Delft University of Technology
Department of Control & Simulation

Technical Note 1.2: IDHP extensions

W. Chan

Date:	01/03/2024
Subject:	RL for Flying-V v2.0
Supervisors:	Dr. E.J. van Kampen
Project term:	01/2024 - 08/2024
E-mail:	w.y.chan@student.tudelft.nl

This document derives the augmented update rules for the IDHP algorithm to make use of more previous information during the improvement of critic or actor networks, extending the algorithm to a multi-step version and an eligibility trace version.

The IDHP algorithm is made up of 3 components:

1. **The critic**, which estimates the gradient of the cost-to-go function, almost analogous to the state-value function in the MDP framework.
2. **The actor**, which provides actions as a function of observed states.
3. **The system model**, represents the dynamics of the system, analogous to the environment dynamics in the MDP framework. It takes the form of a linear but time-varying state space model and is identified using RLS online, which is the characteristic mark of IDHP [1].

The augmentations that extend IDHP with multi-step and eligibility traces only need to be made to **the critic** and **the actor**.

1 Multi-step Extension

The multi-step extension is originally inspired by Wang et al. [2], which in turn was an improvement of and inspired by Luo et al. [3]. The extensions presented henceforth, however, borrow little from these publications and instead borrow from the n-step TD algorithms [4], using the idea of taking multiple time steps worth of observation for estimate updates and applying it to IDHP.

Augmentation to the Critic:

The original TD error for IDHP Equation 1 is augmented with a summation of discounted costs as done in Equation 2, which should have the effect of "grounding" the TD error estimate as more real signals are used to calculate the error.

$$TD = \mathcal{J}(\mathbf{x}_t) - c_t - \gamma\mathcal{J}(\mathbf{x}_{t+1}) \tag{1}$$

$$TD_n = \mathcal{J}(\mathbf{x}_t) - \gamma^n\mathcal{J}(\mathbf{x}_{t+n}) - \sum_{k=0}^{n-1}\gamma^k c_{t+k} \tag{2}$$

$$\text{E.g. when } n=2: \quad TD_2 = \mathcal{J}(\mathbf{x}_t) - c_t - \gamma c_{t+1} - \gamma^2\mathcal{J}(\mathbf{x}_{t+2})$$

$$\text{when } n=1: \quad TD_1 = \mathcal{J}(\mathbf{x}_t) - c_t - \gamma\mathcal{J}(\mathbf{x}_{t+1})$$

$$\text{Where} \quad c_t = \frac{1}{2}(\mathbf{x}_t - \mathbf{x}_{t,ref})^\top(\mathbf{x}_t - \mathbf{x}_{t,ref})$$

$$\therefore \frac{\partial c_t}{\partial \mathbf{x}_t} = (\mathbf{x}_t - \mathbf{x}_{t,ref})^\top \tag{3}$$

Subsequently, to update the parameters of the function approximator in the critic, a quadratic error function that measures the error between estimated and observed costs is formulated in Equation 4, following the example of what was done in several implementations of ADP methods [5, 6, 7]. Formulating a quadratic error function allows it to be easily differentiable, thus the weights may be updated through gradient descent. More importantly, a quadratic cost function has only one global optimum, which helps with the convergence of optimization routines.

$$E_n(t) = \frac{1}{2}\mathbf{e}_n(t)^\top \mathbf{e}_n(t) \tag{4}$$

$$\begin{aligned}
\text{Where} \quad & \mathbf{e}_n(t) = \frac{\partial [TD_n]}{\partial \mathbf{x}_t} \\
\text{E.g. when } n = 2: \quad & \mathbf{e}_2(t) = \lambda(\mathbf{x}_t) - \frac{\partial c_t}{\partial \mathbf{x}_t} - \gamma \frac{\partial c_{t+1}}{\partial \mathbf{x}_{t+1}} \frac{\partial \mathbf{x}_{t+1}}{\partial \mathbf{x}_t}\bigg|_t - \gamma^2 \lambda(\mathbf{x}_{t+2}) \frac{\partial \mathbf{x}_{t+2}}{\partial \mathbf{x}_{t+1}}\bigg|_t \frac{\partial \mathbf{x}_{t+1}}{\partial \mathbf{x}_t}\bigg|_{t+1} \\
\text{when } n = 1: \quad & \mathbf{e}_1(t) = \lambda(\mathbf{x}_t) - \frac{\partial c_t}{\partial \mathbf{x}_t} - \gamma \lambda(\mathbf{x}_{t+1}) \frac{\partial \mathbf{x}_{t+1}}{\partial \mathbf{x}_t}\bigg|_t
\end{aligned}$$

The partial derivatives of the cost c_t are calculated using Equation 3, the partial derivative of the system state $\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_{t-1}}$ is taken from the RLS identified system models, and the value function derivatives $\lambda(\mathbf{x}_t)$ are simply the critic function evaluated at previous time steps.

The critic function approximator parameters are then updated using gradient descent as shown in Equation 5.

$$\begin{aligned}
\mathbf{w}_C(t+1) &= \mathbf{w}_C(t) - \eta \frac{\partial E_n(t)}{\partial \mathbf{w}_C(t)} \qquad (5) \\
\text{Where} \quad \frac{\partial E_n(t)}{\partial \mathbf{w}_C(t)} &= \frac{\partial E_n(t)}{\partial \mathbf{e}(t)} \frac{\partial \mathbf{e}(t)}{\partial \lambda(\mathbf{x}_t)} \frac{\partial \lambda(\mathbf{x}_t)}{\partial \mathbf{w}_C(t)} \\
&= \frac{1}{2}\mathbf{e}(t)^\top \frac{\partial \lambda(\mathbf{x}_t)}{\partial \mathbf{w}_C(t)}
\end{aligned}$$

The partial derivative $\frac{\partial \lambda(\mathbf{x}_t)}{\partial \mathbf{w}_C(t)}$ can be analytically evaluated when the function approximator used for the critic is differentiable, which is the case for a neural network or multi-layer perceptron. In the critic update equation, the effect of making the algorithm multi-step comes from the error term $\mathbf{e}_c(t)$, which is constituted of a varying number of environment observations depending on how many steps n are taken in Equation 2.

Augmentation to the Actor:

A similar derivation can be done for the actor weight updates. Starting with the original loss function used to compute the accuracy of the actor Equation 6, additional observations can be added to the actors' loss function which yields Equation 7.

$$L(t) = c_t + \gamma \mathcal{J}(\mathbf{x}_{t+1}) \tag{6}$$

$$L_n(t) = \gamma^n \mathcal{J}(\mathbf{x}_{t+n}) + \sum_{k=0}^{n-1} \gamma^k c_{t+k} \tag{7}$$

$$\text{E.g. when } n = 2: \quad L_2(t) = c_t + \gamma c_{t+1} + \gamma^2 \mathcal{J}(\mathbf{x}_{t+2})$$

The multi-step loss function Equation 7 is then used in formulating the gradient descent step used for optimizing the actor to push it towards minimizing the loss function Equation 8.

$$\mathbf{w}_A(t+1) = \mathbf{w}_A(t) - \eta \frac{\partial L_n(t)}{\partial \mathbf{w}_A(t)} \tag{8}$$

$$\text{Where} \quad \frac{\partial L_n(t)}{\partial \mathbf{w}_A(t)} = \frac{\partial L_n(t)}{\partial \mathbf{x}(t+1)} \frac{\partial \mathbf{x}(t+1)}{\partial \mathbf{a}(t)} \frac{\partial \mathbf{a}(t)}{\partial \mathbf{w}_A(t)}$$

$$\text{E.g. when } n=2: \quad \frac{\partial L_2(t)}{\partial \mathbf{w}_A(t)} = \underbrace{\left[\frac{\partial c_t}{\partial \mathbf{x}_{t+1}} + \gamma \frac{\partial c_{t+1}}{\partial \mathbf{x}_{t+1}} + \gamma^2 \lambda(\mathbf{x}_{t+2}) \frac{\partial \mathbf{x}_{t+2}}{\partial \mathbf{x}_{t+1}} \bigg|_{t+1} \right]}_{\text{partial of } L_2(t) \text{ w.r.t. } \mathbf{x}(t+1)} \frac{\partial \mathbf{x}(t+1)}{\partial \mathbf{a}(t)} \bigg|_t \frac{\partial \pi(t, \mathbf{w}_A(t))}{\partial \mathbf{w}_A(t)}$$

$$\text{when } n=1: \quad \frac{\partial L_1(t)}{\partial \mathbf{w}_A(t)} = \underbrace{\left[\frac{\partial c_t}{\partial \mathbf{x}_{t+1}} + \gamma \lambda(\mathbf{x}_{t+1}) \right]}_{\text{partial of } L_1(t) \text{ w.r.t. } \mathbf{x}(t+1)} \frac{\partial \mathbf{x}(t+1)}{\partial \mathbf{a}(t)} \bigg|_t \frac{\partial \pi(t, \mathbf{w}_A(t))}{\partial \mathbf{w}_A(t)}$$

A special note needs to be made regarding the term $\frac{\partial c_t}{\partial \mathbf{x}_{t+1}}$ from the last 2 lines, in Casper's and Stefan's work this term is defined to be Equation 3, whereas in JunHyeon's work this is defined to be 0. Thus effectively a choice could be made on how to evaluate this derivative, with Stefan's implementation showing perfect tracking performance, whereas both Casper and JunHyeon had slightly diminished tracking performance.

2 Eligibility Trace Extension

The idea of extending the IDHP algorithm with eligibility traces came naturally from trying to extend it with multi-step observations, as they result in the same performance effects and serve the very similar purpose of improving convergence/sample efficiency [4].

Both critic and actor updates will be augmented in the same manner. Whereas the original parameter update equations follow the form of Equation 9, where the original parameters are added with a term comprising of the learning rate η, some form of target δ, and the gradient of the function approximator $\frac{\partial f(t)}{\partial \mathbf{w}(t)}$. In the case of the critic and the actor, this function approximator is denoted as λ and π respectively. Instead of using the function approximator gradient, an eligibility trace is used in the update term. This eligibility trace term is defined in Equation 11, where a new parameter λ is introduced which controls the decay rate of the eligibility trace, γ is the same cost discount rate constant as used in the previous equations.

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \eta\delta \frac{\partial f(t)}{\partial \mathbf{w}(t)} \tag{9}$$

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \eta\delta E(t) \tag{10}$$

$$\text{Where} \quad E(t) = \lambda\gamma E(t-1) + \frac{\partial f(t)}{\partial \mathbf{w}(t)} \tag{11}$$

Eligibility traces can be thought of as moving average or low-pass filters of the approximator gradient, which continue to extend one gradient update step over time but with decaying magnitudes. It is noted that the eligibility trace extension is a lot simpler than the multi-step augmentations, both in their derivation and implementation.

Eligibility traces seem to be more commonly applied to only updates of value function estimates, i.e. for critic updates [7, 4, 8, 9]; however, a case of eligibility traces being used for critic and actor updates seems to also exist [10]. Thus to be conservative, it would be wise to augment the critic update with eligibility traces first, and test out augmenting actor update thereafter.

References

[1] Ye Zhou, Erik-Jan Van Kampen, and Q.P. Chu. "Incremental model based online dual heuristic programming for nonlinear adaptive control". In: *Control Engineering Practice* 73 (2018), pp. 13–25. DOI: 10.1016/j.conengprac.2017.12.011.

[2] Ding Wang, Jiangyu Wang, Mingming Zhao, Peng Xin, and Junfei Qiao. "Adaptive Multi-Step Evaluation Design With Stability Guarantee for Discrete-Time Optimal Learning Control". In: *IEEE/CAA Journal of Automatica Sinica* 10.9 (2023), pp. 1797–1809. DOI: 10.1109/JAS.2023.123684.

[3] Biao Luo, Derong Liu, Tingwen Huang, Xiong Yang, and Hongwen Ma. "Multi-step heuristic dynamic programming for optimal control of nonlinear discrete-time systems". In: *Information Sciences* 411 (2017), pp. 66–83. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2017.05.005.

[4] R. Sutton and A. Barto. *Reinforcement Learning, An Introduction*. 2nd ed. Cambridge, Massachusetts: The MIT Press, 2018.

[5] Ganesh Venayagamoorthy, R.G. Harley, and Donald Wunsch. "Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neurocontrol of a turbogenerator". In: *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council* 13 (2002), pp. 764–73. DOI: 10.1109/TNN.2002.1000146.

[6] Tao Li, Dongbin Zhao, and Jianqiang Yi. "Heuristic Dynamic Programming strategy with eligibility traces". In: *2008 American Control Conference*. 2008, pp. 4535–4540. DOI: 10.1109/ACC.2008.4587210.

[7] Jun Ye, Yougang Bian, Biao Xu, Zhaobo Qin, and Manjiang Hu. "Online Optimal Control of Discrete-Time Systems Based on Globalized Dual Heuristic Programming with Eligibility Traces". In: *2021 3rd International Conference on Industrial Artificial Intelligence (IAI)*. 2021, pp. 1–6. DOI: 10.1109/IAI53119.2021.9619346.

[8] Jun Rao, Jingcheng Wang, Jiahui Xu, and Shangwei Zhao. "Optimal control of nonlinear system based on deterministic policy gradient with eligibility traces". In: *Nonlinear Dynamics* 111.21 (2023), pp. 20041–20053. DOI: 10.1007/s11071-023-08909-6.

[9] Simone Baldi, Zichen Zhang, and Di Liu. "Eligibility traces and forgetting factor in recursive least-squares-based temporal difference". In: *International Journal of Adaptive Control and Signal Processing* 36.2 (2022), pp. 334–353. DOI: https://doi.org/10.1002/acs.3282.

[10] Taisuke Kobayashi. "Adaptive and multiple time-scale eligibility traces for online deep reinforcement learning". In: *Robotics and Autonomous Systems* 151 (2022), p. 104019. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2021.104019.