Course Notes On Dynamic Optimization (Fall 2023)
Lecture 3: Imperfect State Information

Instructor: Daniel Russo

Email: djr2174@gsb.columbia.edu

Graduate Instructor: David Cheikhi
Email: d.cheikhi@columbia.edu

These notes are based of scribed notes from a previous edition of the class. I have done some follow up light editing, but there may be typos or errors.

Topics:

- Problems with imperfect state observations.
- Examples: linear quadratic Gaussian (LQG) systems, bandit problems, recommender systems.
- Reduction to perfect state information problems: beliefs as state and a connection to RNNs.
- LQG and the separation principle

1 Problems with imperfect state information

Consider a dynamic system that evolves according to $z_{k+1} = f_Z(z_k, u_k, w_k)$ where the disturbances $\{w_k\}$ are independent. Now instead of seeing the latent state z_k, what is observed in time k is $y_k = O_k(z_k, u_{k-1}, \xi_k)$ where $\{\xi_k\}$ are independent. Effectively,

$$y_k | z_k, u_{k-1}, \cdots, z_0, u_0 \sim q_{z_k, u_{k-1}}(\cdot)$$

The objective is to solve

$$\inf_{\pi} \mathbb{E}^{\pi}_{\{w_K\},\{\epsilon_k\}} \left[\sum_{k=0}^{N-1} g_k(z_k, u_k, w_k) + g_N(z_N) \right]$$

over policies $\pi = (\mu_0, ..., \mu_{N-1})$. Whereas in previous classes we concluded that it was sufficient for μ_k to map the state of the system to a control, here the latent state is unobservable so it is (at least initially) unclear how we would implement such a policy. Instead we take μ_k to be a map $H_k := (y_0, u_0, \cdots, y_{k-1}, u_{k-1}, y_k) \mapsto \mu_k(H_k) \in U$. In other words, μ_k maps from the information available to the controller at time k, i.e. H_k, to some action in the control set U. [*A generalization allows the set of feasible controls to be $U_k(H_k)$. We'll skip that extra generality.*]

2 Examples

The next example has had a profound impact on control systems in use throughout the world.

Example 1 (Linear Quadratic Gaussian (LQG))**.** *As in last class, we consider a control problem with linear dynamics $z_{k+1} = Az_k + Bu_k + w_k \;\; k = 0, ..., N-1$. But now the controller does not have access to the current latent state z_k. Instead it receives at the beginning of each period k an observation of the form $y_k = Cz_k + \xi_k$ where $\{w_k\}, \{\xi_k\}$ are independent sequences and are independent of z_0. The objective is to minimize the total cost:*

$$\inf_{\pi} \mathbb{E}^{\pi} \left[\sum_{k=1}^{N-1} (z_k^T Q z_k + u_k^T R u_k + z_N^T Q z_N) \right]$$

Next we describe an example where —unlike the LQG example—there are no 'dynamics' to latent state. The problem dynamics are solely about the changing information available to the decision-maker and not about some underlying physical system evolving.

Example 2 (Multi-armed bandit problems)**.** *There are m advertisements indexed by $i \in \{1, \ldots, m\}$. The latent state of the system is a vector $Z_k \in [0,1]^m$ where $Z_{k,i}$ denotes the click through rate of the ad in the population. The control decision $u_k \in \{1, \ldots, m\}$ indicates a choice of ad to display when the next user visits the website. (In this model, the ad is not targeted to this specific user, but to a population.) The observation $y_k \in \{0, 1\}$ indicates whether the ad was clicked on, with $y_k | H_k, u_k, Z_k \sim Z_{k,u_k}$. The goal of the decision maker is to solve*

$$\sup_{\pi} \mathbb{E}^{\pi} \left[\sum_{k=1}^{N-1} y_{k,u_k} \right] = \sup_{\pi} \mathbb{E}^{\pi} \left[\sum_{k=1}^{N-1} Z_{k,u_k} \right].$$

The latent click rate Z_0 is a random variable, thought to be drawn from some prior distribution. We assume the system is stationary, so $Z_0 = Z_1 = \ldots Z_N$ almost surely; the click-through-rate of the ad is not changing across time.

If the decision-maker knew the click-through rates, they would pick the ad $\arg\max_i Z_{0,i}$ *and display it in every period. But since these are unobservable, the DM faces a tension between exploring ads to learn about their latent click-rate and exploiting past observations to get clicks right now.*

Remark (On the role of the prior)**.** *Discussion we had in class: think of the prior as a way to share data across tasks/experiments. A company that optimized ad placement would do this on a recurring basis, across time and across many websites. This structure is extremely common and that is no accident: usually you only invest in a sophisticated algorithm for solving some decision problem if you have to solve variants of it repeatedly*[1]*. Questions around enforcing robustness to distribution shift, etc., are very interesting but we will not tackle them.*

The next example is more complex and not as mathematically clean as the previous ones. Still it reflects how our general problem formulation might provide *a way of thinking* about more complex scenarios.

Example 3 (A recommender system)**.** *Consider a recommender system. The system interacts with many users and leveraging cross-user data is essential. But many systems (e.g. Netflix, Spotify, TikTok, etc.) also*

[1]The field of decision-analysis is full of important non-recurring decisions. The logic of dynamic programming plays a central role, but there is also an emphasis of careful elicitation of a decision-maker's prior beliefs, their assessment of costs, etc.

have recurring interactions with individual users. Here we think of modeling recurring interactions with an individual user whose latent state evolves across time. Different problem formulations highlight different elements of the problem:

1. *(Cold-start/Exploration) The latent state Z_k encodes the user's long-running tastes. While tastes do evolve, it is still reasonably interesting to model them as being fixed but latent, in which case $Z_0 = Z_1 = \ldots, Z_N$ as in Example 2. Models of this type isolate the problem of efficiently learning about a user's tastes so that future recommendations can be tailored to match them.*

2. *(Contextual/Sequential RecSys) The latent state Z_k encodes the user's mood or context. Think "when I'm at the gym I like to listen to workout music." The app doesn't know you're at the gym and doesn't influence whether you're at the gym, but your recent listening behavior might reflect that you're in the mood for workout music. This problem is similar to (1) in that recommendation actions can reveal information about the latent state but do not actively influence it. It differs from (1) in that the latent state is changing across time.*

3. *(Reinforcement learning in RecSys) The latent state can also encode things like habits, boredom, etc, which are actively influenced by. In "optimizing audio recommendations for the long term: a reinforcement learning perspective", one we looked at users' habitual listening of individual podcasts. When recommending a user try a new podcast show, one is not just reflecting their latent tastes, but potentially impacting their listening habit months into the future. [Many podcast shows release episodes daily, and have listeners who engage frequently across a long timespan.]*

3 Reduction to case of the perfect state observations

Here we review ways of reducing a problem with imperfect state information to an equivalent problem with a fully observable state variable. Because of this reduction, all results an algorithms we develop for problems with perfect state information can be applied – at least in principle — to problems with imperfect state information.

These notes will use the variable x_k to denote a fully observable state, thus maintaining consistency with the previous notes. In all cases, that x_k is observable means $x_k = \text{function}(H_k)$ is some function or "compression" of the history that retains sufficient information.

3.1 An observable Markov state

From the decision-maker's perspective, z_k is not "the state"[2]! The state variable x_k should

1. Be observable: the DM be able to compute the state from observed data, i.e. there is some function ϕ such that $x_k = \phi(H_k)$.

2. Be sufficient: given the state, the DM doesn't need to remember the history. Formally, if $c_k \equiv g_k(z_k, u_k, w_k)$ denote the cost at stage k, then

$$\mathbb{P}(x_k + 1 = x \wedge c_k = c \mid H_k, u_k) = \mathbb{P}(x_k + 1 = x \wedge c_k = c \mid X_k, u_k). \tag{1}$$

[2] I will call it "the latent state", though this is not universally accepted.

3.2 Alternative sufficient conditions (Dan's custom definitions)

Some of you may find the Markov condition in (1) to be difficult to parse. A potential difficulty is that x appears on both sides of (1); the state variable must be sufficient for predicting states. Encoding more information in the state both helps (since x_k contains more of the history) and hurts (since x_{k+1} is now more complex and harder to predict). I will provide an set of sufficient conditions which I find easier to digest.

Observable costs: For this subsection, I will assume that costs are observable. That is the stage cost $c_k = \text{someFunction}(u_k, y_{k+1})$ is computable given observations.

Remark (When are costs observable.)**.** *Costs/rewards are observable in Example 2. In Example 3, costs/rewards are not observable if the goal is to maximize "user satisfaction", but are observable if the goal is to maximize some proxy like engagement, retention etc.*

As long as the state-observation noise is independent of control decisions, LQG problem in Example 1 can be rewritten as a problem with observable costs by noting that

$$\mathbb{E}^\pi[\sum_{k=0}^{N-1}(z_k^T Q z_k + u_k^T R u_k + z_N^T Q z_N)] = \mathbb{E}^\pi\left[\sum_{k=1}^{N}(y_k^T (C^{-1})^\top Q C^{-1} y_k)\right] + \sum_{k=0}^{N-1} u_k^T R u_k + y_N^T (C^{-1})^T Q C^{-1} y_N) + \underbrace{(\mathbb{E}\left[y_0^T (C^{-1})^\top Q C^{-1} y_0\right] + \sum_{k=0}^{N}\mathbb{E}[\xi_k^\top Q \xi_k]}_{\text{indep. of } u\text{'s}}.$$

Sufficient conditions for a state variable. How do we formalize this? Here are some sufficient conditions.

1. (Recursive updating) There is a function f such that $x_{k+1} = f(x_k, u_k, y_{k+1})$
2. (Sufficient for predicting observables) $\mathbb{P}(y_{k+1} = y \mid u_k, x_k) = \mathbb{P}(y_{k+1} = y \mid u_k, H_k)$ for any y.

Here we require x_k is sufficient to predict observable, whereas the Markov property requires x_k is contains all information relevant to predicting future states — requiring a kind of circular thinking. Of course, we can't really get away from that circular thinking; here hidden it in the requirement that x_k can be recursively updated. The next lemma confirms that x_k is indeed a Markov state assuming it satisfies these conditions.

Lemma 1. *If costs are observable and conditions 1,2 above hold, then*

$$\mathbb{P}(x_{k+1} = x \wedge c_k = c \mid H_k, u_k) = \mathbb{P}(x_k + 1 = x \wedge c_k = c \mid X_k, u_k).$$

Proof. Focus only on showing the Markov property for x_k, as a the argument for c_k is similar. We have

$$\mathbb{P}(x_{k+1} = x \mid H_k, u_k) = \mathbb{P}(f(x_k, u_k, y_{k+1}) \mid H_k, u_k) = \mathbb{P}(f(x_k, u_k, y_{k+1}) \mid x_k, u_k) = \mathbb{P}(x_{k+1} = x \mid x_k, u_k),$$

where the first and final equalities use property (1) and the second equality uses (2). Note that it is important in the proof that H_k contains all the information in x_k, so we are only taking hte conditional expectation over y_{k+1} and not x_k. □

3.3 Examples of state variables

3.3.1 History as state

By definition, one can satisfy (1) and (2) by taking $x_k = H_k$ to be the full history. (The function f is then one that appends the most recent observation and control decision to the history).

3.3.2 Posterior distribution as state

For concreteness, suppose the latent state $z_k \in \{1, \dots m\}$ takes on finite number of possible values. Let

$$x_{k,i} = \mathbb{P}(z_k = i \mid H_k) \; i \in [m].$$

This is sufficient since

$$\begin{aligned}\mathbb{P}(y_{k+1} = y \mid H_k, u_k) = \sum_{i=1}^{n} \mathbb{P}(z_k = i \mid H_k, u_k)\mathbb{P}(y_{k+1} = y \mid H_k, u_k, z_k = i) &= \sum_{i=1}^{n} x_{k,i} q_{i,u_k}(y) \\ &= \mathbb{P}(y_{k+1} = y \mid x_k, u_k).\end{aligned}$$

Moreover, the posterior distribution can be updated recursively by sequential Bayesian updating:

$$x_{k+1,i} = \frac{x_{k,i} q_{i,u_k}(y_{k+1})}{\sum_{j=1}^{n} x_{k,j} q_{j,u_k}(y_{k+1})}.$$

The same results apply when the latent state is continuous, though in practice you need some way to store the posterior distribution. Sometimes it is enough to track sufficient statistics, as is illustrated in the next example.

Example 4 (Revisiting the MAB)**.** *Consider the MAB problem in Example 2. Since $z_0 = \dots z_{N-1}$ almost surely, simplify notation by writing $z \equiv z_k$. Suppose there exist prior hyperparameters $\alpha_0 \in \mathbb{R}_+^m$ and $\beta_0 \in \mathbb{R}_+^m$ such that $z \sim \text{Beta}(\alpha_0, \beta_0)$. Over time, one can update these parameters as*

$$\alpha_{k+1,i} = \alpha_{k,i} + \mathbb{1}(u_k = i, y_k = 1) \qquad \beta_{k+1,i} = \alpha_{k,i} + \mathbb{1}(u_k = i, y_k = 0).$$

The posterior distribution is $z \mid H_k \sim \text{Beta}(\alpha_k, \beta_k)$. In this problem, one choose the state variable $x_k = [\alpha_k, \beta_k]$.

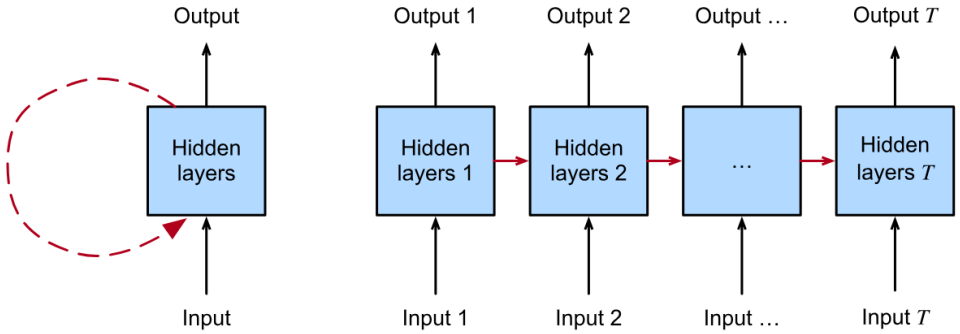
3.4 Recurrent neural network

Consider a recurrent neural network where the input at time k is (y_k, u_k). The model maintains a recursively updates a hidden state h_k, as

$$h_k = f_w(h_{k-1}; [y_k, u_k])$$

where w denotes trainable weights of tte neural network. The output $\hat{y}_{k+1} = \text{someFunction}(h_k, [y_k, u_k])$ is optimized to minimize error in predicting y_{k+1}. In the context of Example 3, you should imagine that the weights w are trained on data collected across many users, whereas our problem formulation focuses on optimizing interactions with a single user (who is randomly chosen from the population).

Define the approximate state variable $x_k = (h_{k-1}, y_k)$. This satisfies our recursive updating requirement (1) and is trained to approximate the sufficiency condition (2).



4 Linear quadratic control and the separation principle

Consider the LQ control problem $z_{k+1} = Az_k + Bu_k + w_k \;\; k = 0, ..., N-1$ we treated in the previous class session. We no longer the controller does not have access to the current latent state z_k. Instead it receives at the beginning of each period k an observation of the form $y_k = Cz_k + \xi_k$ where $\{w_k\}, \{\xi_k\}$ are independent sequences and are independent of z_0. The objective is to minimize the total cost

$$\inf_{\pi} \mathbb{E}^{\pi}[\sum_{k=1}^{N-1} (z_k^T Q z_k + u_k^T R u_k + z_N^T Q z_N)]$$

over policies $\pi = (\mu_0, ..., \mu_{N-1})$ where $u_k = \mu_k(H_k)$ that map sequences of past we saw to the action.

Recall that when there are prefect state observations ($y_k = z_k$), the optimal policy sets

$$\mu_k^\star(H_k) = L_k z_k$$

where

$$L_i = -(B^T K_{i+1} B + R)^{-1} B^T K_{i+1} A \quad i = 0, \cdots, N-1$$

and

$$K_i = A^T (K_{i+1} - K_{i+1} B (B^T K_{i+1} B + R)^{-1} B^T K_{i+1}) A Q$$

The next proposition shows that this solution structure extends to the case of imperfect state observations.

Proposition 1. *(Separation principle) The optimal policy of the LQ control with imperfect state information is* $\pi^\star = (\mu_0^\star, \cdots, \mu_{N-1}^\star)$ *where*

$$\mu_k^\star(H_k) = L_k \mathbb{E}[z_k | H_k]$$

The matrices L and K are the same as above.

The separation principle is quite satisfying. Imagine you know about how to solve control problems (hopefully!) and your friend is a statistician. You team up and agree to divide the work — you write the code for a controller and your friend codes up a procedure to estimate the latent state. Neither of you understand what the other did. Proposition 1 says that (at least in principle), this separation of tasks comes at no price.

How do you estimate latent states? In the important special case where the disturbances $\{w_k\}$, $\{\xi_k\}$ and the initial state x_0 are independent Gaussian vectors, Kalman filtering provides a way to recursively update the parameters of the (Gaussian) posterior distribution of the latent state. See Appendix E of the textbook for a precise introduction.

Remark (How do estimate something unobservable?)**.** *It is worth noting that the state-estimation part of this result is subtle. How do you train a model to estimate latent states if you never observe latent states? One plausible answer is that it may be possible to fairly precisely estimate the latent state of a physical control system by (1) employing expensive sensors and (2) using data that is only available in hindsight. One can then train a model to predict these more accurate state readings from the noisy measurements available with cheap sensors and real-time information.*

When the separation principle completely fails. The proof of Proposition 1 of the separation principle can feel like a bunch of algebra. To really digest what is going on, it may be helpful to understand what breaks the separation principle. Basically, the separation principle is the grandest form of the 'certainty equivalence' property we saw in the last class: it is optimal to select a control in each period as if there was no noise in the dynamics and as if your state estimate were correct. This idea fails completely if a decision-maker's uncertainty is a major driver of how they should act, such as:

1. (Risk aversion) A patient who is pre-diabetic is provided a cheap device which helps monitor their blood glucose level based on thumb pricks. Data from these measurements is then visible to their doctor, who needs to determine whether the patient should be called in for a closer examination and a more expensive/inconvenient laboratory test. Should the doctor behave as if the blood glucose measurements are correct? If the measurements are pretty noisy, then obviously not! A key feature of this example is that there is a huge downside to missing diabetes but a comparatively small cost to having a patient take a lab test.

2. (Information gathering) Consider the MAB problem in Example 4. The separation principle suggests to behave as if each ad i's click through rate were the posterior mean $\alpha_{k,i}/(\alpha_{k,i}+\beta_{k,i})$. This results in a so-called greedy policy, which selects the arm $\arg\max_i \alpha_{k,i}/(\alpha_{k,i}+\beta_{k,i})$ in any period k. This maximizes immediate expected reward myopically, failing to explore poorly-understood arms. The optimal policy should actively gather information about arms with high upside, and accounting for uncertainty about an arm's latent quality plays an essential role in that consideration.

Why it works here (1): 'predicting' the mean minimizes quadratic costs. The discussion above suggests that the separation principle may not even in single period problems due to risk-aversion. We can get some intuition as to why using the conditional mean might nevertheless be optimal in the LQ control case. Consider the optimization problem below with a quadratic estimation loss and a quadratic penalty

$$\min_u \mathbb{E}_Z[(Z-u)^T Q(Z-u)+u^T R u],$$

where $Q, R \succ 0$. It is easy to show that the minimizer to this is

$$u^\star = (Q+R)^{-1}Q\mathbb{E}[z],$$

which is a linear function of the mean. When $R = 0$, the optimal objective value is $\mathbb{E}[(Z - \mathbb{E}[Z])^T Q(Z - \mathbb{E}[Z])]$, which penalizes the variance of estimation error. Otherwise, the objective value separates into the sum of two terms: one of which depends on the variance of x and one which depends on the mean, which influences the energy cost $u^\top Ru$. The proof of the separation principle relies on a similar decomposition of the cost-to-go functions.

Why it works here (2): Control decisions cannot reduce estimation errors The discussion above suggests that the separation principle may not hold in dynamic optimization problems in which costly information gathering can improve future performance. The structure of our LQ control problem rules this out; control decisions can alter the state of the system by a fixed, known amount, but this does nothing to resolve uncertainty about the random noise terms $\{w_k\}$, $\{\xi_k\}$ which additively impact latent state dynamics and observations. This is formalized in the following lemma, which states that the state estimation error, $z_k - \mathbb{E}[z_k|H_k]$ is independent of the control choice.In particular, z_k and $\mathbb{E}(z_k|H_k)$ contain the same linear terms in $(u_0, \cdots, u_{k-1})$, which cancel each other out.

Lemma 2. *For every k, the estimation error $z_k - \mathbb{E}(z_k|H_k)$ does not depend on $u_1, \cdots, u_{k-1}$.*

Proof. Since there is no control when $k = 0$, the claim is obviously true. For $k > 0$, write z_k recursively

$$\begin{aligned} z_k &= Az_{k-1} + Bu_{k-1} + w_{k-1} \\ &= A(Az_{k-2} + Bu_{k-2} + w_{k-2}) + Bu_{k-1} + w_{k-1} \\ &= \cdots \\ &= A^k z_0 + \sum_{i=0}^{k-1} A^i B u_i + \sum_{i=0}^{k-1} A^{k-1-i} w_i \end{aligned}$$

$$z_k - \mathbb{E}[z_k|H_k] = A^k(z_0 - \mathbb{E}[z_0|H_k]) - \sum_{i=0}^{k-1} A^{k-1-i}(w_i - \mathbb{E}[w_i|H_k])$$

Hence does not depend on $u_1, \cdots, u_{N-1}$.

□

Completing the proof. Now we are ready to prove Proposition 1.

Proof. For $k_N = Q$ and $P_N = 0$, we write the cost-to-go function as mean cost plus estimation variance (does not depend on the controls)

$$J_N(H_N) = \mathbb{E}[z_N^T Q z_N|H_N] + \mathbb{E}[e_N^T P_N e_N|H_N]$$

where $e_N = z_N - \mathbb{E}(z_N|H_N)$. Continuing in this way

$$J_{N-1}(H_{N-1}) = \min_u l(H_{N-1}, u)$$

where

$$\begin{aligned} l(H_{N-1}, u) = u^T R u &+ \mathbb{E}[z_{N-1}^T Q z_{N-1} | H_{N-1}] \\ &+ \mathbb{E}[e_N^T P_N e_N | H_{N-1}, u_{N-1} = u] \\ &+ \mathbb{E}[(Az_{N-1} + Bu_{N-1} + w_{N-1})^T K_N (Az_{N-1} + Bu_{N-1} + w_{N-1}) | H_{N-1}, u_{N-1} = u] \end{aligned}$$

i.e. the expected accumulated cost-to-go at stage $N-1$ conditional on history H_{N-1} and action u is the instantaneous cost plus the cost-to-go. The cost-to-go at next stage is the sum of expectation of measurement error (not affected by action) and expectation of quadratic cost of next state, given by the linear dynamics, conditional on the history.

Differentiate with respect to u we get

$$\mu^\star(H_{N-1}) = L_{N-1} \mathbb{E}[z_{N-1} | H_{N-1}]$$

where

$$L_{N-1} = -(R + B^T k_N B)^{-1} B^T k_N A$$

Plug the linear policy back into the quadratic function

$$\begin{aligned} l(H_{N-1}, L_{N-1}\mathbb{E}[z_{N-1}|H_{N-1}]) = \mathbb{E}[w_{N-1}^T Q w_{N-1}] &+ \mathbb{E}[e_N^T P_N e_N | H_{N-1}] \\ &+ \mathbb{E}[z_{N-1}^T (Q + A^T k_N A) z_{N-1}^T | H_{N-1}] \\ &- \mathbb{E}[z_{N-1}|H_{N-1}]^T P_{N-1} \mathbb{E}[z_{N-1}|H_{N-1}] \end{aligned}$$

where

$$P_{N-1} = A^T K_N B (R + B^T K_N B)^{-1} B K_N A$$

Notice that can write the last term as

$$\mathbb{E}[z_{N-1}|H_{N-1}]^T P_{N-1} \mathbb{E}[z_{N-1}|H_{N-1}] = \mathbb{E}[z_{N-1}^T P_{N-1} z_{N-1} | H_{N-1}] - \mathbb{E}[e_{N-1}^T P_{N-1} e_{N-1} | H_{N-1}]$$

Plug this back into the original one we have

$$\begin{aligned} J_{N-1}(H_{N-1}) = &\mathbb{E}[z_{N-1}^T K_{N-1} z_{N-1} | H_{N-1}] \\ &+ \mathbb{E}[e_{N-1}^T P_{N-1} e_{N-1} | H_{N-1}] \\ &+ \mathbb{E}[e_N^T P_N e_N | H_{N-1}] \\ &+ C_{N-1} \end{aligned}$$

Thus the cost-to-go function is equal to a quadratic function of state taking expectation over state, plus a bunch of terms that is not affected by the control decision. □