修士論文

数理モデルの構築と効率的なアルゴリズムの開発

指導教員　　情報太郎　教授
　　工学次郎　助教

数理　三郎

京都大学大学院情報学研究科

数理工学専攻

令和３年２月

摘要

本研究では，修士論文を執筆する学生の数理モデルを構築し，このモデルを用いて効率的に修士論文を作成するためのアルゴリズムを開発した．また，このアルゴリズムを用いて，実際に本論文を作成した．その結果，従来の自分で執筆する方法に比べて，65536 倍効率的に修士論文を作成することが可能であることが確認された．

目次

1　序論

本研究では，修士論文を執筆する学生の数理モデルを構築し，このモデルを用いて効率的に修士論文を作成するためのアルゴリズムを開発することを目的とする．先行研究 [**?**] を改良し，より学生の実態に合致したモデルを構築する．また，このアルゴリズムを用いて，実際に本論文を作成する実験を行う．

2　はじめに

(TODO: 研究のモチベーションの整理)

3　方策勾配を用いた強化学習

3.1　強化学習の基礎知識

マルコフ決定過程 M を $M = \{S, A, T, d_0, r, \gamma\}$ として与える。ここで S, A はそれぞれ状態, 行動集合, $T(s^{'}|s, a)$ は状態遷移確率を表す. また $d_0, r(s, a), \gamma \in [0, 1]$ はそれぞれ初期状態分布, 報酬, 割引率を示す.

さて, 強化学習の目的は

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, J(\pi) \tag{1}$$

を求めることである. ここで,

$$V^\pi(s) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)|_{a_t=\pi(s_t)}, s_0 = s \tag{2}$$

$$J(\pi) = \mathbb{E}_{s_0 \sim d_0}[V^\pi(s_0)] \tag{3}$$

であり, $J(\pi), V^\pi(s)$ をそれぞれ評価関数, (状態) 価値関数とよぶ.

強化学習を解析するツールとして有用な関数として Q 関数がある.

$$\begin{aligned} Q^\pi(s, a) &= r(s, a) + \gamma \sum_{t=1}^{\infty} \gamma^t r(s_t, a_t)|_{a_t=\pi(s_t)} \\ &= r(s, a) + \gamma V^\pi(s') \end{aligned} \tag{4}$$

式 (4) より, Q 関数は開始時刻において自由に行動 a を選択して, 次ステップから方策 π に従った時の価値を表す. したがって, Q 関数は行動価値関数という別名がある.

3.2 方策反復法

式 (1) を達成するためのアルゴリズムとして方策反復法というものがある. これは以下の 2 つのステップを繰り返すというものである.

1. 方策評価: 行動価値関数 $Q^\pi(s,a)$ を求め (または, 近似す) る.
2. 方策改善: 求めた $Q^\pi(s,a)$ に従って, $\pi(s) = \underset{a}{\operatorname{argmax}} Q^\pi(s,a)$ と方策を更新する.

以上の 2 ステップを繰り返すことで最適方策 π^* が得られることが知られている.(方策改善定理)

3.3 状態空間, 行動空間の特性に合わせたアルゴリズム

状態空間も行動空間も離散値をとる場合, $Q^\pi(s,a)$ をテーブルに保存しておくことで, 前節で登場した $\pi(s) = \underset{a}{\operatorname{argmax}} Q^\pi(s,a)$ を容易に求めることができる.

では, 状態空間が連続値の場合はどうか. 状態 s が無限種類の値をとることになるため, テーブルに保存することができない. そこで Minh ら [2] は, $Q^\pi(s,a)$ をニューラルネットワークを用いてパラメトライズして近似するアプローチをとった. ここでも, 行動空間は離散であるため $\underset{a}{\operatorname{argmax}} Q^\pi(s,a)$ を求めることは可能である.

最後に, 状態空間も行動空間も連続値である場合は $\underset{a}{\operatorname{argmax}} Q^\pi(s,a)$ を求めるのに膨大なコストがかかるという問題点がある. したがって, これまでは方策 π は Q 関数によって定めていたが, 両空間が連続値の場合にはこのアプローチはとれない. そこで, 方策関数も独立に π_θ のようにパラメトライズした関数として用意し, パラメータ θ を勾配法などで更新する手法が取られることが多い.

3.4 方策勾配による方策関数のパラメータの更新

Silver ら [3] は, 方策が $\pi(s)$ のように決定論的に定めるものとして与えた場合に, 評価関数 $J(\pi_\theta)$ に対する勾配を計算する手法を発見した. この勾配は決定論的方策勾配 (DPG:Deterministic Policy Gradient) とよばれ, 以下のように計算することができる.

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{s\sim\rho^{\pi_\theta}}[\nabla_\theta \pi_\theta(s) \nabla_a Q^{\pi_\theta}(s,a)|_{a=\pi_\theta(s)}] \tag{5}$$

ただし,

$$\rho^{\pi_\theta}(s) = \int_S \sum_{t=0}^{\infty} \gamma^t d_0(s_0) \Pr(s_0 \to s, t, \pi_\theta) \mathrm{d}s_0 \tag{6}$$

を割引分布という.

この方策勾配を深層強化学習の枠組みに取り入れたアルゴリズムが DDPG(Deep DPG)[4] である. これは Actor-Critic 構造を採用しており, Q^{π_θ} を近似する critic ネットワーク $Q(s,a|\omega)$ と, 方策 π を表現する actor ネットワーク $\pi(s|\theta)=\pi_\theta$ をそれぞれ学習する手法である. 以下に, この近似勾配を用いた actor と critic の更新アルゴリズムを記す.

DDPG はミニバッチ学習を用いる. 過去の経験データ (s_t,a_t,r_t,s_{t+1}) を保存しておき, その中から N 個のデータを取り出し (ミニバッチ, E と記載する), そのデータ集合に対する最適化を行う. まず, critic の更新から記す. critic の目的は Q^π を近似することである. Q 関数は式 (4) のように分解することが可能で有るため, $Q(s,a|\omega)$ もこれを満たすように更新すればよい. そのために TD(Temporal Difference) 誤差

$$\mathrm{TD}=Q(s,a|\omega)-\{r(s,a)+\gamma Q(s,\pi(s)|\omega)\} \tag{7}$$

が最小となる方向に ω を更新する. 全ての (s,a) についてこれを一度に行うことはできないので, 作成したミニバッチ E に対する平均二乗誤差

$$Loss=\frac{1}{N}\sum_{s\in E}\mathrm{TD}^2 \tag{8}$$

を Loss 関数として, この Loss 関数を減らすようにアルゴリズムを働かせる.

さて, 上記の critic の更新方法は教師あり学習そのものである. 従ってミニバッチに含まれるデータは i.i.d. であることが要求される. もしミニバッチ E がエージェントが経験した直近の N ステップのデータを用いると, これらは独立ではなくなってしまう. 従って, エージェントは環境とのインタラクションによって得られた経験データを experience replay に保存しておき, そこから無作為に N このデータを選びとる手法でデータの分散を上げている.

次に actor の更新を記す. actor は方策関数 $\pi(s)$ を表現するものであり, パラメータの更新には方策勾配を用いる. ただし, DDPG では式 (5) のように正しい Q 関数を用いることができないので,

$$\mathbb{E}_{s\sim\rho^\pi}[\nabla_\theta\pi_\theta(s)\nabla_a Q(s,a|\omega)|_{a=\pi_\theta(s)}]\simeq\nabla_\theta J(\pi_\theta) \tag{9}$$

のように critic ネットワークを用いて近似した方策勾配を用いる. さらに, 期待値に関して,

$$\mathbb{E}_{s\sim\rho^\pi}[\nabla_\theta\pi_\theta(s)\nabla_a Q(s,a|\omega)|_{a=\pi_\theta(s)}]\simeq\frac{1}{N}\sum_{s\in E}[\nabla_\theta\pi_\theta(s)\nabla_a Q(s,a|\omega)|_{a=\pi_\theta(s)}] \tag{10}$$

という近似も行っている. 従って, この近似勾配 $g=\frac{1}{N}\sum_{s\in E}[\nabla_\theta\pi_\theta(s)\nabla_a Q(s,a|\omega)|_{a=\pi_\theta(s)}]$ は critic の近似精度とミニバッチの分布によって大きく性能を落としてしまう可能性があり, 大きな問題点であると言える.

4　最適セルフトリガー制御問題に対する強化学習

4.1　セルフトリガー制御

図 1 のような制御系を考える.

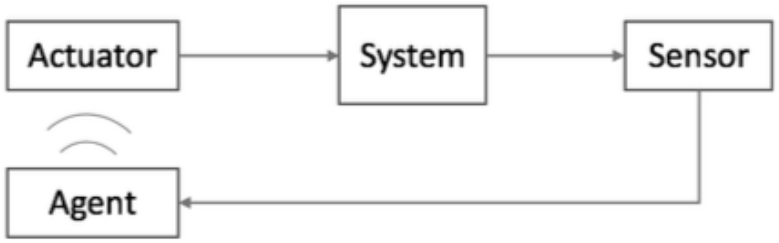

図 1　制御系

これに対するフィードバック制御を考える. 状態変数 s を観測してアクチュエータに入力信号を送信することを「インタラクション」と呼ぶと, セルフトリガー制御では, 連続的なインタラクションは行わずに, 次のインタラクションを何秒後に行うかをエージェントが決定する. それを数式上で表すため, エージェントの制御則 $\pi(s)$ は 2 つの要素からなるベクトル値関数であるとし, 1 つ目の要素はアクチュエータに送信する入力 a, 2 つ目の要素は次にインタラクションを行うまでの時間間隔 τ (s : 秒) を表すものとする. また, 次のインタラクションを行う時刻までは 1 つ前のインタラクションで送信した入力 a を加え続けるものとする (ZOH 制御).

4.2　最適セルフトリガー制御

最適なセルフトリガー制御則 π^* を, 以下のように定義する.

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, J(\pi) \tag{11}$$

$$J(\pi) = \mathbb{E}_{s_0 \in d_0}[V^\pi(s_0)] \tag{12}$$

$$V^\pi(s_0) = \sum_{i=0}^{\infty} \gamma^i r_i^\pi \tag{13}$$

$$r_i^\pi = -\int_{T_i}^{T_{i+1}} s(t)^\top Q s(t) \mathrm{d}t + \tau_i a_i^\top R a_i + \lambda \tau_i,\ T_i = \sum_{l=0}^{i} \tau_l \tag{14}$$

ここで, i はインタラクションの回数を示し, a_i, τ_i はそれぞれ i 回目のインタラクションでの方策 π の出力であるとする.

さて, 一般的に強化学習では, 1 ステップ 1 ステップの行動の良し悪しを評価して方策を更新していく. インタラクションとインタラクションの間の区間を「インターバル」と呼ぶと, 式 (13) より, この問題は各インターバルを 1 ステップとした強化学習問題であると考えることができる.

5 数値実験

上記の問題は, 一般的な強化学習のフレームワークにそのまま落とし込むことができるため, 数値実験を行うことが可能である. 数値実験の結果を通して, 最適セルフトリガー制御の強化学習にどのような課題点があるのかを考察していく.

実験環境は Open-AI Gym の pendulum で, 初期状態が $\theta \sim N(0, \pi), \dot{\theta} \sim U(-\pi, \pi)$ となるような環境で行った. また replay buffer の分散を上げるため, 10 秒間の制御を 1 エピソードと定義し, エピソードを重ねることで経験データを増やしていく. また, pendulum は以下のようなダイナミクスに従っており入力アフィン系の非線型システムである.

$$\frac{\mathrm{d}}{\mathrm{dt}} \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \dot{\theta} \\ \frac{3g}{2l} \sin\theta + \frac{3}{ml^2} a \end{pmatrix} \tag{15}$$

5.1 方策関数の表現モデル

セルフトリガー制御では, エージェントは各ステップにおいて入力信号 u と, それを何秒間加えるかを決める τ を決定する必要がある. 制御則を状態フィードバックとして与えるならば,

$$\pi(s) = \begin{bmatrix} u(s) & \tau(s) \end{bmatrix}^\top \tag{16}$$

のようになる.

方策 $\pi_\theta(s)$ を表現するモデルとして, 図 2 のようなニューラルネットワークを用いる.

5.2 初期方策

学習の初期方策として

$$\begin{cases} u(s) = lqr(s) \\ \tau(s) = 0.2 \end{cases} \tag{17}$$

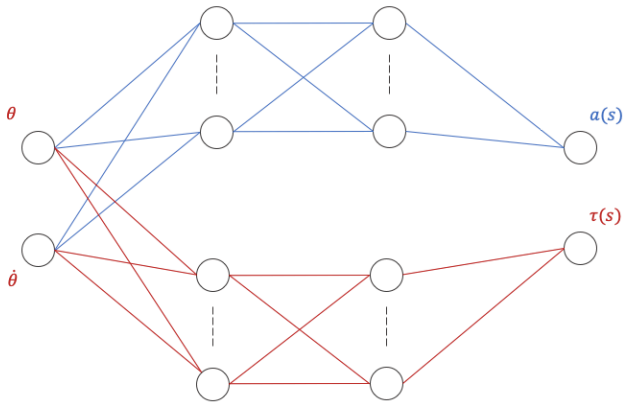

図 2　方策 $\pi_\theta(s)$ の表現モデル

とする方策 π_{init} を用いる. ただし, $lqr(s)$ はシステム (15) を, 離散化幅 $\delta_t = 0.05$ として離散化したシステムを原点付近で安定化する制御則として与える.

5.3　学習によって得られた方策

前節まで述べてきた設定により, DDPG を用いて強化学習を行った. その結果得られた方策 π_{RL} の制御性能を図 3 に示す.

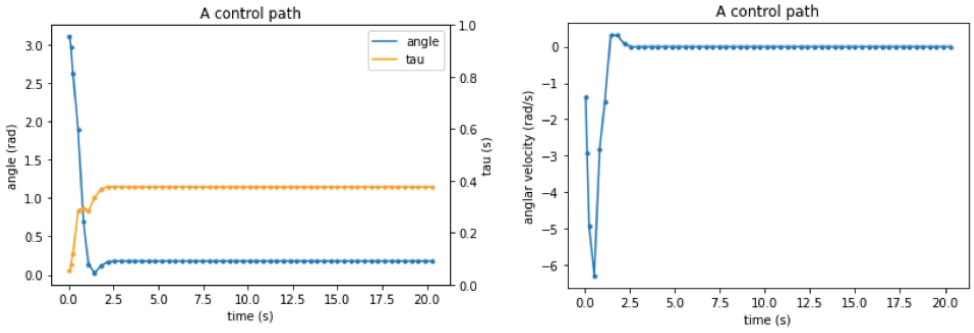

図 3　学習方策 π_{RL} による制御例

図 3 の左図には, 制御時間に対する角度と通信間隔 τ の変化をプロットしている. また, 通信が行われた時刻に点を打っている. この図から, 角度の絶対値が大きい時には細かく通信を行い, 角度が 0 に近い時には通信間隔を大きくするような制御を行っていることがわかる.

さて, この方策 π_{RL} は, 初期方策に比べて評価関数 $J(\pi)$ を大きくできているのか確認す

る. $J(\pi) = \mathbb{E}_{s_0}[V^{\pi}(s_0)]$ であったので, 初期状態 s_0 を 500 個発生させて $V^{\pi}(s_0)$ を計算した時の平均をとる作業を 1000 回行った. その時の結果は

$$J(\pi_{\text{init}}) = 4.23 \pm 0.083,\ J(\pi_{\text{RL}}) = 37.12 \pm 0.071 \tag{18}$$

となり $J(\pi_{\text{init}}) < J(\pi_{\text{RL}})$ であることから, 方策改善ができていると結論づけられる.

6 課題点の抽出

6.1 方策の最適性

前節で述べた方策 π_{RL} は最適解なのだろうか. もし最適解であれば方策勾配が $\mathbf{0}$ となる必要があるため, 式 (5) より,

$$\nabla_a Q^{\pi_{\text{RL}}}(s,a)|_{a=\pi_{\text{RL}}(s)} = \mathbf{0} \tag{19}$$

となるはずである.

さて, 強化学習の最中は方策が各ステップで変化し続けるため, critic の近似対象となる $Q^{\pi}(s,a)$ も絶えず変化していく. そのため, 正しく $Q^{\pi_{\text{RL}}}(s,a)$ を近似できている保証はない. そこで, 方策を固定してインタラクションすることによって発生させたデータのみを用いて, critic を教師あり学習して $Q^{\pi_{\text{RL}}}(s,a)$ を学習させた. この関数を $Q^{\pi_{\text{RL}}}_{\text{ap}}(s,a)$ とする.

π_{RL} が最適方策であれば, $\nabla_a Q^{\pi_{\text{RL}}}_{\text{ap}}(s,a) = \mathbf{0}$ となるはずなので, その計算を行ってみると

$$\nabla_a Q^{\pi_{\text{RL}}}_{\text{ap}}(s,a) = \big[0.03, 0.07\big]$$

となり, 最適解ではないことがわかった.

6.2 Q 関数の近似精度

Q 関数は方策勾配

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{s\sim\rho^{\pi_\theta}}[\nabla_\theta \pi_\theta(s) \nabla_a Q^{\pi_\theta}(s,a)|_{a=\pi_\theta(s)}]$$

を計算するために用いられるのであった. したがって, Q 関数の a に関する勾配の $a = \pi(s)$ における値が, ρ^{π_θ} のサポートとなる状態 s に対して近似できていれば良い.

強化学習によって得られた critic ネットワークの Q 関数の近似が, 上記の性質を満たしているか確認する為, 図 4 に, $Q(s,a|\omega)$ と $Q^{\pi}_{\text{ap}}(s,a)$ のそれぞれに対して $s = [0,0]$ における関数値をヒートマップとして表し, $a = \pi(s)$ における勾配を黄矢印にて示す.
図 4 の左図は $Q(s,a|\omega)$, 右図は $Q^{\pi_{\text{RL}}}_{\text{ap}}(s,a)$ を示している. 左右の黄矢印を比較すると, critic は Q 関数の a 勾配を近似できていないことが見て取れる.

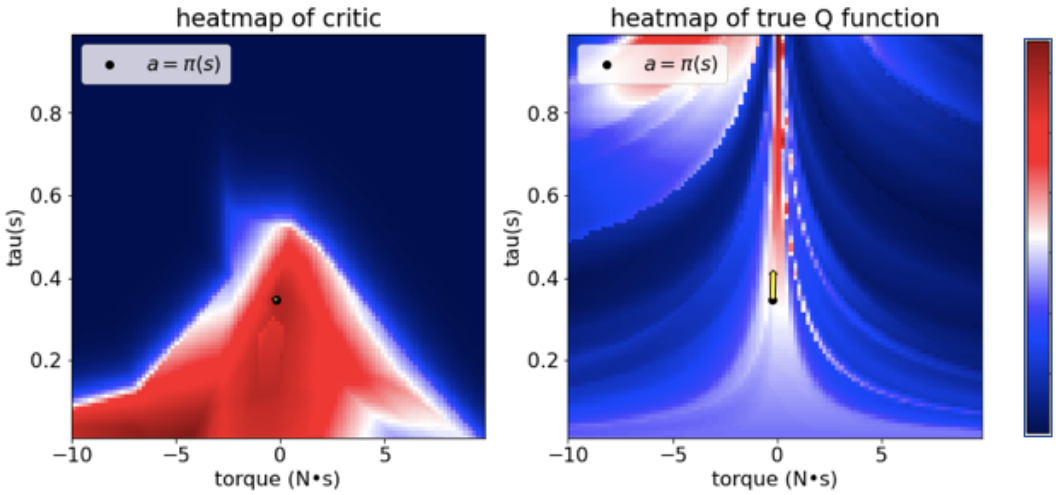

図 4　$s = [0, 0]$ に対する Q 関数の近似精度

6.3　Q 関数の近似精度が低い理由

前節で確認した, Q 関数の近似精度の低さの原因はどこにあるのだろうか. まず, critic は損失関数

$$L = \frac{1}{N} \sum_{(s,a)\in E} \{Q(s, a|\omega) - \{r(s, a) + \gamma Q(s', \pi(s')|\omega)\}^2$$

を小さくするように, パラメータ更新をするのであった. この最適化はデータ集合 E に含まれる (s, a) にのみ働くため, その分布に存在しない領域の (s, a) に対する近似精度は当然低くなる.

では, 実際に critic の教師あり学習に用いられたデータの分布を確認してみる. (s, a) は 4 次元のデータであるから, 可視化のために図 5 では $s = [0, 0]$ の付近で加えられた入力 a の分布を示している.

図 5 から, 非常に狭小な領域のデータを用いて関数近似がなされていたことがわかる. これは, データ収集の際に, 状態 $s = [0, 0]$ で加えた入力が偏っていたことが原因である. 方策勾配の計算には

$$\nabla_a Q(s, a|\omega)|_{a=\pi_\theta(s)}$$

の近似精度が肝要なので, 図 5 において橙色の点で表した $a = \pi_\theta(s)$ 周りのデータが不足していることが問題となる.

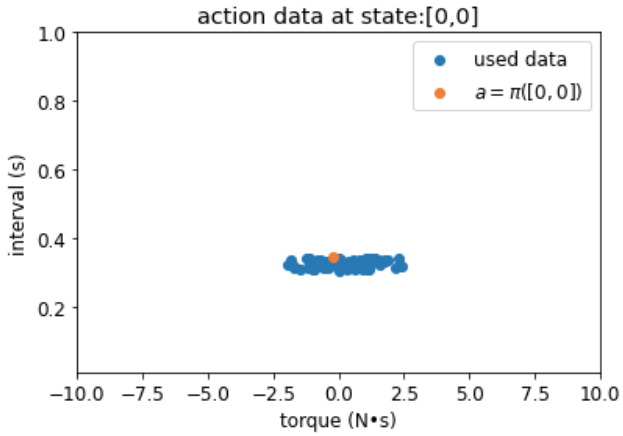

図 5　critic の学習に用いられた $s = [0, 0]$ に対する行動 a の分布

7　探索手法の工夫

前節で過去の経験データのにおける $a = \pi_\theta(s)$ 周りのデータが欠損に不足があることを確認した．経験データの分散を上げる為に，各状態で多様な入力を試してみることを探索といい，まずはその探索について考えていく．

7.1　探索

強化学習における探索手法として「ε-greedy アルゴリズム」などがよく知られているが，本稿で考えているような連続行動空間の強化学習では

$$a = \pi_\theta(s) + e \tag{20}$$

のように，方策関数の出力にノイズ e を付加して決定された行動でデータ収集を行うのが一般的である．(ノイズを付加した方策を，学習中の方策と区別するために行動方策とよぶ.) 直感的に，このノイズが大きければ大きいほど探索の幅が広がるため，$a = \pi_\theta(s)$ 周りのデータの分布が広がることで，Q 関数の a に関する勾配の近似精度が向上することが予想できる．では強化学習アルゴリズム全体に対して，探索ノイズを大きくすることはどのような影響を与えるのかについて次節で考察していく．

7.2 探索ノイズの大きさ

探索ノイズを大きくすれば，Q 関数の a に関する勾配の近似精度を向上させられることを予想した．では, 探索ノイズを大きくすればそれで良いのか．答えは NO である．大きな探索ノイズを加えれば, 経験データ E の分布は割引分布 $\rho^{\pi}(s)$ から大きく乖離することになる．これはどのような問題を引き起こすのか．

actor の更新に用いる方策勾配の計算には, Q^{π} の critic による近似の他にも, 以下のように状態 s に対する期待値のサンプル平均による近似も用いられる．

$$\mathbb{E}_{s\sim\rho^{\pi_\theta}}[\nabla_\theta \pi_\theta(s) \nabla_a Q(s,a|\omega)|_{a=\pi_\theta(s)}] \simeq \frac{1}{N}\sum_{s\in E}[\nabla_\theta \pi_\theta(s) \nabla_a Q(s,a|\omega)|_{a=\pi_\theta(s)}]$$

よって, 経験データ E の分布は割引分布 $\rho^{\pi_\theta}(s)$ に等しいことが望ましい．従って, 入力に大きなノイズを加えることは, 方策勾配の近似に対しては望ましくない操作であると言える．

7.3 理想ケースとそれを達成する工夫

これまで考察してきたことをまとめると, DDPG において最も理想的な経験データ集合 E の分布は

1. 状態 s の分布は, 割引分布 $\rho^{\pi}(s)$ に近い
2. 各状態 s に対して，行動 a の分散が大きい

を満たすものであると考えられる. そこで「入力 a を変えても, 次ステップの状態 s' が大きく変わらない場合」は大きなノイズを加え，「入力 a を変えると, 次ステップの状態 s' が大きく変わってしまう場合」はノイズは小さくすれば, 上記の要求を満たす経験データ集合を構築できるのではないか．

例えば, 探索ノイズを正規乱数 $\mathcal{N}(0,1)$ のスカラー倍として与える場合に, そのスカラー k(以下ノイズスケールと記載) を適応的に変化させることを考える．次ステップの状態 s' は, 現在の状態 s と入力信号 u および通信間隔 τ の関数として与えられるので,

$$\frac{\partial s'}{\partial u}, \frac{\partial s'}{\partial \tau}$$

の大きさによって，u, τ それぞれに加えるノイズスケール k_u, k_τ を適応的に与えれば良い．

7.4　提案手法による実験結果と考察

7.3 節で提案した探索手法を用いて強化学習を行い, それにより得られた方策を $\pi_{\mathrm{prop}}(s)$ を表記すると, その評価関数 $J(\pi_{\mathrm{prop}})$ の値は,

$$J(\pi_{\mathrm{prop}}) = 68.10 \pm 0.018 \tag{21}$$

となり, 式 (18) における $J(\pi_{\mathrm{RL}})$ からの改善が見られた.

7.5　提案手法の妥当性の検討

7.3 節で提案した探索手法は, 次ステップの状態が方策 $\pi_\theta(s)$ で制御を行った場合から, 大きく変化することを避けることを目的としていた. これは新たな状態を経験しないことになるので, 果たして探索と言えるのであろうか.

3.3 節で述べたように, 連続行動空間では

$$\pi(s) \leftarrow \mathrm{argmax}_a Q^\pi(s,a) \tag{22}$$

として方策改善をすることは難しい. そこで方策勾配法を用いて方策改善を行う手法をとっていた. したがって, 連続行動空間の強化学習においては, エージェントは「いち早く、最適な状態行動組を経験する」ことよりも, 「方策勾配を正しく計算する」ことを優先すべきである. よって, 提案手法には妥当性があると考えられる.

8　まとめ

本稿では, 最適セルフトリガー制御を定式化し, その強化学習による解法を試した. それにより得られた方策や, 方策勾配の近似に用いられた関数などを解析することにより, 課題点の抽出とその原因の特定を行った. さらに, 抽出した課題点に対する解決策の提案も行った.

しかしながら, 本稿で提案した手法は強化学習一般の課題点を解決するためのものであり, またその提案手法も, 本稿で定式化した最適セルフトリガー制御にユニークな特徴を活かしていない. 今後は有効な特徴がないか調査していきたいと考える.

また方向では, DDPG における「探索」のモチベーションを, $\nabla_a Q^\pi(s,a)|_{a=\pi(s)}$ の近似のみにあると仮定して議論を行った. この妥当性についても検討していきたい.

謝辞

本研究に取り組むにあたって助言をいただいた情報太郎教授と工学次郎助教に深く感謝する.

参考文献

[1] C. J. Watkins, and P. Dayan. Q-learning. *Machine Learning*, vol. 8, no. 3-4, pp. 279-292, 1992.

[2] V. Minh, K. Kavukcouglu, D. Silver. et al.. “Human-level control through deep reinforcement learning.” *Nature 518*, pp.529-533, 2015.

[3] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, et al.. “Deterministic Policy Gradient Algorithms.” *ICML Beijing, China.*, 2014, Beijing.

[4] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N.Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *International Conference on Learning Representations*, 2015.

[5] T. Degris, M. White and R. Sutton. “Off-Policy Actor-Critic.” *ICML Edinburgh, United Kingdom*, 2012.

[6] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. ”Policy gradient methods for reinforcement learning with function approximation.” *In Advances in Neural Information Processing Systems*, 2000.

[7] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization.” *arXiv preprint arXiv: 1412.6980*, 2014.

付録 A　aho

修士論文

数理モデルの構築と効率的なアルゴリズムの開発

指導教員　　情報太郎　教授
　　工学次郎　助教

数理　三郎

京都大学大学院情報学研究科

数理工学専攻

令和３年２月

数理モデルの構築と効率的なアルゴリズムの開発

数理　三郎

令和三年二月

数理モデルの構築と効率的なアルゴリズムの開発

数理　三郎

摘要

本研究では，修士論文を執筆する学生の数理モデルを構築し，このモデルを用いて効率的に修士論文を作成するためのアルゴリズムを開発した．また，このアルゴリズムを用いて，実際に本論文を作成した．その結果，従来の自分で執筆する方法に比べて，65536 倍効率的に修士論文を作成することが可能であることが確認された．