Question 1:

Part A)

Used bvp4c [1], with support from an example from [2].

Part B)

Used [3] as support for solving the HJB equation.

Part C)

The results from part A and B can be compared to satisfy the theory. The open-loop solution solved numerically by the two-point boundary-value problem in part A and the closed-loop solution, solved by integrating the differential equation backwards in part B, are visualized in Figures 1-3.

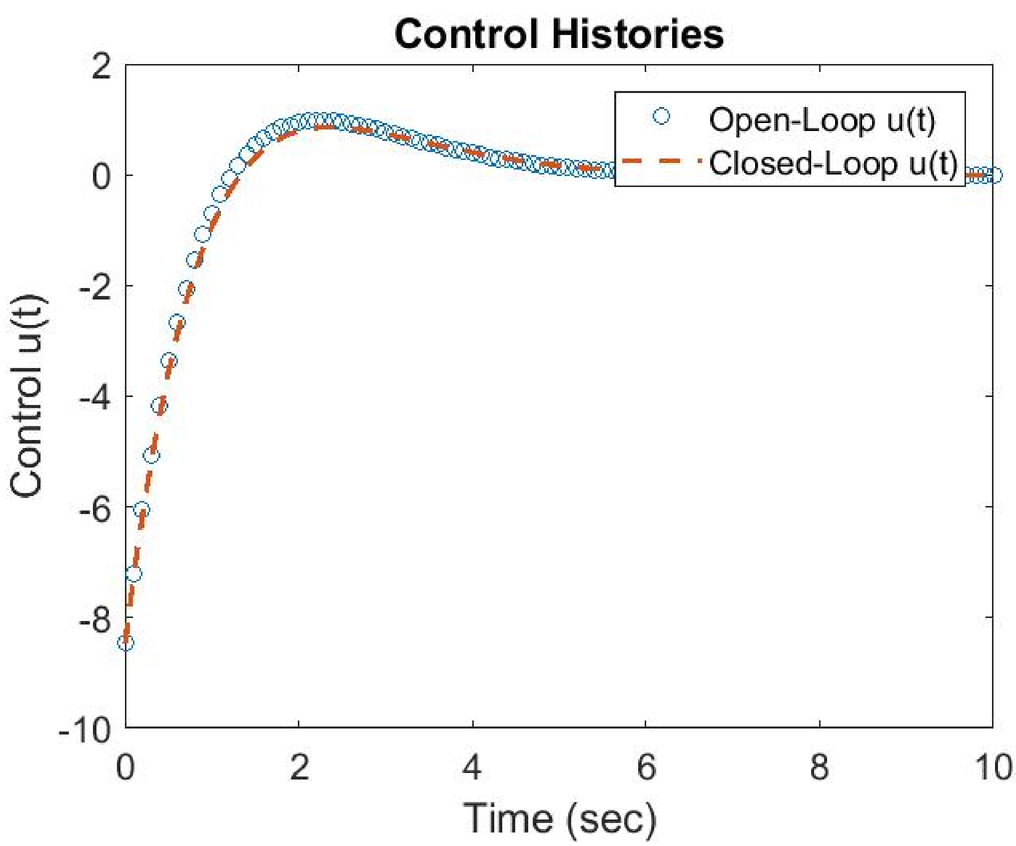


Figure 1: Control Histories of Optimal Open- and Closed- Loop u(t)

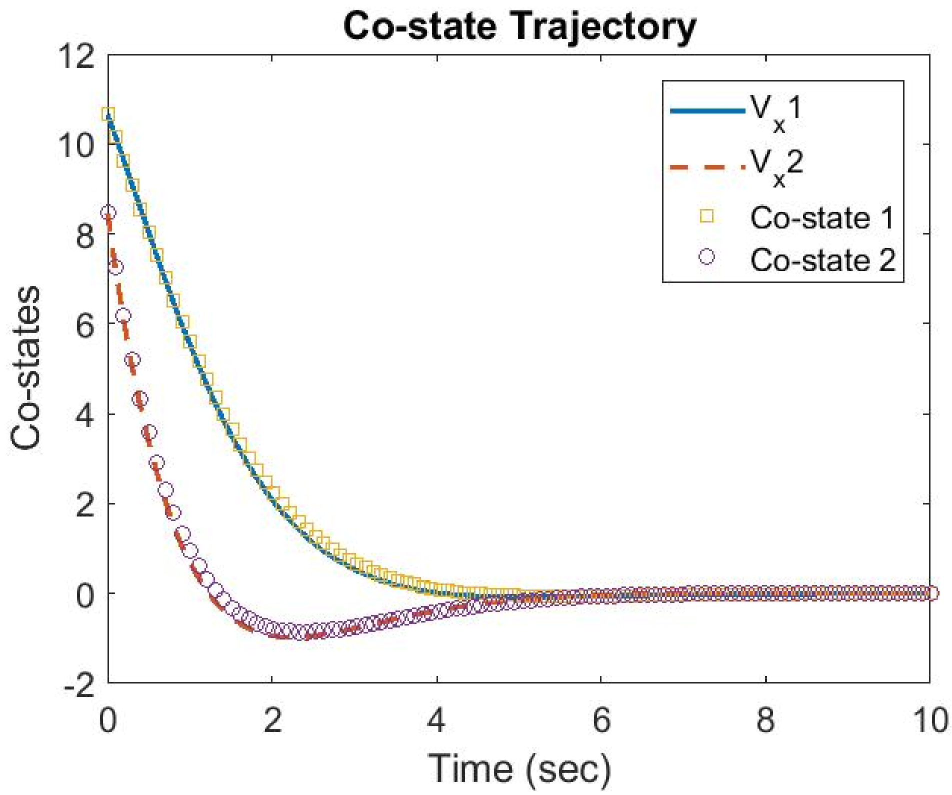


Figure 2: Co-state Trajectories of Optimal Open- and Closed-Loop Solutions

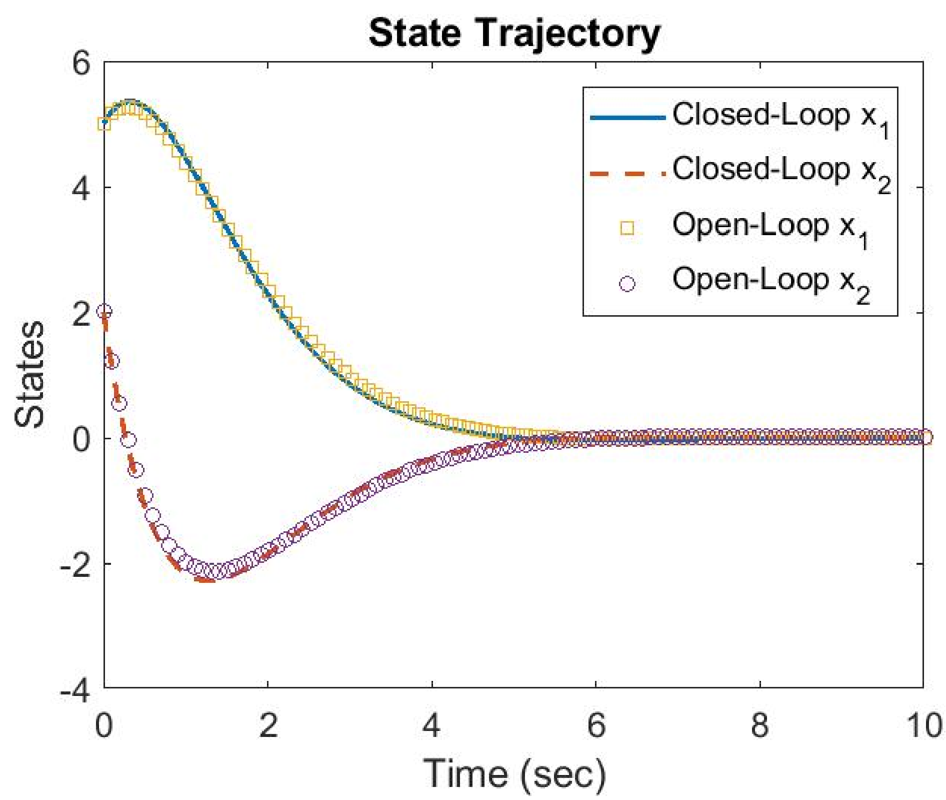


Figure 3: State Trajectories of Optimal Open- and Closed-Loop Solutions

Question 2

(SEE HANDWRITTEN RESULTS)

Question 3

Question 3 involved completing the discrete DDP code for the cartpole problem. In order to complete the code, the section for backpropagation the value function needed to be filled. The completed code is shown here:

```
% compute feedforward control (l_k) and feedback gains matrix (L_k)
 Q_uu(:,:,j) = B(:,:,j)'*Vxx(:,:,j+1)*B(:,:,j) + Luu(:,:,j); %
 Q_u(:,j) = Vx(:,j+1)'*B(:,:,j) + Lu(:,j); %
 Q_xu(:,:,j) = A(:,:,j)'*Vxx(:,:,j+1)*B(:,:,j)+Lux(:,:,j)'; %
 Q_ux(:,:,j) = Q_xu(:,:,j)'; %
 Q_x(:,j) = Vx(:,j+1)'*A(:,:,j)+Lx(:,j)'; %
 Q_xx(:,:,j) = A(:,:,j)'*Vxx(:,:,j+1)*A(:,:,j)+Lxx(:,:,j); %

 l_k(:,j) = -inv(Q_uu(:,:,j))*Q_u(:,j); %
 L_k(:,:,j)= -inv(Q_uu(:,:,j))*Q_ux(:,:,j); %

% compute value function and its first and second derivatives
V(j) = V(j+1) +  …
l_k(:,j)'*Q_u(:,j)+(1/2)*l_k(:,j)'*Q_uu(:,:,j)*l_k(:,j); %
Vx(:,j) = Q_x(:,j) + L_k(:,:,j)'*Q_u(:,j) + Q_xu(:,:,j)*l_k(:,j) + …
L_k(:,:,j)'*Q_uu(:,:,j)*l_k(:,j); %
Vxx(:,:,j) = Q_xx(:,:,j) + …
L_k(:,:,j)'*Q_ux(:,:,j)+Q_xu(:,:,j)*L_k(:,:,j)+L_k(:,:,j)'*Q_uu(:,:,
j)*L_k(:,:,j); %
```

The default problem parameters were not changed and are shown as follows:

Initial Configuration: [0 0 π 0.2]

Target Configuration: [3 0 0 0]

In order to get an appropriate solution some parameters had to be changed as shown in Table 1.

Table 1: Modified Parameters in DDP Code

Parameter	Current Value
# of Iteratations	300
Final State Weights	`10*diag([100 10 50 10])`
State Weights	`diag([10 10 10 10]);`
Learning Rate	0.1

The updated parameters and completed code led to a successful cartpole maneuver as shown by the last image of the animation shown in Figure 1. The complete animation was uploaded to canvas.

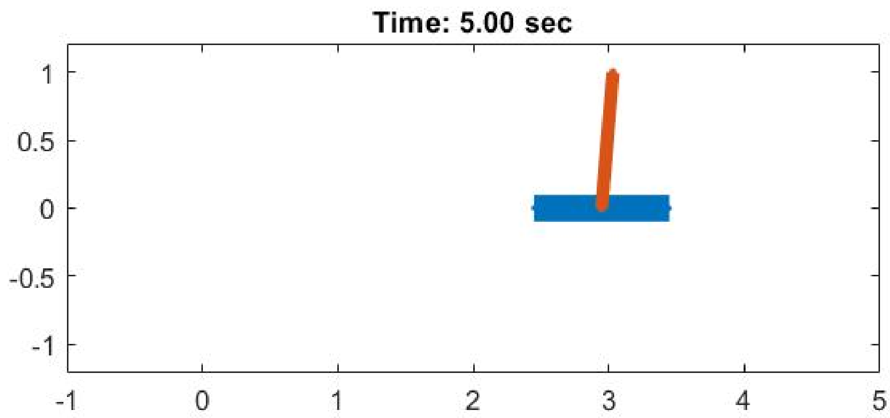


Figure 4: Final Image from DDP Cartpole Animation

The results for the state trajectories, cost, and controller gains are shown in Figures 5-7.

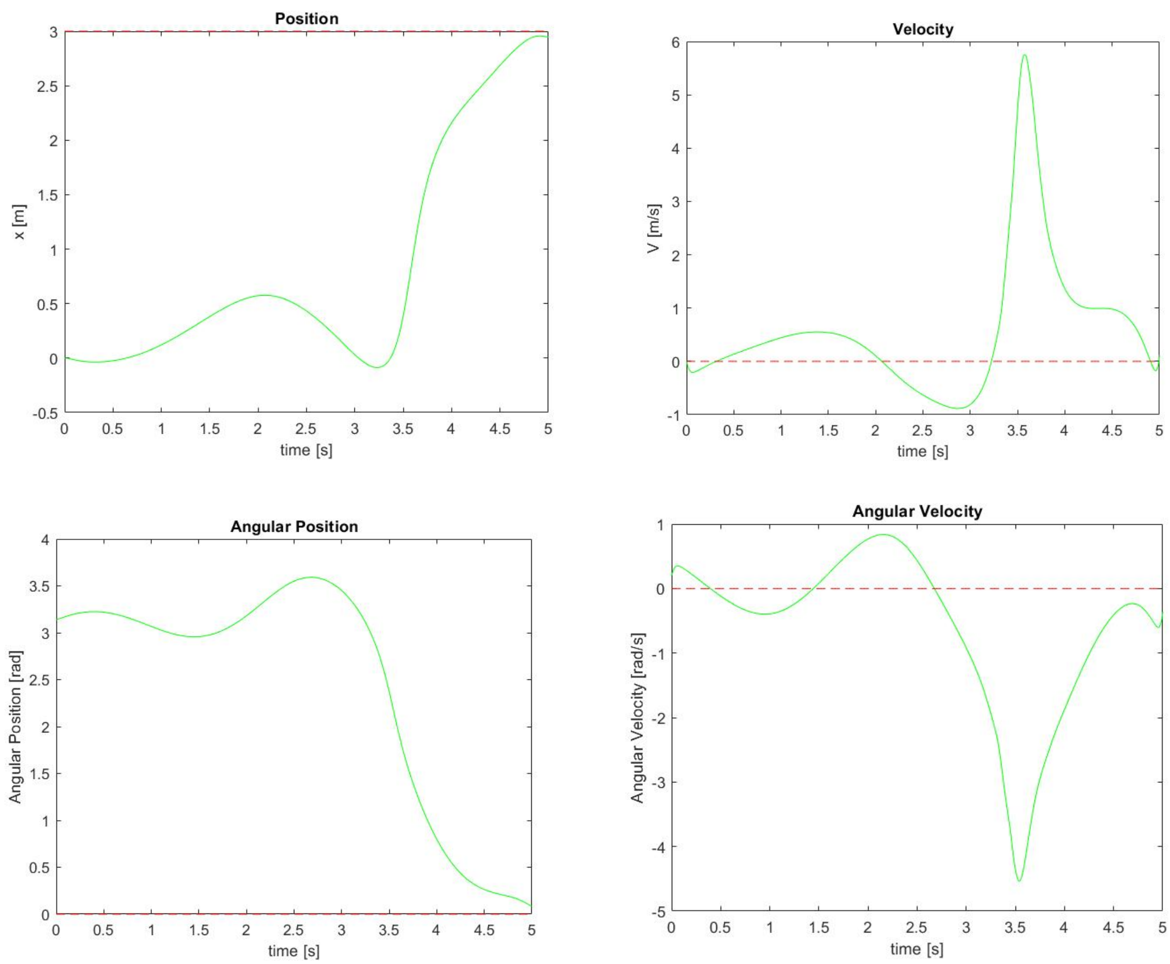
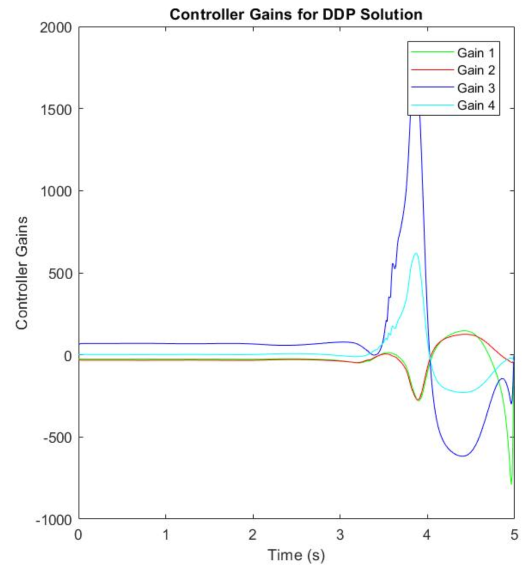


Figure 5: State trajectories and target state of DDP solution



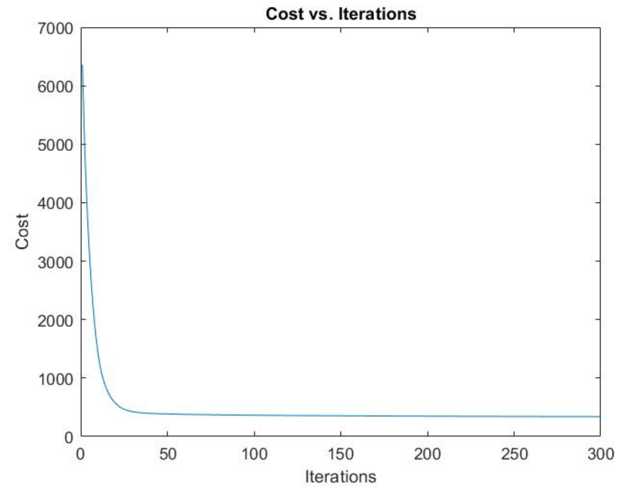


Figure 6: Controller gains and the cost for DDP solution

Homework #1 | AE 8803 TSI | Fall 2019

① Given: Continuous-time, linear-quadratic problem.

$$\dot{x} = F(x(t), u(t), t) = Ax(t) + Bu(t) \qquad t \in [0, t_F]$$

$$J(x_0; u(\cdot)) = \frac{1}{2} x^T(t_F) S x(t_F) + \frac{1}{2} \int_0^{t_F} [x^T(t) Q x(t) + u^T R u(t)] dt$$

with initial conditions: ~~[illegible]~~, $x(0) = [5, 2]^T$ and $t_F = 10$

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1, \quad S = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$

---

a) Show that the optimal open-loop control problem is given by:

$$u(t) = -R^{-1} B^T \lambda(t)$$

This is shown by the Necessary Conditions for a local minimum, included in the four conditions of Pontryagin's Maximum Principle.

Define Hamiltonian

$$H(x(t), u(t), \lambda(t), t) = L(x(t), u(t), t) + \lambda^T(t) F(x(t), u(t), t)$$

$$\phi(x(t_F), t_F) = \frac{1}{2} x^T(t_F) S x(t_F)$$

$$L(x(t), u(t), t) = \frac{1}{2} x^T(t) Q x(t) + u^T(t) R u(t)$$

$$F(x(t), u(t), t) = Ax(t) + Bu(t) \Rightarrow \dot{x} + Ax + Bu = 0$$

Necessary Conditions:

1) $\dot{\lambda}^T(t) = -\partial H / \partial x$

2) $\partial H / \partial u = 0$

3) $\partial \phi / \partial x (x(t_F), t_F) = \lambda^T(t_F)$

4) $\partial \phi / \partial t (x(t_F), t_F) = -H(x(t_F), u(t_F), \lambda(t_F), t_F)$

We need condition 2:

$$\partial H / \partial u = \frac{1}{2}[2Ru(t)] + B^T \lambda(t)$$

$$0 = Ru(t) + B^T \lambda(t)$$

$$\underline{u(t) = -R^{-1} B^T \lambda(t)}$$

To solve the two-point boundary-value problem in the (x, λ) space we solve the rest of the necessary conditions:

Condition 1: $\dot{\lambda}^T(t) = -\partial H / \partial x$

$$\partial H / \partial x = \frac{1}{2}[2Qx(t)] + A^T \lambda(t) = -\dot{\lambda}^T(t)$$

$$\underline{\dot{\lambda}(t) = -A^T \lambda(t) - Qx(t)}$$

Condition 3: $\partial \phi / \partial x (x(t_f), t_f) = \lambda^T(t_f)$

$$\partial \phi / \partial x (x(t_f), t_f) = \frac{1}{2}(2) S x(t_f) = S x(t_f)$$

$$\underline{\lambda(t_f) = S x(t_f)}$$

Condition 4: t_f known, not free

Plug in optimal control: ~~$\dot{x}(t) =$~~ $Ax(t) + Bu(t)$

$$\dot{x}(t) = Ax(t) + -BR^{-1}B^T\lambda(t)$$

We can now form the combined system of $x(t)$ + $\lambda(t)$ into the Hamiltonian matrix:

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \underbrace{\begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix}}_{H} \begin{bmatrix} x \\ \lambda \end{bmatrix}$$

$$x(0) = x_0$$
$$\lambda(t_f) = S x(t_f)$$

This results in a two-point boundary-value problem which can be solved using bvp4c in MATLAB.

b) Solve the Hamilton-Jacobi-Bellman equation assuming: $V(t,x) = \frac{1}{2} x^T P(t)\, x$

The HJB equation is defined as:

$$\frac{\partial J^*(x,t)}{\partial t} + \underbrace{H\left(x, u^*\left(x, \frac{\partial J^*}{\partial x}, t\right), \frac{\partial J^*}{\partial x}, t\right)}_{} = 0$$

$$= \min_u H\left(x, u, \frac{\partial J^*}{\partial x}, t\right) = L(x,u,t) + \frac{\partial J^*}{\partial x}(x,t)\, f(x,u,t)$$

Define $J^* = V = \frac{1}{2} x^T P(t)\, x$

First we can find optimal feedback control:

$$\min_u H = \left[\tfrac{1}{2} x^T Q x + \tfrac{1}{2} u^T R u + \partial/\partial x\left(\tfrac{1}{2} x^T P(t)\, x\right)(Ax + Bu)\right]$$

$$= \tfrac{1}{2} x^T Q x + \tfrac{1}{2} u^T R u + (P(t)\, x)(Ax + Bu)$$

$$\min_u \Rightarrow \partial H/\partial u = 0 \Rightarrow Ru + P(t)\, x\, B$$

$$Ru = -B^T P(t)\, x$$

$$\underline{u^* = -R^{-1} B^T P(t)\, x}$$

Now, the full HJB is solved by substituting u^*

$$\frac{\partial J^*}{\partial t} = \frac{\partial}{\partial t}\left(\tfrac{1}{2} x^T P(t)\, x\right) = \tfrac{1}{2} x^T \dot{P}(t)\, x$$

$$H\left(x, u^*, \frac{\partial J^*}{\partial x}, t\right) = \tfrac{1}{2} x^T Q x + x^T P(t) A x(t)$$
$$+ \tfrac{1}{2}\left(-R^{-1} B^T P(t)\, x\right)^T R \left(-R^{-1} B^T P(t)\, x\right)$$
$$- x^T P(t) B R^{-1} B^T P(t)\, x$$

HJB
$$0 = \tfrac{1}{2} x^T \left[\dot{P}(t) + P(t) A + A^T P(t) - P(t) B R^{-1} B^T P(t) + Q\right] x$$

Therefore, in-order for $V(t,x) = \frac{1}{2} x^T P(t)\, x$ to solve HJB, $P(t)$ must satisfy the following matrix differential equation, known as the continuous-time Ricatti equation:

$$\dot{P}(t) = -P(t) A - A^T P(t) + P(t) B R^{-1} B^T P(t) - Q$$

with boundary condition: $P(t_f) = Q_f$

(2) Part of proofs expressing Principle of Optimality (PoP)

a) Continuous

The cost function is defined as:

$$J(x_0, t_0 ; u(\tau)_{t_0 \le \tau \le t_f}) = \Phi(x(t_f), t_f) + \int_{t_0}^{t_f} L(x(\tau), u(\tau), \tau) d\tau$$

We express the PoP when rewriting as:

$$J(x, t ; u(\tau)_{t \le \tau \le t_f}) = \Phi(x(t_f), t_f) + \int_{t}^{t_f} L(x(\tau), u(\tau), \tau) d\tau$$

Represented as such:

(optimal) J^*: "cost-to go"

b) Discrete

Implementing using dynamic programming for discrete case:

$$J = \underbrace{\Phi(x_N)}_{\text{Terminal cost}} + \underbrace{\sum_{k=0}^{N-1} L_k(x_k, u_k)}_{\text{Running cost}}$$

The optimal cost J^* is the min. of J

$$J^*(x_0) = \min_u J(x_0 ; u) = \min_u \left[\Phi(x_N) + \sum_{k=0}^{N-1} L_k(x_k, u_k) \right]$$

PoP is included now, as we introduce a "cost-to-go"

$$J_k^*(x) = V(x, k) = \min_u \left(\Phi(x_N) + \sum_{j=k}^{N-1} L_j(x_j, u_j) \right)$$

The method of forming the HJB equation follows a similar pattern for both continuous + discrete when the PoP is used.

References:

1. "bvp4c Matlab Reference," Mathworks, 2019, https://www.mathworks.com/help/matlab/ref/bvp4c.html.
2. "Principles of Optimal Control: Lecture 7," MIT OpenCourseWare, Spring 2008, url: https://ocw.mit.edu/courses/aeronautics-and-astronautics/16-323-principles-of-optimal-control-spring-2008/lecture-notes/lec7.pdf.
3. Bertsekas, Dimitri, **Dynamic Programming and Optimal Control**, Volume 1, 4th Edition, 2017.