制御システム論分野研究会資料

2020 年 12 月 15 日

M2 竹内 維吹

進捗報告 12.2

1 はじめに

近年, 機械学習分野で注目を集めているトピックとして強化学習というものがある. これは, ロボットなどが環境とのインタラクションを通じて, 最適な制御方策を見つけ出していくというものである. その中でも, 強化学習にニューラルネットワークを組み込んだ深層強化学習は, その適応能力の高さから多くの研究がなされている. しかしながら, 深層強化学習にはハイパーパラメータへの強い依存性など, 依然として課題点が多くまだまだ発展途上にある.

本稿では, 対象となる問題を最適セルフトリガー制御問題というものに絞り, 強化学習の課題点に対して有効な特性の調査に向けた準備を行う. まず, 深層強化学習の基礎についての確認を行った後, 最適セルフトリガー制御問題を定式化し, その強化学習に特有の課題点を抽出する. 最後に, その課題点の解決に向けた今後の展望についても簡単に述べる.

2 方策勾配を用いた強化学習

2.1 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.

2.2 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)

2.3 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 [**?**] 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.

2.4 方策勾配による方策関数のパラメータの更新

Silver ら [**?**] 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.

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

ただし,

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

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)[**?**] 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.

3 最適セルフトリガー制御問題に対する強化学習

3.1 セルフトリガー制御

図 1 のような制御系を考える.

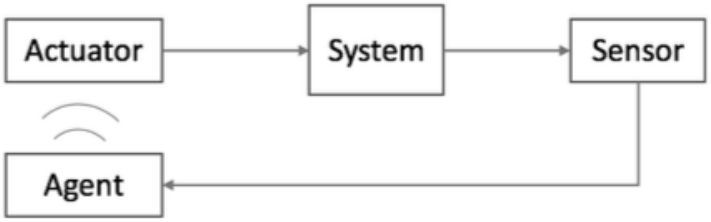

図 1 制御系

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).

3.2 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.

4　数値実験による課題点の抽出

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.

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

4.1　初期方策

学習の初期方策として

$$\begin{cases} a(s) = a^*_{0.05}(s) \\ \tau(s) = 0.2 \end{cases} \tag{16}$$

とする方策 π_{init} を用いる. ただし, $a^*_{0.05}(s)$ 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.

4.2　学習によって得られた方策

方策を表現するニューラルネットワークモデルや, 学習率などを工夫することで, 学習を安定的に進めることができた. 方策 π_{init} から 1000000 ステップの学習を行うことで得られた方策を π_{RL} と書くと, π_{RL} による, とある初期状態からの制御の様子は以下のようになる.

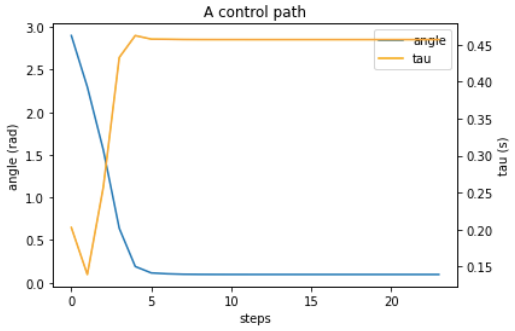

図 2　π_{RL} による制御ログ (角度 (rad) と通信間隔 (s))

図 2 からは, 原点付近では通信間隔を大きく, そうでない場合は通信間隔を小さく, というふうに適応的に通信間隔を決定している様子が見られる.

この時の評価関数 $J(\pi_{\mathrm{RL}})$ の値は

$$J(\pi_{\mathrm{RL}}) = 45 \tag{17}$$

となり, 初期方策 π_{init} に対する $J(\pi_{\mathrm{init}}) = -14$ を上回った.

4.3　学習の様子

DDPG では, 方策 π を表現する actor と, 方策 π の良し悪しを特徴付ける $Q^{\pi}(s,a)$ を表す critic をそれぞれ更新していく. critic は過去の経験データを教師データとして用いる教師あり学習でパラメータを学習する. また, actor は critic が正しく $Q^{\pi}(s,a)$ を近似できていれば (方策勾配の割引分布に関する期待値を除けば) 評価関数 $J(\pi)$ を大きくする方向に正しくパラメータを更新することができる. したがって critic の学習における損失関数の値を追うことで, DDPG アルゴリズムが方策改善を行えているかどうかおおよそ知ることができる.

方策 π_{init} から方策 π_{RL} を得る 1000000 ステップの学習では, critic の損失関数がある一定値のまわりの値を取りながら振動しており, 収束していく様子は見られなかった. この原因を考察していく.

actor のパラメータ更新には方策勾配 (5) が用いられる. したがって $\nabla_a Q(s,a|\omega)$ の値は非常に重要である. (データは後で載せますが,) 方策 π_{RL} でエピソードを数回経験すると, 原点付近の状態を数多く経験していることが確認できる. これは方策勾配の状態 s に関する期待値が, 原点付近にあることを示しているので, 特に $s=[0,0]$ での critic $Q(s,a|\omega)$ の, $u=0$ の断面を見てみると,

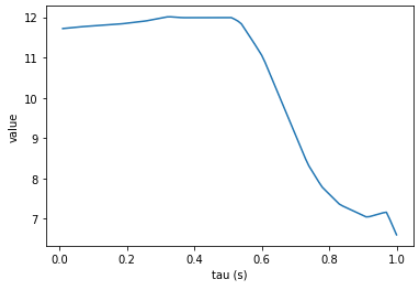

図 3　原点で何も入力を加えない時の critic と τ のグラフ

のようになった. このため, τ に関する $Q(s,a|\omega)$ の微分値がほぼ 0 となるため, オプティマイザ Adam の性質により, この台の上を振動してしまう. ただし, 微分値がほぼ 0 となったり, $\tau > 0.5$ の領域で急激に減少するのは探索不足が原因である (探索雑音を加えすぎると発散してしまう (探索と活用のジレンマ)).

従って actor が収束しないことになるので, critic も収束しない. そのことを, critic $Q(s,a|\omega)$ の a に $\pi_{\mathrm{RL}}(s)$ を代入した関数 (以下 $V_c^{\pi}(s)$ と記載) と, 定義 (2) 通りの価値関数 $V^{\pi}(s)$ を図示して比較することで確認していく.

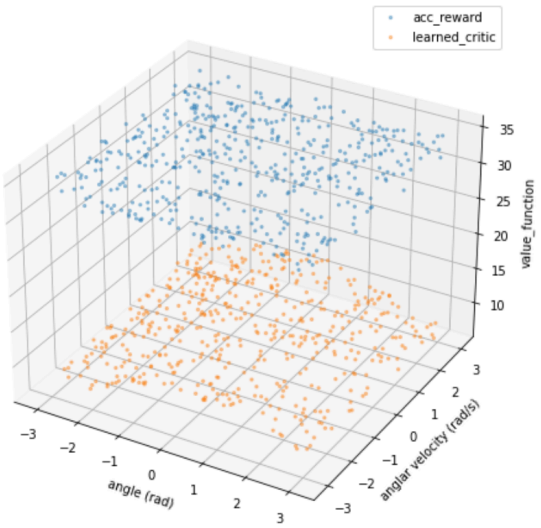

図 4　DDPG で得られた critic による価値関数 V^{π} の近似

図 4 では橙色に V_c^π, 青色に価値関数 V^π の関数形を示している (同一色の点を複数プロットすることにより, 関数曲面を表現しているとみてください.) critic が Q 関数をよく近似できていれば, 橙色の曲面は青色の曲面に一致するため, 近似できているとは結論づけられない.

5　critic からリアプノフ関数へ

前節では actor が収束しないため, critic が Q 関数を近似する限界があることを示唆した. では, actor を固定して Q 関数を教師あり学習すれば, critic を収束させられるか確認してみた.

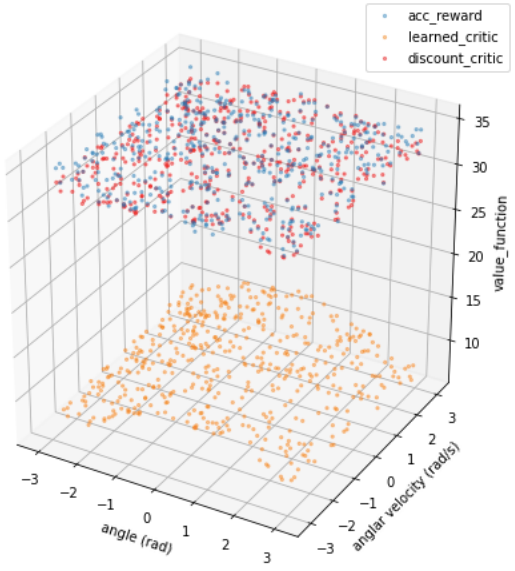

図 5　教師あり学習で追加学習した critic による価値関数 V^π の近似

図 5 は, 図 4 に教師あり学習で学習した critic に対する V_c^π を赤色で追加した図である. 赤色と青色の曲面を比べると, 近いものが得られているので, (価値関数 V は) 近似が可能であると結論づけられる.

6　Q 関数と critic ネットワークの比較による, 課題点の抽出

6.1　Q 関数の近似精度

方策勾配は,

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

と書けるので, その計算に用いられるのは $Q^\pi(s,a)$ の a 勾配である. 実際の方策勾配の計算には $Q^\pi(s,a)$ の近似である critic ネットワーク $Q(s,a|\omega)$ を用いるので, $Q^\pi(s,a)$ と $Q(s,a|\omega)$ の a に関する勾配が等しいことが望ましい. その性質を満たしているか確認する為, $Q(s,a|\omega)$ と $Q^\pi(s,a)$ に対して, $s=[0,0]$ とした時の関数形を図 6 に示す.
ただし, Q^π は漸化式

$$Q^\pi(s,a) = r(s,a) + \gamma V^\pi(s')$$

における V^π に, 図 5 で得た価値関数を用いることで計算した. 図 6 から, 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. この原因として考えられるのは,

1. critic の教師あり学習に用いたデータ組 $\{s,a\}$ の偏り
2. ニューラルネットワークの表現力不足

が挙げられる. 特に 1 つ目について, 考察していく.

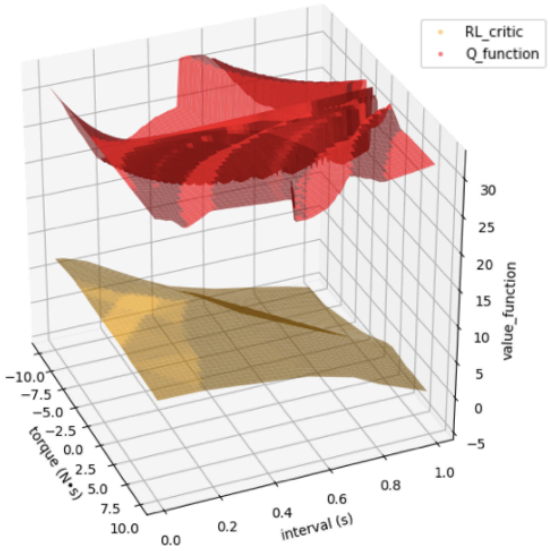

図 6　$s = [0, 0]$ に対する Q 関数の近似精度

6.2　critic の近似と経験データの分散

critic の近似精度の低さの原因の一つに, critic の教師あり学習に用いたデータ組 (s, a) の偏りを挙げた. 教師あり学習における教師データの分布がある領域に偏ってしまうと, 当然その分布外の (s, a) における関数近似は行うことができない. 経験データの分散を上げる為に多様な入力を試してみることを探索というが, 連続行動空間の強化学習では

$$a = \pi_\theta(s) + e \tag{18}$$

のように入力にノイズ e を付加してデータ収集を行うのが一般的である. しかしながら, 実際に実験で加えていたノイズは, 非常に小さいものであった. このことが経験データの分布に偏りをもたらしてしまっていると考えられる.

さて, 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. ここで, E は過去の経験データを表す為, その分布に偏りがあったことが, 低い近似精度の原因の一つであると考えられる.

6.3　割引分布と探索ノイズ

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.

6.4　探索と利用のジレンマ

前節と前々節で確認した探索ノイズに対する 2 つの要求は互いに相反するのものであり, 強化学習における有名な課題である. 最適セルフトリガー制御の強化学習に対する最後の課題として, このジレンマの最小化に取り組みたい. より具体的には, 効率的な探索ノイズについて考察する.

効率的な探索とは, 以下の二つの要求を満たすような探索である.

1. 経験する状態 s の分布は, 割引分布 $\rho^{\pi}(s)$ に近いものにしたい
2. 各状態 s に対して, 様々な入力 a を経験したい

前述の通り, 連続行動空間に対する探索には, データ収集時の入力にノイズを加えるという手法が用いられる. このノイズの大きさを適応的に制御することによって, 上記のジレンマを解決する手法を考えたい. 以下ではそれに対する考察を行う.

6.5　適応的探索ノイズ

前節で述べたジレンマは, 「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ることによって, 最小化できるのではないかと考えている.

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.

しかしながら, s' の計算にはシステム (15) が既知である必要があるので, 議論の一般性を失う. そこで, システムに対する必要な知識を緩和することを考える. [1] では, 入力アフィン系

$$\dot{s} = f(s) + g(s)u \tag{19}$$

における f, g のリプシッツ連続という性質のみを用いて,

$$\|s' - s\| \leq \frac{1}{L}\|f(s) + g(s)u\|(e^{L\tau} - 1) = \bar{r}(s, u, \tau) \tag{20}$$

のように, セルフトリガー制御における状態変化に上界を与えた. これはつまり, 次ステップの状態 s' が

$$s' \in B, B = \{s' : \|s' - s\| \leq \bar{r}\} \tag{21}$$

を満たすことを保証したことになる (図 7 参照).

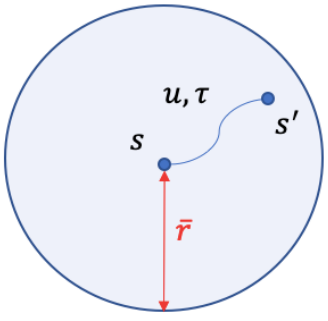

図 7　セルフトリガー制御における状態変化の上界

この半径 $\bar{r}$ に対して,

$$\frac{\partial \bar{r}}{\partial u}, \frac{\partial \bar{r}}{\partial \tau}$$

を用いることにより, ノイズの適応的な制御ができないかを考えていきたい.

参考文献

[1] G. Yang, C. Belta, and R. Tron. “Self-triggered Control for Safety Critical Systems Using Control Barrier Functions.” *In American Control Conference (ACC) Philadelphia, USA*, 2019.