計測自動制御学会東北支部 第229回研究集会 (2006.6.9)
資料番号 229-11

連続状態空間における強化学習

Reinforcement Learning in Continuous State Spaces

○釜谷博行*，北山数行**，藤村敦子*，阿部健一***

○Hiroyuki Kamaya*, Kazuyuki Kitayama**, Atsuko Fujimura*, Kenichi Abe***

*八戸工業高等専門学校, **豊橋技術科学大学, ***日本大学

* Hachinohe National College of Technology,
** Toyohashi University of Technology,
*** Nihon University

キーワード： 連続状態空間(Continuous State Spaces), 強化学習 (Reinforcement Learning), RBFネットワーク(Radial Basis Function Network)

連絡先： 〒039-1192 八戸市田面木字上野平16–1 八戸工業高等専門学校 電気情報工学科
釜谷博行，Tel.: (0178)27-7283，Fax.: (0178)27-9379，E-mail: kamaya-e@hachinohe-ct.ac.jp

1. はじめに

数理心理学の分野で動物の学習行動を記述する数学モデルが種々提案され，そのモデルが学習機能をもつ工学システムの構築に応用されるようになった。強化学習[1)]もそのモデルのひとつである。すなわち，強化学習は報酬(あるいは罰)という特別な情報を手掛かりに，エージェントが環境との相互作用を通してあらかじめ定められた明確なゴールを達成するための行動決定戦略を自律的に獲得する学習システムと捉えることができる(Fig. 1)。

強化学習では，一般的に環境モデルが未知で，しかも正解を教えてくれるような教師は存在しない。また，不確実性のある環境，報酬に遅れが存在する環境にも適用可能であるという特徴をもつ。

強化学習を実問題へ適用する場合，連続状態を扱う必要がある。これには，連続状態を離散化しテーブル形式で表現する方法が考えられる。しかし，細分化しすぎると状態数が多くなり，学習が遅くなる。一方，荒く量子化すると部分観測環境となり，環境にマルコフ性を仮定しているQ-学習などは，その学習性能が悪化するなどの問題が生じる。そこで本研究では，関数近似を用いる方法としてRBF(Radial Basis Function)ネットワーク[2)]に着目し，シミュレーション実験を通してその有効性を確認する。

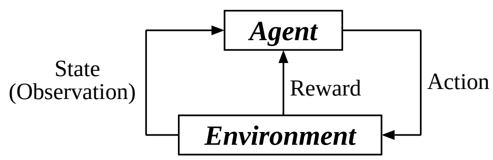


Fig. 1 強化学習モデル

2. 強化学習

強化学習では，行動する毎に得られる報酬の合計を最大化するための方策を学習する。各時間ス

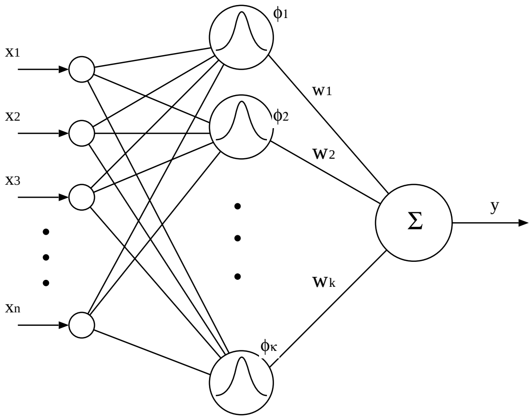


Fig. 2　RBFネットワーク

テップでエージェントは状態sを観測し，これに基づいて行動aを選択する。その結果として報酬rを受け取り，新しい状態s'を観測する。強化学習では，エージェントがある状態においてある行動を行うと，将来的に報酬がどれだけ期待できるかを価値関数$Q(s,a)$として表現する。

Q値の大きさに応じて，エージェントは状態sにおいて実行すべき行動aを決定する。環境と対峙したエージェントは試行錯誤を繰り返しながら，割引期待利得$E\{\sum_{t=0}^{\infty}\gamma^t r_t\}$の最大化を目的として，各時点$t$で得られる報酬$r_t$に基づいて$Q$値を更新していく。ここで，$\gamma(0 \leq \gamma \leq 1)$は割引率を表わす。

3.　RBFネットワーク

各状態および行動に対するQ値を推定するために，関数近似器としてFig. 2に示すRBF(Radial Basis Function)ネットワークを用いる。

入力パターン$\boldsymbol{x} \in \{x_1, x_2, \cdots, x_n\}$が与えられると，ネットワークの出力$y(\boldsymbol{x})$は次式で計算される。

$$y(\boldsymbol{x}) = \sum_k w_k \phi_k(\boldsymbol{x}) + b \tag{1}$$

ただし，w_k：k番目のユニットの重み，$\phi_k(\boldsymbol{x})$：k番目のユニットの出力

$$\phi_k(\boldsymbol{x}) = \exp(-\frac{\|\boldsymbol{x}-\boldsymbol{\mu}_k\|}{\rho\sigma_k^2}), \tag{2}$$

b：ベースライン，$\boldsymbol{\mu}_k$： k番目のユニットの中心位置，σ_k：k番目のユニットの広がり，ρ：広がり係数を表す。Fig.3に入力が2次元の時の単一ユニットの出力例（ガウス関数）を示す。

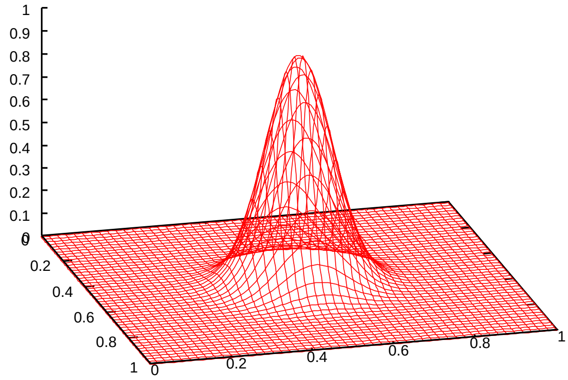


Fig. 3　単一ユニットの出力例（ガウス関数）

学習初期のユニット数$K=0$である。初期ユニットの生成は次のように行う。

$$\boldsymbol{\mu}_K \leftarrow \boldsymbol{x},\ \sigma_K \leftarrow \kappa d_{max}, \tag{3}$$

$$w_K \leftarrow \eta\delta,\ b \leftarrow \eta\delta,\ K \leftarrow K+1 \tag{4}$$

ここで，κは重なり度合いを調整するパラメータ，d_{max}は距離に関するしきい値パラメータの最大値である。

新たなユニット追加の条件は，次式で与える。

$$|\delta| > \epsilon_{th} \text{ and } l > d_{th} \tag{5}$$

ただし，δはTD(temporal-difference)誤差，ϵ_{th}は推定誤差に関するしきい値，lは入力パターン$\boldsymbol{x}$と最近傍ユニット間の距離で次式を用いて求める。

$$l \leftarrow \min_k \|\boldsymbol{x}-\boldsymbol{\mu}_k\| \tag{6}$$

また，d_{th}は距離に関するしきい値を表す。ユニット追加時の各パラメータ値は次のように設定する。

$$\boldsymbol{\mu}_K \leftarrow \boldsymbol{x},\ \sigma_K \leftarrow \kappa l, \tag{7}$$

$$w_K \leftarrow \eta\delta,\ b \leftarrow b+\eta\delta,\ K \leftarrow K+1 \tag{8}$$

一方，ネットワーク内のパラメータの調整は次式で与える。

$$w_k \leftarrow w_k + \eta\delta\phi_k(\boldsymbol{x}) \tag{9}$$

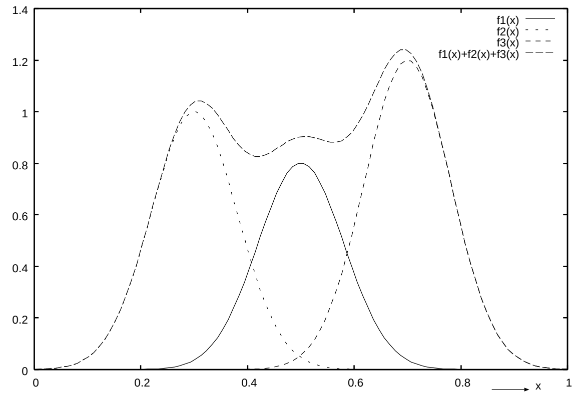


Fig. 4　RBFネットワークによる関数近似の例

$$\mu_{ki} \leftarrow \mu_{ki} + \eta_\mu \delta \phi_k(\boldsymbol{x}) w_k \frac{x_i - \mu_{ki}}{\sigma_k} \tag{10}$$

$$b \leftarrow b + \eta\delta \tag{11}$$

ただし，η：学習率，η_μ：ユニット中心の学習率を表す。

距離に関するしきい値$d_{th} = [d_{min}, d_{max}]$を学習ステップとともに減少させるためにつぎの更新式を用いる。

$$d \leftarrow d \exp(-\frac{1}{\tau}),\ \text{if } d > d_{min} \tag{12}$$

ただし，τは時定数を表す。

ネットワーク内に生成できる最大ユニット数をK_{max}とする。RBFネットワークによる関数近似の例をFig.4に示す。

今回，RBFネットワークをQ値の推定に利用するにあたり，入力パターンは$\boldsymbol{x} = [\boldsymbol{s}, u]^T$ とし，各要素は$[0,1]$に正規化するものとした。ただし，$\boldsymbol{s}$：状態ベクトル，$u \in \{a_1, a_2, \cdots, a_m\}$：行動を表す。

4.　学習アルゴリズム

ここでは，政策オフ型のアルゴリズムであるQ-学習に基づいて学習システムを構築する。以下にRBFネットワークを用いた強化学習アルゴリズムの概略を示す。

1) RBFネットワークの初期化($K = 0, d_{th} = d_{max}, b = 0$)
2) **Repeat** (for each trial)
3) 状態$\boldsymbol{s}$を観測する
4) **Repeat** (for each step of trial)
5) 　RBFネットワークを用いて$Q(\boldsymbol{s}, u)$を推定する
6) 　ϵ-greedy法を用いて行動aを選択する($a \leftarrow Q(\boldsymbol{s}, u)$)
7) 　行動aを実行，報酬rを取得し，新たな状態$\boldsymbol{s}'$を観測する
8) 　新しい状態$\boldsymbol{s}'$において，RBFネットワークを用いて$Q(\boldsymbol{s}', u)$を推定する
9) 　TD誤差δを用いて，RBFネットワークを更新する
10) 　$\boldsymbol{s} \leftarrow \boldsymbol{s}'$
11) 　**until** s is the terminal

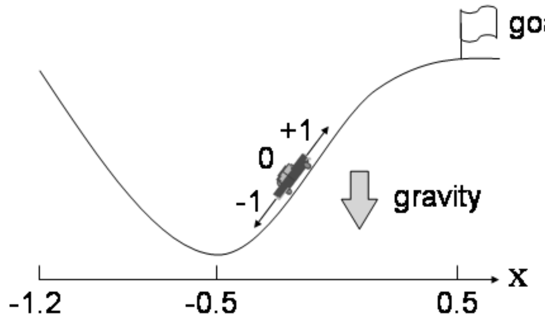


Fig. 5　mountain-car問題

学習初期において，RBFネットワークにユニットが存在しない場合の出力は0である。エージェントの経験組$< s, a, r, s' >$に対するTD誤差の計算式を以下に示す。

$$\delta = r + \gamma \max_b Q(s', b) - Q(s, a) \tag{13}$$

ここで，γ $(0 \le \gamma \le 1)$は割引率を表す。

本実験では，行動選択にϵ-greedy法を用いる。ϵ-greedy法は，確率 ϵでランダムな行動を，確率 $1-\epsilon$で最大のQ値をもつ greedy　な行動を選択するものである。なお，学習終了時に決定的な方策を得るために，ϵを初期値ϵ_0から0まで徐々に減少させる。

5.　問題設定

本稿では，学習問題としてmountain-carタスク[1)]を取り上げる。これはFig. 5に示すように，急な坂道をパワー不足の車が登ってゴールに辿り着

く問題である。ここでは，車のエンジン出力よりも重力の方が大きく，急な坂道ではフルスロットルでも加速できない。急な坂道を登りきるための唯一の解決策は，一度ゴールとは逆方向に加速し，その後ゴールへ向かって加速し続けることである。

報酬として1ステップ毎に-1が与えられ，車が坂道の頂上のゴール位置を通過するか，最大ステップ数に達したときに1回の試行が終了する。エージェントはフルスロットル前進$(+1)$，ゼロスロットル(0)，フルスロットル後退(-1)の3つの行動aから1つを選択する。エージェントが観測できる車の状態は，位置xと速度vの2種類で，次式で更新される。

$$v \leftarrow v + 0.001a - 0.0025\cos(3x), \tag{14}$$

$$x \leftarrow x + v \tag{15}$$

ただし，$-1.2 \leqq x \leqq 0.5$，$-0.07 \leqq v \leqq 0.07$である。なお，この問題の最適ステップ数は103である。

6. 実験結果

RBFネットワークのパラメータは，学習率$\eta = 0.1$，ユニット中心の学習率$\eta_\mu = 10^{-4}$，距離に関するしきい値$d_{th} = [d_{min}, d_{max}] = [0.07, 0.7]$，時定数$\tau = 50$，推定誤差しきい値$\epsilon_{th} = 0.2$，$\rho = 2.67$，$\kappa = 0.87$とした。また，強化学習のパラメータは$\epsilon_0 = 0.1$，$\gamma = 1$とし，最大ステップ数$t_{max} = 3,000$，試行回数$T_{max} = 1,000$とした。

6.1 ユニット中心の更新

ユニット中心を固定した場合($\eta_\mu = 0.0$)と可変にした場合($\eta_\mu = 10^{-4}$) の学習性能について比較する。実験は，乱数のシード値を変えた独立な20シミュレーションにより行った。ユニット中心を固定した場合の結果を，Fig. 6とTable 1に，ユニット中心を可変にした場合の結果を，Fig. 7とTable 2に示す。

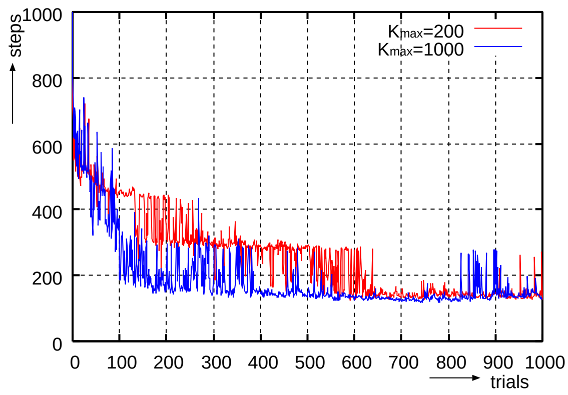


Fig. 6　ユニット中心固定時の平均学習性能

Table 1　ユニット中心固定時の学習後の性能

最大ユニット数	平均ステップ数	標準偏差	成功率(%)
100	423.1	—	80
200	126.3	20.0	100
300	136.2	36.1	100
1000	130.2	40.8	100

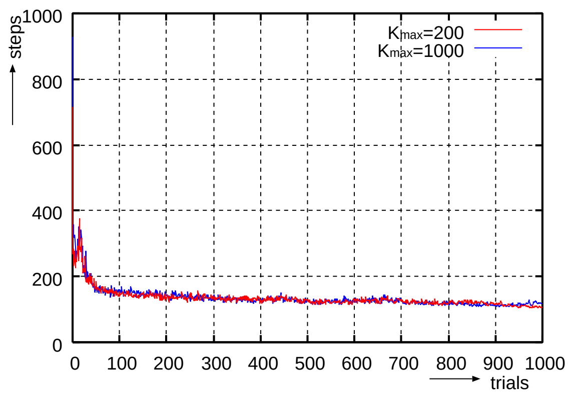


Fig. 7　ユニット中心可変時の平均学習性能

Table 2　ユニット中心可変時の学習後の性能

最大ユニット数	平均ステップ数	標準偏差	成功率(%)
50	156.0	58.1	100
100	114.9	18.9	100
200	105.4	3.2	100
300	106.5	3.3	100
1000	118.2	20.8	100

学習性能曲線において，横軸は試行回数を，縦軸はゴールまでのステップ数を表わす。グラフを見ると，試行回数とともにステップ数が減少し，学習性能が向上していることがわかる。

ユニット中心固定時には，$K_{max} = 1000$の収束特性が良い。しかし，学習後の性能を見ると，$K_{max} = 200$の結果が良い。これに対して，ユニット中心可変時には，収束特性が非常に良く，$K_{max} = 50$の場合でも学習することができた。特に，$K_{max} = 200$〜300付近の結果が，ゴールまでの平均ステップ数が105〜107ぐらいで良好であった。

6.2 学習後のRBFネットワークの評価

各行動毎にユニットの中心位置を示したのがFig. 8である。状態遷移の軌道に沿ってユニットが分布していることがわかる。

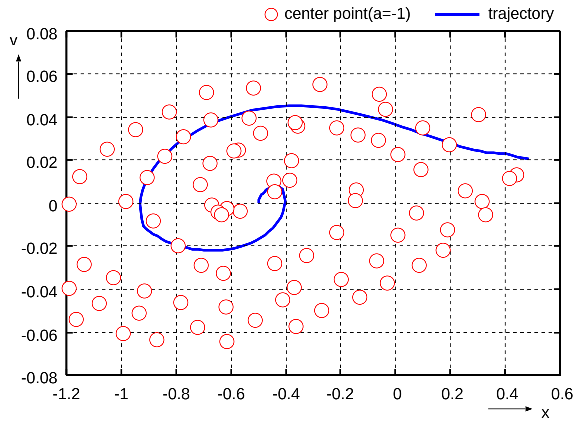
価値関数を3次元表示したのが，Fig. 9である。状態価値の低い初期状態から状態価値の高いゴールに向かって，状態遷移している様子がわかる。

各ステップにおける位置x，速度v，行動a，状態価値を表したのがFig. 10〜 13である。特に，Fig. 13では，車(エージェント)が状態価値の山を登っていく様子が読みとれる。

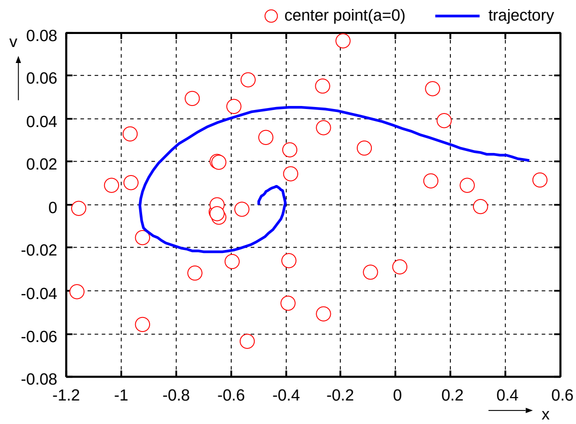
6.3 入力次元の追加

RBFネットワーク($K_{max} = 1000$)において，状態とは無関係な入力次元を1次元〜3次元まで追加して実験を行った。実験結果をFig. 14，Table 3に示す。これらは，10回のシミュレーションの平均値である。

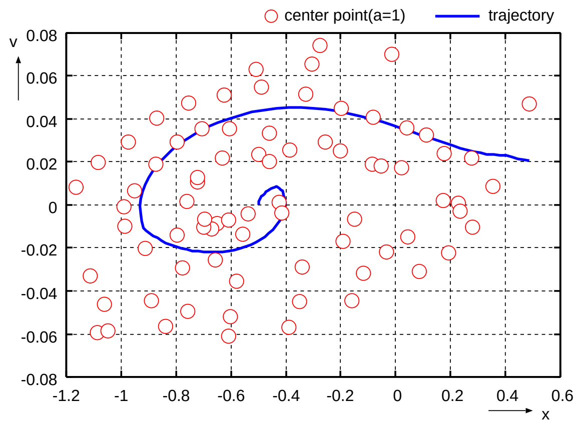
ノイズ次元が2までは，うまく学習していることがわかる。3次元になると，学習が不安定になることが確認された。これは，ユニット数不足のため，価値関数をうまく近似できていないことによるものと考えられる。



(a) action = -1 (ユニット数81)



(b) action = 0 (ユニット数39)



(c) action = 1 (ユニット数80)

Fig. 8　ユニットの分布

7. おわりに

本稿では，RBFネットワークを用いた関数近似による強化学習アルゴリズムの有効性を確認した。今後は，高次元状態空間を扱う問題への適用，他の関数近似手法[3)4)]との比較などが必要である。

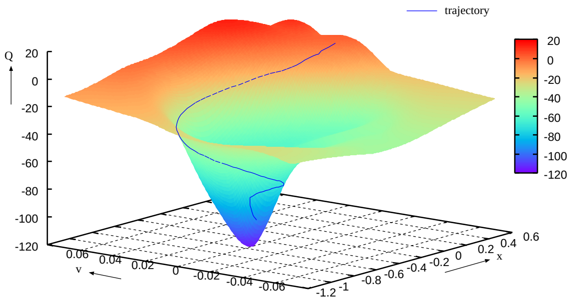


Fig. 9　価値関数と状態遷移

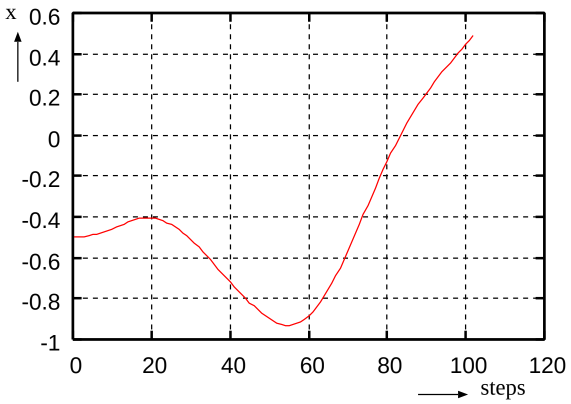


Fig. 10　各ステップにおける位置xの変化

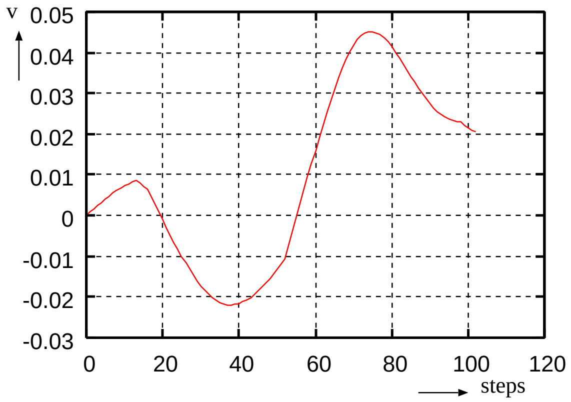


Fig. 11　各ステップにおける速度vの変化

Table 3　ノイズ次元追加時の学習後の性能

追加次元	平均ステップ数	標準偏差(%)
1次元	121.0	16.9
2次元	139.6	15.4
3次元	223.3	197.5

参考文献

1) Sutton, R. S. and Barto, A. G.: *Reinforcement Learning : An Introduction.*, MIT Press, 1998.

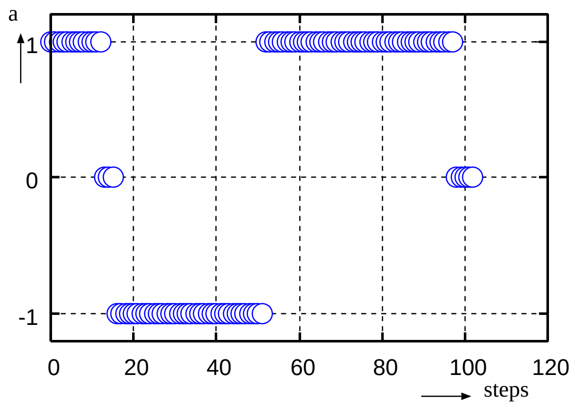


Fig. 12　各ステップで選択した行動a

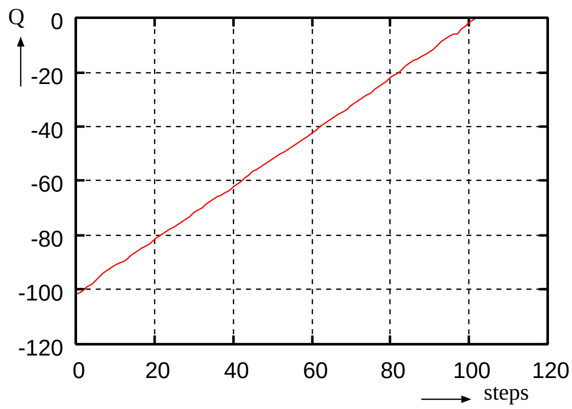


Fig. 13　各ステップにおける状態価値

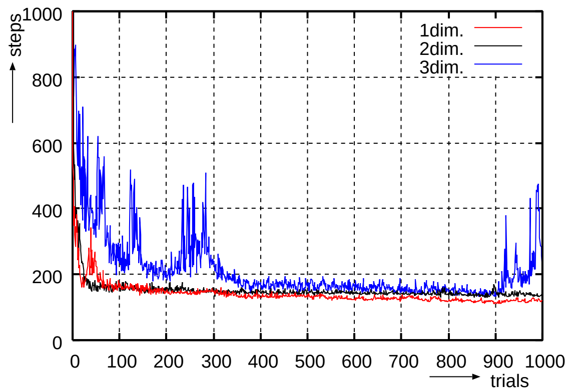


Fig. 14　ノイズ次元追加時の平均学習性能

2) Platt, J.: "A Resource-Allocating Network for Function Interpolation," Neural Computation, vol.3, no.2, pp. 213–225, 1991.

3) C. G. Atkeson, A. W. Moore, and S. Schall, "Locally Weighted Learning," Artificial Intelligence Review, vol.11, 1997.

4) 釜谷博行, 山火和也, 阿部健一: "関数近似を用いた強化学習," 平成17年度電気関係学会東北支部連合大会講演論文集, p. 325, 2005