．Q-learning に対して線形アーキテクチャを使用して学習を行うと，近似した価値関数の大域最適解を獲得できることが保証されている [2].

5.2 実験の環境

実験の環境として強化学習のベンチマークテスト問題である Puddle World[3] を使用する．Puddle World は，エージェントの初期位置から目的地までの池を通らない最短経路を導出する問題である．Puddle World には，状態が連続値であるという特徴がある．本実験では，状態だけではなく行動も連続値に拡張した Puddle World を環境とする．ここで，本実験の環境を図 4 に示す．

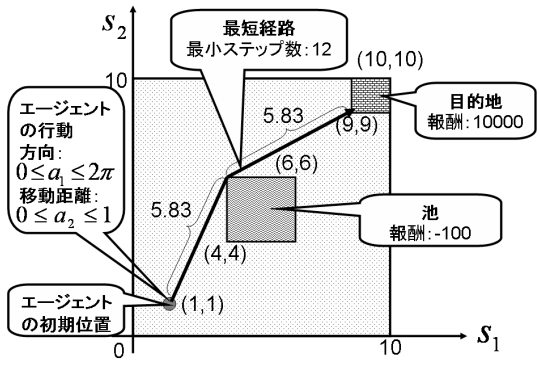


図 4: 本実験の環境

図 4 で示した環境では，エージェントの状態は現在位置 $(0 \le s_1 \le 10, 0 \le s_2 \le 10)$，行動は移動方向 $(0 \le a_1 \le 2\pi)$ と移動距離 $(0 \le a_2 \le 1)$ で定義する．報酬は目的地 (9〜10,9〜10) に到達した場合に 10,000，池 (4〜6,4〜6) に到達した場合に -100 とする．池を通らない最短経路は図 4 に示されるように，エージェントの初期位置 (1,1) から (4,6) を経由し目的地 (9,9) までとなる．また，図 4 には示していないが，初期位置から (6,4) を経由して目的地に行く経路も最短経路となる．そして，1 ステップの行動で最大 1 しか進むことができないので，最短経路長の 11.66 における最小ステップ数は 12 となる．以下の実験では，初期位置から目的地に到達するまでのステップ数 (到達ステップ数) を用いて，学習率 α と特徴数 n の変化による性能評価を行う．また，到達ステップ数が小さければ小さいほど，池を通らない経路では対応する Q 値 (期待報酬) が大きい．ここで，解の評価は到達ステップ数とし，学習により得られた解が池を通っていないことを確認する．

そして，初期位置から目的地に到達するまでを 1 エピソードとする．強化学習では，このエピソードをある設定値まで繰り返し，これを 1 試行と呼ぶ．強化学習は確率的探索であるため，本実験では試行を複数回行う．

5.3 実験の設定

実験では，三角分布と RBF の両方を使用し特徴数 n を 3,4,5,6 と変化させて実験を行い，特徴数における学習の性能評価を行う．強化学習アルゴリズムは，Q-learning を使用する．特徴の中心 c_i は状態空間と行動

空間を次元ごとに，特徴数分で等間隔になるように設定し，三角分布で補間範囲を示す d_i は各隣接する c_i 間の距離とし，RBF で補間範囲を示す σ_i は各隣接する c_i 間の距離の半分とした．すなわち，$c_i = \frac{D}{n-1} \times (i-1)$，$d_i = \frac{D}{n-1}$，$\sigma_i = \frac{D}{2(n-1)}$ となる．ただし，$i = 1, 2, \ldots, n$，D は各次元の範囲を示す．また，状態 s_1, s_2，行動 a_1, a_2 に対して，D はそれぞれ $10, 10, 2\pi, 1$ である．

そして，行動選択法は学習中一定の探索率を保つ ϵ-greedy 選択を使用する．ϵ の値は 10 腕テスト問題で，より早く最適行動を見つけることができる示されている $\epsilon = 0.1$ を使用する [2]．割引率 γ は一般的に $\gamma = 0.9$ 〜0.99 ぐらいの値を設定することが多い [4]．したがって，実験では割引率 $\gamma = 0.9$ とする．

池を通るか通らないかは，池の形状，池を通ることにより与えられる負の報酬，目的地に到達したことにより与えられる報酬，及び割引率 γ によって決定される．池を通らないような設定を，理論的に求めることは難しい．そこで本レポートでは，予備実験により，実験の設定によって得られる経路が池を通っていないことを確認した．したがって，到達ステップ数は 5.2 節により示される最小ステップ数 12 に近い方が良い解である．また，学習率 α は一般的に 0.1 ぐらいの値を設定することが多い [4] が，予備実験よりエピソード数 10,000 で最も最適解に近い解を得ることができた 0.001 を使用する．

5.4 実験の結果

5.3 節の設定を使用して，実験を行った結果を図 5，図 6 に示す．そして，学習後 (10,000 エピソード後) の到達ステップ数，計算時間，学習終了時 (10,000 エピソード目) の傾きを表 1 に示す．また，特徴数 3 の三角分布の場合は，収束が確認できなかったため，データの記述を省略する．

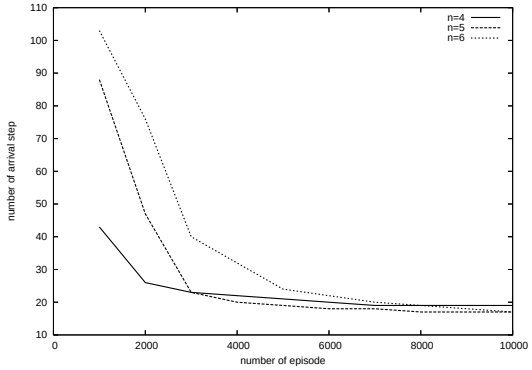


図 5: 三角分布を用いた場合の結果

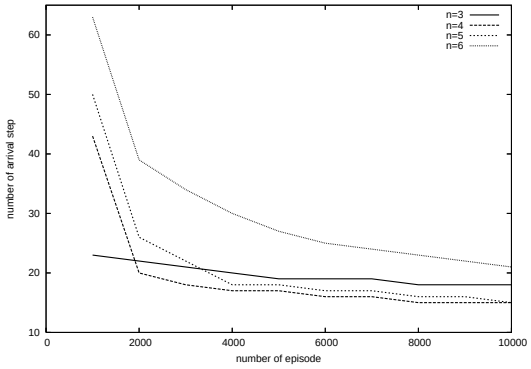


図 6: RBF を用いた場合の結果

表 1: 実験の結果

三角分布を用いた場合の結果

特徴数	学習後の到達ステップ数	計算時間 (秒)	学習終了時の傾き
3	-	-	-
4	18	9,113	$-1.89 \times 10^{-}3$
5	16	21,046	$-5.92 \times 10^{-}3$
6	16	43,499	$-9.62 \times 10^{-}3$

RBF を用いた場合の結果

特徴数	学習後の到達ステップ数	計算時間 (秒)	学習終了時の傾き
3	16	1,584	$-2.54 \times 10^{-}4$
4	14	6,866	$-9.61 \times 10^{-}4$
5	14	15,710	$-1.21 \times 10^{-}3$
6	16	38,776	$-1.67 \times 10^{-}3$

図 5，図 6 の横軸はエピソード数，縦軸は各エピソードでの 5 回試行の平均到達ステップ数を表している．また，表 1 の学習後 (10,000 エピソード後) の到達ステップ数とは，その時点での Q 値を使用して，各状態で最も Q 値が大きい行動を選択した (greedy 選択) 結果である．そして，学習終了時の傾きは，最小自乗法で近似した曲線の学習終了時の傾きより求めた．

6 考察

6.1 考察 1：計算時間

特徴数 n が増えることで，計算時間が増加することが示された．線形アーキテクチャにおける Q-learning の更新処理式 (4) に示されるように，一度の更新処理で更新する重み w_i の数が特徴数 n の次元乗で増える．本問題では，状態と行動が共に 2 次元であり，次元数は合計 4 となる．ここで，表 1 の計算時間に対して，3 次及び 4 次の曲線に最小自乗法を用いて近似した結果，計算時間は 4 次の曲線との相関が高いことが確認された．これにより，計算時間は特徴数 n の 4 乗で増加することがわかる．

6.2 考察 2：学習後の性能

図 5，図 6 に示されるように，三角分布では 2,800 エピソード前後まで，RBF では 1,800 エピソード前後までは，特徴数 n が少ない方が到達ステップ数が少ない結果が得られた．そして，学習が終盤に進むにつれて特徴数 n が多い方が到達ステップ数が少なくなる傾向が見られた．表 1 により，特徴数 $n = 6$ の RBF を除いて，特徴数 n が多い方が到達ステップ数が少ないことが示された．特徴数 $n = 6$ の RBF では到達ステップ数が最も多い．ここで，表 1 の学習終了時の傾きがマイナス方向に大きいことから，他の特徴数の設定より学習が進行していると考えられる．したがって，更にエピソード数を増やし学習を行うことで，他の特徴数の設定より到達ステップ数が少なくなることが予想される．しかし，エピソード数 10,000 では約 11 時間かかるため，今回は学習終了時の傾きで確認することとした．したがって，実験より十分なエピソード数が確保される場合は，特徴数 n が多ければ多いほど，到達ステップ数が少なくなることが言える．これは，特徴数 n が多くなると，重み w_i の数が増え，真の価値関数をより近似でき，より良い政策の獲得が期待できるからである．

6.3 考察 3：三角分布と RBF の近似性能

表 1 より同じ特徴数 n に対して，RBF を用いた方が三角分布より学習後の到達ステップ数が少ないか同等であることが示されている．これにより，RBF の方が三角分布より学習性能が良いと言える．ここで，三角分布と RBF を用いた近似価値関数の近似性能を比較することにより，考察を行う．政策 π のもとで状態 s_t において行動 a_t を選択したときの Q 値 $Q^\pi(s_t, a_t)$ は，

$$\begin{aligned} Q^\pi(s_t, a_t) &= E_\pi\{R_t|s_t, a_t\} \\ &= E_\pi\{\Sigma_{k=0}^{t_{end}}\gamma^k r_{t+k+1}|s_t, a_t\} \end{aligned} \quad (6)$$

で表される．ただし，R_t は状態 s_t で行動 a_t を実行した後，政策 π に従って行動することで得られる割引報酬 $\gamma^k r_{t+k+1}$ の合計である．また，$E_\pi\{\}$ は期待値，t_{end} は目的地に到達するまでのステップ数 (最終ステップ数) を表している．しかし，状態や行動が多次元及び連続値空間では真の価値関数を直接求めることが難しい．そこで，状態 s_t は離散で 1 次元，行動 a_t は各状態に対して 1 つだけの環境において考え，エージェントは状態 $s_0 \to s_1 \to \cdots \to s_{t_{end}}$ と順に遷移し，状態 $s_{t_{end}}$ で報酬が与えられるとすると，真の価値関数は式 (6) より

$$Q^\pi(s_t, a_t) = \gamma^{t_{end}-t-1} r_{t_{end}}|s_t, a_t \quad (\text{ただし，} t = 0, 1, \ldots, t_{end} - 1) \quad (7)$$

となる．ここで，最終ステップ数 $t_{end} = 12$，特徴数 $n = 5$，報酬 $r_{t_{end}} = 10,000$，割引率 $\gamma = 0.9$ として，三角分布と RBF を用いた式 (3) による近似価値関数を最小自乗法を用いて，式 (7) に示される真の価値関数にそれぞれ当てはめて相関係数 R を求めた結果，三角分布が $R = 0.997$，RBF が $R = 0.998$ となった．また，特徴数 n が 4,6 及び最終ステップ数 t_{end} が 12 以上の場合も，三角分布より RBF を用いた方が相関係数が高いことを確認した (例：$n = 5$，$t_{end} = 30$ の場合は，三角分布が $R = 0.989$，RBF が $R = 0.995$ である)．5.2 節に示されるように，最小ステップ数は 12 であり，学習過程において学習により得られる到達ステップ数は 12 以上となる．したがって，学習全過程で三角分布より RBF を用いた方が真の価値関数に対する近似性能が良いと言える．近似性能が高いほど学習により得られる行動が最適行動に近いと考えられる．以上の理由から，実験 2 に示されるように同じ特徴数の場合，RBF による到達ステップ数が三角分布による到達ステップ数より少なくなる結果が得られた．

7 むすび

本論文では，強化学習を連続値へ適用するために線形アーキテクチャの特徴ベクトル $\vec{x}$ に対応する三角分布や RBF を使用して関数近似を行い，特徴数 n における学習性能の変化を考察するために実験を行った．

実験結果より，十分なエピソード数を確保できる場合は，特徴数 n は多くすれば多くするほど到達ステップ数が少なくなることが確認された．しかし，特徴数 n が大きくなると n^d(d は次元数) で計算時間が増加する．以上より，学習により得られる解の良さと計算時間は互いにトレードオフとなる為，問題に合わせて特徴数 n を適切に設定する必要があると言える．また、本実験の環境では、三角分布より RBF を使用した方が価値関数の近似性能が高いことが確認された。

今後更に，他の環境においても同様に実験し，考察することが必要がある．また，強化学習アルゴリズムには，Q-learning 以外に最適政策を得られることが保証されていないが学習が高速である Profit Sharing[4] や，状態と行動を別々に評価し，一般的に連続値への適用に適していると言われる Actor-Critic[1] がある．こ

れらの学習アルゴリズムに対して連続値へ適用し，特徴数 n における学習性能の変化を実験し，考察することも今後の課題である．

参考文献

[1] 三上貞芳, 皆川雅章共訳,Richard S.Sutton,Andrew G.Barto 著書,”強化学習”, 森北出版 (2005)

[2] 山村雅幸, 宮崎和光, 小林重信,”エージェントの学習”, 人工知能学会誌,Vol.10,No.5,pp.683-689(1985)

[3] ”強化学習”, 東京工業大学　総合理工学研究科　知能システム科学専攻　小林重信研究室,http://www.fe.dis.titech.ac.jp/research/rl/index.html

[4] 高玉圭樹,”マルチエージェント学習”, コロナ社 (2003)

[5] ”連続な空間における強化学習”, 九州大学　工学府　海洋システム工学専攻　木村元研究室,http://sysplan.nams.kyusyu-u.ac.jp/gen/index.html