残差強化学習を用いた最適制御との効率的な差分学習による制御

M2021SC006 真野翼
指導教員：坂本登

1 はじめに

現在産業で広く用いられているロボットの制御手法としてはPID制御, モデルベース手法が主流であるが, 複雑なモデルの場合や現実とのギャップにより上手くいかない場合が存在する. それに対して強化学習 (Reinforcement Learning:RL)[1] による制御は環境への相互作用の中で適応的に制御方策を学習する為, 複雑な環境でも学習次第では人間の想定する性能を超えることもあり, 現在注目されている手法である. しかし実機での強化学習の実装は安全制約上や, 学習コストの重さから現実の限られたサンプル数では難しいなどの実世界上の課題により成功例は限られたものとなっている [2]. そうした課題解決の一環として学習効率の改善手法が近年は研究されており, 中ではResidual Reinforcement Learning(残差強化学習) といった既存制御手法との残差の学習を行う事で効率を改善するといった手法が用いられている.[3][4] 特に [5] では振子の安定化に線形最適制御との差分を学習しモデル精度のばらつきに対応している. しかしこの手法では線形領域での評価に留まっている. 本論では残差強化学習を非線形特性を持つ振り子の振り上げ安定化タスクに適用し, 最適制御との差分を学習し本手法の有効性を検証する.

2 問題設定

2.1 Pendulum-v1

本論では数値実験の環境として OpenAIGym の Pendulum-v1 環境 [6] を用いて検証する.Gym とは強化学習の学習を円滑に行えるシミュレーション環境を提供しているライブラリであり,Pendulum-v1 環境では振り子を中央のアクチュエータにトルク τ を加えて, 振り子を垂直軸上の平衡状態で安定させることを目的とする.(ただしトルクには制限があり振り上げにははずみが必要なことに注意する) 今回はこれを用いて提案手法の検証を行う. また観測できる状態は振子中心の x 座標,y 座標, 角度の微分の三つであり, すなわち 3 次元の状態空間となる. また行動空間は一次元の 範囲内で制限されているジョイントに適用されるトルク τ $(-2 \leq \tau \leq 2)$ である.

2.2 制御対象のモデル

Pendulum-v1 環境の振り子は以下の運動方程式に従う.

$$I\ddot{\theta} = -\frac{1}{2}mgL\sin\theta - \mu\dot{\theta} + \tau \tag{1}$$

ここで慣性モーメント $I = \frac{1}{3}\mathrm{kgm}^2$ 振り子の質量 m=1kg, 振子の長さL=1m重力加速度 $g = 9.81\mathrm{m/s}$, 粘性摩擦係数 $\mu = 0.03$, 振子の角度 θ とその時間微分 $\dot{\theta}$ を用いて状態 $\boldsymbol{x} = [\theta\, \dot{\theta}]$ とする. これを変形し, 原点近傍で線形化を行う事により以下の状態方程式 (2) 式を得る.

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 1 \\ -3g/2L & -3\mu/mL^2 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 3/mL^2 \end{bmatrix} \tau \tag{2}$$

2.3 報酬設定

学習に用いる報酬は環境デフォルトの式 (3) の報酬を用いる.

$$-(\theta^2 + 0.1\dot{\theta}^2 + 0.001\tau^2) \tag{3}$$

一般に Pendulum-v1 では望ましい挙動が学習できていれば一連の動作の合計報酬は 1 エピソード当たり-130 程度に収束する.

3 準備

3.1 強化学習

強化学習とは基本的には環境内のある問題に対し, 環境内で行動を行う agent が得られる報酬の最大化ができるような行動方策を学ぶことを目的である. また, 一般的な問題はそのままでは取り扱いが難しいことからマルコフ性を満たすと仮定する. 本節では文献 [9] に習い問題をマルコフ決定過程 (Markov decision process:MDP) としてモデル化する.MDP は $\langle S, A, p, R, \gamma \rangle$ で表される. このとき状態行動空間 S, 行動空間 A とし, 状態遷移確率 $p : S \times A \times S \to [0, 1]$ はある時刻での, 状態 $s_t \in S$ と行動 $a_t \in A$ について状態 s_{t+1} に遷移する条件付き確率である. また報酬関数 $R : S \times A \to \mathbb{R}$ はある状態遷移について得られる即時報酬である. 方策 π_θ は確率的であり $\pi_\theta : S \to P(A)$ で示され, ここで $P(A)$ は A 上の確率測度の集合,$\theta \in \mathbb{R}^n$ は n 個のパラメータを持つベクトルである. このとき π_θ は s_t に対する条件付き確率であり,s_t に対して確率的に行動 a_t を生成する関数として定義できる.$\gamma (0 \leq \gamma \leq 1)$ は割引率であり, 割引率を増減させることによって長期的な報酬と短期的な報酬のどちらを重視するかを調整することが出来る. これを用いてある時刻における累積報酬和 ${r_t}^\gamma$ は, 以下の式 (4) で与えられる.

$${r_t}^\gamma = \sum_{k=t}^{\infty} \gamma^{k-t} R(s_k, a_k) \tag{4}$$

ここで強化学習における初期時刻からの累積報酬の最大化を行うには,MDP のある時刻以降の状態遷移はそれ以前の状態遷移に依存しない性質より, 各時刻における累積報酬和の期待値を増大させるように確率的な方策 π_θ の確率を変えていけばよい. 次節では累積報酬和の期待値の導出の為, ベルマン方程式について説明する.

3.2 ベルマン方程式

最適な方策の評価の為には, ある方策を決定した上で, その方策の元行動する agent が獲得できる累積報酬和を

計算する必要がある. 有限区間の元でここで状態価値関数 $V^{\pi}(s_t)$ を式 (5), 行動価値関数 $Q^{\pi}(s,a)$ を式 (6) として定義する. 強化学習の多くはこの $V^{\pi}(s_t)$ か $Q^{\pi}(s,a)$ を目的関数としてその最大化を行う方策の学習問題に帰着する. 状態価値関数 $V^{\pi}(s_t)$ はある状態で得られる累積報酬和の期待値, 行動価値関数 $Q^{\pi}(s,a)$ ある状態で特定の行動を選んだ際に得られる累積報酬和の期待値を意味する.

$$V^{\pi}(s) = \mathbb{E}_{\pi}[{r_t}^{\gamma}|s_t = s] \tag{5}$$

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}[{r_t}^{\gamma}|s_t = s, a_t = a] \tag{6}$$

式 (5),(6) はベルマン方程式を満たし, その解は動的計画法を解くことで効率的に得ることが出来る. 具体的には無限区間の累積報酬和ではなく 1 ステップ先のみを考慮する累積報酬和を考えてみればよい. 行動空間が離散の場合最も高い即時報酬を得られるような行動を選べば求めることが出来る,2 ステップ先を考慮する累積報酬和は先程求めた 1 ステップ累積報酬和で計算が出来, 再帰的に繰り返せば n ステップの累積報酬和を求めることが出来,$V^{\pi}(s)$ を導出できる.[7] また, 特に Q-learning[8] と呼ばれる反復手法では最適な方策をとった時の評価値に概収束するという性質があり, 行動空間の中で最適行動価値関数 $\dot{Q}^{\pi}(s,a)$ が最も高いものを選べば最適方策になるという非常に優れた方法である. しかし高次元の状態, 行動では計算量の観点で実装が難しい, また連続空間に対応していないという問題から $V^{\pi}(s_t)$,$Q^{\pi}(s,a)$ の近似手法の研究が行われ, 優れた関数近似性能をもつニューラルネットワークを用いた深層強化学習による研究が現在は主流となっている. 次節では本論で用いる深層強化学習アルゴリズムについて説明する.

3.3 Deep Deterministic Policy Gradient

本論で用いる強化学習アルゴリズムの Deep Deterministic Policy Gradient(DDPG)[9] は方策勾配法をベースに発展した actor-critic な深層強化学習手法である. 方策勾配法では方策をパラメータ θ によるニューラルネットワークで直接表現し, 前述の $V^{\pi}(s_t)$ のような目的関数について勾配を計算しパラメータ θ を更新することでより良い方策を直接求めることが出来る. 特に actor-critic は方策の近似に加えて行動価値関数 $V^{\pi}(s_t)$ の近似も行いより効率的な学習を行っている. 方策勾配法では学習率 α, 方策勾配 $\nabla_{\theta}J(\theta)$ を用いて,(7) 式のようにパラメータ θ を更新している.

$$\hat{\theta} = \theta + \alpha\nabla_{\theta}J(\theta) \tag{7}$$

$$\nabla_{\theta}J(\theta) = \mathbb{E}_{\pi}\left[\nabla_{\theta}\ln\pi_{\theta}(a|s)Q^{\pi}(s,a)\right] \tag{8}$$

勾配計算には式 (8) に記す方策勾配定理 [10] を用いる. 特に DDPG では (9) 式の近似を利用している.[9]

$$\begin{aligned}\nabla_{\theta\mu}J(\theta) \simeq \frac{1}{N}\sum_{i}[\nabla_a Q(s,a|\theta^Q)|_{s=s_i,a=\mu(s_i)} \\ \nabla_{\theta\mu}\mu(s|\theta_{\mu})|_s = s_i]\end{aligned} \tag{9}$$

3.4 残差強化学習

残差強化学習では環境で実行される行動 $\hat{a}$ は, ベースとなる制御則による制御入力 u_t と方策 π_{θ} により (10) 式で定義される,

$$\hat{a}_t = u_t + \pi_{\theta}(s_t) \tag{10}$$

ここで用いる制御入力 u_t 及び方策 π_{θ} は理論上は任意のものを用いることが出来る. ここでは最適制御及び非線形最適制御をベース制御,DDPG を方策の学習に用いた.

3.5 線形最適制御

線形システムではリッカチ方程式を解くことで以下の (11) 式の評価関数を最小化するような状態フィードバックコントローラのゲインを得る.

$$\int_0^{\infty}(\boldsymbol{x}(t)^T Q\boldsymbol{x}(t) + \boldsymbol{u}(t)^T R\boldsymbol{u}(t))dt \tag{11}$$

ここで 2 と合わせ,$\boldsymbol{Q} = diag(150,1), R = 1$ として与えると. 得られたフィードバックゲイン K は (12) 式であった.

$$K = \begin{bmatrix}8.2287 & 2.5467\end{bmatrix} \tag{12}$$

3.6 非線形最適制御

非線形最適制御 (NonlinearOptionalControl:NOC) を行う手法は, 動的計画法により導かれるハミルトン・ヤコビ方程式を級数展開法により解く手法 [11], 及び安定多様体法によりその近似解を得る手法 [12] などが挙げられるが, ここでは簡単なオイラー・ラグランジュ方程式による二点境界値問題を数値解法の勾配法により解く手法 [13] を用いる. (1) 式に従う以下のような非線形状態方程式を考える.

$$\dot{x} = f(x) + g(x)u \tag{13}$$

ここで非線形状態方程式 f を制約としたラグランジュ乗数として λ を導入すると, 最小化したい評価関数は $\bar{J}$, 終端条件を考慮しない場合 (14) 式で表される.

$$\bar{J} = \int_{t_0}^{t_f}(\boldsymbol{x}(t)^T Q\boldsymbol{x}(t) + \boldsymbol{u}(t)^T R\boldsymbol{u}(t) + \lambda^T(f - \dot{x}))dt \tag{14}$$

ただし t_0,t_f はそれぞれ初期時刻, 終端時刻である. ここで非線形最適制御の為にハミルトン関数 H を (15) 式で定義する.

$$H(y,u,\lambda,t) = \left(x^T Qx + Ru^2 + \lambda^T f\right) \tag{15}$$

このとき評価関数 $\bar{J}$ を最小化する最適な制御入力 $u(t)$ が存在するとき対応する最適軌道 $x(t)$ とすれば,n 次元ベクトル $\lambda(t)$ が存在し (16) 式のオイラーラグランジュ方程式を満たす.

$$\begin{aligned}&\dot{x}(t) = \left(\tfrac{\partial H}{\partial\lambda}\right)^{\mathrm{T}}(x,u,\lambda,t) = f(x(t),u(t),t) \\ &x(t_0) = x_0 \\ &\dot{\lambda}(t) = -\left(\tfrac{\partial H}{\partial x}\right)^{\mathrm{T}}(x,u,\lambda,t) \\ &\lambda(t_f) = \left(\frac{\partial\phi}{\partial x}\right)^{\mathrm{T}}(x(t_f)) \\ &\tfrac{\partial H}{\partial u}(x,u,\lambda,t) = 0\end{aligned} \tag{16}$$

$\bar{J}$ の微小変化は $\lambda(t)$ が (16) 式を満たすなら, 下式が成り立ち,

$$\delta\bar{J} = \int_{t_0}^{t_f} \frac{\partial H}{\partial u}\delta u \; dt \tag{17}$$

最急降下法の為に, 微小入力 δu を,

$$\delta u = -\alpha\frac{\partial H}{\partial u} \tag{18}$$

と与える.(ただし $\alpha > 0$ は学習率) こうすることで, $\frac{\partial H}{\partial u} \neq 0$ である限り,(19),(20) 式の通り $\bar{J}$ を減少させるように制御入力 u を変化させることが出来る.

$$\delta\bar{J} = -\alpha\int_{t_0}^{t_f} \left\|\frac{\partial H}{\partial u}\right\|^2 \; dt < 0 \tag{19}$$

$$\bar{u} = u - \alpha\frac{\partial H}{\partial u} \tag{20}$$

ここでは学習率 $\alpha = 0.0003$, 反復数は 2000 回で最適入力 $u(t)$ を求めた. また, この時の $\boldsymbol{Q} = diag(150, 150), R = 1$ とした. この入力応答及びハミルトン関数 H の微小変化を図 1 に示す. $\frac{\partial H}{\partial u}$ が 0 近傍にあることがわかる.

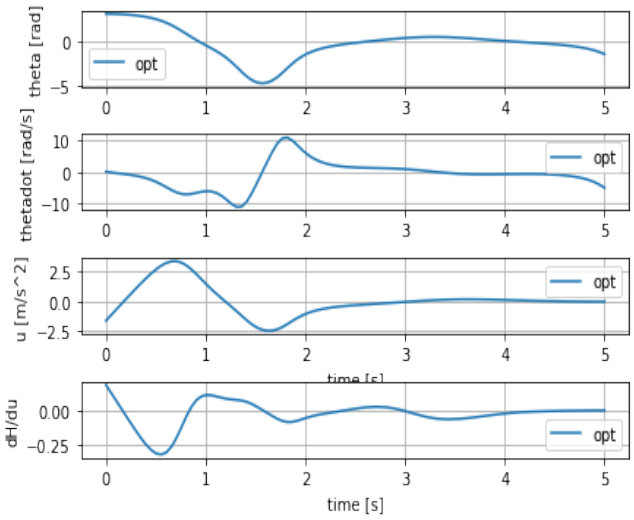


図 1: 非線形最適制御の入力応答

3.7 実装

前節で述べた線形最適制御及び非線形最適制御のその差分を残差強化学習により学習する. 線形最適制御による制御入力 u_t はトルクの制限と一致することが望ましい. その為 u_t は $-2 \leq u_t \leq 2$ 内になるようにクリッピングを行う. ここで方策 π_θ の出力をトルクの制限に合わせると非線形領域での両出力の和が打ち消しあい学習が進まない可能性が存在する. 最終的な制御出力は適応的に学習する方策関数が支配的であることが望ましい為,$-4 \leq \pi_\theta \leq 4$ を方策関数の取りうる範囲とする. 以上の二つの制御出力の和を用いて学習を行った.

4 実験

Pendulum-v1 で DDPG のみによる学習,DDPG+LQR 及び DDPG+NOC での残差強化学習の三つを行い性能を比較する. 使用したハイパーパラメータを表 1 に示す. パラメータは文献 [9] の実装に習って決定した. 学習終了の閾値は 10episode の平均報酬が-130 以上の場合に学習が成功したと判定し, 終了することとした. また学習環境は Google Colaboratory pro を用いて行った. 学習時の GPU は NVIDIA Tesla P100 であった.

表 1: Parameters

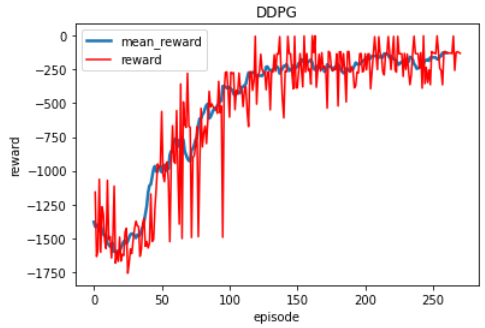
Hyper parameters	DDPG
Policy net	$400 - 300$
Q function net	$400 - 300$
Discount factor(γ)	0.9
Target net param	0.001
Adam Actor learning rate	0.0001
Adam Critic learning rate	0.001
Replay buffer size	1000000
batch size	64
Activation function	elu

5 結果

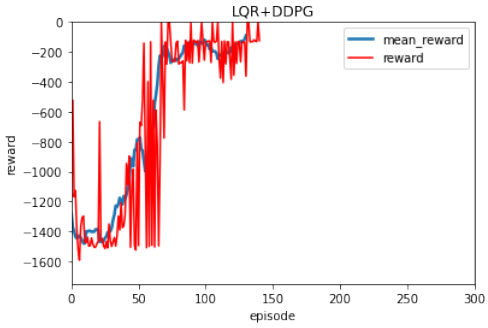
数値実験の結果を図 2 に示す. 縦軸を獲得した episode ごとの報酬, 横軸を学習した episode の数とした. 赤線は獲得報酬, 青線は 10episode 平均の獲得報酬である.DDPG のみによる学習は 270episode,LQR+DDPG による学習は 140episode,NOC+DDPG による学習は 129episode をそれぞれ学習完了までに必要とした. 三つの手法での学習曲線を比較すると DDPG では 120episode までにかけてなだらかに学習が収束しているのに対し,LQR+DDPG では 70episode で収束している他, 報酬の分散が低減している. 他に NOC+DDPG では開始時点高い平均報酬を獲得し, 収束までの時間も短くなっている. この結果から良いベース手法を用いる学習手法の方が効率的な学習が出来ていることが言える.

6 考察

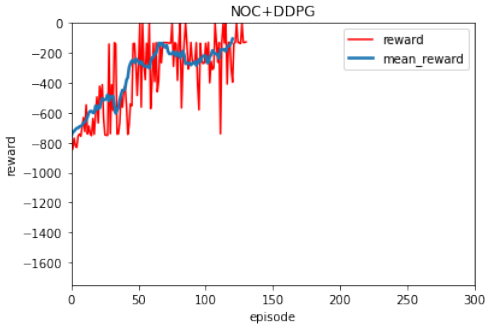
この結果から言える点として, 我々の既知の制御手法による入力を加えたシステムについてうまくいくような制御を考える問題は, 元の問題より簡単な強化学習の問題に帰着できるという可能性があるということである. 学習の効率化に寄与した点としてまず挙げられるものとして, そもそも振り子の振り上げの問題は線形領域での安定化と非線形領域の振り上げというタスクの二つに分解できる為,LQR を加えたシステムは非線形領域の振り上げだけを解く問題に簡単化できたということである. この結果は実世界の運用で強化学習を用いる場合に, 簡単な PID 制御程度のものでも加えた上で学習するだけで学習効率の向上が期待できる為, 実装コストの観点からも実用上の活用が期待できる. 次に残差強化学習で期待される学習効率の向上に寄与する点として探索範囲の減少が挙げられる, しかし LQR との残差は非線形領域では LQR との出力と大幅に異なるような補正が必要となる為, 本来期待されていた効果は余り発揮されていない可能性が挙げられる.NOC との残差は LQR と同程度の学習効率だったものの開始時点で大きく報酬平均が上がっていた. これは振り上げ動作



(a) DDPG による学習曲線



(b) LQR+DDPG による学習曲線



(c) NOC+DDPG による学習曲線

図 2: 実験結果

が NOC 部分に既に含まれていたということが考えられる. しかし初期位置に関わらず一定のフィードフォワード入力である NOC が有効に機能していることは興味深い事例だと考えられる.

7 おわりに

本論文では最適制御との残差強化学習による制御手法を提案した。また、これらの組み合わせが目標性能の達成までの学習量の減少に寄与し、効率的な学習を行っていることをシミュレーション環境での数値実験において発揮していることを示した. 今後の課題としてはディレイ, 外乱などの実機を想定したより困難な環境で検証を行っていくことでより提案手法の有効性を検証することが考えられる.

参考文献

[1] R. S. Sutton and A. G. Barto. Reinforcement learning: An introduction. MIT press, 2018

[2] Gabriel Dulac-Arnold, Nir Levine, Daniel J. Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, Todd Hester. An empirical investigation of the challenges of real-world reinforcement learning. arXiv arXiv:2003.11881 ,2021

[3] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, “Residual reinforcement learning for robot control,” in ICRA, 2019, pp. 6023–6029.

[4] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling, “Residual policy learning,” arXiv preprint arXiv:1812.06298, 2018.

[5] Y. Okawa, T. Sasaki and H. Iwane, ”Control Approach Combining Reinforcement Learning and Model-Based Control,” 2019 12th Asian Control Conference (ASCC), 2019, pp. 1419-1424.

[6] OpenAi Gym `https://www.gymlibrary.dev/environments/classic_control/`

[7] Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). MIT Press, Cambridge (2005) pp 421–436

[8] Watkins, C. J. C. H. and Dayan, P.: Technical Note: Qlearning, Machine Learning, Vol. 8, pp. 279–292 (1992)

[9] 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,” in International Conference on Learning Representations (ICLR), 2016.

[10] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, Martin Riedmiller, ”Deterministic Policy Gradient Algorithms” Proceedings of the 31st International Conference on Machine Learning, PMLR 32(1):387-395, 2014.

[11] D. L. Lukes: Optimal Regulation of Nonlinear Dynamical Systems, SIAM J. Control Optim., 7–1, 75/100 (1969)

[12] N. Sakamoto and A.J. van der Schaft: Analytical approximation methods for the stabilizing solution of the Hamilton-Jacobi equation, IEEE Trans. Automat. Control, 53-10, 2335/2350 (2008)

[13] 大塚敏之：非線形最適制御入門，コロナ社 (2011)