Robotics
Exercise 5

Lecturer: Jim Mainprice
TAs: Philipp Kratzer, Janik Hager, Yoojin Oh
Machine Learning & Robotics lab, U Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

June 10, 2021

1 Riccati equation in discrete time (5 points)

$V(x) = \min_u \left[C(x,u) + V(f(x,u)) \right]$

Consider the time discrete linear quadratic system

$$\overset{n\times 1}{f(x_t,u_t)} = \overset{n\times n \; n\times 1}{Ax_t} + \overset{n\times r \; r\times 1}{Bu_t}$$

$$c(x_t,u_t) = x_t^\top Q x_t + u_t^\top R u_t$$

$x^T P x = \min_u \left[x_t^T Q x_t + u_t^T R u_t + f^T P f \right]$

$\frac{d}{du} \quad 0 \overset{!}{=} 2u_t^T R + 2 f^T P B$

$= 2u_t^{*T} R + 2(Ax_t + Bu_t)^T P B$

$-2x_t^T A^T P B = 2u_t^{*T} R + 2u_t^{*T} B^T P B$

$u_t^* = -(R + B^T P B)^{-T} B^T P A x_t$

with the cost function

$$J^\pi = \sum_{t=0}^{\infty} c(x_t,u_t) \ .$$

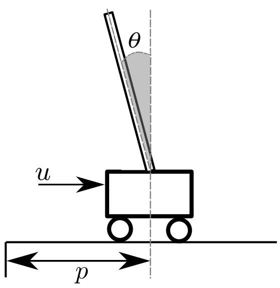
The Bellman equation (slide 04:15) for this infinite-horizon discrete time system is

$$V(x) = \min_u \left[c(x,u) + V(f(x,u)) \right] \ .$$

Start with the Bellman equation and derive the Riccati equation for the system. Similar to the continuous case, you can assume a value function of the form $V(x) = x^\top P x$ with a symmetric matrix P.

pls. see below.

2 Cart Pole Control (7 points)



In the last exercise we calculated the local linearization of the cart-pole around $x^* = (0,0,0,0)$. The solution is

$$\dot{x} = Ax + Bu \ , \quad A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{c_2 g}{\frac{4}{3}l - c_2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{g}{\frac{4}{3}l - c_2} & 0 \end{pmatrix} , \quad B = \begin{pmatrix} 0 \\ c_1 + \frac{c_1 c_2}{\frac{4}{3}l - c_2} \\ 0 \\ \frac{-c_1}{\frac{4}{3}l - c_2} \end{pmatrix}$$

with $g = 9.8ms^2$ the gravitational constant, $l = 1m$ the pendulum length and constants $c_1 = (M_p + M_c)^{-1}$ and $c_2 = lM_p(M_p + M_c)^{-1}$ where $M_p = M_c = 1kg$ are the pendulum and cart masses respectively.

a) We assume a stationary infinite-horizon cost function of the form

$$J^\pi = \int_0^\infty c(x(t),u(t)) \, dt$$

$$c(x,u) = x^\top Q x + u^\top R u$$
$$Q = \text{diag}(c, 0, 1, 0)\ , \quad R = \mathbf{I}\ . \qquad \begin{pmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{pmatrix}$$

That is, we penalize position offset $c\|p\|^2$ and pole angle offset $\varrho\|\theta\|^2$. Choose $c = \varrho = 1$ to start with.

Solve the Algebraic Riccati equation

$$0 = A^\top P + P^\top A - PBR^{\text{-}1}B^\top P + Q$$

by initializing $P = Q$ and iterating using the following iteration:

$$P_{k+1} = P_k + \epsilon[A^\top P_k + P_k^\top A - P_k B R^{\text{-}1} B^\top P_k + Q]$$

Choose $\epsilon = 1/1000$ and iterate until convergence (at least 10.000 iterations). Output the gains $K = -R^{\text{-}1}B^\top P$. (3 P)

b) Solve the same Algebraic Riccati equation analytically using some math library (python, octave, matlab, ...). For Python, install `scipy` (using *'python3 -m pip install scipy --user'*), use `from scipy import linalg` to import scipy in the python script and use `P=linalg.solve_continuous_are(A,B,Q,R)` to solve the ARE. (2 P)

(The solution is $K = (1.0000, 2.6088, 52.9484, 16.5952)$.)

c) Implement the optimal Linear Quadratic Regulator $u^* = Kx$ on the cart pole simulator in the function testMove().

1. For python please install pygame and pyopengl (using *'python3 -m pip install pygame pyopengl --user'*), then you can run: *'jupyter-notebook course1_Lectures/04-riccati/riccati.ipynb'*
2. For C++ run: *'cd course1_Lectures/04-riccati'*, *'make'*, *'./x.exe'*

Simulate the optimal LQR and test it for various noise levels (by changing the variable *dynamicsNoise*). (2 P)

Discrete. infinite Riccati

$$V(x) = \min_u [C(x,u) + V(f(x,u))]$$

$$x^T P x = \min_u [x^T Q x + u^T R u + f^T P f]$$

$$\frac{d}{du} \quad 0 \overset{!}{=} 2u^T R + 2 f^T P B$$
$$= 2u^{*T} R + 2(Ax + Bu)^T P B$$
$$-2x^T A^T P B = 2u^{*T} R + 2u^{*T} B^T P B$$
$$u^* = -(R + B^T P B)^{-1} B^T P A x$$

take $k = (R + B^T P B)^{-1}$, $z = B^T P A$. $u = -kzx$

$$x^T P x = x^T Q x + x^T z^T k^T R k z x + (Ax + Bu)^T P (Ax + Bu)$$
$$= x^T Q x + x^T z^T k^T R k z x + x^T A^T P A x - x^T z^T k z x$$
$$- x^T z^T B^T P A x + x^T z^T k^T (B^T P B) k z x$$

$$u^T \overset{z}{B^T P A} x$$
$$x^T \overset{z^T}{A^T P B} u$$

$$x^T P x = x^T \Big(Q + A^T P A + z^T k^T \overset{k^{-1}}{(R + B^T P B)} k z - z^T k^T z - z^T k^T z \Big) x$$
$$= x^T \Big(Q + A^T P A - z^T k^T z \Big) x$$
$$= x^T \Big(Q + A^T \big[P - P B (R + B^T P B)^{-1} B^T P \big] A \Big) x$$

$$\therefore \ P = Q + A^T [P - PB(R + B^T P B)^{-1} B^T P] A$$