Matlab/Simulink assignment: Grid-based quantization of dynamical systems

Preliminaries

Consider a nonlinear, deterministic, discrete-time system:

$$x(k+1) = f\big(x(k), u(k)\big) \tag{1}$$

where $x \in X$ (the state space), and $u \in U$ (the action space). For simplicity let X be an n-dimensional hyperbox centered on 0, so that x is an n-dimensional vector with:

$$x_d \in [-x_{d,\max}, x_{d,\max}] \tag{2}$$

for the dimension index $d = 1, \ldots, n$, where the vector $x_{\max} \in \mathbb{R}^n$ describes the state space boundary. Similarly, let U be an m-dimensional hyperbox centered on 0, with the bounds given by $u_{\max} \in \mathbb{R}^m$. We will use c to denote action dimensions, with $c = 1, \ldots, m$.

Define now the *state quantization grids*:

$$-x_{d,\max} = q_{d,1} < q_{d,2} < \cdots < q_{d,N_d} < q_{d,N_d+1} = x_{d,\max}$$

for $d = 1, \ldots, n$. The quantization grid along dth dimension has $N_d + 1$ elements (and thus N_d intervals), where the first and the last element are fixed at the boundaries of the domain. Given these grids, we say that the *quantized state* corresponding to a continuous state x is $i = [i_1, \ldots, i_n]^{\mathrm{T}}$ if:

$$x_d \in \begin{cases} [q_{d,i_d}, q_{d,i_d+1}) & \text{when } i_d < N_d \\ [q_{d,i_d}, q_{d,i_d+1}] & \text{when } i_d = N_d \end{cases} \tag{3}$$

for all d, where the second branch is a special case in order to include the right boundary of the domain. Note that $i_d \in \{1, \ldots, N_d\}$, and there are in total $\boldsymbol{N} = \prod_{d=1}^{n} N_d$ quantized states. Similarly, define the action quantization grids:

$$-u_{c,\max} = p_{c,1} < p_{c,2} < \cdots < p_{c,M_c} < p_{c,M_c+1} = u_{c,\max}$$

The quantization grid along action dimension c has $M_c + 1$ elements and M_c intervals. Quantized actions $j = (j_1, \ldots, j_m)$ are defined entirely similarly to the quantized states, with $j_c \in \{1, \ldots, M_c\}$ and a total of $\boldsymbol{M} = \prod_{c=1}^{m} M_c$ quantized actions.

Given the continuous-variable system (1), the goal of this assignment is to build a quantized approximation of this system, which describes how the quantized state evolves as (quantized) actions are applied. The quantized dynamics will be stochastic, because the transitions depend on the actual continuous position of the state and action within the quantization boxes:

$$P\big(i(k+1) = i' | i(k) = i, j(k) = j\big) = F(i, j, i') \tag{4}$$

i.e., F describes the probability of reaching a quantized next state i' when starting from a quantized current state i and applying a quantized action j. So, for each i, j, the function F must satisfy $\sum_{i'} F(i, j, i') = 1$.

Assignment

Part 1. First, implement in Simulink a simulation model of an underactuated inverted pendulum. The continuous-time model is:

$$\ddot{\alpha} = 1/J \cdot [mgl \sin(\alpha) - b\dot{\alpha} - K^2\dot{\alpha}/R + Ku/R]$$

where $J = 1.91 \cdot 10^{-4}\,\mathrm{kgm}^2$, $m = 0.055\,\mathrm{kg}$, $g = 9.81\,\mathrm{m/s}^2$, $l = 0.042\,\mathrm{m}$, $b = 3 \cdot 10^{-6}\,\mathrm{Nms/rad}$, $K = 0.0536\,\mathrm{Nm/A}$, $R = 9.5\,\Omega$. The angle α varies in the interval $[-\pi, \pi)\,\mathrm{rad}$, with $\alpha = 0$ pointing up, and 'wraps around' so that, e.g., a rotation of $3\pi/2$ corresponds to $\alpha = -\pi/2$. The state vector is $x = [\alpha, \dot{\alpha}]^{\mathrm{T}}$. The control action u is constrained to $[-3, 3]\,\mathrm{V}$, and the velocity $\dot{\alpha}$ must be restricted to $[-10\pi, 10\pi]\,\mathrm{rad/s}$, using saturation. The sampling time is $T_{\mathrm{s}} = 0.005\,\mathrm{s}$. Note that discrete-time transitions of the form (1) will have to be obtained from the continuous-time dynamics.

Part 2. Implement in Matlab a function that estimates online the transition probability function F, using sample transitions $(x(k), u(k), x(k+1))$ of the continuous-variable underlying system. This function must provide at any given time an estimate of F consistent with all the transitions observed so far, i.e., the elements of F should be computed as empirical averages of the quantized transitions observed. On the implementation side, the function:

- Should be general enough to work for any dynamic system, having any state and action dimension.
- Should be implemented efficiently, as it is intended for real-time use.

Part 3. Show that your function works properly. For this, use simulation experiments with trajectories generated by the inverted pendulum model implemented above. Save the results of the experiments in data file(s). Simulate some trajectories of the continuous-variable system and compare them with trajectories generated using the quantized, stochastic dynamics F found. Note: quantize the actions used to control the continuous-variable system, as well as the initial state. Optionally, implement a way to graphically visualize (parts of) the estimate F, in comparison to the original dynamics.

Useful hints

1. Simulating a single, very long trajectory is unlikely to provide sufficient information to estimate F, as this trajectory will be limited to a subregion of the state space. Instead, use limited-length trajectories, and at the beginning of each such trajectory, reset the system to some random initial state. Additionally, use random action sequences to properly explore the action space.
2. The quantized dynamics F can be conveniently represented in Matlab as a tensor (three-dimensional array) $\boldsymbol{F} \in [0, 1]^{\boldsymbol{N} \times \boldsymbol{M} \times \boldsymbol{N}}$. Each element $\boldsymbol{F_{i,j,i'}}$ holds the probability of reaching $\boldsymbol{i'}$ after applying $\boldsymbol{j}$ in $\boldsymbol{i}$. Here, $\boldsymbol{i}, \boldsymbol{i'}$ are single-dimensional indices corresponding to the n-dimensional quantized states (indices) i, i'; and similarly for $\boldsymbol{j}$. To transform between n or m-dimensional and single-dimensional indices, use the included `ndi2lin` and `lin2ndi` functions.
3. Interface the Simulink model with Matlab using the `To Workspace` and `From Workspace` blocks. Call the Simulink model programmatically from Matlab.
4. The state space of the pendulum does not directly satisfy (2); instead, the angle interval is open to the right. This issue can trivially be addressed when the angle variable α is quantized, by using the right-open interval $[q_{d,i_d}, q_{d,i_d+1})$ instead of the closed interval $[q_{d,i_d}, q_{d,i_d+1}]$ in the second branch of (3).