制御システム論分野研究会資料

2020 年 9 月 8 日

M2 竹内 維吹

進捗報告 9.7

1 前提

対象のシステムが

$$\dot{s} = f(s) + g(s)a \tag{1}$$

と書かれており, これを離散化したシステムが

$$s_{t+1} = f_d(s_t) + g_d(s_t)a_t \tag{2}$$

であるとする. 本研究ではこのような入力アフィン系のセルフトリガー制御を考える.

1.1 倒立振子による実験

倒立時の振子の角度を $\theta = 0$ とし, 加えられる入力が $A = [-10\mathrm{N}\cdot\mathrm{m}, 10\mathrm{N}\cdot\mathrm{m}]$ と制限されるような倒立振子を考える. この倒立振子のダイナミクスは, 以下のように与えられる.

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

コンピュータで強化学習を行う場合, これを離散化したシステムについて計算を行う必要がある. 上記の状態方程式を離散化すると以下のようになる.

$$\theta_{t+1} = \theta_t + \dot{\theta}_t\delta_t + \frac{3g}{2l}\sin\theta_t\delta_t^2 + \frac{3}{ml^2}a\delta_t^2 \tag{2a}$$

$$\dot{\theta}_{t+1} = \dot{\theta}_t + \frac{3g}{2l}\sin\theta_t\delta_t + \frac{3}{ml^2}a\delta_t \tag{2b}$$

ただし, δ_t は離散化定数であり $\delta_t = 0.005$ とする.

2 現状確認

2.1 セルフトリガー制御

図 1 のような制御系を考える.

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, 毎時刻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ステップ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 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).

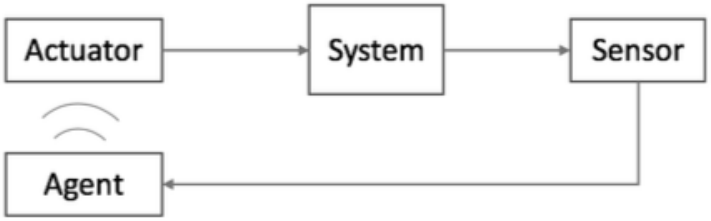

図 1　制御系

2.2　目標点の確認

研究を通しての目標は「安全性を確保しながら, 最適セルフトリガー制御則 π^* の強化学習を実現させること」である。ここで

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, J(\pi) \tag{3}$$

$$J(\pi) = \mathbb{E}_{s_0 \in d_0}[V^\pi(s_0)] \tag{4}$$

$$V^\pi(s_0) = \sum_{i=0}^{\infty} \gamma^i C_i^\pi \tag{5}$$

$$C_i^\pi = -\int_{T_i}^{T_{i+1}} (s_t^\top Q s_t + \tau_i a_i^\top R a_i)\mathrm{d}t + \lambda \tau_i,\ T_i = \sum_{l=0}^{i} \tau_l \tag{6}$$

であり, π_1, π_2 は π の第 1, 第 2 成分である. また, k は通しのステップ数, 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.

以下では方策 π を θ でパラメトライズし, $\theta^* = \underset{\theta}{\operatorname{argmax}}\, J(\pi_\theta)$ を解くことによって $\pi^* = \underset{\pi}{\operatorname{argmax}}\, J(\pi)$ を得るものとする. 方策勾配を用いた強化学習では $\nabla_\theta J(\pi)$ を用いて θ^* を求める. その際方策勾配 $\nabla_\theta J(\pi)$ の近似のため, 実環境とのインタラクションによって得られたデータ組 $\{s, a, r, s'\}$ を用いる. 「学習中の安全」という言葉を, 「このデータ組の収集を決められた安全領域 $\mathcal{C}$ の内部でのみ行うこと」と定義する.

2.3　実現可能性の検証:サンプル値系での実験

上記の目標を達成する見込みがあるのかを検証するために, サンプル値系での実験を行う. サンプル値系では, セルフトリガー制御と同様に毎ステップのインタラクションは行わない. セルフトリガー制御との違いは, インタラクションの間隔がエージェントによって状態 s 依存で決定するのではなく, 制御問題の設定として定数 t_{int} で与えられる点である. したがってサンプル値系での制御方策 π_{sample} は, アクチュエータに送信する入力信号 a のみを出力する関数として与える.

サンプル値系での実験により, $t_{\mathrm{int}} = 1$ のサンプル値系での最適方策

$$\begin{cases} \pi_{\mathrm{sample},1} = \underset{\pi_{\mathrm{sample}}}{\operatorname{argmax}} \sum_{i=0}^{\infty} \gamma^i (-s_i^\top Q s_i - a_i^\top R a_i) \\ a_i = \pi_{\mathrm{sample}}(s_i) \end{cases} \tag{7}$$

を初期方策として, $t_{\mathrm{int}} = 2$ のサンプル値系での最適方策

$$\begin{cases} \pi_{\mathrm{sample},2} = \underset{\pi_{\mathrm{sample}}}{\operatorname{argmax}} \sum_{i=0}^{\infty} \gamma^i (-s_i^\top Q s_i - a_i^\top R a_i) \\ a_i = \begin{cases} \pi_{\mathrm{sample}}(s_i)\ (\text{if } i \text{ is odd}) \\ a_{i-1}(\text{otherwise}) \end{cases} \end{cases} \tag{8}$$

を学習中の安全性を満たしながら学習できるかを検証する.

2.4　セルフトリガー制御への発展

前節での検証によって, インタラクション間隔を大きくしても安全強化学習を行うことが可能であることを確認できたとする. サンプル値系での制御則は入力信号 a のみを出力する関数であったので, 入力信号 a とインタラクション間隔 τ の二つの要素を出力する必要があるセルフトリガー制御の初期方策として方策 $\pi_{\mathrm{sample},1}$ をそのまま用いることはできない.

そこで代替策として,

$$\begin{cases} \pi_1(s) = \pi_{\mathrm{sample},1}(s) \\ \pi_2(s) = 1 \end{cases} \tag{9}$$

とする方策 π_{init} をセルフトリガー制御の強化学習のための初期方策として用いる.

2.5　現状持ち合わせているもの

現在, $\pi_{\mathrm{sample},1}$ と, その学習プログラムは作成しているが, π_{init} を作成することはまだできていない.

その他, 持っている知識として CBF と ECBF[1] がある. CBF を用いることによって, 状態 s に対する安全な入力集合を与えることができる. また ECBF によって, ZOH 制御における, 安全領域を出ないような最大のインタラクション間隔 $\tau_{\max}$ を与えることができる. これらを踏まえて, 今後の研究の道筋を次節に記す.

3　今後の方針

3.1　CBF を用いた安全入力について

3.1.1　制御バリア関数 (CBF)

強化学習ではデータの収集に環境とのインタラクションを行う必要がある. DDPG と呼ばれるアルゴリズムは方策オン型の強化学習とよばれ, データの収集方策に学習中の暫定最適方策を用いる. したがって, 学習初期の方策では安全性が保証されないことがしばしばある. この課題を解決するために, 制御バリア関数を用いる.

関数 $h(s)$ が以下の条件を満たす時, システム (3) に対する制御バリア関数であるという.

$$\sup_{a \in A} \left\{ \frac{\partial h}{\partial s}(f(s) + g(s)a) + K(h(s)) \right\} \geq 0 \tag{10}$$

ただし, $K(s)$ はクラス K 関数である.

さて, 2.2 節にて登場した安全領域 $\mathcal{C}$ を

$$\mathcal{C} = \{ s \in S \mid h(s) \geq 0 \} \tag{11}$$

として与える. このとき $h(s)$ が制御バリア関数であるならば, 状態 $s \in C$ を初期状態とした時, それ以降の全時刻において, 状態 s が $s \in C$ を満たすようにする入力が存在することを保証する. そのような入力集合は現時刻での状態 s に依存し,

$$U(s) = \left\{ a \in A \mid \frac{\partial h}{\partial s}(f(s) + g(s)a) + K(h(s)) \geq 0 \right\}, \forall s \in C \tag{12}$$

としてその集合を与える.

学習中の安全性を確保するために, 図 2 のようにエージェントの出力を $U(s)$ の要素に射影するレイヤーを設ける.

当面は, エージェントの出力 a_π に最も近い $U(s)$ の元への射影を考える.

3.1.2　離散化による影響

さて, 制御バリア関数による安全性保証は連続システムに対して行われるものであり, コンピュータ上でのシミュレーション環境である (2) のような離散システムに対しては安全性の保証はできない. つまり, $\mathcal{C}$ の淵の状

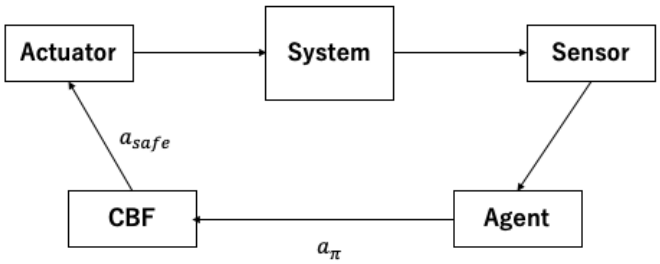

図 2　制御系

態に対して, 次ステップでの状態が $\mathcal{C}$ を出てしまう可能性があるということである. そこで, 離散化が理由で多少 $\mathcal{C}$ が出てしまった場合, 最も効率の良い入力 (例えば $\theta < 0$ なら $a = 10$) を用いて, 人為的に $\mathcal{C}$ の内部に戻すものとする.

また, この問題は制御バリア関数による入力の補償を行う必要がある状況で起こりうるものであるため, その補償が執行されるのを回避するように学習させる方法はないか模索・検証したい.

3.2　サンプル値系での検証

2.3 節で記述した通り.

3.3　セルフトリガー制御の強化学習

2.4 節で記した初期方策 π_{init} から, 安全性を保証しながら最適セルフトリガー制御則 π^* の学習を試みる. その際, CBF による安全入力集合への射影に加えて, ECBF による τ に対する安全性保証も行う.

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.