Online Learning of Markov Jump Dynamic Systems

Ritabrata Ray

July 30, 2021

Cornell University

A two mode system

Example: Washing Machine with washing and spinning modes.

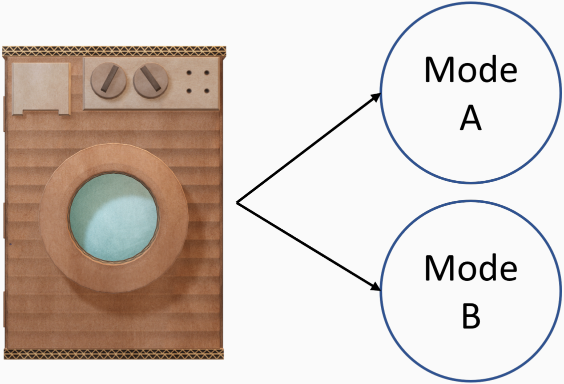

Figure 1: A jump dynamic system with two modes

Markov chain for the two-mode system

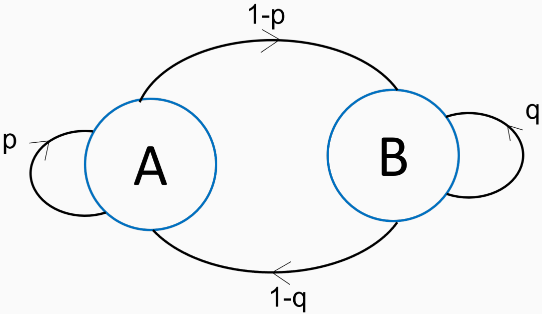

Figure 2: The underlying Markov chain

Markov Jump Linear Dynamic System

1. Multiple modes.
2. Each mode a linear dynamical system with its own state transition parameters (A_i, B_i, σ_i). When in mode i at time t, state transitions according to:

   $$x_{t+1} = A_i x_t + B_i u_t + w_i,$$

   with $w_i \sim \mathcal{N}(0, \sigma_i^2 I)$.
3. Mode switches with time according to a Markov chain.
4. Mode switch introduces non-linearity.

Example: A failure-prone production system

The production system has two modes:

1. Production mode: Apply control u_t units of production per day
$$x_{t+1} = x_t + u_t - d$$
2. Failure mode: No production occurs and the demand piles up
$$x_{t+1} = x_t - d$$

constant demand of d units per day and the state variable x is the net production - net demand.

Example: Quadratic cost motivation

1. $x >> 0$ implies overproduction and wastage.
2. $x << 0$ demands are not met, likely customer dissatisfaction.
3. $u >> 0$ implies high production cost.
4. u cannot be negative.

Natural to have Quadratic cost:

$$c_t = ax_t^2 + bu_t^2$$

with suitably chosen $a, b > 0$.

Analytical setting

State space: $x \in \mathbb{R}^d$, Action space: $u \in \mathbb{R}^k$: At time t, mode is r_t:

$$x_{t+1} = A_{r_t} x_t + B_{r_t} u_t + w_{r_t}$$

where

$$w_t \sim \mathcal{N}(0, \sigma_{r_t}^2 I)$$

Analytical setting: cost

Cost:

$$c_t = x_t^\intercal Q x_t + u_t^\intercal R u_t,$$

where Q and R are symmetric positive semi-definite matrices known to the learner.

Analytical Setting: Markov jump

1. m modes of the system, m known to the learner.
2. Mode may switch at every time step according to a stationary Markov chain.
3. Markov chain transition probability:q_{ij} from mode i to mode j.
4. q_{ij} are unknown to the learner.
5. System parameters (A_i, B_i) unknown to the learner, but noise parameter σ_i known to the learner for each mode $i \in [m]$.

Analytical Setting: Goal

Goal is to minimize the total cost in expectation:

$$J_T = \sum_{t=1}^{T} \mathbb{E}[c_t]$$

For the infinite horizon version, the goal is to minimize:

$$J = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[c_t]$$

Algebraic Ricatti Equations for single mode LQR

The optimal policy which minimizes J for the LQR system (infinite horizon) is the stationary linear policy $K^* \in \mathbb{R}^{k \times d}$ given by:

$$K^* = -(B^\intercal P^* B + R)^{-1}(B^\intercal P^* A),$$

where P^* is a solution to the algebraic Ricatti equation:

$$P^* = A^\intercal P^* A + Q - A^\intercal P^* B(B^\intercal P^* B + R)^{-1} B^\intercal P^* A$$

Optimal Cost for LQR

The optimal policy is played as:

$$u_t = K^* x_t$$

at every time step t and the optimal expected cost when this policy is played is given by:

$$J^* = \sigma^2 \operatorname{Tr}(P^*)$$

.

Optimal policy for Markov jump linear dynamic system

The optimal policy for minimizing J_T in the Markov jump linear dynamic system can be computed using dynamic programming as follows:

$$u_t = -(R + B_i^\intercal P_{i,t+1} B_i)^{-1} B_i^\intercal P_{i,t+1} A_i x_t,$$

where $P_{i,t}$ is given by the recursion:

$$P_{i,t} = \sum_{j \in [m]} q_{ij} M_{j,t},$$

where

$$M_{j,t} = Q + A_j^\intercal P_{j,t+1} A_j - A_j^\intercal P_{j,t+1}^\intercal B_j (R + B_j^\intercal P_{j,t+1} B_j)^{-1} B_j^\intercal P_{j,t+1} A_j,$$

and,

$$P_{i,T+1} = Q, \quad \forall i \in [m].$$

Optimal expected total cost for the system

When the above policy is followed, the optimal expected total cost is given by:

$$\sum_{j\in[m]} \mathbb{E}[x_1^\intercal P_{j,1} x_1 | r_1 = j] \pi_{j,1} \ + \ \sum_{t=1}^{T} \sum_{i\in[m]} \pi_{i,t} \sigma_i^2 \operatorname{Tr}(P_{i,t+1}),$$

where $\pi_{i,t}$ is the probability of mode i at time t.

Optimal cost for stationary Markov chain in the infinite horizon version

For a stationary Markov chain with stationary probabilities π_i^* for the mode i, the optimal time averaged cost in expectation is thus given by:

$$J^* = \sum_{i \in [m]} \pi_i^* \sigma_i^2 \operatorname{Tr}(P_i^*).$$

We use this optimal cost as the benchmark for our regret.

Regret

For the above optimal cost per round the sum-regret of any online learning algorithm for the Markov jump linear dynamic system is defined as:

$$R_T = \sum_{t=1}^{T} (x_t^\intercal Q x_t + u_t^\intercal R u_t - J^*)$$

We also define the running average regret as:

$$r_t = \frac{R_t}{t}.$$

An online algorithm is called a "no-regret" learning algorithm if it achieves:

$$\lim_{t \to \infty} r_t = 0$$

.

Stability of a policy

Given a mode with parameters (A, B), a policy K is called stable with respect to that mode if:

$$\rho(A + BK) < 1,$$

$\rho(M)$ denotes the spectral radius of any matrix M , i.e., the largest absolute value of its singular values.

Strong stability of a policy

A policy K is (κ, γ) strongly-stable for a mode with parameters (A, B) if we could write

$$A + BK = HLH^{-1},$$

$$\text{where} \quad \|H\| \leq \alpha, \quad \left\|H^{-1}\right\| \leq \frac{1}{\beta},$$

$$\text{with} \;\; \frac{\alpha}{\beta} \leq \kappa,$$

$$\|K\| \leq \kappa \;\; \text{and} \quad \|L\| \leq 1 - \gamma,$$

with $\kappa \geq 0$ and $\gamma \in (0, 1]$.

It can be shown that any stable policy is $(\kappa^{'}, \gamma^{'})-$stable for some $\kappa' \geq 0$ and some $\gamma' \in (0, 1]$.

Sequential-strong stability

Given a mode with parameters (A, B), a sequence of policies $K_1, K_2, \ldots$ is called (κ, γ)- sequentially strongly stable for the mode if we can write $\forall t$:

$$A + BK_t = H_t L_t H_t^{-1} \text{ such that :}$$

(i) $\|L_t\| \leq 1 - \gamma$ and $\|K_t\| \leq \kappa$;

(ii) $\|H_t\| \leq B_0, \left\|H_t^{-1}\right\| \leq 1/b_0$ with $\kappa = B_0/b_0$;

(iii) $\left\|H_{t+1}^{-1} H_t\right\| \leq 1 + \gamma/2$.

SDP formulation for the single mode LQR

The problem of finding an optimal policy for the single mode LQR control could be formulated as the following semi-definite program:

$$
\begin{aligned}
&\text{minimize} \quad \begin{pmatrix} Q & 0 \\ 0 & R \end{pmatrix} \bullet \Sigma \\
&\text{subject to:} \\
&\Sigma_{xx} = (A \quad B)\Sigma(A \quad B)^\intercal + W, \\
&\Sigma \succcurlyeq 0,
\end{aligned}
$$

$\Sigma_{xx} \in \mathbb{R}^{d\times d}$, $\Sigma_{xu} \in \mathbb{R}^{d\times k}$, $\Sigma_{ux} \in \mathbb{R}^{k\times d}$, $\Sigma_{uu} \in \mathbb{R}^{k\times k}$, and $W = \sigma^2 I$, where I is the $d \times d$ identity matrix. $\Sigma \in \mathbb{R}^{n\times n}$ is the state-action co-variance matrix at steady state for the optimal policy, where

$$n = d + k.$$

Here, we denote $\mathrm{Tr}(X^\intercal Y)$ as $X \bullet Y$ for any two matrices X and Y.

Assumptions on the jump dynamic system

1. $\exists \alpha_0, \alpha_1, \vartheta, \nu > 0$ such that,

$$\alpha_0 I \preccurlyeq Q \preccurlyeq \alpha_1 I,$$

$$\alpha_0 I \preccurlyeq R \preccurlyeq \alpha_1 I,$$

$$\|A_i B_i\| \leq \vartheta, \ \forall i \in [m],$$

$$J^* \leq \nu,$$

   where $\|AB\|$ denotes the maximum singular value norm of the augmented matrix AB.
2. There exists a stable policy, $K_{0,i} \in^{k \times d}$ for each mode $i \in [m]$ known to the learner.

Warm-up algorithm

The online learning algorithm requires *good* initial estimates of the system parameters (A, B) for each mode. So, we run the following warm-up algorithm first:

Algorithm 1 Warm-up using stable policies for jump dynamic system

---

Input: $(\kappa_{0,i}, \gamma_{0,i})$ strongly-stable policy $K_{0,i}$ for each mode $i \in [m]$, and horizon T_0.

for $t = 1, \dots, T_0$ **do**

  observe state x_t and mode i.

  play: $u_t \sim \mathcal{N}(K_{0,i}x_t, 2\sigma_i^2 \kappa_{0,i}^2 I)$

  record: x_t in Z_i matrix and x_{t+1} in X_i matrix which would be used for ridge-regression for the mode i to learn the system parameter estimates $(A_{0,i}B_{0,i})$ for each mode $i \in [m]$.

end

Warm-up algorithm guarantee

Run the warm-up algorithm for T_0 rounds,

$$T_0 = O(poly(n, \sigma_1, \kappa_{0,1}, \gamma_{0,1}^{-1}, \dots, \sigma_m, \kappa_{0,m}, \gamma_{0,m}^{-1}, \vartheta, \log(1/\delta), \log(1/\epsilon))),$$

ridge-regression step of the above algorithm gets us parameter estimates $(A_{0,i}, B_{0,i})$ for each mode i, satisfying:

$$\|(A_i B_i) - (A_{0,i} B_{0,i})\|_F^2 \leq \epsilon \;\; \forall i \in [m],$$

with probability at least $1 - \delta$.

Optimistic Semi-definite programming for jump dynamic system

Algorithm 4: OSLO for jump dynamic system.

Input: parameters $\alpha_0, \alpha_1, \sigma_1^2, \ldots, \sigma_m^2, \sigma^2 = \max_{i\in[m]} \sigma_i^2, \vartheta, \nu > 0; \delta \in (0,1)$;

initial estimates $(A_{0,i}B_{0,i})$ such that $\left\|(A_iB_i) - (A_{0,i}B_i\right\|_F^2 \leq \epsilon, \forall i \in [m]$.

Initialize: $\mu = 5\vartheta\sqrt{T}$, $V_{1,i} = \lambda I$, $t_i = 1 \ \ \forall i \in [m]$, where $\lambda = \frac{2^{11}\nu^5\vartheta\sqrt{T}}{\alpha_0^5\sigma^{10}}$ and

$\beta = \frac{2^{18}\nu^4 n^2}{\alpha_0^4\sigma^6}\log(\frac{T}{\delta})$

for $t = 1, \ldots, T$ **do**

receive state x_t and mode i and record $x_{t_i,i} = x_t$.

if $det(V_{t_i,i}) > 2det(V_{\tau_i,i})$ *or* $t_i = 1$ **then**

start new episode: $\tau_i = t_i$.

estimate system parameters: Let (A_tB_t) be a minimizer of

$$\frac{1}{\beta}\sum_{s=1}^{t_i-1}\left\|(AB)z_{s,i} - x_{s+1,i}\right\|^2 + \lambda\left\|(AB) - (A_{0,i}B_{0,i})\right\|_F^2 \tag{3.1}$$

over all matrices $(AB) \in \mathbb{R}^{d\times n}$.

compute policy: Let $\Sigma_{t,i}$ be an optimal solution to the following semi-definite program:

$$\begin{aligned}
&\text{minimize} \begin{pmatrix} Q & 0 \\ 0 & R \end{pmatrix} \bullet \Sigma \\
&\text{subject to:} \\
&\Sigma_{xx} = (A_{t,i} \;\; B_{t,i})\Sigma(A_{t,i} \;\; B_{t,i})^{\mathsf{T}} + \sigma_i^2 I - \mu \Sigma \bullet V_{t,i}^{-1} I, \\
&\Sigma \succcurlyeq 0.
\end{aligned} \tag{3.2}$$

set $K_{t,i} = (\Sigma_{t,i})_{ux}(\Sigma_{t,i})_{xx}^{-1}$

else

set $K_{t,i} = K_{t-1,i}, A_{t,i} = A_{t-1,i}, B_{t,i} = B_{t-1,i}$.

play: $u_t = K_{t,i} x_t$

update: $z_{t,i} = \begin{pmatrix} x_t \\ u_t \end{pmatrix}$, $t_i = t_i + 1$ and $V_{t+1,i} = V_{t,i} + \frac{1}{\beta} z_{t,i} z_{t,i}^{\mathsf{T}}$

Implementation details

Performed numerical simulation of the above algorithm on a two mode system with two dimensional state and two dimensional action space with a horizon length of 3×10^6. When the transition probabilities are:

$$q_{11} = p, q_{12} = 1 - p, q_{21} = 1 - q, q_{22} = q.$$

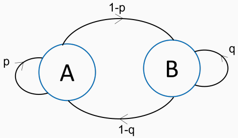

Figure 3: The underlying Markov chain

Implementation details continued

The steady state probabilities are:

$$\pi_1 = \frac{1-q}{2-p-q}$$

$$\pi_2 = \frac{1-p}{2-p-q}$$

On performing dynamic programming we found the matrices $P_{i,t}$'s to quickly converge to P_i^* for each mode $i \in [m]$. The benchmark used for regret is thus:

$$J^* = \sum_{i \in [m]} \pi_i^* \sigma_i^2 \operatorname{Tr}(P_i^*).$$

Experimental expectations

1. Expect to get the running average regret converge to 0 with time. This would imply a sub-linear regret for the algorithm.
2. Also expect the mode parameters to be learnt more precisely with each passing episode for the mode.
3. The state variable to remain bounded and in fact converge towards 0.

Numerical Simulation Results - Average Regret

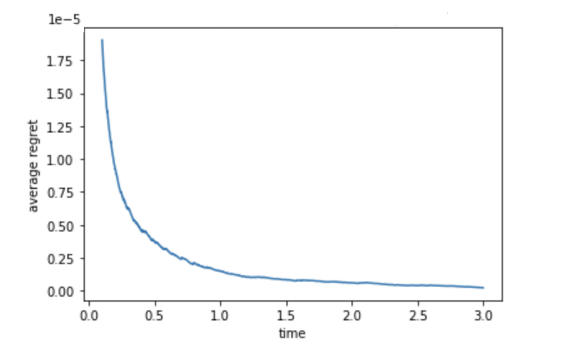

Figure 4: Running average regret r_t against time

Parameter estimation for Mode 1

Here, we plot $\|(A_{t,1}\ B_{t,1}) - (A_1\ B_1)\|_F$ against time:

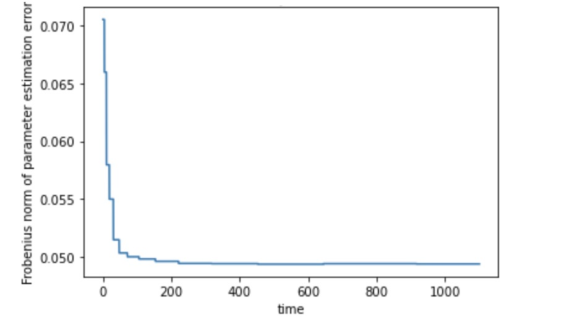

Figure 5: Learning the parameters for Mode 1

Parameter estimation for Mode 2

Here, we plot $\|(A_{t,2} \ B_{t,2}) - (A_2 \ B_2)\|_F$ against time:

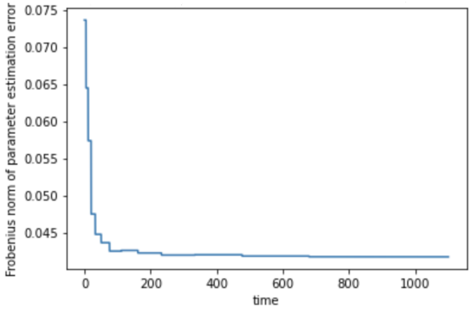

Figure 6: Learning the parameters for Mode 2

Conclusion and future work

The numerical simulation meets all the expectations.

Aim is to obtain a regret analysis for the algorithm.

Challenge lies in proving that the state variable x will remain bounded.This is verified experimentally from the simulation.

The sequence of policies generated by the algorithm is an intermixing of m sequences all of which are themselves sequentially strongly stable for their respective mode.

Questions . . .

End

Thank you.