Meta Learning the Step Size in Policy Gradient Methods

L. Sabbioni, F. Corda and M. Restelli
{luca.sabbioni, marcello.restelli}@polimi.it, francesco.corda@mail.polimi.it
POLITECNICO MILANO 1863

Motivation

▶ We want to apply **Hyperparameter Optimization** for Policy Gradient methods in Reinforcement Learning.

▶ A dynamic learning rate may allow a **fast and stable** learning process.

▶ If the process depends on **external variables**, hyperparameters can be adapted to the context.

Contributions

1. **Meta-MDP**: Definition of an abstract class of hyperparameter decision processes called **Meta-MDP**, a framework for modeling meta learning;

2. **Bounds for Lipschitz Contextual-MDPs**: Derivation of **general guarantees** on the return with respect to the parameterization of the context under smoothness assumptions;

3. **FQI on Meta-MDP**: **Fitted Q-Iteration** algorithm [Ernst et al., 2005] on the meta-MDP, using Natural Gradient Ascent as update rule, used to choose an adaptive step size throughout the learning process.

Properties

Assumptions: Lipschitz MDPs

$$\mathcal{K}\left(P_\omega(\cdot|s,a), P_\omega(\cdot|\bar{s},\bar{a})\right) \le L_P\, d_{\mathcal{S}\times\mathcal{A}}\left((s,a),(\bar{s},\bar{a})\right),$$
$$|r_\omega(s,a) - r_\omega(\bar{s},\bar{a})| \le L_r\, d_{\mathcal{S}\times\mathcal{A}}\left((s,a),(\bar{s},\bar{a})\right).$$

Assumptions: Lipschitz Policy

$$\mathcal{K}\left(\pi(\cdot|s), \pi(\cdot|\bar{s})\right) \le L_\pi\, d_{\mathcal{S}}\left(s,\bar{s}\right),$$

- These assumptions allow the Value Function to be Lipschitz continuous with constant L_V [Pirotta et al., 2015]:

Assumptions: Context Lipschitz Continuity

$$\mathcal{K}\left(P_\omega(\cdot \mid s,a), P_{\widehat{\omega}}(\cdot \mid s,a)\right) \le L_{\omega_P} d_\Omega(\omega,\widehat{\omega})$$
$$\left|r_\omega(s,a) - r_{\widehat{\omega}}(s,a)\right| \le L_{\omega_r} d_\Omega(\omega,\widehat{\omega});$$

Results: Lipschitz Value functions/return

$$\left|Q_\omega(s,a) - Q_{\widehat{\omega}}(s,a)\right| \le L_{\omega_Q} d_\Omega(\omega,\widehat{\omega}); \qquad \left|j_\omega - j_{\widehat{\omega}}\right| \le L_{\omega_Q} d_\Omega(\omega,\widehat{\omega})$$

$$L_{\omega_Q} = \frac{L_{\omega_r} + \gamma L_{\omega_P} L_V}{1-\gamma}$$

References

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005.

Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Policy gradient in lipschitz markov decision processes. *Machine Learning*, 100(2):255–283, 2015.

Meta MDP

CMDP

Contextual Markov Decision Process [Hallak et al., 2015]:

$$\mathcal{M}(\omega) = \langle \mathcal{S}, \mathcal{A}, P_\omega, R_\omega, \gamma, \mu \rangle$$

- Transition and reward processes depend on context ω
- *Expected return* $j_\omega(\boldsymbol{\theta}) = \underset{P_\omega,\pi_{\boldsymbol{\theta}}}{\mathbb{E}}\left[\sum \gamma^t R_\omega(s_t,a_t)\right]$

Policy Gradient Update

$$\nabla_{\boldsymbol{\theta}} j_\omega(\boldsymbol{\theta}) = \mathbb{E}\left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) Q^\pi_\omega(s,a)\right],$$

- Stochastic Gradient Ascent (SGA) with step size h

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + h \nabla j_\omega(\boldsymbol{\theta}_t)$$

Meta-MDP

Meta Observation:	Task and policy info	$x_t = \langle \boldsymbol{\theta}_t, \omega, \nabla j_\omega(\boldsymbol{\theta}_t)\rangle$
Meta Action:	Hyperparameter choice	h
Meta reward:	One step Learning	$l = j_\omega(\boldsymbol{\theta}_{t+1}) - j_\omega(\boldsymbol{\theta}_t)$
Meta discount factor:		$\widetilde{\gamma}$

General framework:
▶ Update rule f
▶ Hyperparameter h

$$\boldsymbol{\theta}_{t+1} = f(\boldsymbol{\theta}_t, h)$$

$$l = j_\omega(f(\boldsymbol{\theta}_t, h)) - j_\omega(\boldsymbol{\theta}_t)$$

Dataset generation

$\mathcal{F} = \{\}$
For $k = 1, \dots, K$
  Sample context ω
  Sample initial policy θ_0
  Estimate $j_\omega(\theta_0)$, $\nabla j_\omega(\theta_0)$ in $\mathcal{M}_\omega$
  For $t = 0, \dots, T$
    Sample hyperparameter h_t;
    Update $\theta_{t+1} = \theta_t + h_t \nabla j_\omega(\theta_t)$
    Estimate $j_\omega(\theta_{t+1})$, $\nabla j_\omega(\theta_{t+1})$ in $\mathcal{M}_\omega$
    Set x_t, x_{t+1} and $l = j_\omega(\theta_{t+1}) - j_\omega(\theta_t)$
    Append (x_t, h_t, x_{t+1}, l) to $\mathcal{F}$
  end for
end for

SGA step

SGA trajectory

Double Clipped Fitted Q Iteration

Set regression algorithm f (Extra-trees);
Initialize $Q^0_1, Q^0_2, N = 0$;
While *stopping conditions not reached* **do**:

  Build training set $TS = \{(i,o)\}$ from $\mathcal{F}$ where:
    $\underline{Q}(h) = \min(Q^N_1(x_{t+1},h), Q^N_2(x_{t+1},h))$,
    $\overline{Q}(h) = \max(Q^N_1(x_{t+1},h), Q^N_2(x_{t+1},h))$,
    $i = (x_t, h_t);\ o = l + \widetilde{\gamma} \max_h \left[\lambda\, \underline{Q}(h) + (1-\lambda)\, \overline{Q}(h)\right]$;
  Split TS in TS_1, TS_2;
  Perform regressions with f on $TS_{1,2}$ to learn Q^{N+1}_1, Q^{N+1}_2.

Double FQI iteration

end while

Empirical Results