V46.

Bingheng Wang

W. Jin et al "Pontryagin Differentiable Programming: An End-to-end Learning and Control Framework" NeurIPS, 2020.

02 - Sept - 2023.

* Discussion of Why we cannot solve NeuroMHE using PDP method. *

This discussion arise from a concern of why we set $\lambda_T^* = 0$ by definition.
Can we follow the steps / computation methods presented in PDP paper to solve our DMHE?

- To match the convention in PDP, here we adopt: $\lambda'_{t+1}(f_t - x_{t+1})$

  System dynamics at time-step t.

  By contrast, in our formulation, $\lambda'_t(x_{t+1} - f_t)$

  This is the main difference.

  So, the index of λ is different.

Ours	PDP
λ_t (t: $0 \sim n-1$)	λ_t: (t: $1 \sim n$) n: horizon.

Hamiltonian: $H_t = C_t + f'_t \cdot \lambda_{t+1}$ $(0<t<T)$ $\quad C_t = C_t(x_t, w_t) \quad h_0 = h_0(x_0), \; h_T = h_T(x_T)$

Lagrangian: $L = h_0 + \sum_{t=0}^{T-1} C_t + h_T + \sum_{t=0}^{T-1} \lambda'_{t+1}(f_t - x_{t+1})$

First-order optimality conditions: $\frac{\partial L}{\partial \lambda_{1:T}} = f_t - x_{t+1} = 0$ (system dynamic model)

Note that x_t $(0<t<T)$ appears twice in L: $-\lambda'_t \cdot x_t$ & $\lambda'_{t+1} f_t(x_t, w_t)$. However, x_0 and x_T only appear once!

$$\frac{\partial L}{\partial x_{1:T-1}} = \frac{\partial C_t}{\partial x_t} + \frac{\partial f'_t}{\partial x_t} \cdot \lambda_{t+1} - \lambda_t = 0$$

$$\frac{\partial L}{\partial w_{0:T-1}} = \frac{\partial C_t}{\partial w_t} + \frac{\partial f'_t}{\partial w_t} \cdot \lambda_{t+1} = 0.$$

2 boundary conditions
$$\begin{cases} \frac{\partial L}{\partial x_0} = \frac{\partial h_0}{\partial x_0} + \frac{\partial C_0}{\partial x_0} + \frac{\partial f'_0}{\partial x_0} \cdot \lambda_1 = 0 & (b1) \\ \frac{\partial L}{\partial x_T} = \frac{\partial h_T}{\partial x_T} - \lambda_T = 0. & (b2) \end{cases}$$

* in PDP, $\frac{\partial L}{\partial x_0}$ is not needed as x_0^* should be x_0! ($x_0^* = x_0$ in MPC) That is why they only need to deal with λ_T

- By contrast, in our formulation, we only need to deal with the initial boundary condition (b1) as $\lambda_T^* = 0$ by definition.

In PDP, differentiating (b2) in both sides w.r.t the parameters θ yields the following condition:

$$\frac{\partial \lambda_T^*}{\partial \theta} = \Lambda_T^* = \frac{\partial^2 h_T}{\partial x_T^2} \cdot \underbrace{\frac{\partial x_T}{\partial \theta}}_{X_T} + \frac{\partial^2 h_T}{\partial x_T \partial \theta} = H_T^{xx} \cdot X_T + H_T^{x\theta} \qquad (d1)$$

Based on (d1), the authors in PDP paper assume that a general form of Λ_t^* $(1 \le t \le T)$ satisfies:

$$\Lambda_t^* = P_t \cdot X_t + W_t \qquad (g1)$$

$P_t := Q_t + A'_t(I + P_{t+1}R_t)^{-1} P_{t+1} A_t \quad W_t := A'_t(I + P_{t+1}R_t)^{-1}(W_{t+1} + P_{t+1}M_t) + N_t \quad \frac{\partial f}{\partial x_t} = F \quad \frac{\partial f}{\partial w_t} = G$

$R_t = G_t (H_t^{ww})^{-1} G'_t$; $Q_t = H_t^{xx} - H_t^{xw}(H_t^{ww})^{-1} H_t^{wx}$, $N_t = H_t^{x\theta} - H_t^{xw}(H_t^{ww})^{-1} H_t^{w\theta}$, $A_t = F_t - G_t (H_t^{ww})^{-1} H_t^{wx}$

Differentiating (b1) in both sides w.r.t θ and plugging $\frac{\partial W_0}{\partial \theta}$ into the resulting derivative yield:

(g2) $$\Lambda_1 = \frac{\partial \lambda_1}{\partial \theta} = -[F_0 - L_0^{xw}(L_0^{ww})^{-1}G_0]^{-1}[L_0^{xx} - L_0^{xw}(L_0^{ww})^{-1}L_0^{wx}] X_0 - [F_0 - L_0^{xw}(L_0^{ww})^{-1}G_0]^{-1}[L_0^{x\theta} - L_0^{xw}(L_0^{ww})^{-1}L_0 \ldots$$

(g1) and (g2) share a similar structure, but the coefficient matrices are different.

Another difference is that in PDP Λ_t^* is expressed in (g1) while it is recursively calculated backward in time in our method where a general form of Λ_t^* may not take the same structure of (g1).

The fundamental reason behind these differences is that (b1) (and its derivative (g1)) is explicitly used in our method to compute the initial value of X_t (i.e., $\frac{\partial x_0}{\partial \theta}$). However, in PDP $\frac{\partial x_0}{\partial \theta} = 0$!!!

Bingheng Wang

Kalman Filter.

Fundamental Comparison With LQR

	Kalman Filter	LQR
Riccati Recursion	$P'_{k+1} = \Phi[I - P'_k H^T (H P'_k H^T + R)^{-1} H] P'_k \Phi^T + Q$ Forward in time * In MHE P_0 is ~~the~~ related to the weighting matrix in the arrival cost!	$P'_k = (\Phi - BK_k)^T P'_{k+1} (\Phi - BK_k) + Q + K_k^T \cdot R \cdot K_k$ Backward in time * In LQR, P_T or (P_N) is related to the terminal cost

	MHE	LQR
Dynamic Programming	$\underset{(x_{n+1})}{V_{n+1}} = \min_{x_n} \{ V_n(x_n) + J_{n+1}(x_n) \}$ V_{n+1} is obtained by minimizing over x_n and becomes a function of x_{n+1} after optimization for the next computation at $n+1$ Forward in time	$\underset{(u_{i-1})}{V_i} = \min_{u_i} \{ J_i(u_i) + V_{i+1}(u_i) \}$ V_i is obtained by minimizing over u_i and becomes a function of u_{i-1} after optimization for the next computation at $i-1$ Backward in time.

* Detailed formulation of $V_n(x_n)$, refer to Page 20 of this Notebook.