Optimal Control and Estimation

Heng Yang

2023-07-26

Contents

Preface

This is the textbook for Harvard ES/AM 158: Introduction to Optimal Control and Estimation. Information about the offerings of the class is listed below.

2023 Fall

Time: Mon/Wed 2:15 - 3:30pm

Location: Science and Engineering Complex, Room TBD

Instructor: Heng Yang

Teaching Fellow: Weiyu Li

Syllabus

Acknowledgment

Chapter 1

The Optimal Control Formulation

1.1 The Basic Problem

Consider a discrete-time dynamical system

$$x_{k+1} = f_k(x_k, u_k, w_k), \quad k = 0, 1, \dots, N-1 \tag{1.1}$$

where

- $x_k \in \mathbb{X} \subseteq \mathbb{R}^n$ is the *state* of the system,
- $u_k \in \mathbb{U} \subseteq \mathbb{R}^m$ is the *control* we wish to design,
- $w_k \in \mathbb{W} \subseteq \mathbb{R}^p$ a random *disturbance* or noise (e.g., due to unmodelled dynamics) which is described by a probability distribution $P_k(\cdot \mid x_k, u_k)$ that may depend on x_k and u_k but not on prior disturbances $w_0, \dots, w_{k-1}$,
- k indexes the discrete time,
- N denotes the horizon,
- f_k models the transition function of the system (typically $f_k \equiv f$ is time-invariant, especially for robotics systems; we use f_k here to keep full generality).

Remark (Deterministic v.s. Stochastic). When $w_k \equiv 0$ for all k, we say the system (1.1) is *deterministic*; otherwise we say the system is *stochastic*. In the following we will deal with the stochastic case, but most of the methodology should carry over to the deterministic setup.

We consider the class of *controllers* (also called *policies*) that consist of a sequence of functions

$$\pi = \{\mu_0, \dots, \mu_{N-1}\},$$

where $\mu_k(x_k) \in \mathbb{U}$ for all x_k, i.e., μ_k is a *feedback* controller that maps the state to an admissible control. Given an initial state x_0 and an admissible policy π, the state *trajectory* of the system is a sequence of random variables that evolve according to

$$x_{k+1} = f_k(x_k, \mu_k(x_k), w_k), \quad k = 0, \dots, N-1 \tag{1.2}$$

where the randomness comes from the disturbance w_k.

We assume the state-control trajectory $\{u_k\}_{k=0}^{N-1}$ and $\{x_k\}_{k=0}^{N}$ induce an *additive cost*

$$g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k) \tag{1.3}$$

where $g_k, k = 0, \dots, N$ are some user-designed functions.

With (1.2) and (1.3), for any admissible policy π, we denote its induced *expected cost* with initial state x_0 as

$$J_\pi(x_0) = \mathbb{E}\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k)) \right\}, \tag{1.4}$$

where the expectation is taken over the randomness of w_k.

Definition 1.1 (Discrete-time, Finite-horizon Optimal Control)**.** Find the best admissible controller that minimizes the expected cost in (1.4)

$$\pi^\star \in \arg\min_{\pi \in \Pi} J_\pi(x_0), \tag{1.5}$$

where Π is the set of all admissible controllers. The cost attained by the optimal controller, i.e., $J^\star = J_{\pi^\star}(x_0)$ is called the optimal *cost-to-go*, or the optimal *value function*.

Remark (Open-loop v.s. Closed-loop). An important feature of the basic problem in Definition 1.1 is that the problem seeks *feedback policies*, instead of numerical values of the controls, i.e., $u_k = \mu_k(x_k)$ is in general a function of the state x_k. In other words, the controls are executed sequentially, one at a time after observing the state at each time. This is called closed-loop control, and is in general better than open-loop control

$$\min_{u_0, \dots, u_{N-1}} \mathbb{E}\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k) \right\}$$

where all the controls are planned at $k = 0$. Intuitively, a closed-loop policy is able to utilize the extra information received at each timestep (i.e., it observes x_{k+1} and hence also observes the disturbance w_k) to obtain a lower cost than an open-loop controller. Example 1.2.1 in (Bertsekas, 2012) gives a concrete application where a closed-loop policy attains a lower cost than an open-loop policy.

In deterministic control (i.e., when $w_k \equiv 0, \forall k$), however, a closed-loop policy has no advantage over an open-loop controller. This is obvious because at $k = 0$, even the open-loop controller predicts perfectly the consequences of all its actions and there is no extra information to be observed at later time steps. In fact, even in stochastic problems, a closed-loop policy may not be advantageous, see Exercise 1.27 in (Bertsekas, 2012).

1.2 Dynamic Programming and Principle of Optimality

We now introduce a general and powerful algorithm, namely *dynamic programming* (DP), for solving the optimal control problem 1.1. The DP algorithm builds upon a quite simple intuition called the *Bellman principle of optimality.*

Theorem 1.1 (Bellman Principle of Optimality)**.** *Let $\pi^\star = \{\mu_0^\star, \mu_1^\star, \dots, \mu_{N-1}^\star\}$ be an optimal policy for the optimal control problem 1.1. Assume that when using $\pi^\star$, a given state x_i occurs at timestep i with positive probability (i.e., x_i is reachable at time i).*

Now consider the following subproblem where we are at x_i at time i and wish to minimize the cost-to-go from time i to time N

$$\min_{\mu_i,\dots,\mu_{N-1}} \mathbb{E}\left\{ g_N(x_N) + \sum_{k=i}^{N-1} g_k(x_k, \mu_k(x_k)) \right\}.$$

Then the truncated policy $\{\mu_i^\star, \mu_{i+1}^\star, \dots, \mu_{N-1}^\star\}$ must be optimal for the subproblem.

Theorem 1.1 can be proved intuitively by contradiction: if the truncated policy $\{\mu_i^\star, \mu_{i+1}^\star, \dots, \mu_{N-1}^\star\}$ is not optimal for the subproblem, say there exists a different policy $\{\mu_i', \mu_{i+1}', \dots, \mu_{N-1}'\}$ that attains a lower cost for the subproblem starting at x_i at time i. Then the combined policy $\{\mu_0^\star, \dots, \mu_{i-1}^\star, \mu_i', \dots, \mu_{N-1}'\}$ must attain a lower cost for the original optimal control problem 1.1 due to the additive cost structure, contradicting the optimality of $\pi^\star$.

The Bellman principle of optimality is more than just a principle, it is also an algorithm. It suggests that, to build an optimal policy, one can start by solving the last-stage subproblem to obtain $\{\mu_{N-1}^\star\}$, and then proceed to solve the subproblem containing the last two stages to obtain $\{\mu_{N-2}^\star, \mu_{N-1}^\star\}$. The recursion continues until optimal policies at all stages are computed. The following theorem formalizes this concept.

Theorem 1.2 (Dynamic Programming)**.** *The optimal value function $J^\star(x_0)$ of the optimal control problem 1.1 (starting from any given initial condition x_0) is*

equal to $J_0(x_0)$, which can be computed backwards and recursively as

$$J_N(X_N) = g_N(x_N) \tag{1.6}$$

$$J_k(x_k) = \min_{u_k \in \mathbb{U}} \mathbb{E}_{w_k \sim P_k(\cdot|x_k,u_k)} \left\{ g_k(x_k, u_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \right\}, \ k = N-1, \dots, 1, 0. \tag{1.7}$$

Moreover, if $u_k^\star = \mu_k^\star(x_k)$ is a minimizer of (1.7) for every x_k, then the policy $\pi^\star = \{\mu_0^\star, \dots, \mu_{N-1}^\star\}$ is optimal.

Proof. For any admissible policy $\pi = \{\mu_0, \dots, \mu_{N-1}\}$, denote $\pi^k = \{\mu_k, \dots, \mu_{N-1}\}$ the last-$(N-k)$-stage truncated policy. Consider the sub-problem consisting of the last $N-k$ stages starting from x_k, and let $J_k^\star(x_k)$ be its optimal cost-to-go. Mathematically, this is

$$J_k^\star(x_k) = \min_{\pi^k} \mathbb{E}_{w_k, \dots, w_{N-1}} \left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, \mu_i(x_i)) \right\}, \quad k = 0, 1, \dots, N-1. \tag{1.8}$$

We define $J_N^\star(x_N) = g(x_N)$ for $k = N$.

Our goal is to prove the $J_k(x_k)$ computed by dynamic programming from (1.7) is equal to $J_k^\star(x_k)$ for all $k = 0, \dots, N$. We will prove this by induction.

Firstly, we already have $J_N^\star(x_N) = J_N(x_N) = g(x_N)$, so $k = N$ holds automatically.

Now we assume $J_{k+1}^\star(x_{k+1}) = J_{k+1}(x_{k+1})$ for all x_{k+1}, and we wish to induce $J_k^\star(x_k) = J_k(x_k)$. To show this, we write

$$J_k^\star(x_k) = \min_{\pi^k} \mathbb{E}_{w_k, \dots, w_{N-1}} \left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, \mu_i(x_i)) \right\} \tag{1.9}$$

$$= \min_{\mu_k, \pi^{k+1}} \mathbb{E}_{w_k, \dots, w_{N-1}} \left\{ g_k(x_k, \mu_k(x_k)) + g_N(x_N) + \sum_{i=k+1}^{N-1} g_i(x_i, \mu_i(x_i)) \right\} \tag{1.10}$$

$$= \min_{\mu_k} \left[\min_{\pi^{k+1}} \mathbb{E}_{w_k, \dots, w_{N-1}} \left\{ g_k(x_k, \mu_k(x_k)) + g_N(x_N) + \sum_{i=k+1}^{N-1} g_i(x_i, \mu_i(x_i)) \right\} \right] \tag{1.11}$$

$$= \min_{\mu_k} \mathbb{E}_{w_k} \left\{ g_k(x_k, \mu_k(x_k)) + \min_{\pi^{k+1}} \left[\mathbb{E}_{w_{k+1}, \dots, w_{N-1}} \left\{ g_N(x_N) + \sum_{i=k+1}^{N-1} g_i(x_i, \mu_i(x_i)) \right\} \right] \right\} \tag{1.12}$$

$$= \min_{\mu_k} \mathbb{E}_{w_k} \left\{ g_k(x_k, \mu_k(x_k)) + J_{k+1}^\star(f_k(x_k, \mu_k(x_k), w_k)) \right\} \tag{1.13}$$

$$= \min_{\mu_k} \mathbb{E}_{w_k} \left\{ g_k(x_k, \mu_k(x_k)) + J_{k+1}(f_k(x_k, \mu_k(x_k), w_k)) \right\} \tag{1.14}$$

$$= \min_{u_k \in \mathbb{U}} \mathbb{E}_{w_k} \left\{ g_k(x_k, \mu_k(x_k)) + J_{k+1}(f_k(x_k, \mu_k(x_k), w_k)) \right\} \tag{1.15}$$

$$= J_k(x_k), \tag{1.16}$$

where (1.9) follows from definition (1.8); (1.10) expands $\pi^k = \{\mu_k, \pi^{k+1}\}$ and $\sum_{i=k}^{N-1} g_i = g_k + \sum_{i=k+1}^{N-1}$; (1.11) writes the joint minimization over μ_k and π^{k+1} as equivalently first minimizing over π^{k+1} and then minimizing over μ_k; (1.12) is the key step and holds because g_k and w_k depend only on μ_k but not on π^{k+1}; (1.13) follows again from definition (1.8) with k replaced by $k+1$; (1.14) results from the induction assumption; (1.15) clearly holds because any $\mu_k(x_k)$ belongs to $\mathbb{U}$ and any element in $\mathbb{U}$ can be chosen by a feedback controller μ_k; and lastly (1.16) follows from the dynamic programming algorithm (1.7).

By induction, this shows that $J_k^\star(x_k) = J_k(x_k)$ for all $k = 0, \dots, N$. □

The careful reader, especially from a robotics background, may soon become disappointed when seeing the DP algorithm (1.7) because it is rather conceptual than practical. To see this, we only need to run DP for $k = N-1$:

$$J_{N-1}(x_{N-1}) = \min_{u_{N-1} \in \mathbb{U}} \mathbb{E}_{w_{N-1}} \left\{ g_{N-1}(x_{N-1}, u_{N-1}) + J_N(f_{N-1}(x_{N-1}, u_{N-1}, w_{N-1})) \right\}. \tag{1.17}$$

Two challenges immediately show up:

- How to perform the minimization over u_{N-1} when $\mathbb{U}$ is a continuous constraint set? Even if we assume g_{N-1} is convex[1] in u_{N-1}, J_N is convex in x_N, and the dynamics f_{N-1} is also convex in u_{N-1} (so that the optimization (1.17) is convex), we may be able to solve the minimization *numerically* for each x_{N-1} using a convex optimization solver, but rarely will we be able to find an analytical policy $\mu_{N-1}^\star$ such that $u_{N-1}^\star = \mu_{N-1}^\star(x_{N-1})$ for every x_{N-1} (i.e., the optimal policy $\mu_{N-1}^\star$ is implict but not explict).
- Suppose we can find an anlytical optimal policy $\mu_{N-1}^\star$, say $\mu_{N-1}^\star = K x_{N-1}$ a linear policy, how will plugging $\mu_{N-1}^\star$ into (1.17) affect the complexity of $J_{N-1}(x_{N-1})$? One can see that even if $\mu_{N-1}^\star$ is linear in x_{N-1}, J_{N-1} may be highly nonlinear in x_{N-1} due to the composition with g_{N-1}, f_{N-1} and J_N. If $J_{N-1}(x_{N-1})$ becomes too complex, then clearly it becomes more challenging to perform (1.17) for the next step $k = N-2$.

Due to these challenges, only in a very limited amount of cases will we be able to perform *exact dynamic programming*. For example, when the state space $\mathbb{X}$ and control space $\mathbb{U}$ are discrete, we can design efficient algorithms for exact DP. For another example, when the dynamics f_k is linear and the cost g_k is quadratic, we will also be able to compute $J_k(x_k)$ in closed form (though this sounds a bit surprising!). We will study these problems in more details in Chapter 2.

For general optimal control problems with continuous state space and control space (and most problems we care about in robotics), unfortunately, we will have to resort to *approximate dynamic programming*, basically variations of the DP algorithm (1.7) where approximate value functions $J_k(x_k)$ and/or control

[1] You may want to read Appendix B if this is your first time seeing "convex" things.

policies $\mu_k(x_k)$ are used (e.g., with neural networks and machine learning).[2] We will introduce several popular approximation schemes in Chapter 3. We will see that, although exact DP is not possible anymore, the Bellman principle of optimality still remains one of the most important guidelines for designing approximation algorithms. Efficient algorithms for approximate dynamic programming, preferrably with performance guarantees, still remain an active area of research.

[2]Another possible solution is to discretize continuous states and controls. However, when the dimension of state and control is high, discretization becomes too expensive in terms of memory and computational complexity.

Chapter 2

Exact Dynamic Programming

2.1 Linear Quadratic Regulator

Chapter 3

Approximate Dynamic Programming

3.1 Introduction

The limitations of classical deterministic dynamic programming (DP) were mentioned in Chapter 1, particularly its inefficiency in fields such as robotics where both the state and control spaces are typically large and continuous. The process of discretization in such contexts is not only challenging but also costly. Even when discretization is achievable, the resultant state and control spaces tend to be extraordinarily vast and often high-dimensional, leading to prohibitive computational demands. This issue, commonly called the *curse of dimensionality*, renders the use of classical DP unfeasible. Add time complexity analysis here. To circumvent the constraints of traditional DP algorithms in such contexts, a pragmatic approach involves the adoption of a suboptimal control scheme. This compromises between the ease of implementation and adequate performance. The principal objective of this chapter is to find such suboptimal control. In this chapter, we will spend most of the time discussing finite horizon problems with discrete state and control space, which is the classical scenario. We will also mention the infinite horizon problem and continuous state and control spaces scenario later.

Broadly, two categories of approximation are used in the context of DP-based suboptimal control. The first is *approximation in value space*, where we aim to approximate the optimal cost function or the cost function of a given policy. The second is *approximation in policy space*, where we select the policy by using optimization over a suitable class of policies.

3.2 Approximation in value space

Let us first recap the iteration process of the generic form of DP as mentioned in theorem 1.2. We can obtain the cost-to-go function J_k, which means the cost-to-go value at time k, thereby defining corresponding control u_k or policy μ_k.

$$J_k(x_k) = \min_{u_k \in \mathbb{U}} \mathbb{E}\left\{g_k(x_k, u_k) + J_{k+1}(f_k(x_k, u_k, w_k))\right\}, \ k = N-1, \dots, 1, 0. \quad (3.1)$$

By using the *approximation in value space* methods, we could replace the optimal cost-to-go function J_k with some other functions $\tilde{J}_k$. In other words, the suboptimal policy $\tilde{\mu}_k(x_k)$ (and the corresponding control) is obtained from the one-step lookahead minimization

$$\tilde{J}_k(x_k) = \min_{u_k \in \mathbb{U}_k(x_k)} \mathbb{E}\left\{g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, w_k))\right\} \quad (3.2)$$

$$\tilde{\mu}_k(x_k) = \arg \min_{u_k \in \mathbb{U}_k(x_k)} \mathbb{E}\left\{g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, w_k))\right\} \quad (3.3)$$

The major issue in value space approximation is how to compute the approximate cost-to-go functions $\tilde{J}_{k+1}$ in (3.3). We will consider three types of methods:

1. *Problem approximation*
2. *Parametric cost approximation*
3. *Online approximate optimization*

In approximation in value space, we may also distinguish between *online* and *offline* methods.

1. *Offline* methods, where the entire function $\tilde{J}_{k+1}$ in (3.3) is computed for every k before the control process begins. The advantage of this is that most of the computation is done offline. Once the control process starts, the only thing we have to do is one-step lookahead minimization. These methods are well-suited for settings where there are strict time constraints for the online computation of the control, and where there is no need for online replanning.
2. *Online* methods, where most of the computation is performed just after the current state x_k becomes known, the values $\tilde{J}_{k+1}(x_{k+1})$ are computed only at the relevant next states x_{k+1} and are used to compute u_k via (3.3). These methods require the computation of control only for the N states actually encountered in the control process. These methods are well-suited for online replanning.

3.2.1 Problem Approximation

The functions $\tilde{J}_{k+1}$ are obtained (by exact DP, or other methods) as the optimal or nearly optimal cost functions of a simplified version of the original problem. The problem is how to simplify the problem, which is more convenient for computation. There are three widely-used approaches to simplify the initial problem:

1. *Simplifying the structure of the problem through enforced decomposition.*
2. *Simplifying the probabilistic structure of the problem*, such as replacing the stochastic problem with a deterministic one by *certainty equivalence*. To be more specific, the original stochastic system contains the disturbance term $w_k(x_k, u_k)$. To simplify the probabilistic structure of the problem, we could fix the disturbances at some "typical" values and transform the stochastic problem into a deterministic one.
3. *Aggregation*, where the original problem is approximated with a new problem with fewer states, makes it easier to obtain the cost-to-go function. The state in this new problem is the "combination" of the states in the initial problem. It is worth noting that the discretization of continuous state space and action space could be viewed as a kind of aggregation.

3.2.2 Parametric cost approximation

For discrete problems, it is natural to consider using the tabular method to represent the $\tilde{J}_k$ functions. However, if the number of the state space is large this method's memory cost will be overwhelming. On the other hand, for tabular representation, it is not convenient to optimize the function, while we can only update the value of *one* state at a time, but in many circumstances, there is a cluster of states that have similar attributes, which means their corresponding $\tilde{J}_k$ are also similar. It is inconvenient to update them one by one. In this part, we will discuss an alternative approach to represent $\tilde{J}_k$ function, whereby $\tilde{J}_k$ are chosen to be members of a parametric class of functions, with the parameters "optimized" or "trained" by using some algorithms.

To be more specific, the $\tilde{J}_k$ functions could be described as $\tilde{J}_k(x_k, r_k)$ that for each k, depend on the current state x_k and a vector $r_k = (r_{1,k}, ..., r_{m,k})$ of m "tunable" scalar parameters, also called *weights*. By adjusting the weights, one can change the "shape" of $\tilde{J}_k$ so that it is a reasonably close approximation to the true cost-to-go function J_k. In order to train those weights, we can use some cost functions to measure the accuracy of the approximation. The most common cost function is *least squares*. Training the parameters r_k using least squares as the cost function is sometimes referred to as *fitted value iteration*. Value iteration could be viewed as a special form of dynamic programming, where the parameter vectors r_k are determined sequentially, starting from the end of the horizon and proceeding backward. The algorithm samples the state space for each stage k and generates a large number of states x_k^s, $s = 1, ..., q$. It

then determines sequentially the parameter vectors r_k to obtain a good "least square fit" to the DP algorithm.

$$\beta_k^s = \min_{u \in \mathbb{U}_k(x_k^s)} E\left\{ g(x_k^s, u, w_k) + \tilde{J}_{k+1}(f_k(x_k^s, u, w_k), r_{k+1}) \right\} \tag{3.4}$$

$$r_k = \arg\min_r \sum_{s=1}^{q} (\tilde{J}_k(x_k^s, r) - \beta_k^s)^2 \tag{3.5}$$

The next question is how to choose the most suitable class of functions, which is called *approximation architecture*. It is obvious that approximation architecture can greatly affect the performance of the approximation and the difficulty of training. The most popular architecture is *neural networks*, which are widely used in *reinforcement learning*, but the optimization process is difficult and the optimality is not guaranteed. We will start with a simpler linear feature-based approximation architecture.

3.2.2.1 Linear feature-based architecture

In this architecture, the $\tilde{J}_k$ function could be parameterized as follows:

$$\tilde{J}_k(x_k, r_k) = r_k^T \phi_k(x_k) \tag{3.6}$$

Here T means the transpose of the matrix, $\phi_k(x_k)$ is pre-selected and called the (non-linear) feature vector associated with x_k at time k. The scalar components of the feature vector are called *features*. Common examples of features include polynomials and radial basis functions. The notion of feature is commonly used in the theory of computer vision, where x_k could be interpreted as an image, and the ϕ_k function extracts the critical features such as angles and points that could be used for object recognition or image alignment. By using this architecture, the fitted value iteration (3.4), (3.5) greatly simplifies and admits a closed-form solution.

Example 3.1 (Swinging up a pendulum using feature-based method)**.** EXPERIMENT HERE

It is worth noting that the cost-to-go function in Linear Quadratic Regulator is also using the linear architecture mentioned above.

$$J(x, S) = x^T S x \tag{3.7}$$

It is quadratic in x but linear in S. Therefore, except for using the Recatti equation to solve for S, we could also try to use fitted value iteration to handle it.

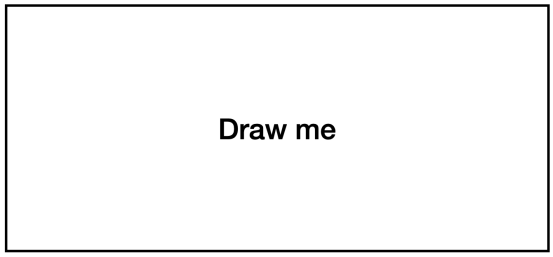

Figure 3.1: example of swinging up a pendulum using feature-based value function approximation

3.2.2.2 Neural networks

The selection of features is frequently manually crafted, relying on human intellect, intuition, or experience, and can pose considerable challenges. The utilization of a neural network as the approximation architecture has emerged as a popular approach in recent years. In this context, the parameter r_k may correspond to the weights of the neural networks. A diverse range of machine learning techniques can then be implemented to manage the training problem, steering the approximation toward the optimal value.

However, the optimization process of the weights is more difficult due to the non-convexity. According to the equation (3.3), the J_{k+1} is non-convex which makes the entire equation hard to optimize.

Example 3.2 (Swinging up a pendulum using NN-based method)**.** EXPERIMENT HERE

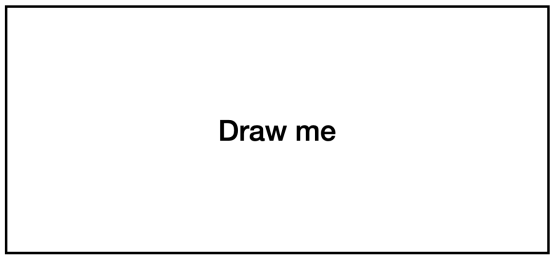

Figure 3.2: example of swinging up a pendulum using NN-based value function approximation

3.2.3 Online approximate optimization

Different from previous sections, in this section, we will discuss *online* approaches for computing the one-step lookahead control u_k just after the current state x_k becomes known. Here, to compute u_k, the values $\tilde{J}_{k+1}(x_{k+1})$ need only

be computed at the relevant next states x_{k+1} (the ones that can occur following application of u_k).

A particularly effective online approach is *rollout*. In rollout algorithm, $\tilde{J}_{k+1}(x_{k+1})$ is calculated by a *suboptimal policy*, or *base policy*. $\tilde{J}_{k+1}$ could be calculated either analytically or by Monte Carlo simulation. This part is interconnected with *model predictive control (MPC)*, which we will also discuss at the end of this section.

3.2.3.1 Rollout algorithm

The essence of the rollout is policy improvement, which generates a better policy on top of the base policy. In the rollout algorithm, $\tilde{J}_{k+1}$ is the cost-to-go of some known suboptimal policy $\pi = \{\mu_0, ..., \mu_{N-1}\}$, referred to as *base policy*. The policy $\bar{\pi} = \{\bar{\mu}_0, ..., \bar{\mu}_{N-1}\}$ thus obtained is called the *rollout policy* based on π. In short, *the rollout* policy is the one-step lookahead policy, with the optimal cost-to-go approximated by the cost-to-go of the base policy_.

Definition 3.1 (One-step Rollout Algorithm) We can get an improved policy from the base policy π

$$\bar{\mu}_k(x_k) = \arg \min_{u_k \in \mathbb{U}_k(x_k)} E\left\{g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, w_k))\right\} \tag{3.8}$$

where $\tilde{J}_{k+1}$ is the corresponding cost-to-go function of the base policy π. If we use H_{k+1} to represent the cost-to-go function of the base policy π, the rollout algorithm will be:

$$\bar{\mu}_k(x_k) = \arg \min_{u_k \in \mathbb{U}_k(x_k)} E\left\{g_k(x_k, u_k, w_k) + H_{k+1}(f_k(x_k, u_k, w_k))\right\} \tag{3.9}$$

In the control system, after the current state x_k is revealed, we calculate the cost-to-go function H_{k+1} of the known base policy π and conduct one-step lookahead minimization to find $\bar{\mu}_k(x_k)$ and feed it into the system immediately.

Note that it is also possible to define the rollout policy that makes use of multistep lookahead. While such multistep lookahead involves much more online computation, it will likely yield better performance than its one-step counterpart. In what follows, we concentrate on rollout policy with one-step lookahead.

Theorem 3.1 (Cost improvement property of rollout algorithm)**.** *It is possible to show that the rollout policy's performance is no worse than the one of the base policy, while some special conditions must hold to guarantee this cost improvement property. Here we introduce the* sequential improvement *condition. We say that the base policy has sequential improvement property if, for all x_k and k, we have*

$$\min_{u_k \in \mathbb{U}_k(x_k)} \{g_k(x_k, u_k) + H_{k+1}(f_k(x_k, u_k))\} \leq H_k(x_k) \tag{3.10}$$

where $H_k(x_k)$ denotes the cost of the base policy starting from x_k. Here we use deterministic problems to make our proof concise. Sometimes people also use the Q factor *mentioned below:*

$$\tilde{Q}_k(x_k, u_k) = g_k(x_k, u_k) + H_{k+1}(f_k(x_k, u_k)) \tag{3.11}$$

so now the sequential improvement property could also be written as:

$$\min_{u_k \in \mathbb{U}_k(x_k)} \tilde{Q}_k(x_k, u_k) \leq H_k(x_k) \tag{3.12}$$

We will now show that the rollout algorithm obtained with a base policy with sequential improvement property yields no worse cost than the base policy. In particular, consider the rollout policy $\tilde{\pi} = \{\tilde{\mu}_0, ..., \tilde{\mu}_{N-1}\}$, and let $J_{k,\tilde{\pi}}(x_k)$ denote the cost obtained with $\tilde{\pi}$ starting from x_k. We claim that

$$J_{k,\tilde{\pi}}(x_k) \leq H_k(x_k), \textit{for all } x_k \textit{ and } k \tag{3.13}$$

Proof. We prove this inequality by induction. Clearly it holds for $k = N$, since $J_{N,\tilde{\pi}} = H_N = g_N$. Assume it holds for index $k + 1$. We have:

$$\begin{aligned}
\tilde{J}_{k,\tilde{\pi}}(x_k) &= g_k(x_k, \tilde{\mu}_k(x_k)) + J_{k+1,\tilde{\pi}}(f_k(x_k, \tilde{\mu}_k(x_k))) && (3.14)\\
&\leq g_k(x_k, \tilde{\mu}_k(x_k)) + H_{k+1}(f_k(x_k, \tilde{\mu}_k(x_k))) && (3.15)\\
&= \min_{u_k \in \mathbb{U}_k(x_k)} [g_k(x_k, u_k) + H_{k+1}(f_k(x_k, u_k))] && (3.16)\\
&= \min_{u_k \in \mathbb{U}_k(x_k)} \tilde{Q}_k(x_k, u_k) && (3.17)\\
&\leq H_k(x_k) && (3.18)
\end{aligned}$$

where:

a. The first equality is the DP equation for the rollout policy $\tilde{\pi}$.

b. The first inequality holds by the induction hypothesis.

c. The second equality holds by the definition of the rollout algorithm.

d. The second inequality holds by the sequential improvement property.

This completes the induction proof of the cost improvement property (3.13). □

Computational issues in rollout algorithms. In the rollout algorithm, the cost-to-go function H_{k+1} of the base policy is required to be computed online at all possible next states $f_k(x_k, u_k, w_k)$. However, the real-time constraint will be a critical problem, for the corresponding cost-to-go function of a given base policy is not easy to calculate in real-time. In most cases, we will use the approximate version of the cost-to-go $\tilde{H}_{k+1}$ to simplify the calculation. So the rollout algorithm will be:

$$\bar{\mu}_k(x_k) = \arg \min_{u_k \in \mathbb{U}_k(x_k)} E\left\{g_k(x_k, u_k, w_k) + \tilde{H}_{k+1}(f_k(x_k, u_k, w_k))\right\} \tag{3.19}$$

There are two variants to handle the computational difficulties, deterministic case, and stochastic case.

1. *Deterministic case.* If the problem is deterministic, the calculation is greatly simplified.
2. *Stochastic case.* In these cases, the $\tilde{H}_{k+1}$ are evaluated online by Monte Carlo simulation for all $u_k \in \mathbb{U}_k(x_k)$.

Truncated rollout algorithm with multistep lookahead and terminal cost approximation. We may incorporate multistep lookahead into the rollout framework. Let us start with a two-step lookahead for deterministic problems. Suppose that after k steps we have reached state x_k. We then consider the set of all two-step-ahead states x_{k+2}, run the base policy starting from each of them, and compute the two-stage cost to get from x_k to x_{k+2}, plus the cost of the base policy from x_{k+2}. We select the state, say $\tilde{x}_{k+2}$, that is associated with minimum cost, compute the controls $\tilde{u}_k$ and $\tilde{u}_{k+1}$ that lead from x_k to $\tilde{x}_{k+2}$, and choose $\tilde{u}_k$ as the next rollout control and $x_{k+1} = f_k(x_k, \tilde{u}_k)$ as the next state.

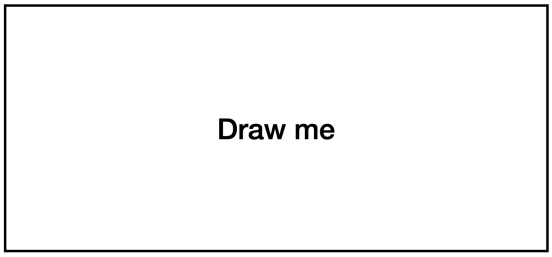

Figure 3.3: Illustration of truncated rollout with two-step lookahead

The extension of the algorithm to lookahead of more than two steps is straightforward: instead of the two-step-ahead states x_{k+2} we run the base policy starting from all the possible l-step ahead states x_{k+l}, etc.

An important variation for problems with a large number of stages is *truncated rollout with terminal cost approximation*. Here the rollout trajectories are obtained by running the base policy from the leaf nodes of the lookahead tree, and

they are truncated after a given number of steps, while a terminal cost approximation is added to the policy cost to compensate for the resulting error. One possibility that works well for many problems is to simply set the terminal cost approximation to zero. Alternatively, the terminal cost function approximation may be obtained by problem approximation or by using some sophisticated offline training process that may involve an approximation architecture such as a neural network.

3.2.3.2 Model predictive control (MPC)

In this section, we will discuss a popular control algorithm called *model predictive control (MPC)*. We will start by considering the case where the objective is to keep the state close to the origin (or more generally some point of interest, called the *set point*, or *fixed point*); this is called the *regulation problem*. Similar approaches have been developed for the problem of maintaining the state of a non-stationary system along a given state trajectory, and also, with appropriate modifications, to control problems involving disturbances. In particular, in some cases, the trajectory is treated like a sequence of set points, and the subsequently described algorithm is applied repeatedly.

We will consider a deterministic system

$$x_{k+1} = f_k(x_k, u_k) \tag{3.20}$$

whose state x_k and control u_k are vectors that consist of a finite number of scalar components. The cost per stage is assumed nonnegative

$$g_k(x_k, u_k) \geq 0, for\ all\ (x_k, u_k) \tag{3.21}$$

(e.g., a quadratic cost). We impose state and control constraints

$$x_k \in \mathbb{X}_k, u_k \in \mathbb{U}_k(x_k), k = 0, 1, ... \tag{3.22}$$

We also assume that the system can be kept at the origin at zero cost, i.e.,

$$f_k(0, \bar{u}_k) = 0, g_k(0, \bar{u}_k) = 0 \tag{3.23}$$

for some control $\bar{u}_k \in \mathbb{U}_k(0)$. This is a characteristic that all fixed points possess.

For a given initial state $x_0 \in \mathbb{X}_0$, we want to obtain a sequence $\{u_0, u_1, ...\}$ such that the states and controls of the system satisfy the state and control constraints with a small total cost.

The MPC algorithm. Let us describe the MPC algorithm for the deterministic problem just described. At the current state x_k:

1. MPC solves an l-step lookahead version of the problem, which requires that $x_{k+l} = 0$.
2. If $\{\tilde{u}_k, ..., \tilde{u}_{k+l-1}\}$ is the optimal control sequence of this problem, MPC applies $\tilde{u}_k$ and discards the other controls $\tilde{u}_{k+1}, ..., \tilde{u}_{k+l-1}$.
3. At the next stage, MPC repeats this process, once the next state x_{k+1} is revealed.

In some literature, this MPC algorithm is also called *Receding Horizon Control* algorithm, or RHC for short. One obvious drawback of this method is the online computation time limit. The MPC algorithm needs to solve an optimization problem online, which is time-consuming and does not guarantee a solution.

To make the connection between MPC and rollout, we first recap the case of the truncated rollout algorithm. In a truncated rollout algorithm with multistep lookahead and terminal cost approximation,

$$\min_{u_k \in \mathbb{U}_k(x_k), ..., u_{k+l} \in \mathbb{U}_{k+l}(x_{k+l})} \left\{ \sum_{i=k}^{k+l} g_i(x_i, u_i) + \sum_{i=k+l+1}^{k+l+m} g_i(x_i, \mu_i(x_i)) + \tilde{J}(x_{k+l+m+1}) \right\} \tag{3.24}$$

such that

$$x_{i+1} = f_i(x_i, u_i) \tag{3.25}$$

The control u_k will be used as the control at step k (online current step). All the x_i are admissible states. Here $\tilde{J}$ means the terminal cost approximation, which can be obtained through offline computation or sometimes be set to zero. l means the number of lookahead steps, and m means the number of steps that the base policy runs to evaluate the cost-to-go function H_{k+l+1}. Let us discuss a special case, where the $\tilde{J}$ is set to zero while m is also zero. So now the rollout algorithm becomes:

$$\min_{u_k \in \mathbb{U}_k(x_k), ..., u_{k+l} \in \mathbb{U}_{k+l}(x_{k+l})} \sum_{i=k}^{k+l} g_i(x_i, u_i) \tag{3.26}$$

while u_k still be used as the current online control, and all other optimized controls are discarded. We can see that now it is *almost* the case of model predictive control, without the terminal state constraint (in this case the terminal state constraint is $x_{k+l+1} = 0$). This constraint is also called *recursive feasibility*, for it guarantees the optimization will not suddenly encounter a situation where the solver returns "infeasible".

3.3 Approximation in policy space

A major alternative to approximation in value space is *approximation in policy space*, whereby we select the policy from a suitably restricted class of policies, usually a parametric class of some form. In particular, we can introduce a parametric family of policies

$$\mu_k(x_k, r_k), k = 0, ..., N-1 \tag{3.27}$$

where r_k is a parameter, such as a family represented by a neural network, and then estimate the parameters r_k using some type of optimization.

An important advantage of approximation in policy space is that the computation of controls during the online operation of the system is often much easier compared with the lookahead minimization (3.3). In this section, we will present two distinct approaches for computing r: *training by cost optimization* and *training by using an expert*.

3.3.1 Training by using an expert

This approach is pretty similar to *supervised learning* in machine learning. We r_k by "training" on a large number of sample state-control pairs $(x_k^s, u_k^s), s = 1, ..., q$, such that for each s, u_k^s is a "good" control at state x_k^s. This can be done for example by solving for each k the least squares problem

$$\min_{r_k} \sum_{s=1}^{q} \left\| u_k^s - \tilde{\mu}_k(x_k^s, r_k) \right\|^2 \tag{3.28}$$

(possibly with added regularization). In particular, we may determine u_k^s by a human or a software "expert" that can choose "near-optimal" controls at the given states x_k^s, so $\tilde{\mu}_k$ is trained to match the behavior of the expert. Of course, in the expert training approach, we cannot expect to obtain a controller that performs better than the expert with which it is trained.

The "near-optimal" controls of sampled states $x_k^s, s = 1, ..., q$ could also be calculated from one-step lookahead minimization with a suitable approximation $\tilde{J}_{k+1}$.

$$u_k^s = \arg \min_{u_k \in \mathbb{U}_k(x_k)} \mathbb{E}\left\{ g_k(x_k^s, u_k, w_k) + \tilde{J}_{k+1}(f_k(x_k^s, u_k, w_k)) \right\} \tag{3.29}$$

3.3.2 Training by cost optimization

POLICY GRADIENT

3.4 Extension

It is possible for a suboptimal control scheme to employ both types of approximation: in policy space and in value space, with a distinct architecture for each case. This is known as the simultaneous use of a "policy network" (or "actor network") and a "value network" (or "critic network"), each with its own set of parameters. Simultaneous approximation in policy space and value space through the use of deep neural networks are central in AlphaGo and AlphaZero, DeepMind's Go and chess playing programs.

Chapter 4

Stability Analysis

Optimal control formulates a control problem via the language of mathematical optimization. However, there are control problems, and sometimes even the very basic control problems, that cannot be easily stated in the optimal control formulation.

For example, suppose our goal is to *swing up a pendulum to the upright position and stabilize it there*. You may want to formalize the problem as

$$\min_{u(t)\in\mathbb{U}} \int_0^\infty \|x(t) - x_d\|^2 dt, \quad \text{subject to} \quad \dot{x} = f(x,u), x(0) = x_0, \tag{4.1}$$

where x_d is the desired upright position for the pendulum. However, does the solution of problem (4.1), if exists, guarantee the stabilization of the pendulum at the upright position? The answer is unclear without a rigorous proof.

However, after a slight change of perspective, the optimal control problem may be formulated to better match the goal. Suppose there exists a region, Ω, in the state space such that as long as the pendulum enters Ω, there always exists a sequence of control to bring the pendulum to the goal state x_d, then we can simply formulate a different optimal control problem

$$\min_{u(t)\in\mathbb{U}} \int_0^T \|u(t)\|^2 dt, \quad \text{subject to} \quad x(0) = x_0, x(T) \in \Omega, \dot{x} = f(x,u), \tag{4.2}$$

where now it is very clear, if a solution exists to problem (4.2), then we will definitely achieve our goal. This is because the constraint $x(T) \in \Omega$ guarantees that we will be able to stabilize the pendulum, and the cost function of (4.2) simply encourages minimum control effort along the way.

This highlights that, sometimes the formulation of a problem may deserve more thoughts than the actual solution. Of course the formulation (4.2) may be much

more difficult to solve. In fact, does the set Ω exist, and if so, how to describe it?

This is the main focus of this chapter: to introduce tools that can help us analyze the *stability* of uncontrolled and controlled nonlinear systems. Specifically, we will introduce the notion of *stability certificates*, which are conditions that, if hold, certify the stability of the system (e.g., in the set Ω). Interestingly, you will see that the notion of stability certificates is intuitive and easy, but what is really challenging is to *find* and *compute* the stability certificates. We will highlight the power and also limitation of computational tools, especially those that are based on convex optimization (see Appendix B for a review of convex optimization).

4.1 Autonomous Systems

Let us first focus on autonomous systems, i.e., systems whose dynamics do not depent on time (and control). We introduce different concepts of stability and ways to certify them.

4.1.1 Concepts of Stability

Consider the autonomous system

$$\dot{x} = f(x) \tag{4.3}$$

where $x \in \mathbb{X} \subseteq \mathbb{R}^n$ is the state and $f : \mathbb{R}^n \to \mathbb{R}^n$ is the (potentially nonlinear) dynamics.

Before talking about concepts of stability, we need to define an *equilibrium point*.

Definition 4.1 (Equilibrium Point)**.** A state $x^\star$ is called an equilibrium point of system (4.3) if $f(x^\star) = 0$, i.e., once the system reaches $x^\star$, it stays at $x^\star$.

For example, a linear system

$$\dot{x} = Ax$$

has a single equilibrium point $x^\star = 0$ when A is nonsingular, and an infinite number of equilibrium points when A is singular (those equilibrium points lie in the kernel of matrix A).

When analyzing the behavior of a dynamical system around the equilibrium point, it is often helpful to "shift" the dynamics equation so that 0 is the equilibrium point. For example, if we are interested in the behavior of system (4.3) near the equilibrium point $x^\star$, we can create a new variable

$$z = x - x^\star,$$

so that

$$\dot{z} = \dot{x} = f(x) = f(z + x^\star). \tag{4.4}$$

Clearly, $z^\star = 0$ is an equilibrium point for the shifted system (4.4).

Let us find the equilibrium points of a simple pendulum.

Example 4.1 (Equilibrium Points of A Simple Pendulum)**.** Consider the dynamics of an uncontrolled pendulum

$$\begin{cases} \dot{\theta} = \dot{\theta} \\ \ddot{\theta} = -\frac{1}{ml^2}(b\dot{\theta} + mgl\sin\theta) \end{cases} \tag{4.5}$$

where θ is the angle between the pendulum and the vertical line, and $x = [\theta, \dot{\theta}]^T$ is the state of the pendulum (m, g, l, b denote the mass, gravity constant, length, and damping constant, respectively).

To find the equilibrium points of the pendulum, we need the right hand sides of (4.5) to be equal to zero:

$$\dot{\theta} = 0, \quad -\frac{1}{ml^2}(b\dot{\theta} + mgl\sin\theta) = 0.$$

The solutions are easy to find

$$x^\star = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} \pi \\ 0 \end{bmatrix},$$

corresponding to the bottomright and upright positions of the pendulum, respectively.

The pendulum dynamics has two equilibrium points, but our physics intuition tells us these two equilibrium points are dramatically different. Specifically, the bottomright equilibrium $x^\star = [0, 0]^T$ is such that if you perturb the pendulum around the equilibrium, the pendulum will go back to that equilibrium; the upright equilibrium $x^\star = [\pi, 0]^T$ is such that if you perturb the pendulum (even just a little bit) around the equilibrium, it will diverge from that equilibrium.

This physical intuition is exactly what we want to formalize as the concepts of stability.

In the following, we focus on the nonlinear autonomous system (4.3) with $f(0) = 0$, i.e., $x^\star = 0$ is an equilibrium point. We now formally define the different concepts of stability.

Definition 4.2 (Lyapunov Stability)**.** The equilibrium point $x = 0$ is said to be *stable in the sense of Lyapunov* if, for any $R > 0$, there exists $r > 0$ such that if $\|x(0)\| < r$, then $\|x(t)\| < R$ for all $t \geq 0$. Otherwise, the equilibrium point is unstable.

For a system that is Lyapunov stable around $x = 0$, the definition says that, if we want to constrain the trajectory of the system to be within the ball $B_R = \{x \mid \|x\| < R\}$, then we can always find a smaller ball $B_r = \{x \mid \|x\| < r\}$ such that if the system starts within B_r, it will remain in the larger ball B_R.

On the other hand, if the system is not Lyapunov stable at $x = 0$, then there exists at least one ball B_R, such that no matter how close the system's initial condition is to the origin, it will eventually exit the ball B_R. The following exercise is left for you to verify the instability of the Van der Pol oscillator.

Exercise 4.1 (Instability of the Van der Pol oscillator)**.** Show that the Van der Pol oscillator

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -x_1 + (1 - x_1^2)x_2 \end{cases}$$

is unstable at the equilibrium point $x = 0$.

Lyapunov stability does not guarantee the system trajectory will actually converge to $x = 0$. Instead, asymptotic stability will ask the system trajectory to converge to $x = 0$.

Definition 4.3 (Asymptotic Stability and Domain of Attraction)**.** The equilibrium point $x = 0$ is said to be *asymptotically stable* if (i) it is Lyapunov stable, and (ii) there exists some $r > 0$ such that $x(0) \in B_r$ implies $x(t) \to 0$ as $t \to 0$.

The domain of attraction (for the equilibrium $x = 0$) is the largest set of points in the state space such that trajectories initiated at those points will converge to the equilibrium point. That is,

$$\Omega(x^\star) = \{x \in \mathbb{X} \mid x(0) = x \Longrightarrow \lim_{t\to\infty} x(t) = x^\star\}.$$

The ball B_r is *a* domain of attraction for the equilibrium point $x = 0$, but not necessarily the largest domain of attraction.

You may immediately realize that in the definition of asymptotic stability, we require Lyapunov stability to hold first. Is this necessary? i.e., does there exist a system where trajectories eventually converge to zero, but is not stable in the sense of Lyapunov? You should work out the following exercise.

Exercise 4.2 (Vinograd System)**.** Show that for the Vinograd dynamical system (Vinograd, 1957)

$$\begin{cases} \dot{x} = \frac{x^2(y-x)+y^5}{(x^2+y^2)(1+(x^2+y^2)^2)} \\ \dot{y} = \frac{y^2(y-2x)}{(x^2+y^2)(1+(x^2+y^2)^2)} \end{cases},$$

all system trajectories converge to the equilibrium point $(x, y) = 0$, but the equilibrium point is not stable in the sense of Lyapunov.

(Hint: the system trajectories will behave like the following plot.)

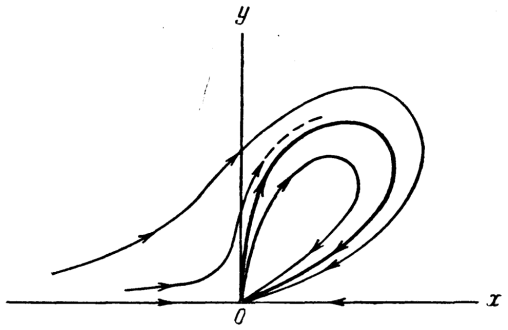

Figure 4.1: Trajectories of the Vinograd system. Copied from the original article of Vinograd.

In many cases, we want the convergence of the system trajectory towards $x = 0$ to be fast, thus bringing in the notion of exponential stability.

Definition 4.4 (Exponential Stability)**.** An equilibrium point $x = 0$ is said to be exponentially stable, if there exists a ball B_r such that as long as as $x(0) \in B_r$, then

$$\|x(t)\| \leq \alpha \|x(0)\| e^{-\lambda t}, \quad \forall t,$$

for some $\alpha > 0$ and $\lambda > 0$ (λ is called the rate of exponential convergence).

Exponential stability implies asymptotic stability (and certainly also Lyapunov stability). What is nice about exponential stability is that we can quantify the distance of the system trajectory to the equilibrium point as a function of time (as long as we know the constants $\alpha, \|x(0)\|, \lambda$). In many safety-critical applications, we need such performance guarantees. For example, in Chapter 5.1, we will see the application of exponential stability in observer-feedback control.

All the concepts of stability we have mentioned so far only talk about the stability of the system *locally* around the equilibrium point $x = 0$ (via arguments like B_r and B_R). It would be much nicer if we can guarantee stability of the system *globally*, i.e., no matter where the system starts in the state space $\mathbb{X}$, its trajectoy will converge to $x = 0$.

Definition 4.5 (Global Asymptotic and Exponential Stability)**.** The equilibrium point $x = 0$ is said to be globally asymptotically (exponentially) stable if asymptotic (exponential) stability holds for any initial states. That is,

$$\forall x \in \mathbb{X}, \quad x(0) = x \Longrightarrow \begin{cases} \lim_{t\to\infty} x(t) = 0 & \text{global asymptotic stability} \\ \exists \alpha, \lambda > 0, \text{ s.t. } \|x(t)\| \leq \alpha \|x(0)\| e^{-\lambda t} & \text{global exponential stability} \end{cases}$$

This concludes our definitions of stability for nonlinear systems (Definition 4.2-4.5). It is worth mentioning that the concepts of stability are complicated (refined) here due to our focus on nonlinear systems. For linear systems, the concepts of stability are simpler. Specifically, all local stability properties of linear systems are also global and asymptotic stability is equal to exponential stability. In fact, for a linear time-invariant system $\dot{x} = Ax$, it is either asymptotically (exponentially) stable, or marginally stable, or unstable. Moreover, we can fully characterize the stability property by inspecting the eigenvalues of A (you can find a refreshment of this in Appendix A.1).

How do we characterize the stability property of a nonlinear system? If someone gave me a nonlinear system (4.3), how can I provide a certificate to her that the system is stable or unstable (I cannot use eigenvalues anymore in this case)? Let us describe some of these certificates below.

4.1.2 Stability by Linearization

A natural idea is to linearize, if possible, the nonlinear system (4.3) at a given equilibrium point $x^\star$ and inspect the stability of the linearized system (for which we can compute eigenvalues). Therefore, the key question here is how does the stability and instability of the linearized system relate to the stability and instability of the original nonlinear system.

Theorem 4.1 (Stability by Linearization)**.** *Assume $x = 0$ is an equilibrium point of system* (4.3) *and f is continuously differentiable. Let*

$$\dot{x} = Ax, \quad A = \frac{\partial f}{\partial x}\Big|_{x=0} \tag{4.6}$$

be the linearized system at $x = 0$. The following statements are true about the stability relationship between (4.3) *and* (4.6)*.*

- *If the linearized system* (4.6) *is strictly stable (i.e., all eigenvalues of A have strictly negative real parts), then the original system* (4.3) *is asymptotically stable at $x = 0$.*
- *If the linearized system* (4.6) *is unstable (i.e., at least one eigenvalue of A has strictly positive real part), then the original system* (4.3) *is unstable at $x = 0$.*
- *If the linearized system* (4.6) *is marginally stable (i.e., all eigenvalues of A have nonpositive real parts, and at least one eigenvalue has zero real part), then the stability of the original system* (4.3) *at $x = 0$ is indeterminate.*

Theorem 4.1 is actually quite useful when we want to quickly examine the local stability of a nonlinear system around a given equilibrium point, as we will show in the next example.

Example 4.2 (Stability of A Simple Pendulum by Linearization)**.** Consider the simple pendulum dynamics (4.5) in Example 4.1. Without loss of generality, let $m = 1, l = 1, b = 0.1$. The Jacobian of the nonlinear dynamics reads

$$A = \frac{\partial f}{\partial x} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l}\cos\theta & -\frac{b}{ml^2} \end{bmatrix}.$$

At the bottomright equilibrium point $\theta = 0, \dot{\theta} = 0$, the matrix A has two eigenvalues

$$-0.0500 \pm 3.13i,$$

and hence the pendulum is asymptotically stable at the bottomright equilibrium point.

At the upright equilibrium point $\theta = \pi, \dot{\theta} = 0$, the matrix A has two eigenvalues

$$3.08, \quad -3.18,$$

and hence the pendulum is unstable at the upright equilibrium point.

The linearization method is easy to carry out. However, it tells us nothing about global stability or exponential stability. Moreover, when the linearized system is marginally stable, the stability of the orignal system is inconclusive. In the next, we will introduce a more general, and perhaps the most popular framework for analyzing the stability of nonliear systems.

4.1.3 Lyapunov Analysis

The basic idea of Lyapunov analysis is quite intuitive: if can find an "energy-like" scalar functin for a system such that the scalar function is zero at an equilibrium point and positive everywhere else, and the time-derivative of the scalar function is zero at the equilibrium point but negative otherwise, then we know that the energy of the system will eventually converge to zero, and hence the state trajectory will converge to the equilibrium point. Lyapunov analysis was originally inspired by the energy function of a mechanical system: the total energy of a mechanical system (potental energy plus kinetic energy) will settle down to its minimum value if it is constantly dissipated (e.g., due to damping). However, the concept of a Lyapunov function is much broader than the energy function, i.e., it can be an arbitrary abstract function without any physical meaning.

Let us now introduce the concept of a Lyapunov function.

Definition 4.6 (Positive Definite Function)**.** A scalar function $V(x)$ is said to be locally positive definite in a ball B_R if

$$V(0) = 0 \quad \text{and} \quad V(x) > 0, \forall x \in B_R \backslash \{0\},$$

and globally positive definite if

$$V(0) = 0 \quad \text{and} \quad V(x) > 0, \forall x \in \mathbb{X}\backslash\{0\},$$

where $\mathbb{X}$ is the entire state space.

A function $V(x)$ is said to be negative definite if $-V(x)$ is positive definite.

A function $V(x)$ is said to be positive semidefinite if the "$>$" sign is replaced by the "$\geq$" sign in the above equations.

A function $V(x)$ is said to be negative semidefinite if $-V(x)$ is positive semidefinite.

For example, when $\mathbb{X} = \mathbb{R}^2$, the function $V(x) = x_1^2 + x_2^2$ is positive definite, but the function $V(x) = x_1^2$ is only positive semidefinite.

Definition 4.7 (Lyapunov Function)**.** In the ball B_R, if a function $V(x)$ is positive definite, and its time derivative along any system trajectory

$$\dot{V}(x) = \frac{\partial V}{\partial x} f(x)$$

is negative semidefinite (we assume the partial derivative $\frac{\partial f}{\partial x}$ exists and is continuous), then $V(x)$ is said to be a Lyapunov function for system (4.3). Note that $\dot{V}(x^\star) = 0$ at any equilibrium point $x^\star$ by definition.

With the introduction of positive definite and Lyapunov functions, we are now ready to use them to certify different concepts of stability.

Theorem 4.2 (Lyapunov Local Stability)**.** *Consider the nonlinear system* (4.3) *in a ball B_R with equilibrium point $x = 0$, if there exists a scalar function $V(x)$ (with continuous partial derivatives) such that*

- *$V(x)$ is positive definite (in B_R)*
- *$\dot{V}(x)$ is negative semidefinite (in B_R)*

then the equilibrium point $x = 0$ is stable in the sense of Lyapunov (cf. Definition 4.2).

Moreover,

- *if $\dot{V}(x)$ is negative definite in B_R, then the equilibrium point is asymptotically stable (cf. Definition 4.3).*
- *if $\dot{V}(x) \leq -\alpha V(x)$ for any $x \in B_R$, then the equilibrium point is exponentially stable (cf. Definition 4.4).*

Let us apply Theorem 4.2 to the simple pendulum.

Example 4.3 (Lyapunov Local Stability for A Simple Pendulum)**.** Consider the pendulum dynamics (4.5). The total energy of a pendulum is

$$V(x) = \frac{1}{2}ml^2\dot{\theta}^2 + mgl(1-\cos\theta). \tag{4.7}$$

Clearly, $V(x)$ is positive definite on the entire state space, and the only point where $V(x) = 0$ is the equilibrium point $\theta = 0, \dot{\theta} = 0$.

Let us compute the time derivative of $V(x)$:

$$\dot{V}(x) = ml^2\dot{\theta}\ddot{\theta} + mgl\sin\theta\dot{\theta} = ml^2\dot{\theta}\left(-\frac{1}{ml^2}(b\dot{\theta} + mgl\sin\theta)\right) + mgl\sin\theta\dot{\theta} = -b\dot{\theta}^2,$$

which is clearly negative semidefinite. In fact, $\dot{V}(x)$ is precisely the energy dissipation rate due to damping. By Theorem 4.2 we conclude that the equilibrium point is stable in the sense of Lyapunov.

Note that with this choice of $V(x)$ as in (4.7), we actually cannot certify asymptotic local stability of the bottomright equilibrium point. So a natural question is, can we find a better Lyapunov function that indeed certifies asymptotic stability?

The answer is yes. Consider a different Lyapunov function

$$\tilde{V}(x) = \frac{1}{2}ml^2\dot{\theta}^2 + \frac{1}{2}ml^2\left(\frac{b}{ml^2}\theta + \dot{\theta}\right)^2 + 2mgl(1-\cos\theta), \tag{4.8}$$

which is positive definite and admits a single zero-value point $\theta = 0, \dot{\theta} = 0$ that is also the bottomright equilibrium point. Simplifying $\tilde{V}(x)$ we can get

$$\tilde{V}(x) = ml^2\dot{\theta}^2 + 2mgl(1-\cos\theta) + \frac{1}{2}ml^2\left(\frac{b^2}{m^2l^4}\theta^2 + \frac{2b}{ml^2}\theta\dot{\theta}\right) \tag{4.9}$$

$$= 2V(x) + \frac{1}{2}ml^2\left(\frac{b^2}{m^2l^4}\theta^2 + \frac{2b}{ml^2}\theta\dot{\theta}\right). \tag{4.10}$$

The time derivative of the new function $\tilde{V}(x)$ is

$$\dot{\tilde{V}}(x) = 2\dot{V}(x) + \frac{ml^2}{2}\left(\frac{2b^2}{m^2l^4}\theta\dot{\theta} + \frac{2b}{ml^2}(\dot{\theta}^2 + \theta\ddot{\theta})\right) \tag{4.11}$$

$$= 2\dot{V}(x) + b\dot{\theta}^2 + \left(\frac{b^2}{ml^2}\theta\dot{\theta} + b\theta\left(-\frac{1}{ml^2}(b\dot{\theta} + mgl\sin\theta)\right)\right) \tag{4.12}$$

$$= -b\left(\dot{\theta}^2 + \frac{g}{l}\theta\sin\theta\right). \tag{4.13}$$

$\dot{\tilde{V}}(x)$ is negative definite locally around the equilibrium point (locally $\sin\theta \approx \theta$). Therefore, with the new Lyapunov function $\tilde{V}(x)$ we can certify asymptotic stability.

Interestingly, $V(x)$ is intuitive (the total energy of the pendulum system), but it fails to certify asymptotic local stability (as least by just using Theorem 4.2). $\tilde{V}(x)$ does not have any physical intuition, but it successfully certifies local asymptotic stability.

In Section 4.1.4, we will see that when using $V(x)$ with the invariant set theorem, we can actually still certify the asymptotic stability of the pendulum around the bottomright equilibrium.

In many applications, we desire to certify the global stability of an equilibrium point. The following theorem states that if in addition the scalar function $V(x)$ is *radially unbounded*, then global stability can be certified.

Theorem 4.3 (Lyapunov Global Stability)**.** *For the autonomous system* (4.3)*, suppose there exists a scalar function $V(x)$ with (continuous partial derivatives) such that*

- *$V(x)$ is positive define;*
- *$\dot{V}(x)$ is negative define;*
- *$V(x) \to \infty$ as $\|x\| \to \infty$,*

then the equilibrium point $x = 0$ is globally asymptotically stable (cf. Definition 4.5).

Moreover, if in addition to the three conditions above

- *$\dot{V}(x) \leq -\alpha V(x)$ for some $\alpha > 0$, then the equilibrium point is globally exponentially stable.*

4.1.4 Invariant Set Theorem

Through Theorem 4.2, Theorem 4.3, and Example 4.3, we see that in order to certify asymptotic stability, the time derivative $\dot{V}(x)$ is required to be positive definite. However, in many cases, with Example 4.3 being a typical one, $\dot{V}(x)$ is only negative semidefinite, which makes it difficult to certify asymptotic stability.

In this section, we will introduce the invariant set theorem that can help us reason about asymptotic stability even when $\dot{V}(x)$ is only negative semidefinite.

Let us first introduce the notion of an invariant set.

Definition 4.8 (Invariant Set)**.** A set G is an invariant set for a dynamical system (4.3) if every system trajectory that starts within G remains in G for all future time. Formally,

$$x(0) \in G \Longrightarrow x(t) \in G, \forall t.$$

A trivial invariant set if the entire state space $\mathbb{X}$. Another example of an invariant set is the singleton $\{x^\star\}$ with $x^\star$ being an equilibrium point. A nontrivial invariant set is the domain of attraction of an equilibrium point (cf. Definition 4.3).

We now state the local invariant set theorem.

Theorem 4.4 (Local Invariant Set)**.** *Consider the autonomous system* (4.3)*, and let $V(x)$ be a scalar function with continuous partial derivatives. Assume that*

- *the sublevel set $\Omega_\rho = \{x \in \mathbb{X} \mid V(x) < \rho\}$ is bounded for some $\rho > 0$, and*
- *$V(x) \leq 0$ for all $x \in \Omega_\rho$.*

Let $\mathcal{R}$ be the set of all points within Ω_ρ such that $\dot{V}(x) = 0$, and $\mathcal{M}$ be the largest invariant set in $\mathcal{R}$. Then, every trajectory that starts in Ω_ρ will converge to $\mathcal{M}$ as $t \to \infty$.

With this theorem, we can now revisit the pendulum example 4.3.

Example 4.4 (Revisiting the Local Stability of A Simple Pendulum)**.** In Example 4.3, using the Lyapunov function

$$V(x) = \frac{1}{2}ml^2\dot{\theta}^2 + mgl(1 - \cos\theta),$$

with time derivative

$$\dot{V}(x) = -b\dot{\theta}^2,$$

we were only able to verify the stability of the bottomright equilibrium point in the sense of Lyapunov.

Now let us use the invariant set theorem 4.4 to show the asymptotic stability of the bottomright equilibrium point.

First it is easy to see that the sublevel set of $V(x)$ is bounded. For example, with $\rho = \frac{1}{4}mgl$,

$$V(x) < \frac{1}{4}mgl \Rightarrow \frac{1}{2}ml^2\dot{\theta}^2 < \frac{1}{4}mgl \Rightarrow \dot{\theta}^2 < \frac{1}{2}\frac{g}{l} \tag{4.14}$$

$$V(x) < \frac{1}{4}mgl \Rightarrow mgl(1 - \cos\theta) < \frac{1}{4}mgl \Rightarrow \cos\theta > \frac{3}{4} \Rightarrow \theta \in (-\arccos\frac{3}{4}, \arccos\frac{3}{4}). \tag{4.15}$$

The set $\mathcal{R}$, including all the points in Ω_ρ such that $\dot{V}(x) = 0$ is

$$\mathcal{R} = \{x \in \Omega_\rho \mid \dot{\theta} = 0\}.$$

We now claim that the largest invariant set $\mathcal{M}$ in $\mathcal{R}$ is just the single equilibrium point $x = [0, 0]^T$. We can prove this by contradiction. Suppose there is a different point $x' = [\theta, 0]^T$ with $\theta \neq 0$ also belonging to the invariant set $\mathcal{M}$, then

$$\ddot{\theta} = -\frac{1}{ml^2}(b\dot{\theta} + mgl\sin\theta) = -\frac{g}{l}\sin\theta \neq 0,$$

which means $\dot{\theta}$ will immediately become nonzero, and hence the trajectory will exit $\mathcal{R}$ and also $\mathcal{M}$. So that point cannot belong to the invariant set.

Now by Theorem 4.4, we conclude the bottomright equilibrium point is asymptotically stable.

Note that through this analysis we also obtain Ω_ρ as a domain of attraction for the bottomright equilibrium point.

Similarly, with the addition of the radial unboundedness of $V(x)$, we have a global version of the invariant set theorem.

Theorem 4.5 (Global Invariant Set)**.** *For the autonomous system* (4.3)*, let* $V(x)$ *be a scalar function with continuous partial derivatives that satisfies*

- $V(x) \to \infty$ *as* $\|x\| \to \infty$*, and*
- $\dot{V}(x) \leq 0$ *over the entire state space.*

Let $\mathcal{R} = \{x \in \mathbb{X} \mid \dot{V}(x) = 0\}$*, and* $\mathcal{M}$ *be the largest invariant set in* $\mathcal{R}$*. Then all system trajectories asymptotically converge to* $\mathcal{M}$ *as* $t \to \infty$*.*

4.1.5 Computing Lyapunov Certificates

All the Theorems we have stated so far (Theorems 4.2, 4.3, 4.4, and 4.5) are very general and powerful tools for certifying stability of nonlinear systems. However, the key requirement for applying the results is a Lyapunov function $V(x)$ that verifies different types of nonnegativity constraints.

How to find these functions?

In Example 4.3, we have seen that physical intuition can help us find a good Lyapunov function (4.7). Nevertheless, it did not quite give us what we want in terms of asymptotic stability. Instead, a hand-crafted function (4.8) helped us certify local asymptotic stability.

Wouldn't it be cool that we can design an algorithm to find the Lyapunov certificates for us?

A closer look at the Theorems 4.2, 4.3, 4.4, and 4.5 tells us the key property of a Lyapunov certificate is that it needs to satisfy the positivity (or negativity) constraint for all states inside a set. This is a nontrivial and difficult requirement, because even if we were given a function $V(x)$, naively evaluting if $V(x)$ is nonnegative inside a set requires enumeration over all the states in the set,

which is impractical given that the set is continuous and has infinite number of states.[1] When the dynamics (4.3) is linear, searching for Lyapunov functions is well understood and presented in Appendix A.1. However, when the dynamics is nonlinear, things can get very complicated.

In the next, I want to introduce a general framework for searching Lyapunov certificates for nonlinear systems that is based on convex optimization.

This framework, although having deep connections with many other disciplines such as algebraic geometry, theoretical computer science, and mathematical optimization, is based on a very simple intution that we all have since high school.

Example 4.5 (A Simple Example for Certifying Nonnegativity)**.** Suppose I give you a polynomial of a single variable $x \in \mathbb{R}$

$$p(x) = x^2 + 2x + 1$$

and ask you if $p(x) \geq 0$ for all x. You would not hesitate to answer "yes", because you know

$$p(x) = (x+1)^2$$

is the square of $x+1$ and hence must be nonnegative.

Let me make it more challenging. Suppose I give you a different polynomial

$$p(x) = -x^4 + 2x^2 + x + 1$$

and ask you if $p(x)$ is nonnegative for any $x \in [-1, 1]$ (instead of any $x \in \mathbb{R}$). At first glance, it seems much harder to answer this question because (i) we have a constraint set $x \in [-1, 1]$, and (ii) the polynomial $p(x)$ has a higher degree and it is not a polynomial that we are very famliar with (compared to $p(x) = x^2 + 2x + 1$).

However, if I show you that $p(x)$ can be written as

$$p(x) = -x^4 + 2x^2 + 2x + 1 = (x+1)^2 + x^2(1 - x^2), \tag{4.16}$$

it becomes easy again to certify that $p(x)$ is nonnegative for any $x \in [-1, 1]$. Why?

1. First notice that $(x+1)^2 \geq 0$ for any $x \in \mathbb{R}$,
2. Then notice that $1 - x^2 \geq 0$ for any $x \in [-1, 1]$, and $x^2 \geq 0$ for any x. Therefore, $x^2(1 - x^2) \geq 0$ for any $x \in [-1, 1]$.

Combining the above two reasonings, it becomes clear $p(x)$ is nonnegative for any $x \in [-1, 1]$.

[1] In fact, many of the recent works verify "neural" Lyapunov certificates (and other types of certificates) using this idea, see for example (Dawson et al., 2023).

What we have learned from this simple example is that

> Given a polynomial $p(x)$ and a constraint set $x \in \mathcal{X} \subseteq \mathbb{R}^n$, if we can write $p(x)$ as a sum of a finite number of products
>
> $$p(x) = \sum_{i=1}^{K} \sigma_i(x) g_i(x)$$
>
> where $\sigma_i(x)$ is a polynomial that we know is always nonnegative for any $x \in \mathbb{R}^n$ (just like $(x+1)^2$ and x^2 in (4.16)), and $g_i(x)$ is a polynomial that we know is always nonnegative for any x in the constraint set $\mathcal{X}$ (just like $1 - x^2$ for the set $[-1, 1]$ in (4.16)), then we have a certificate that $p(x) \geq 0$ for any $x \in \mathcal{X}$.

With this simple intuition, let me now formalize the framework of sum of squares (SOS) certificates for proving nonnegativity (also known as *positivstellensatz*, or in short P-satz).

Positivstellensatz, Sum of Squares, and Convex Optimization

Basic Semialgebraic Set. Let $x = [x_1, \dots, x_n] \in \mathbb{R}^n$ be a list of variables, we define a *basic semialgebraic set* as

$$\mathcal{X} = \{x \in \mathbb{R}^n \mid p_i(x) = 0, i = 1, \dots, l_{\text{eq}}; p_i(x) \geq 0, i = l_{\text{eq}} + 1, \dots, l_{\text{eq}} + l_{\text{ineq}}\} \tag{4.17}$$

where $p_i(x), i = 1, \dots, l_{\text{eq}} + l_{\text{ineq}}$ are polynomial functions in x. In other words, the set $\mathcal{X}$ is a subset of $\mathbb{R}^n$ that is defined by l_{eq} equality constraints and l_{ineq} inequality constraints.

Observe that a basic semialgebraic set can capture a lot of the common constraint sets, such as a unit sphere, a unit ball, and a box (try this for yourself).

Positivstellensatz. We are now given the same question as in Example 4.5. Suppose I give you another polynomial function $p_0(x)$, how can you tell me if $p_0(x)$ is nonnegative for any x in the basic semialgebraic set $\mathcal{X}$? That is, to verify if

$$p_0(x) \geq 0, \quad \forall x \in \mathcal{X}.$$

Formalizing the intution obtained from Example 4.5, you will say if someone can produce a decomposition of $p_0(x)$ as

$$p_0(x) = \sigma_0(x) + \sum_{i=1}^{l_{\text{ineq}}} \sigma_i(x) p_{i+l_{\text{eq}}}(x) + \sum_{i=1}^{l_{\text{eq}}} \lambda_i(x) p_i(x), \tag{4.18}$$

where $\sigma_0, \sigma_1, \dots, \sigma_{l_{\text{ineq}}}$ are "some type of" polynomials that we know are always nonnegative (for any $x \in \mathbb{R}^n$), and $\lambda_1, \dots, \lambda_{l_{\text{eq}}}$ are arbitrary polynomials. Then I have a "certificate" that $p_0(x) \geq 0$ for any $x \in \mathcal{X}$.

Why? The reasoning is exactly the same as before.

1. $\sigma_0(x) \geq 0$ for any x,
2. $\sigma_i(x)p_{i+l_{\text{eq}}}(x) \geq 0, i = 1, \ldots, l_{\text{ineq}}$ for any $x \in \mathcal{X}$, because (a) $\sigma_i(x) \geq 0$ for any x, and (b) $p_{i+l_{\text{eq}}}(x) \geq 0$ for any $x \in \mathcal{X}$ by definition of the basic semialgebraic set (4.17),
3. $\lambda_i(x)p_i(x) = 0, i = 1, \ldots, l_{\text{eq}}$ for any $x \in \mathcal{X}$ by definition of the basic semialgebraic set (4.17).

We call σ_i's "nonnegative polynomial multipliers", and λ_i's "polynomial multipliers".

Sum-of-Squares. Now it comes the key question: what type of polynomials should we choose as the nonnegative polynomial multipliers? Ideally, this type of polynomials should

a. be always (trivially) nonnegative, and
b. have a nice representation for its unknown parameters (coefficients).

Looking back at our choice of multipliers, i.e., $(x+1)^2$ and x^2 in Example 4.5, it is natural to come up with the choice of a "sum-of-squares" (SOS) polynomial.

Definition 4.9 (Sum-of-Squares Polynomial)**.** A polynomial $\sigma(x)$ is called an SOS polynomial if

$$\sigma(x) = \sum_{i=1}^{k} q_i^2(x),$$

i.e., $\sigma(x)$ can be written as a sum of k squared polynomials.

OK, an SOS polynomial is trivially nonnegative (satisfying requirement (a) above), but does it have a nice representation for its parameters? The following Lemma gives us an affirmative answer.

Lemma 4.1 (SOS Polynomial and Positive Semidefinite Matrix)**.** *A polynomial $\sigma(x)$ is SOS if and only if*

$$\sigma(x) = [x]_d^T Q [x]_d$$

for some $Q \succeq 0$, where $[x]_d$ is the vector of monomials in x of degree up to d. For example, if $x \in \mathbb{R}^2$ and $d = 2$, then

$$[x]_2 = [1, x_1, x_2, x_1^2, x_1x_2, x_2^2]^T.$$

With the choice of $\sigma(x)$ as SOS polynomials, we are now ready to explicitly

search for a nonnegativity certificate in the form of (4.18):

$$\begin{aligned} \text{find} \quad & \{\sigma_i\}_{i=0}^{l_{\text{ineq}}}, \{\lambda_i\}_{i=1}^{l_{\text{eq}}} \\ \text{subject to} \quad & p_0(x) = \sigma_0(x) + \sum_{i=1}^{l_{\text{ineq}}} \sigma_i(x) p_{i+l_{\text{eq}}}(x) + \sum_{i=1}^{l_{\text{eq}}} \lambda_i(x) p_i(x), \\ & \sigma_i \text{ is SOS}, i = 0, \dots, l_{\text{ineq}}, \\ & \lambda_i \text{ is polynomial}, i = 1, \dots, l_{\text{eq}} \\ & \deg(\sigma_0) \le 2\kappa, \deg(\sigma_i p_{i+l_{\text{eq}}}) \le 2\kappa, i = 1, \dots, l_{\text{ineq}}, \\ & \deg(\lambda_i p_i) \le 2\kappa, i = 1, \dots, l_{\text{eq}}. \end{aligned} \tag{4.19}$$

Bounding the Degree. The careful reader realizes that in (4.19) we have added constraints on the degrees of the polynomial multipliers σ_i's and λ_i's.[2] Precisely, we choose an integer κ, which we call the relaxation order, such that

$$2\kappa \ge \max\{\deg(p_i(x))\}_{i=0}^{l_{\text{eq}}+l_{\text{ineq}}},$$

and restrict the products $\sigma_i p_{i+l_{\text{eq}}}$'s and $\lambda_i p_i$'s to have degrees at most 2κ. With this, we are explicitly limiting the degrees of the multipliers σ_i's and λ_i's, and hence asking the formulation (4.19) to search for a finite number of parameters (otherwise, if the degree of the multipliers is unbounded, then the number of parameters to be searched is infinite).

Convex Optimization. The last crucial (and surprising) observation is that the problem (4.19) is a convex optimization! This is due to the following three reasons

a. The polynomial multipliers λ_i's can be fully parametrized by their coefficients, and these coefficients can be arbitrary vectors. Precisely, if $\lambda(x)$ is a polynomial with degree up to d, then

$$\lambda(x) = c^T [x]_d,$$

where $[x]_d$ is the vector of monomials in x of degree up to d, and c is the vector of coefficients.

b. The SOS multipliers σ_i's can be fully parametrized by their coefficients, and these coefficients are positive semidefinite matrices, according to Lemma 4.1.

c. The equality constraint of decomposing $p_0(x)$ as a sum of products in (4.19) therefore becomes a set of affine equality constraints on the parameters of λ_i's and σ_i's, by matching coefficients of the monomials on the left-hand size and the right-hand side.

[2]The degree of a monomial is the sum of its exponents. For example, $\deg(x_1 x_2^4 x_3^2) = 1 + 4 + 2 = 7$. The degree of a polynomial is the maximum degree of its monomials. For example, the polynomial $p(x) = 1 + x_2 + x_1^2 x_2^3$ has three monomials with degrees 0, 1, and 5, respectively. Therefore, $\deg(p) = 5$.

Therefore, the problem (4.19) is a convex semidefinite program (SDP). There are multiple software packages, e.g., SOSTOOLS, YALMIP, SumOfSquares.py, that allow us to model our problem in the form of (4.19), convert the formulation into SDPs, and pass them to SDP solvers (such as MOSEK). We will see an example of this soon.

Extensions. I want to congratulate, and welcome you to enter the world of SOS relaxations! Like I said before, this is an active area of research and the framework I just introduced is just a tip of the iceberg. Therefore, before I end this tutorial, I want to point out several extensions of the SOS framework.

- **Necessary Condition**. We have seen that a decomposition in the form of (4.19) is a *sufficient* condition to prove the nonnegativity of $p_0(x)$. Is it also a *necessary* condition? That is, for any $p_0(x)$ that is nonnegative on the set $\mathcal{X}$, does it admit a decomposition in the form of (4.19)? In general, the answer is no, and there exist nonnegative polynomials that cannot be written in the form of SOS decompositions (e.g., the Motzkin's polynomial). However, with certain assumptions on the set $\mathcal{X}$, the decomposition (4.19) is also necessary for nonnegativity! A well-known assumption is called the Archimedean condition (which, roughly speaking, requires the set $\mathcal{X}$ to be compact)). I suggest you to read (Blekherman et al., 2012) for more details.

- **Global Polynomial Optimization**. The SOS framework can be used for global optimization of polynomials in a straightforward way. Consider the polynomial optimization problem (POP)

$$\min_{x\in\mathcal{X}} p_0(x),$$

where one seeks the global minimum of the polynomial $p_0(x)$ on the set $\mathcal{X}$. A POP is generally a nonconvex optimization problem, and it is difficult to obtain a globally optimal solution. However, with a slight change of perspective, we can write the problem above equivalently as

$$\begin{aligned} \max \quad & \gamma \\ \text{subject to} \quad & p_0(x) - \gamma \geq 0, \quad \forall x \in \mathcal{X}. \end{aligned} \tag{4.20}$$

Basically I want to push the lower bound γ as high as possible. The constraint in (4.20) asks $p_0(x)-\gamma$ to be nonnegative on $\mathcal{X}$. With the SOS framework introduced above, we can naturally relax it to

$$\begin{aligned} \max \quad & \gamma \\ \text{subject to} \quad & p_0(x) - \gamma \quad \text{is SOS on} \quad \mathcal{X}, \end{aligned} \tag{4.21}$$

where the "SOS on $\mathcal{X}$" constraint is exactly the problem (4.19). Therefore, we have relaxed the nonconvex optimization (4.20) into a convex problem (4.21)! Moreover, by increasing the relaxation order κ, we obtain a sequence of lower bounds that asymptotically converge to the true global

optimum of the nonconvex problem (4.20). This is called Lasserre's hierarchy of moment-SOS relaxations, originally proposed by Lasserre in the seminal work (Lasserre, 2001). As this name suggests, the dual problem to the SOS relaxation (4.21) is called the moment relaxation. Lasserre's hierarchy has recently gained a lot of attention due to the empirical observation in many engineering disciplines that the convergence to global optimum is finite, i.e., by solving the convex problem (4.21) at a finite relaxation order κ, an exact global optimizer of the original nonconvex problem (4.20) can be extracted. For a pragmatic introduction to the moment relaxation, I suggest to read Section 2.2 of (Yang and Carlone, 2022). For more applications of Lasserre's hierarchy, please refer to (Lasserre, 2009).

- **Scalability**. I have to warn you that there is no free lunch. The fact that so many challenging problems can be relaxed or restated as convex optimization problems should send you an alert. Does this mean that we can use convex optimization to solve all the challenging problems? Well, although we hope this is the case, in practice we are limited by the computational resources. The caveat is that the problem (4.19) and (4.21), despite being convex, grows very large as the dimension n and relaxation order κ increases. Another way of saying this is that, we seek to solve small-to-medium scale nonconvex problems with large-scale convex problems. Unfortunately, today's SDP solver cannot solve all the problems we formulate, and hence a major research direction in the mathematical optimization community is to develop SDP solvers that are more scalable. You can read (Yang et al., 2022) and references therein for more details.

- **Non-SOS Certificates**. Nobody is preventing us to use a different choice of nonnegative polynomial multipliers (other than SOS multipliers) in (4.18). For example, one can use a decomposition as the sum of nonnegative circuit polynomials (Wang, 2022) or signomials (Murray et al., 2021). However, to the best of my knowledge, non-SOS certificates are far less popular than SOS certificates.

There are many other extensions to the SOS framework, and a complete enumeration is beyond the scope of this lecture notes. For the connection between SOS and theoretical computer science, you can see the lecture notes by Boaz Barak and David Steurer. There are also more recent monographs about SOS, for example (Magron and Wang, 2023) and (Nie, 2023). I plan to introduce these in more details in an upcoming graduate-level class at Harvard.

That was a long detour from Lyapunov analysis! The SOS machinery will come back later when we study multiple other topics in optimal control and estimation. But now let us show how to tackle the problem of computing Lyapunov certificates using the SOS machinery.

According to Theorem 4.2, given a set $\mathcal{X}$ that contains an equilibrium point $x^\star$, if we can find a Lyapunov function $V(x)$ such that $V(x)$ is positive definite on $\mathcal{X}$ and $\dot{V}(x)$ is negative definite on $\mathcal{X}$, then the equilibrium point $x^\star$ is locally

asymptotically stable. With the SOS machinery, we can search for a $V(x)$ that is a polynomial as

$$\begin{aligned} \text{find} \quad & V(x) && (4.22)\\ \text{subject to} \quad & V(x) - \epsilon_1 \|x - x^\star\|^2 \quad \text{is SOS on} \quad \mathcal{X} && (4.23)\\ & -\epsilon_2 \|x - x^\star\|^2 - \frac{\partial V(x)}{\partial x} f(x) \quad \text{is SOS on} \quad \mathcal{X} && (4.24)\\ & V(x^\star) = 0, && (4.25) \end{aligned}$$

where $\epsilon_1, \epsilon_2 > 0$ are (small) positive constants. This is a convex optimization problem, just like (4.19) (try to convince yourself my claim is true). Similarly, we can choose a relaxation order κ and solve the above problem. If a solution exists, then we find a valid Lyapunov certificate.

Let us apply it to the simple pendulum to synthesize local stability certificates.

Example 4.6 (Computing Lyapunov Local Stability Certificate for the Simple Pendulum with Convex Optimization)**.** The SOS framework works with polynomials, so let us first write the pendulum dynamics in polynomial form via a change of coordinate $x = [\mathfrak{s}, \mathfrak{c}, \dot{\theta}]^T$ with $\mathfrak{s} = \sin\theta$, $\mathfrak{c} = \cos\theta$:

$$\begin{cases} \dot{\mathfrak{s}} = \mathfrak{c}\dot{\theta} \\ \dot{\mathfrak{c}} = -\mathfrak{s}\dot{\theta} \\ \ddot{\theta} = -\frac{1}{ml^2}(b\dot{\theta} + mgl\mathfrak{s}) \end{cases}.$$

We will use $m = 1, l = 1, b = 0.1$ for our numerical experiment.

We want to find a local Lyapunov certificate in the compact set

$$\theta \in \left[-\arccos\frac{3}{4}, \arccos\frac{3}{4}\right], \quad \dot{\theta} \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]. \tag{4.26}$$

In the new coordinates x, this is equivalent to the semialgebraic set

$$\mathcal{X} = \left\{ x \in \mathbb{R}^3 \mid \mathfrak{s}^2 + \mathfrak{c}^2 = 1, \dot{\theta}^2 \le \frac{\pi^2}{4}, \mathfrak{c} \ge \frac{3}{4} \right\}.$$

Denoting the bottomright equilibrium point as $x_e = [0, 1, 0]^T$, and with $\epsilon_1, \epsilon_2 > 0$ two positive constants, we can seek a Lyapunov function $V(x)$ that satisfies the following conditions

$$\begin{aligned} V(x) &\ge \epsilon_1 (x - x_e)^T (x - x_e), \quad \forall x \in \mathcal{X} && (4.27)\\ \dot{V}(x) = \frac{\partial V}{\partial x}\dot{x} &\le -\epsilon_2 (x - x_e)^T (x - x_e), \quad \forall x \in \mathcal{X} && (4.28)\\ V(x_e) &= 0, \quad \dot{V}(x_e) = 0 && (4.29) \end{aligned}$$

where (4.27) ensures $V(x)$ is positive definite, (4.28) ensures $\dot{V}(x)$ is negative definite, and (4.29) ensures $V(x), \dot{V}(x)$ vanish at the equilibrium point.

To leverage the power of convex optimization, we can relax the positivity constraints as SOS constraints

$$V(x) - \epsilon_1 (x - x_e)^T (x - x_e) \quad \text{is SOS on} \quad \mathcal{X} \tag{4.30}$$

$$-\epsilon_2 (x - x_e)^T (x - x_e) - \frac{\partial V}{\partial x} \dot{x} \quad \text{is SOS on} \quad \mathcal{X} \tag{4.31}$$

$$V(x_e) = 0, \quad \dot{V}(x_e) = 0. \tag{4.32}$$

If we limit the degree of V to 2, choose the relaxation order $\kappa = 2$, and $\epsilon_1 = \epsilon_2 = 0.01$, we obtain a solution

$$V(x) = 2.7982\mathfrak{s}^2 + 0.086248\mathfrak{s}\dot{\theta} + 2.4548\mathfrak{c}^2 + 0.88117\dot{\theta}^2 - 16.6277\mathfrak{c} + 14.1728$$

with the time derivative

$$\dot{V}(x) = 0.68675\mathfrak{s}\mathfrak{c}\dot{\theta} + 0.086248 * \mathfrak{c}\dot{\theta}^2 - 0.84523\mathfrak{s}^2 - 0.65191\mathfrak{s}\dot{\theta} - 0.17623\dot{\theta}^2.$$

Plotting $V(x)$ in the constraint set (4.26) using $(\theta, \dot{\theta})$ coordinates, we get

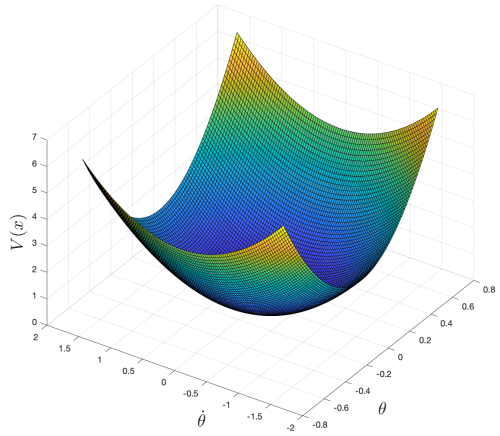

Figure 4.2: Lyapunov local stability certificate computed via convex optimization.

and verify that $V(x)$ is locally positive definite.

Plotting $\dot{V}(x)$ in the constraint set (4.26) using $(\theta, \dot{\theta})$ coordinates, we get

and verify that $\dot{V}(x)$ is locally negative definite.

You should try the code for this example here.

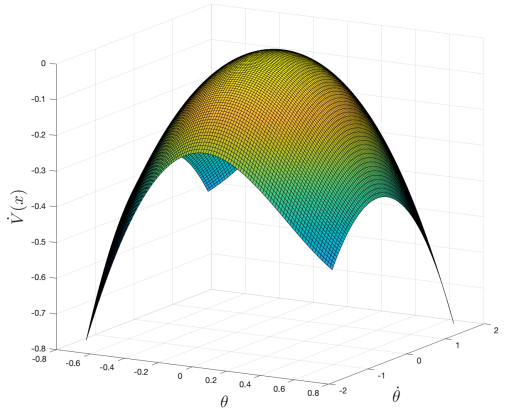

Figure 4.3: Derivative of the lyapunov local stability certificate computed via convex optimization.

4.2 Controlled Systems

4.3 Non-autonomous Systems

Lemma 4.2 (Barbalat's Lemma)**.** *Let $f(t)$ be differentiable, if*

- $\lim_{t\to\infty} f(t)$ *is finite, and*
- $\dot{f}(t)$ *is uniformly continuous,*[3]

then

$$\lim_{t\to\infty} \dot{f}(t) = 0.$$

Theorem 4.6 (Barbalat's Stability Certificate)**.** *If a scalar function $V(x,t)$ satisfies*

- $V(x,t)$ *is lower bounded,*
- $\dot{V}(x,t)$ *is negative semidefinite*
- $\dot{V}(x,t)$ *is uniformly continuous*

then $\dot{V}(x,t) \to 0$ as $t \to \infty$.

Proof. $V(x,t)$ is lower bounded and $\dot{V}$ is negative semidefinite implies the limit of V as $t \to \infty$ is finite (note that $V(x,t) \leq V(x(0),0)$). Then the theorem clearly follows from Barbalat's Lemma 4.2. □

[3] A sufficient condition for this to hold is that $\ddot{f}$ exists and is bounded.

Chapter 5

Output Feedback

Consider a continuous-time dynamical system

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= h(x,u) \end{aligned} \tag{5.1}$$

where $x(t) \in \mathbb{X} \subseteq \mathbb{R}^n$ the state of the system, $u(t) \in \mathbb{U} \subseteq \mathbb{R}^m$ the control (or input), $y(t) \in \mathbb{Y} \subseteq \mathbb{R}^d$ the output (i.e., measurement) of the state and control, and f, g the evolution and measurement functions (which are sufficiently smooth).

5.1 State Observer

For the system (5.1), let us denote

- $X(x_0, t_0; t; u)$ the solution at time t with input u and initial condition x_0 at time t_0; when $t_0 = 0$, we write $X(x_0; t; u)$
- $Y(x_0, t_0; t; u)$ the output at time t with input u and initial condition x_0 at time t_0, i.e., $Y(x_0, t_0; t; u) = h(X(x_0, t_0; t; u), u(t))$; when $t_0 = 0$, we write $y_{x_0,u}(t)$;
- $\mathcal{X}_0$ a subset of $\mathbb{X}$ containing the initial conditions we consider; for any $x_0 \in \mathcal{X}_0$, we write $\sigma^+_{\mathcal{X}}(x_0; u)$ the maximal time of existence of $X(x_0, \cdot; t; u)$ in a set $\mathcal{X}$
- $\mathcal{U}$ the set of all sufficiently many times differentiable inputs $u : [0, +\infty) \to \mathbb{U}$.

The problem of state observation is to produce an estimated state $\hat{x}(t)$ of the true state $X(x_0, t_0; t; u)$ based on knowledge about the system (5.1) and information about the history of inputs $u_{[0,t]}$ and outputs $y_{[0,t]}$, so that $\hat{x}(t)$ asymptotically

converges to $X(x_0, t_0; t; u)$, for any initial condition $x_0 \in \mathcal{X}_0$ and any input $u \in \mathcal{U}$.

There are multiple ways for solving the problem of state observation (see e.g., (Bernard, 2019), (Bernard et al., 2022)). Here we are particularly interested in the approach using a *state observer*, i.e., a dynamical system whose *internal state* evolves according to the history of inputs and outputs, from which a state estimation can be reconstructed that guarantees asymptotic convergence to the true state. We formalize this concept below.

Definition 5.1 (State Observer)**.** A state observer for system (5.1) is a couple $(\mathcal{F}, \mathcal{T})$ such that

1. $\mathcal{F} : \mathbb{R}^l \times \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}^l$ is continuous
2. $\mathcal{T}$ is a family of continuous functions indexed by $u \in \mathcal{U}$ where each $\mathcal{T}_u : \mathbb{R}^l \times [0, +\infty) \to \mathbb{R}^n$ respects the causality condition
$$\forall \tilde{u} : [0, +\infty) \to \mathbb{R}^m, \forall t \in [0, +\infty), u_{[0,t]} = \tilde{u}_{[0,t]} \Rightarrow \mathcal{F}_u(\cdot, t) = \mathcal{F}_{\tilde{u}}(\cdot, t).$$
3. For any $u \in \mathcal{U}$, any $z_0 \in \mathbb{R}^l$, and any $x_0 \in \mathcal{X}_0$ such that $\sigma^+_{\mathbb{X}}(x_0; u) = +\infty$, any solution $Z(z_0; t; u, y_{x_0,u})$[1] to
$$\dot{z} = \mathcal{F}(z, u, y_{x_0,u}) \tag{5.2}$$
initialized at z_0 at time 0 with input u and $y_{x_0,u}$ exists on $[0, +\infty)$ and satisfies
$$\lim_{t \to \infty} \|\hat{X}(x_0, z_0; t; u) - X(x_0; t; u)\| = 0, \tag{5.3}$$
with
$$\hat{X}(x_0, z_0; t; u) = \mathcal{T}_u(Z(z_0; t; u, y_{x_0,u}), t). \tag{5.4}$$
In words, (i) the state observer maintains an internal state (or latent state) $z \in \mathbb{R}^l$ that evolves according to the latent dynamics $\mathcal{F}$ in (5.2), where u and $y_{x_0,u}$ are inputs; (ii) an estimated state can be reconstructed from the internal state using $\mathcal{T}_u$ as in (5.4); and (iii) the error between the estimated state and the true state (defined by a proper distance function $\|\cdot\|$ on $\mathbb{X}$) converges to zero.

If $\mathcal{T}_u$ is the same for any $u \in \mathcal{U}$ and is also time independent, then we say $\mathcal{T}$ is *stationary*.[2] In this case, we can simply write the observer (5.2) and (5.4) as

$$\begin{aligned} \dot{z} &= \mathcal{F}(z, u, y) \\ \hat{x} &= \mathcal{T}(z). \end{aligned} \tag{5.5}$$

[1] We say "any solution" because there may be several solutions to the observer (5.2) due to $\mathcal{F}$ only being continuous. This is not a problem as long as any such solution satisfies the required convergence property.

[2] The time dependence of $\mathcal{T}_u$ enables us to cover the case where the knowledge of the u and $y_{x_0,u}$ is used to construct the estimate from the observer state. In particular, using the output sometimes can reduce the dimension of the observer state (and thus alleviate the computations), thus obtaining a reduced-order observer. For example, see (Karagiannis and Astolfi, 2005) and (Astolfi and Ortega, 2003).

If $\hat{x}$ can be read off directly from z, then we say the observer (5.5) is *in the given coordinates*. A special case of this is when $\hat{x} = z$, i.e., the internal state of the observer is the same as the system state.

5.1.1 General Design Strategy

Theorem 5.1 (Meta Observer)**.** *Let $F : \mathbb{R}^p \times \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}^p$, $H : \mathbb{R}^p \times \mathbb{R}^m \to \mathbb{R}^d$ and $\mathcal{F} : \mathbb{R}^p \times \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}^p$ be continuous functions such that*

$$\dot{\hat{\xi}} = \mathcal{F}(\hat{\xi}, u, \tilde{y}) \tag{5.6}$$

is an observer for

$$\dot{\xi} = F(\xi, u, H(\xi, u)), \quad \tilde{y} = H(\xi, u), \tag{5.7}$$

i.e., for any $\xi_0, \hat{\xi}_0 \in \mathbb{R}^p$ and any $u \in \mathcal{U}$, the solution of the observer (5.6), denoted by $\hat{\Xi}(\hat{\xi}_0; t; u; \tilde{y}_{\xi_0,u})$, and the solution of the true system (5.7), denoted by $\Xi(\xi_0; t; u)$, satisfy

$$\lim_{t\to\infty} \|\hat{\Xi}(\hat{\xi}_0; t; u; \tilde{y}_{\xi_0,u}) - \Xi(\xi_0; t; u)\| = 0. \tag{5.8}$$

Note that the observer (5.6) is stationary and in the given coordinates for system (5.7). Indeed the internal state of the observer is the same as the system state.

Now suppose for any $u \in \mathcal{U}$, there exists a continuous function (i.e., coordinate transformation) $T_u : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^p$ and a subset $\mathcal{X}$ of $\mathbb{X}$ such that

1. *For any $x_0 \in \mathcal{X}_0$ such that $\sigma^+_{\mathbb{X}}(x_0; u) = +\infty$, $X(x_0; \cdot; u)$ remains in $\mathcal{X}$*
2. *There exists a concave $\mathcal{K}$[3] function ρ and a positive number $\bar{t}$ such that*

   $$\|x_a - x_b\| \le \rho(|T_u(x_a, t) - T_u(x_b, t)|), \quad \forall x_a, x_b \in \mathcal{X}, t \ge \bar{t},$$

   i.e., $x \mapsto T_u(x, t)$ becomes injective on $\mathcal{X}$,[4] uniformly in time and space, after a certain time $\bar{t}$.
3. *T_u transforms the system (5.1) into the system (5.7), i.e., for all $x \in \mathcal{X}$ and all $t \ge 0$, we have*

   $$L_{(f,1)}T_u(x, t) = F(T_u(x, t), u, h(x, u)), \quad h(x, u) = H(T_u(x, t), u), \tag{5.9}$$

   where $L_{(f,1)}T_u(x, t)$ is the Lie derivative of T_u along the vector field $(f, 1)$

   $$L_{(f,1)}T_u(x, t) = \lim_{\tau\to 0} \frac{T_u(X(x, t; t + \tau; u), t + \tau) - T_u(x, t)}{\tau}.$$

[3] A function $\rho : \mathbb{R}_+ \to \mathbb{R}_+$ is a $\mathcal{K}$ function if $\rho(0) = 0$, ρ is continuous, and ρ is increasing.

[4] An injective function is a function f that maps distinct elements of its domain to distinct elements. That is, $f(x_a) = f(x_b)$ implies $x_a = x_b$, or equivalently, $x_a \ne x_b$ implies $f(x_a) \ne f(x_b)$.

4. T_u *respects the causality condition*

$$\forall \tilde{u} : [0, +\infty) \to \mathbb{R}^m, \forall t \in [0, +\infty), u_{[0,t]} = \tilde{u}_{[0,t]} \Rightarrow T_u(\cdot, t) = T_{\tilde{u}}(\cdot, t).$$

Then, for any $u \in \mathcal{U}$*, there exists a function* $\mathcal{T}_u : \mathbb{R}^p \times [0, +\infty) \to \mathcal{X}$ *(satisfying the causality condition) such that for any* $t \geq \bar{t}$*,* $\xi \mapsto \mathcal{T}_u(\xi, t)$ *is uniformly continuous on* $\mathbb{R}^p$ *and satisfies*

$$\mathcal{T}_u(T_u(x, t), t) = x, \forall x \in \mathcal{X}.$$

Moreover, denoting $\mathcal{T}$ *the family of functions* $\mathcal{T}_u$ *for* $u \in \mathcal{U}$*, the couple* $(\mathcal{F}, \mathcal{T})$ *is an observer for the system* (5.1) *initialized in* $\mathcal{X}_0$*.*

Proof. See Theorem 1.1 in (Bernard, 2019). □

A simpler version of Theorem 5.1 where the coordinate transformation T_u is stationary and fixed for all u is stated below as a corollary.

Corollary 5.1 (Meta Observer with Fixed Transformation)**.** *Let* $F : \mathbb{R}^p \times \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}^p$*,* $H : \mathbb{R}^p \times \mathbb{R}^m \to \mathbb{R}^d$ *and* $\mathcal{F} : \mathbb{R}^p \times \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}^p$ *be continuous functions such that* (5.6) *is an observer for* (5.7)*.*

Suppose there exists a continuous coordinate transformation $T : \mathbb{R}^p \to \mathbb{R}^n$ *and a compact subset* Ω *of* $\mathbb{R}^n$ *such that*

1. *For any* $x_0 \in \mathcal{X}_0$ *such that* $\sigma^+_{\mathbb{X}}(x_0; u) = +\infty$*,* $X(x_0; \cdot; u)$ *remains in* Ω
2. $x \mapsto T(x)$ *is injective on* Ω
3. T *transforms the system* (5.1) *into system* (5.7)

$$L_f T(x) = F(T(x), u, h(x, u)), \quad h(x, u) = H(T(x), u),$$

where $L_f T(x)$ *is the Lie derivative of* $T(x)$ *along* f

$$L_f T(x) = \lim_{\tau \to 0} \frac{T(X(x, t; t + \tau; u)) - T(x)}{\tau}.$$

Then, there exists a uniformly continuous function $\mathcal{T} : \mathbb{R}^p \to \mathbb{R}^n$ *such that*

$$\mathcal{T}(T(x)) = x, \quad \forall x \in \Omega,$$

and $(\mathcal{F}, \mathcal{T})$ *is an observer for system* (5.1) *initialized in* $\mathcal{X}_0$*.*

Theorem 5.1 and Corollary 5.1 suggest the following general observer design strategy:

1. Find an injective coordinate transformation T_u (that may be time-varying and also dependent on u) that transforms the original system (5.1) with coordinate x into a new system (5.7) with coordinate ξ

2. Design an observer (5.6), $\hat{\xi}$, for the new system
3. Compute a left inverse, $\mathcal{T}_u$, of the transformation T_u to recover a state estimation $\hat{x}$ of the original system.

The transformed systems (5.7) are typically referred to as *normal forms*, or in my opinion, *templates*.

Of course, the general design strategy is rather conceptual, and in order for it to be practical, we have to answer three questions.

- What templates do we have, what are their associated observers, and what are the conditions for the observers to be asymptotically converging?
- What kinds of (nonlinear) systems can be transformed into the templates, and how to perform the transformation?
- How to invert the coordinate transformation? Is it analytical or does it require numerical approximation?

In the following sections, we will study several representative normal forms and answer the above questions. Before presenting the results, let us first introduce several notions of observability.

Definition 5.2 (Observability)**.** Consider an open subset $\mathcal{L}$ of the state space $\mathbb{X} \subseteq \mathbb{R}^n$ of system (5.1). The system (5.1) is said to be

- **Distinguishable** on $\mathcal{L}$ for some input $u(t)$, if for all $(x_a, x_b) \in \mathcal{L} \times \mathcal{L}$,

$$y_{x_a,u}(t) = y_{x_b,u}(t), \forall t \in [0, \min\{\sigma^+_{\mathbb{X}}(x_a; u), \sigma^+_{\mathbb{X}}(x_b; u)\}] \Longrightarrow x_a = x_b$$

- **Instantaneously distinguishable** on $\mathcal{L}$ for some input $u(t)$, if for all $(x_a, x_b) \in \mathcal{L} \times \mathcal{L}$, and for all $\bar{t} \in (0, \min\{\sigma^+_{\mathbb{X}}(x_a; u), \sigma^+_{\mathbb{X}}(x_b; u)\})$,

$$y_{x_a,u}(t) = y_{x_b,u}(t), \forall t \in [0, \bar{t}) \Longrightarrow x_a = x_b$$

- **Uniformly observable** on $\mathcal{L}$ if it is distinguishable on $\mathcal{L}$ for any input $u(t)$ (not only for $u \in \mathcal{U}$)
- **Uniformly instantaneously observable** on $\mathcal{L}$ if it is instantaneously observable on $\mathcal{L}$ for any input $u(t)$ (not only for $u \in \mathcal{U}$).

Moreover, let $\mathcal{X}$ be a subset of $\mathbb{X}$ such that $\mathrm{cl}(\mathcal{X})$, i.e., the closure of $\mathcal{X}$, is contained in $\mathcal{L}$. Then the system (5.1) is said to be

- **Backward $\mathcal{L}$-distinguishable on $\mathcal{X}$** for some input $u(t)$, if for any $(x_a, x_b) \in \mathcal{X} \times \mathcal{X}$ such that $x_a \neq x_b$, there exists $t \in (\max\{\sigma^-_{\mathcal{L}}(x_a; u), \sigma^-_{\mathcal{L}}(x_b; u)\}, 0]$ such that $y_{x_a,u}(t) \neq y_{x_b,u}(t)$, or in words similar to the definition of distinguishable on $\mathcal{L}$, for all $(x_a, x_b) \in \mathcal{X} \times \mathcal{X}$

$$y_{x_a,u}(t) = y_{x_b,u}(t), \forall t \in (\max\{\sigma^-_{\mathcal{L}}(x_a; u), \sigma^-_{\mathcal{L}}(x_b; u)\}, 0] \Longrightarrow x_a = x_b.$$

5.1.2 Luenberger Template

Consider an instance of the normal form (5.7) as follows:

$$\dot{\xi} = A\xi + B(u, y), \quad y = C\xi, \tag{5.10}$$

where A, C are constant matrices, and $B(u, y)$ can depend nonlinearly on u and y.

For this template, we have the well-known Luenberger observer.

Theorem 5.2 (Luenberger Observer)**.** *If the pair (A, C) is detectable (see Theorem A.3), then there exists a matrix K such that $A - KC$ is Hurwitz and the system*

$$\dot{\hat{\xi}} = A\hat{\xi} + B(u, y) + K(y - C\hat{\xi}) \tag{5.11}$$

is an observer for (5.10)*.*

Proof. Define $e(t) = \xi(t) - \hat{\xi}(t)$. In that case,

$$\dot{e}(t) = [A - KC]e(t) \tag{5.12}$$

Solving (5.12), we obtain

$$e(t) = \exp[(A - KC)t]e(0) \tag{5.13}$$

Then, if the real components of the eigenvalues of $A - KC$ are strictly negative (i.e., $A - KC$ is Hurwitz), then $e(t) \to 0$ as $t \to \infty$, independent of the initial error $e(0) = \xi(0) - \hat{\xi}(0)$. From Theorem A.3, we know that (A, C) being detectable implies the existence of K such that $A - KC$ is Hurwitz.

If one is further interested in estimating the convergence rate of the Luenberger observer, then one can use the result from Corollary A.1. Particularly, one can solve the Lyapunov equation

$$(A - KC)^T P + P(A - KC) = -I$$

to obtain P. Then the convergence rate of $\|e\|$ towards zero is $\frac{0.5}{\lambda_{\max}(P)}$. □

The Luenberger observer is an elegant result in observer design (and even in control theory) that has far-reaching impact. In my opinion, the essence of observer design is twofold: (i) to simulate the dynamics when the state estimation is correct, and (ii) to correct the state estimation from observation when it is off. These two pieces of ideas are evident in (5.11): the observer is a copy of the original dynamics ($A\hat{\xi} + B(u, y)$) plus a feedback correction from the difference between the "imagined" observation $C\hat{\xi}$ and the true observation y.

You may think the Luenberger template is restricting because it requires the system to be linear (up to the only nonlinearly in $B(u, y)$). However, it turns out the Luenberger template is already quite useful, as I will show in the following pendulum example.

Example 5.1 (Luenberger Observer for A Simple Pendulum)**.** Consider a simple pendulum dynamics model

$$x = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}, \quad \dot{x} = \begin{bmatrix} \dot{\theta} \\ -\frac{1}{ml^2}(b\dot{\theta} + mgl\sin\theta) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{ml^2} \end{bmatrix} u, \quad y = \theta, \tag{5.14}$$

where θ the angular position of the pendulum from the vertical line, $m > 0$ the mass of the pendulum, $l > 0$ the length, g the gravitational constant, $b > 0$ the dampling coefficient, and u the control input (torque).

We assume we can only observe the angular position of the pendulum in (5.14), e.g., using a camera, but not the angular velocity. Our goal is to construct an observer that can provide a full state estimation.

We first note that the pendulum dynamics (5.14) can actually be written in the (linear) Luenberger template (5.10) as[5]

$$\begin{aligned} \dot{x} &= \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & -\frac{b}{ml^2} \end{bmatrix}}_{=:A} x + \underbrace{\begin{bmatrix} 0 \\ \frac{u - mgl\sin\theta}{ml^2} \end{bmatrix}}_{=:B(u,y)}. \\ y &= \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{=:C} x \end{aligned} \tag{5.15}$$

In order for us to use the Luenberger observer, we need to check if the pair (A, C) is detectable. We can easily find the eigenvalues and eigenvectors of A:

$$A \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 0, \quad A \begin{bmatrix} -\frac{ml^2}{b} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -\frac{b}{ml^2} \end{bmatrix} = -\frac{b}{ml^2} \begin{bmatrix} -\frac{ml^2}{b} \\ 1 \end{bmatrix}.$$

The first eigenvalue has real part equal to 0. However,

$$C \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 1 \neq 0.$$

According to Theorem A.3, we conclude (A, C) is detectable. In fact, the pair (A, C) is more than just detectable, it is indeed observable (according to Theorem A.2). Therefore, the poles of $A - KC$ can be arbitrarily placed.

Finding K. Now we need to find K. An easy choice of K is

$$K = \begin{bmatrix} k \\ 0 \end{bmatrix}, \quad A - KC = \begin{bmatrix} -k & 1 \\ 0 & -\frac{b}{ml^2} \end{bmatrix}.$$

[5] I have to say I was a bit surprised when I arrived at this formulation.

With $k > 0$, we know $A - KC$ is guaranteed to be Hurwitz (the two eigenvalues of $A - KC$ are $-k$ and $-b/ml^2$), and we have obtained an observer!

We can also estimate the convergence rate of this observer. Let us use $m = 1, g = 9.8, l = 1, b = 0.1$ as parameters of the pendulm dynamics. According to Theorem 5.2, we solve the Lyapunov equation

$$(A - KC)^T P + P(A - KC) = -I$$

and $\gamma = \frac{0.5}{\lambda_{\max}(P)}$ will be our best estimate of the convergence rate (of $\|e\| = \|\hat{x} - x\|$ towards zero).

Table 5.1 below shows the convergence rates computed for different values of k. We can see that as k is increased, the convergence rate estimation is also increased. However, it appears that 0.1 is the best convergence rate we can achieve, regardless of how large k is.

Table 5.1: Convergence rate estimation of the Luenberger observer for a simple pendulm.

k	0.1	1	10	100	1000	10000
γ	0.0019	0.0523	0.0990	0.1000	0.1000	0.1000

Optimal K. Is it true that 0.1 is the best convergence rate, or in other words, what is the best K that maximizes the convergence rate γ?

A natural way (and my favorite way) to answer this question is to formulate an optimization problem.

$$\begin{aligned} \min_{P,K} \quad & \lambda_{\max}(P) \\ \text{subject to} \quad & (A - KC)^T P + P(A - KC) = -I \\ & P \succeq 0 \end{aligned} \tag{5.16}$$

The above formulation seeks the best possible K that minimizes $\lambda_{\max}(P)$ which, according to $\gamma = 0.5/\lambda_{\max}(P)$, also maximizes γ.

However, problem (5.16) is not a convex formulation due to the bilinear term PK. Nevertheless, via a simple change of variable $H = PK$, we arrive at the following convex formulation

$$\begin{aligned} \min_{P,H} \quad & \lambda_{\max}(P) \\ \text{subject to} \quad & A^T P - C^T H^T + PA - HC = -I \\ & P \succeq 0 \end{aligned} \tag{5.17}$$

Problem (5.17) is a semidefinite programming problem (SDP), that can be modeled and solved by off-the-shelf tools. We can recover $K = P^{-1}H$ from (5.17) after it is solved.

Interestingly, solving problem (5.17) verifies that the minimum $\lambda_{\max}(P)$ is 5 and the maximum converge rate is 0.1. An optimal solution of (5.17) is

$$P^\star = \begin{bmatrix} 2.4923 & 0 \\ 0 & 5 \end{bmatrix}, \quad K^\star = \begin{bmatrix} 0.2006 \\ 0.4985 \end{bmatrix}.$$

You should check out the Matlab code of this example here.

5.1.3 State-affine Template

Consider an instance of the normal form (5.7) where the dynamics is linear in ξ, but the coefficients are time-varying and dependent on the input and output

$$\dot{\xi} = A(u, y)\xi + B(u, y), \quad y = C(u)\xi. \tag{5.18}$$

The difference between the state-affine template (5.18) and the Luenberger template (5.10) is that the linear matrices A, C are allowed to depend nonlinearly on the input (u, y).

Kalman and Bucy originally proposed an observer for linear time-varying systems (Kalman and Bucy, 1961). The result is later extened by (Besançon et al., 1996) and (Hammouri and de Leon Morales, 1990) to deal with coefficient matrices dependent on the control. The following theorem is a direct extension of the result from (Besançon et al., 1996) and (Hammouri and de Leon Morales, 1990) by considering (u, y) as an augmented control input.

Before presenting the theorem, we need to introduce the following terminology.

Definition 5.3 (Linear Time-Varying System)**.** For a linear time-varying system of the form

$$\dot{\chi} = A(\nu)\chi, \quad y = C(\nu)\chi, \tag{5.19}$$

with input ν and output y, we define

- the *transition matrix* Ψ_ν as the unique solution to

$$\Psi_\nu(t, t) = I, \quad \frac{\partial \Psi_\nu}{\partial \tau}(\tau, t) = A(\nu(\tau))\Psi_\nu(\tau, t).$$

  Note that the transition matrix is used to express the solution to (5.19) because it satisfies

$$\chi(\chi_0, t_0; t; \nu) = \Psi_\nu(t, t_0)\chi_0.$$

- the *observability grammian* as

$$\Gamma_\nu(t_0, t_1) = \int_{t_0}^{t_1} \Psi_\nu(\tau, t_0)^T C(\nu(\tau))^T C(\nu(\tau))\Psi_\nu(\tau, t_0) d\tau.$$

- the *backward observability grammian* as

$$\Gamma_\nu^b(t_0, t_1) = \int_{t_0}^{t_1} \Psi_\nu(\tau, t_1)^T C(\nu(\tau))^T C(\nu(\tau)) \Psi_\nu(\tau, t_1) d\tau.$$

We now introduce the Kalman-Bucy Observer for the state-affine template (5.18).

Theorem 5.3 (Kalman-Bucy Observer)**.** *Let $y_{\xi_0,u}(t) = C(u(t))\Xi(\xi_0; t; u)$ be the output of system* (5.18) *at time t with initialization ξ_0 and control u. Suppose the control u satisfies*

- *For any ξ_0, $t \mapsto A(u(t), y_{\xi_0,u}(t))$ is bounded by $A_{\max}$*
- *For any ξ_0, the augmented input $\nu = (u, y_{\xi_0,u})$ is* regularly persistent *for the dynamics*

$$\dot{\chi} = A(\nu)\chi, \quad y = C(\nu)\chi \tag{5.20}$$

uniformly with respect to ξ_0. That is, there exist strictly positive numbers $t_0, \bar{t}$, and α such that for any ξ_0 and any time $t \geq t_0 \geq \bar{t}$,

$$\Gamma_v^b(t - \bar{t}, t) \succeq \alpha I,$$

where Γ_v^b is the backward observability grammian *associated with system* (5.20)*.*

Then, given any positive definite matrix P_0, there exist $\alpha_1, \alpha_2 > 0$ such that for any $\lambda \geq 2A_{\max}$ and any $\xi_0 \in \mathbb{R}^p$, the matrix differential equation

$$\dot{P} = -\lambda P - A(u, y)^T P - P A(u, y) + C(u)^T C(u) \tag{5.21}$$

initialized at $P(0) = P_0$ admits a unique solution satisfying $P(t) = P(t)^T$ and

$$\alpha_2 I \succeq P(t) \succeq \alpha_1 I.$$

Moreover, the system

$$\dot{\hat{\xi}} = A(u, y)\hat{\xi} + B(u, y) + K(y - C(u)\hat{\xi}) \tag{5.22}$$

with a time-varying gain matrix

$$K = P^{-1} C(u)^T \tag{5.23}$$

is an observer for the state-affine system (5.18)*.*

Let us work out an example of the Kalman-Bucy Observer for nonlinear systems.

Example 5.2 (Kalman-Bucy Observer for A Simple Pendulum)**.** Let us reconsider the pendulum dynamics (5.14) but this time try to design a Kalman-Bucy observer.

We first write the pendulum dynamics in a new coordinate system so that it is in the state-affine normal form (5.18). We choose $\xi = [\mathfrak{s}, \mathfrak{c}, \dot{\theta}]^T$ with $\mathfrak{s} = \sin\theta$ and $\mathfrak{c} = \cos\theta$. Clearly, we will be able to observe $y = [\mathfrak{s}, \mathfrak{c}]^T$ in this new coordinate. The state-affine normal form of the pendulum dynamics reads

$$\dot{\xi} = \begin{bmatrix} \mathfrak{c}\dot{\theta} \\ -\mathfrak{s}\dot{\theta} \\ -\frac{1}{ml^2}(b\dot{\theta} + mgl\mathfrak{s}) + \frac{1}{ml^2}u \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & \mathfrak{c} \\ 0 & 0 & -\mathfrak{s} \\ 0 & 0 & -\frac{b}{ml^2} \end{bmatrix}}_{=:A(u,y)} \xi + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \frac{u-mgl\mathfrak{s}}{ml^2} \end{bmatrix}}_{=:B(u,y)}. \tag{5.24}$$

$$y = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{=:C(u)} \xi$$

Note that $C(u)$ is in fact time-invariant, and $B(u, y)$ only depends on u; but we adopt the same notation as the general state-affine template (5.18).

In order to use the Kalman-Bucy observer in Theorem 5.3, we need to verify the boundedness of $A(u, y)$, and the regular persistence of (5.20).

Boundedness of $A(u, y)$. We can easily show the boundedness of $A(u, y)$ by writing

$$\|A(u,y)\xi\| = \|\xi_3(\mathfrak{c}-\mathfrak{s}-b/ml^2)\| \le |\xi_3|\sqrt{3}\sqrt{\mathfrak{c}^2+\mathfrak{s}^2+b^2/m^2l^4} \le \|\xi\|\sqrt{3+3b^2/m^2l^4}.$$

Therefore, we can take $A_{\max} = \sqrt{3 + 3b^2/m^2l^4}$.

Regular persistence. We write the backward observability grammian of system (5.20)

$$\Gamma^b_\nu(t-\bar{t}, t) = \int_{t-\bar{t}}^{t} \Psi_\nu(\tau,t)^T C^T C \Psi_\nu(\tau,t) d\tau = \int_{t-\bar{t}}^{t} \Psi_\nu(\tau,t)^T \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{=:\tilde{C}} \Psi_\nu(\tau,t) d\tau.$$

$\Gamma^b_\nu(t-\bar{t}, t) \succeq \alpha I$ if and only if

$$w^T \Gamma^b_\nu(t-\bar{t}, t) w \ge \alpha, \quad \forall w \in \mathbb{R}^3, \|w\| = 1.$$

With this, we develop $w^T\Gamma^b_\nu(t-\bar{t}, t)w$

$$\begin{aligned} w^T \Gamma^b_\nu(t-\bar{t}, t) w &= \int_{t-\bar{t}}^{t} s^T \tilde{C} s d\tau, \\ &= \int_{t-\bar{t}}^{t} (s_1^2 + s_2^2)\, d\tau, \quad s = \Psi_\nu(\tau, t) w \end{aligned} \tag{5.25}$$

and observe that $s = \Psi_\nu(\tau, t)w$ is equivalent to

$$w = (\Psi_\nu(\tau, t))^{-1}s = \Psi_\nu(t, \tau)s,$$

that is, w is the solution of $\dot{\xi} = A(\nu)\xi$ at time t with initial condition s at time $\tau \le t$. Equivalently, this is saying s is the initial condition of $\dot{\xi} = A(\nu)\xi$ at time $\tau \le t$ such that its solution at time t is w. Note that from (5.25) it is clearly that $\int_{t-\bar{t}}^{t}(s_1^2 + s_2^2)\,d\tau \ge 0$, and $\int_{t-\bar{t}}^{t}(s_1^2 + s_2^2)\,d\tau = 0$ if and only if $s_1^2 + s_2^2 = 0$, or equivalently $s_1 = s_2 = 0$ for any $\tau \in [t - \bar{t}, t]$.

We then take a closer look at the system $\dot{\xi} = A(\nu)\xi$:

$$\begin{aligned} \dot{\xi}_1 &= \mathfrak{c}\xi_3 \\ \dot{\xi}_2 &= -\mathfrak{s}\xi_3 \\ \dot{\xi}_3 &= -\frac{b}{ml^2}\xi_3. \end{aligned} \tag{5.26}$$

If the solution of ξ_3 at time t is w_3, then

$$\xi_3(\tau) = w_3 e^{\frac{b}{ml^2}(t-\tau)}, \quad \tau \le t.$$

We can now claim it is impossible that $s_1 = s_2 = 0$ at any time $\tau \in [t - \bar{t}, t]$.

We can show this by contradiction. First of all, $s_1 = s_2 = 0$ at $\tau = t$ implies $w_1 = w_2 = 0$ and hence $w_3 = \pm 1$. This implies $\xi_3 \neq 0$ for any $\tau \in [t - \bar{t}, t]$. Then, $s_1 = 0, \forall \tau \in [t - \bar{t}, t]$ implies $\dot{\xi}_1 = 0$ which, due to $\xi_3 \neq 0$, implies $\mathfrak{c} = 0$ for all τ. Similarly, $s_2 = 0, \forall \tau \in [t - \bar{t}, t]$ implies $\dot{\xi}_2 = 0$ and $\mathfrak{s} = 0$. This creates a contradiction because $\mathfrak{c}^2 + \mathfrak{s}^2 = 1$ and $\mathfrak{c}, \mathfrak{s}$ cannot be simultaneously zero.

The above reasoning proves that the backward observability Grammian is positive definite, which is, however, still insufficient for the Kalman-Bucy observer. We need a stronger uniformly positive definite condition on Γ_ν^b, i.e., to find $t_0, \bar{t}$ and $\alpha > 0$ so that $\Gamma_\nu^b(t - \bar{t}, t) \succeq \alpha I$ for all $t \ge t_0$.

If the control u is unbounded, then sadly, one can show that the uniform positive definite condition fails to hold, as left by you to show in the following exercise.

Exercise 5.1 (Counterexample for Kalman-Bucy Observer)**.** Show that, if the control u is unbounded, then for any $\alpha > 0$, $t_0 \ge \bar{t} > 0$, there exists $t \ge t_0$ such that $\Gamma_\nu^b(t - \bar{t}, t) \prec \alpha I$. (Hint: consider a controller that spins the pendulum faster and faster such that in time $\bar{t}$ it has rotated $2k\pi$, in this case the angular velocity becomes unobservable because we are not sure how many rounds the pendulum has rotated.)

Fortunately, if the control u is bounded, then we can prove the uniform positive define condition holds for $\Gamma_\nu^b(t - \bar{t}, t)$. The following proof is given by Weiyu Li.

Without loss of generality, let $\frac{b}{ml^2} = 1$. Assume u is bounded such that the third entry of $B(u, y)$ in (5.24) is bounded by $\beta > 0$

$$\left|\frac{u - mgl\mathfrak{s}}{ml^2}\right| \leq \beta.$$

Assuming the initial velocity of the pendulum is $\dot{\theta}(0) = \dot{\theta}_0$, we know $\dot{\theta}(t)$ is bounded by

$$\dot{\theta}(t) \in [c_1(1-\beta)e^{-t} - \beta, c_2(1-\beta)e^{-t} + \beta],$$

where c_1, c_2 are constants chosen to satisfy the initial condition. Clearly, for all $t > 0$, we see $\dot{\theta}(t)$ is bounded, and hence we know $\dot{\mathfrak{c}}$ and $\dot{\mathfrak{s}}$ are bounded (due to $\mathfrak{c}$ and $\mathfrak{s}$ are bounded). Intuitively, what we have just shown says that when the control is bounded, the measurements $\mathfrak{c}$ and $\mathfrak{s}$ will have bounded time derivatives. (This will help us analyze the auxiliary system (5.26).)

Now back to checking regular persistence of the auxilary system (5.26). We will discuss two cases: (1) $w_3^2 > 1 - \delta$, and (2) $w_3^2 \leq 1 - \delta$, for some constant $\delta < 0.5$ determined later.

1. $w_3^2 > 1 - \delta > 0.5$. In this case we have $w_1^2 + w_2^2 = 1 - w_3^2 < \delta$, and hence $w_1^2 < \delta$, $w_2^2 < \delta$. On the other hand, from (5.26) we have

   $$\dot{\xi}_1^2(\tau) + \dot{\xi}_2^2(\tau) = \xi_3^2 = w_3^2 e^{2(t-\tau)} > w_3^2 > 1 - \delta, \quad \forall \tau < t.$$

   Without loss of generality assume $\dot{\xi}_1(t)^2 > (1-\delta)/2$. As $\dot{\xi}_1 = \mathfrak{c}\xi_3$ and both $\mathfrak{c}$ and ξ_3 have bounded derivatives, we know $\dot{\xi}_1$ will not change sign for some duration T that is independent from the choice of δ (because the time derivatives of $\mathfrak{c}$ and ξ_3 do not depend on δ). That is $|\dot{\xi}_1| > \sqrt{(1-\delta)/2} > 1/2$ for $\tau \in [t - T, t]$. Consequently,

   $$|\xi_1(t-\tau)| > \frac{1}{2}\tau - |w_1| > \frac{1}{2}\tau - \sqrt{\delta}, \quad \tau \in [0, T].$$

   Choosing δ small enough, we have $|\xi_1(t-\tau)| > 0.25\tau$ for $\tau \in [0.5T, T]$. Then we have

   $$\Gamma_\nu^b(t - T, t) \succ [(0.25 \times 0.5T)^2 \times 0.5T]I.$$

2. $w_3^2 \leq 1 - \delta$. In this case $w_1^2 + w_2^2 = 1 - w_3^2 \geq \delta$, and at least one of w_1 and w_2 has absolute value larger than $\sqrt{\delta/2}$. Because the derivatives of ξ_1 and ξ_2 are both bounded, we know ξ_1 and ξ_2 will remain large for some constant time. Thus there is a uniform lower bound.

The intuition of the above proof is simple: when ξ_1 and/or ξ_2 already have large absolute value (case 2), we can find a time window such that ξ_1 and/or ξ_2 remain large in that time window; when ξ_1 and/or ξ_2 are small (case 1), using the observation that their time derivatives are large (because w_3 is large), together with the fact that these derivatives remain large (because the derivative of these derivatives are bounded), we can also find a time window that ξ_1 and/or ξ_2 are large (back in time). Therefore, the backward observability Grammian is uniformly positive definite.

5.1.4 Kazantzis-Kravaris-Luenberger (KKL) Template

In Luenberger's original paper about observer design for linear systems (Luenberger, 1964), the goal was to transform a linear system

$$\dot{x} = Fx, \quad y = Cx$$

into a Hurwitz form

$$\dot{\xi} = A\xi + By \tag{5.27}$$

with A a Hurwitz (stable) matrix. If such a transformation is available, then the following system

$$\dot{\hat{\xi}} = A\hat{\xi} + By,$$

which is nothing but a copy of the dynamics (5.27), is in fact an observer. This is because the error $e = \hat{\xi} - \xi$ evolves as

$$\dot{e} = Ae,$$

which implies that e tends to zero regardless of the initial error $e(0)$. Luenberger proved that when (F, C) is observable, a stationary transformation $\xi = Tx$ with $p = n$, i.e., $T \in \mathbb{R}^{n\times n}$, always exists and is unique, for any matrix A that is Hurwitz and (A, B) that is controllable. This is based on the fact that

$$(AT + BC)x = A\xi + By = \dot{\xi} = T\dot{x} = TFx, \forall x \tag{5.28}$$

$$\iff AT + BC = TF, \tag{5.29}$$

known as the Sylvester equation, admits a unique and invertible solution T.

A natural extension of Luenberger's original idea is to find a transformation that converts the nonlinear system (5.1) into the following form

$$\dot{\xi} = A\xi + B(u, y), \quad y = H(\xi, u), \tag{5.30}$$

with A a Hurwitz matrix (but H can be nonlinear, as opposed to the Luenberger template in Theorem 5.2). If such a transformation can be found, then we can design a similar observer that copies the dynamics (5.30)

$$\dot{\hat{\xi}} = A\hat{\xi} + B(u, y). \tag{5.31}$$

We refer to such a nonlinear Luenberger template the Kazantzis-Kravaris-Luenberger (KKL) template, due to the seminal work (Kazantzis and Kravaris, 1998).

The KKL template, once found, is nice in the sense that (i) the observer (5.31) is a simple copy of the dynamics and also very easy to implement (as opposed to the Kalman-Bucy observer); and (ii) checking if the matrix A is Hurwitz is easy, at least when A has reasonable size, (e.g., compared to checking the regular persistence condition in the state-affine template in Theorem 5.3).

However, the KKL template is difficult to realize in the sense that (i) what kind of nonlinear systems can be converted to (5.30), and (ii) for those systems, how do we find the coordinate transformation?

Recent works have leveraged deep learning to learn the coordinate transformation, for example in (Janny et al., 2021), (Niazi et al., 2023), (Miao and Gatsis, 2023). Before hammering the problem with deep learning, let us look at the fundamentals of the KKL observer.

5.1.4.1 Autonomous Systems

Consider the autonomous version of system (5.1) without control

$$\dot{x} = f(x), \quad y = h(x), \tag{5.32}$$

where $x \in \mathbb{X} \subseteq \mathbb{R}^n, y \in \mathbb{Y} \subseteq \mathbb{R}^d$.

The following result, established by (Andrieu and Praly, 2006), states that the KKL observer exists under mild conditions.

Theorem 5.4 (KKL Observer for Autonomous Systems)**.** *Assum $\mathcal{X}$ and $\mathcal{L}$ are open bounded sets in $\mathbb{X}$ (the state space) such that* $\mathrm{cl}(\mathcal{X})$ *is contained in $\mathcal{L}$ and the system* (5.32) *is backward $\mathcal{L}$-distinguishable on $\mathcal{X}$ (cf. Definition 5.2). Then there exists a strictly positive number γ and a set $\mathcal{S}$ of zero Lebesgue measure in $\mathbb{C}^{n+1}$ such that denoting $\Omega = \{\lambda \in \mathbb{C} \mid \mathrm{Re}(\lambda) < -\gamma\}$, for any $(\lambda_1, \dots, \lambda_{n+1}) \in \Omega^{n+1} \backslash \mathcal{S}$, there exists a function $T : \mathbb{R}^n \to \mathbb{R}^{(n+1)d}$ uniformly injective on $\mathcal{X}$ satisfying*

$$L_f T(x) = AT(x) + B(h(x))$$

with

$$A = \tilde{A} \otimes I_d, \quad B(y) = (\tilde{B} \otimes I_d)y \tag{5.33}$$

$$\tilde{A} = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_{n+1} \end{bmatrix} \quad \tilde{B} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}. \tag{5.34}$$

Moreover, if $\mathcal{X}$ is backward invariant, then T is unique and defined by

$$T(x) = \int_{-\infty}^{0} e^{-A\tau} B(h(X(x, \tau))) d\tau. \tag{5.35}$$

Remark. The function T in Theorem 5.4 takes complex numbers. To simulate the observer

$$\dot{\hat{\xi}} = A\hat{\xi} + B(y),$$

one needs to implement in real numbers, for each λ_i and $j \in [d]$

$$\dot{\hat{\xi}}_{\lambda_i, j} = \begin{bmatrix} -\mathrm{Re}(\lambda_i) & -\mathrm{Im}(\lambda_i) \\ \mathrm{Im}(\lambda_i) & -\mathrm{Re}(\lambda_i) \end{bmatrix} \hat{\xi}_{\lambda_i, j} + \begin{bmatrix} y_j \\ 0 \end{bmatrix}.$$

Therefore, the dimension of the observer is $2 \times d(n+1)$.

Theorem 5.4 states that as long as the system (5.32) is backward distinguishable, then there exists a stationary transformation T that can transform the system to a new coordinate system ξ such that the dynamics in ξ is Hurwitz. A closer look at the structure of A and B reveals that the coordinate transformation needs to satisfy $n+1$ differential equations of the form

$$\frac{\partial T_\lambda}{\partial x}(x)\dot{x} = \lambda T_\lambda(x) + y$$

where each T_λ transforms the state x into a new coordinate having the same dimension of y. Clearly, if $T = (T_\lambda)$, i.e., there is a single λ, then T is not uniformly injective (as the dimension of ξ is $d < n$). Consequently, by choosing

$$T = (T_{\lambda_1}, \dots, T_{\lambda_{n+1}}),$$

the uniform injectivity of T is ensured.

However, the difficulty lies in the computation of T (and T_λ), let alone its inverse (that recovers x from ξ). Even though $\mathcal{X}$ is backward invariant, the formulation (5.35) is difficult to compute. I tried very hard to find a coordinate transformation T that can convert the non-controlled pendulum dynamics into the KKL form but did not succeed. **You should let me know if you were able to find one!** Nevertheless, the following example shows you the flavor of how such a transformation may look like for a different system.

Example 5.3 (KKL Observer for an Oscillator with Unknown Frequency)**.** Consider a harmonic oscillator with unknown frequency

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -x_1 x_3 \\ \dot{x}_3 = 0 \end{cases}, \quad y = x_1$$

Consider the coordinate transformation

$$T_{\lambda_i}(x) = \frac{\lambda_i x_1 - x_2}{\lambda_i^2 + x_3}, \quad \lambda_i > 0, i = 1, \dots, p.$$

We have

$$\frac{\partial T_{\lambda_i}(x)}{\partial x}\dot{x} = \left\langle \begin{bmatrix} \frac{\lambda_i}{\lambda_i^2+x_3} \\ \frac{-1}{\lambda_i^2+x_3} \\ \frac{x_2-\lambda_i x_1}{(\lambda_i^2+x_3)^2} \end{bmatrix}, \begin{bmatrix} x_2 \\ -x_1 x_3 \\ 0 \end{bmatrix} \right\rangle = \frac{\lambda_i x_2 + x_1 x_3}{\lambda_i^2 + x_3} \tag{5.36}$$

$$-\lambda_i T_{\lambda_i}(x) + y = \frac{-\lambda_i^2 x_1 + \lambda_i x_2 + x_1\lambda_i^2 + x_1 x_3}{\lambda_i^2 + x_3} = \frac{\lambda_i x_2 + x_1 x_3}{\lambda_i^2 + x_3} \tag{5.37}$$

Therefore, with

$$\xi = T(x) = [T_{\lambda_1}(x), T_{\lambda_2}(x), \dots, T_{\lambda_p}(x)]^T,$$

we have

$$\dot{\xi} = \underbrace{\begin{bmatrix} -\lambda_1 & & \\ & \ddots & \\ & & -\lambda_p \end{bmatrix}}_{A} \xi + \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} y$$

with A clearly Hurwitz.

With some extra arguments (cf. Section 8.1.1 in (Bernard, 2019)), one can see that the transformation T is injective with $p \geq 4$ distinct λ_i's. Therefore, this is a valid KKL observer.

The final issue that one needs to think about is, since the observer is estimating $\hat{\xi}$, how to recover $\hat{x}$? In this example, there is actually no analytical formula for recovering $\hat{x}$ from $\hat{\xi}$. In this case, one approach is to solve the following optimization problem

$$\hat{x} = \arg\min_x \|\hat{\xi} - T(x)\|^2,$$

which may be quite expensive.

A more general treatment is given in Section 8.2.2 in (Bernard, 2019).

5.1.4.2 Controlled Systems

5.1.5 Triangular Template

5.1.6 Design with Convex Optimization

Consider a nonlinear system

$$\dot{x} = f(x) + \psi(u, y), \quad y = Cx \tag{5.38}$$

where $x \in \mathbb{X} \subseteq \mathbb{R}^n$, $y \in \mathbb{R}^d$, C a constant matrix, and $\psi(u, y)$ a nonlinear function. We assume that $f(x)$ is a polynomial vector map (i.e., each entry of f is a polynomial function in x). Certainly the formulation in (5.38) is not as general as (5.1), but it is general enough to include many examples in robotics.

Recall that I said the essence of observer design is to (i) simulate the dynamics when the state estimation is correct, and (ii) to correct the state estimation from observation when it is off. Therefore, we wish to design an observer for (5.38) in the following form

$$\dot{\hat{x}} = \underbrace{f(\hat{x}) + \psi(u, y)}_{\text{dynamics simulation}} + \underbrace{K(y - \hat{y}, y)(C\hat{x} - y)}_{\text{feedback correction}}, \tag{5.39}$$

where, compared to the Luenberger observer (5.11), we allow the gain matrix K to be nonlinear functions of the true observation y and the estimated observation $\hat{y}$.

With the observer (5.39), the dynamics on the estimation error $e = \hat{x} - x$ becomes

$$\dot{e} = f(x + e) - f(x) + K(Ce, Cx)Ce.$$

If we can find a Lyapunov-like function $V(e)$ so that $V(e)$ is positive definite and $\dot{V}(e)$ is negative definite, then Lyapunov stability theorem 4.3 tells us that $e = 0$ is asymptotically stable. Because we do not know the gain matrix K either, we need to jointly search for V and K (that are polynomials). Mathematically, this is

$$\begin{aligned} \text{find} \quad & V, K \\ \text{subject to} \quad & V(0) = 0, \quad V(e) > 0, \forall e \neq 0 \\ & \dot{V}(e) = \frac{\partial V}{\partial e}\left(f(x+e) - f(x) + K(Ce, Cx)Ce\right) < 0, \forall e \neq 0, \forall x \in \mathbb{X} \\ & V(e) \geq \epsilon \|e\|^2, \forall e \end{aligned} \tag{5.40}$$

where the last constraint is added to make sure $V(e)$ is radially unbounded. Furthermore, if we replace the second constraint by $\dot{V}(e) \leq -\lambda V(e)$, then we can guarantee $V(e)$ converges to zero exponentially.

Problem (5.40), however, is not a convex optimization problem, due to the term $\frac{\partial V}{\partial e}K$ being bilinear in the coefficients of V and K. Nevertheless, as shown in (Ebenbauer et al., 2005), we can use a reparameterization trick to formulate a stronger version of (5.40) as follows.

$$\begin{aligned} \text{find} \quad & V, Q(Ce), M(Ce, Cx) \\ \text{subject to} \quad & V(0) = 0, \quad V(e) > 0, \forall e \neq 0 \\ & \frac{\partial V}{\partial e} = e^T Q(Ce), \quad Q(Ce) \succ 0 \\ & e^T Q(Ce)\left(f(x+e) - f(x)\right) + e^T M(Ce, Cx)Ce < 0, \forall e \neq 0, \forall x \in \mathbb{X} \\ & V(e) \geq \epsilon \|e\|^2, \forall e \end{aligned} \tag{5.41}$$

Clearly, if we can solve problem (5.41), then

$$K = Q(Ce)^{-1} M(Ce, Cx)$$

is the right gain matrix for the formulation (5.40).

Let us bring this idea to action in our pendulum example.

Example 5.4 (Pendulum Observer with Convex Optimization)**.** With $x = [\mathfrak{s}, \mathfrak{c}, \dot{\theta}]^T$ ($\mathfrak{s} = \sin\theta, \mathfrak{c} = \cos\theta$), we can write the pendulum dynamics as

$$\dot{x} = \underbrace{\begin{bmatrix} \mathfrak{c}\dot{\theta} \\ -\mathfrak{s}\dot{\theta} \\ -\frac{b}{ml^2}\dot{\theta} \end{bmatrix}}_{=:f(x)} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \frac{u - mgl\mathfrak{s}}{ml^2} \end{bmatrix}}_{=:\psi(u,y)}, \quad y = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{=:C} x$$

Clearly $f(x)$ is a polynomial. Solving the convex optimization problem (5.41), we obtain a solution

$$V(e) = 0.5954e_1^2 + 0.5954e_2^2 + 0.9431e_3^2$$

$$Q(Ce) = \begin{bmatrix} 0.4603e_2^2 + 1.1909 & -0.4603e_1e_2 & 0 \\ -0.4603e_1e_2 & 0.4603e_1^2 + 1.1909 & 0 \\ 0 & 0 & 1.8863 \end{bmatrix}$$

$$M(Ce, Cx) = \begin{bmatrix} -2.0878e_1^2 - 0.8667e_2^2 - \\ 0.4588(y_1^2+y_2^2) - 0.4885 & -0.8667e_1e_2 \\ -0.8667e_1e_2 & -0.8667e_1^2 - 2.0878e_2^2 - \\ & 0.4588(y_1^2+y_2^2) - 0.4885 \\ -1.1909y_2 & 1.1909y_1 \end{bmatrix}$$

Simulating this observer, we verify that the observer is in fact exponentially converging, as shown in Fig. 5.1.

The Matlab code for formulating and solving the convex optimization (5.41) can be found here. The code for simulating the observer can be found here.

5.2 Observer Feedback

Now that we have good ways to design a state observer, we will see how we can use the observer for feedback control.

Example 5.5 (Pendulum Stabilization with A Luenberger Observer)**.** In Example 5.1, we have written the dynamics of a pendulum, and the dynamics of a Luenberger observer as

$$\dot{x} = Ax + B(u, y) \tag{5.42}$$

$$\dot{\hat{x}} = A\hat{x} + B(u, y) + KC(x - \hat{x}) \tag{5.43}$$

We wish to understand (so we can optimize) the behavior of this system under certain control input u. To do so, let us denote $e = \hat{x} - x$, and write the above dynamics as

$$\dot{x} = Ax + B(u, Cx) \tag{5.44}$$

$$\dot{e} = (A - KC)e \tag{5.45}$$

Denoting $z = [x, e]^T$, we have the augmented dynamics

$$\dot{z} = \underbrace{\begin{bmatrix} A & 0 \\ 0 & A - KC \end{bmatrix}}_{=:F} z + \underbrace{\begin{bmatrix} B(u, Dz) \\ 0 \end{bmatrix}}_{=:G(z,u)}$$

We want to stabilize the system at $z_0 = [\pi, 0, 0, 0]^T$ (the upright position) subject to control bounds $u \in \mathbb{U} = [-u_{\max}, u_{\max}]$.

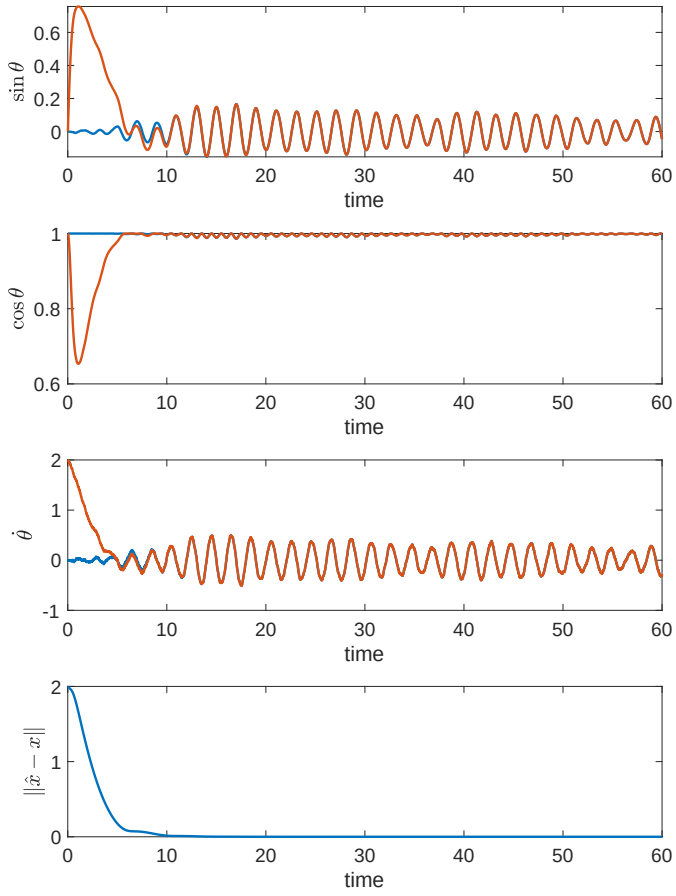

Figure 5.1: Simulation of the pendulum observer design from convex optimization

We need to find a control Lyapunov function (CLF), $V(z)$, that satisfies the following constraints:

$$V(z_0) = 0$$
$$V(z) > 0 \quad \forall z \in \{z : V(z) < \rho, z \neq z_0\}$$
$$\inf_{u \in \mathbb{U}} [L_F V(z) + L_G V(z)] \leq 0 \quad \forall z \in \mathcal{Z}$$

where $L_F V$ and $L_G V$ are the Lie derivatives of V along F and $G(z, u)$, respectively. $\mathcal{Z}$ is the set of all possible augmented states. The CLF will define the set of admissible control inputs U.

$$U = \{u : L_f V(z) + L_g V(z) u \leq 0\}$$

To find the smallest-magnitude control input such that $u \in K$, we may use a quadratic program:

$$\min_{u \in \mathcal{U}} ||u||^2$$
$$\text{s.t.} \quad L_f V(z) + L_g V(z) u \leq -cV(z)$$

where c is some positive constant. The challenge now is in choosing a suitable $V(z)$.

Chapter 6

Adaptive Control

6.1 Model-Reference Adaptive Control

Basic flow for designing an adaptive controller

1. Design a control law with variable parameters
2. Design an adaptation law for adjusting the control parameters
3. Analyze the convergence of the closed-loop system

The control law design at the first step typically requires the designer to know what a good controller is if the true parameters were actually known, e.g., from feedback linearization (Appendix D), sliding control (Appendix E) etc.

The design of the adaptation law typically comes from analyzing the dynamics of the tracking error, which as we will see often appears in the form of Lemma 6.1.

The convergence of the closed-loop system is usually analyzed with the help of a Lyapunov-like function introduced in Chapter 4.

Lemma 6.1 (Basic Lemma)**.** *Let two signals $e(t)$ and $\phi(t)$ be related by*

$$e(t) = H(p)[k\phi(t)^T v(t)] \tag{6.1}$$

where $e(t)$ a scalar output signal, $H(p)$ a strictly positive real (SPR) transfer function, k an unknown real number with known sign, $\phi(t) \in \mathbb{R}^m$ a control signal, and $v(t) \in \mathbb{R}^m$ a measurable input signal.

If the control signal $\phi(t)$ satisfies

$$\dot{\phi}(t) = -\text{sgn}(k)\gamma e(t)v(t) \tag{6.2}$$

with $\gamma > 0$ a positive constant, then $e(t)$ and $\phi(t)$ are globally bounded. Moreover, if $v(t)$ is bounded, then

$$\lim_{t\to\infty} e(t) = 0.$$

Proof. Let the state-space representation of (6.1) be

$$\dot{x} = Ax + b[k\phi^T v], \quad e = c^T x. \tag{6.3}$$

Since $H(p)$ is SPR, it follows from the Kalman-Yakubovich Lemma C.1 that there exist $P, Q \succ 0$ such that

$$A^T P + PA = -Q, \quad Pb = c.$$

Let

$$V(x, \phi) = x^T P x + \frac{|k|}{\gamma}\phi^T\phi,$$

clearly V is positive definite (i.e., $V(0,0) = 0$, and $V(x,\phi) > 0$ for all $x \neq 0, \phi \neq 0$). The time derivative of V along the trajectory defined by (6.3) with ϕ chosen as in (6.2) is

$$\begin{aligned}
\dot{V} &= \frac{\partial V}{\partial x}\dot{x} + \frac{\partial V}{\partial \phi}\dot{\phi} && (6.4)\\
&= x^T(PA + A^T P)x + 2x^T Pb(k\phi^T v) + \frac{2|k|}{\gamma}\phi^T(-\mathrm{sgn}(k)\gamma e v) && (6.5)\\
&= -x^T Q x + 2(x^T c)(k\phi^T v) - 2\phi^T(ev) && (6.6)\\
&= -x^T Q x \leq 0. && (6.7)
\end{aligned}$$

As a result, we know x and ϕ must be bounded ($V(x(t), \phi(t)) \leq V(x(0), \phi(0))$ is bounded). Since $e = c^T x$, we know e must be bounded as well.

If the input signal v is also bounded, then $\dot{x}$ is bounded as seen from (6.3). Because $\ddot{V} = -2x^T Q\dot{x}$ is now bounded, we know $\dot{V}$ is uniformly continuous. Therefore, by Barbalat's stability certificate (Theorem 4.6), we know $\dot{V}$ tends to zero as t tends to infinity, which implies $\lim_{t\to\infty} x(t) = 0$ and hence $\lim_{t\to\infty} e(t) = 0$. □

6.1.1 First-Order Systems

Consider the first-order single-input single-output (SISO) system

$$\dot{x} = -ax + bu \tag{6.8}$$

where a and b are unknown groundtruth parameters. However, we do assume that the sign of b is known. What if the sign of b is unknown too?

Let $r(t)$ be a reference trajectory, e.g., a step function or a sinusoidal function, and $x_d(t)$ be a desired system trajectory that tracks the reference

$$\dot{x}_d = -a_d x_d + b_d r(t), \tag{6.9}$$

where $a_d, b_d > 0$ are user-defined constants. Note that the transfer function from r to x_d is

$$x_d = \frac{b_d}{p + a_d} r$$

and the system is stable. Review basics of transfer function.

The goal of adaptive control is to design a control law and an adaptation law such that the tracking error of the system $x(t) - x_d(t)$ converges to zero.

Control law. We design the control law as

$$u = \hat{a}_r(t) r + \hat{a}_x(t) x \tag{6.10}$$

where $\hat{a}_r(t)$ and $\hat{a}_x(t)$ are time-varying feedback gains that we wish to adapt. The closed-loop dynamics of system (6.8) with the controller (6.10) is

$$\dot{x} = -ax + b(\hat{a}_r r + \hat{a}_x x) = -(a - b\hat{a}_x)x + b\hat{a}_r r.$$

With the equation above, the reason for choosing the control law (6.10) is clear: if the system parameters (a, b) were known, then choosing

$$a_r^\star = \frac{b_d}{b}, \quad a_x^\star = \frac{a - a_d}{b} \tag{6.11}$$

leads to the closed-loop dynamics $\dot{x} = -a_d x + b_d r$ that is exactly what we want in (6.9).

However, in adaptive control, since the true parameters (a, b) are not revealed to the control designer, an adaptation law is needed to dynamically adjust the gains $\hat{a}_r$ and $\hat{a}_x$ based on the tracking error $x(t) - x_d(t)$.

Adaptation law. Let $e(t) = x(t) - x_d(t)$ be the tracking error, and we develop its time derivative

$$\begin{aligned}
\dot{e} &= \dot{x} - \dot{x}_d && (6.12)\\
&= -a_d(x - x_d) + (a_d - a + b\hat{a}_x)x + (b\hat{a}_r - b_d)r && (6.13)\\
&= -a_d e + b\underbrace{(\hat{a}_x - \hat{a}_x^\star)}_{=:\tilde{a}_x} x + b\underbrace{(\hat{a}_r - \hat{a}_r^\star)}_{=:\tilde{a}_r} r && (6.14)\\
&= -a_d e + b(\tilde{a}_x x + \tilde{a}_r r) && (6.15)
\end{aligned}$$

where $\tilde{a}_x$ and $\tilde{a}_r$ are the gain errors w.r.t. the optimal gains in (6.11) if the true parameters were known. The error dynamics (6.15) is equivalent to the following transfer function

$$e = \frac{1}{p + a_d} b(\tilde{a}_x x + \tilde{a}_r r) = \frac{1}{p + a_d} \left(b \begin{bmatrix} \tilde{a}_x \\ \tilde{a}_r \end{bmatrix}^T \begin{bmatrix} x \\ r \end{bmatrix} \right), \tag{6.16}$$

which is in the form of (6.1). Therefore, we choose the adaptation law

$$\begin{bmatrix} \dot{\tilde{a}}_x \\ \dot{\tilde{a}}_r \end{bmatrix} = -\text{sgn}(b)\gamma e \begin{bmatrix} x \\ r \end{bmatrix}. \tag{6.17}$$

Tracking convergence. With the control law (6.10) and the adaptation law (6.17), we can prove that the tracking error converges to zero, using Lemma 6.1. With $\tilde{a} = [\tilde{a}_x, \tilde{a}_r]^T$, let

$$V(e, \tilde{a}) = e^2 + \frac{|b|}{\gamma}\tilde{a}^T\tilde{a} \tag{6.18}$$

be a positive definite Lyapunov function candidate with time derivative

$$\dot{V} = -2a_d e^2 \leq 0.$$

Clearly, e and $\tilde{a}$ are both bounded. Assuming the reference trajectory r is bounded, we know x_d is bounded (due to (6.9)) and hence x is bounded (due to $e = x - x_d$ being bounded). Consequently, from the error dynamics (6.15) we know $\dot{e}$ is bounded, which implies $\ddot{V} = -4a_d e\dot{e}$ is bounded and $\dot{V}$ is uniformly continuous. By Barbalat's stability certificate 4.6, we conlude $e(t) \to 0$ as $t \to \infty$.

It is always better to combine mathematical analysis with intuitive understanding. Can you explain intuitively why the adaptation law (6.17) makes sense? (Hint: think about how the control should react to a negative/positive tracking error.)

Parameter convergence. We have shown the control law (6.10) and the adaptation law (6.17) guarantee to track the reference trajectory. However, is it guaranteed that the gains of the controller (6.10) also converge to the optimal gains in (6.11)?

We will now show that the answer is indefinite and it depends on the reference trajectory $r(t)$. Because the tracking error e converges to zero, and e is the output of a stable filter (6.16), we know the input $b(\tilde{a}_x x + \tilde{a}_r r)$ must also converge to zero. On the other hand, the adaptation law (6.17) shows that both $\dot{\tilde{a}}_x$ and $\dot{\tilde{a}}_r$ converge to zero (due to e converging to zero and x, r being bounded). As a result, we know $\tilde{a} = [\tilde{a}_x, \tilde{a}_r]^T$ converges to a constant that satisfies

$$v^T\tilde{a} = 0, \quad v = \begin{bmatrix} x \\ r \end{bmatrix}, \tag{6.19}$$

which is a single linear equation of $\tilde{a}$ with time-varying coeffients.

- **Constant reference: no guaranteed convergence**. Suppose $r(t) \equiv r_0 \neq 0$ for all t. From (6.9) we know $x = x_d = \alpha r_0$ when $t \to \infty$, where α is the constant DC gain of the stable filter. Therefore, the linear equation (6.19) reduces to

  $$\alpha\tilde{a}_x + \tilde{a}_r = 0.$$

  This implies that $\tilde{a}$ does not necessarily converge to zero. In fact, it converges to a straight line in the parameter space.

- **Persistent excitation: guaranteed convergence**. However, when the signal v satisfies the so-called *persistent excitation* condition, which states

that for any t, there exists $T, \beta > 0$ such that

$$\int_t^{t+T} vv^T d\tau \geq \beta I, \tag{6.20}$$

then $\tilde{a}$ is guaranteed to converge to zero. To see this, we multiply (6.19) by v and integrate it from t to $t+T$, which gives rise to

$$\left(\int_t^{t+T} vv^T d\tau \right) \tilde{a} = 0.$$

By the persistent excitation condition (6.20), we infer that $\tilde{a} = 0$ is the only solution.

It remains to understand under what conditions of the reference trajectory $r(t)$ can we guarantee the persistent excitation of v. We leave it as an exercise for the reader to show, if $r(t)$ contains at least one sinusoidal component, then the persistent excitation condition of v is guaranteed.

Exercise 6.1 (Extension to Nonlinear Systems)**.** Design a control law and an adaptation law for the following system

$$\dot{x} = -ax - cf(x) + bu$$

with unknown true parameters (a, b, c) (assume the sign of b is known) and known nonlinearity $f(x)$ to track a reference trajectory $r(t)$. Analyze the convergence of tracking error and parameter estimation error.

6.1.2 High-Order Systems

Consider an n-th order nonlinear system

$$q^{(n)} + \sum_{i=1}^{n} \alpha_i f_i(x, t) = bu \tag{6.21}$$

where $x = [q, \dot{q}, \ddot{q}, \ldots, q^{(n-1)}]^T$ is the state of the system, f_i's are known nonlinearities, $(\alpha_1, \ldots, \alpha_n, b)$ are unknown parameters of the system (with $\text{sgn}(b)$ known).

The goal of adaptive control is to control the system (6.21) trajectory to follow a desired trajectory $q_d(t)$ despite no knowing the true parameters.

To facilitate the derivation of the adaptive controller, let us divide both sides of (6.21) by b

$$hq^{(n)} + \sum_{i=1}^{n} a_i f_i(x, t) = u \tag{6.22}$$

where $h = 1/b$ and $a_i = \alpha_i / b$.

Control law. Recall that the choice of the control law is tyically inspired by the control design if the true system parameters were known. We will borrow ideas from sliding control (Appendix E).

- **Known parameters**. Let $e = q(t) - q_d(t)$ be the tracking error, and define the following combined error

$$s = e^{(n-1)} + \lambda_{n-2}e^{(n-2)} + \cdots + \lambda_0 e = \Delta(p)e$$

where $\Delta(p) = p^{n-1} + \lambda_{n-2}p^{(n-2)} + \cdots + \lambda_0$ is a stable polynomial with user-chosen coeffients $\lambda_0, \ldots, \lambda_{n-2}$. The rationale for defining the combined error s is that the convergence of e to zero can be guaranteed by the convergence of s to zero (when $\Delta(p)$ is stable). Note that s can be equivalently written as

$$s = (q^{(n-1)} - q_d^{(n-1)}) + \lambda_{n-2}e^{(n-2)} + \cdots + \lambda_0 e \tag{6.23}$$

$$= q^{(n-1)} - \underbrace{\left(q_d^{(n-1)} - \lambda_{n-2}e^{(n-2)} - \cdots - \lambda_0 e\right)}_{q_r^{(n-1)}}. \tag{6.24}$$

Now consider the control law

$$u = hq_r^{(n)} - ks + \sum_{i=1}^{n} a_i f_i(x,t) \tag{6.25}$$

where

$$q_r^{(n)} = q_d^{(n)} - \lambda_{n-2}e^{(n-1)} - \cdots - \lambda_0 \dot{e}$$

and k is a design constant that has the same sign as h. This choice of control, plugged into the system dynamics (6.22), leads to

$$hq^{(n)} + \sum_{i=1}^{n} a_i f_i(x,t) = hq_r^{(n)} - ks + \sum_{i=1}^{n} a_i f_i(x,t) \iff \tag{6.26}$$

$$h\left(q^{(n)} - q_r^{(n)}\right) + ks = 0 \iff \tag{6.27}$$

$$h\dot{s} + ks = 0, \tag{6.28}$$

which guarantees the exponential convergence of s to zero (note that h and k have the same sign), and hence the convergence of e to zero.

- **Unknown parameters**. Inspired by the control law with known parameters in (6.25), we design the adapative control law as

$$u = \hat{h}q_r^{(n)} - ks + \sum_{i=1}^{n} \hat{a}_i f_i(x,t), \tag{6.29}$$

where the time-varying gains $\hat{h}, \hat{a}_1, \ldots, \hat{a}_n$ will be adjusted by an adaptation law.

Adaptation law. Inserting the adapative control law (6.29) into the system dynamics (6.22), we obtain

$$h\dot{s} + ks = \tilde{h}q_r^{(n)} + \sum_{i=1}^{n} \tilde{a}_i f_i(x,t) \Longleftrightarrow \tag{6.30}$$

$$s = \frac{1}{p+k/h}\frac{1}{h}\underbrace{\left(\begin{bmatrix}\tilde{h}\\ \tilde{a}_1\\ \vdots\\ \tilde{a}_n\end{bmatrix}^T \begin{bmatrix}q_r^{(n)}\\ f_1(x,t)\\ \vdots\\ f_n(x,t)\end{bmatrix}\right)}_{=:\phi^T v} \tag{6.31}$$

where $\tilde{h} = \hat{h} - h$ and $\tilde{a}_i = \hat{a}_i - a_i, i = 1, \ldots, n$. Again, (6.31) is in the familiar form of (6.1), which naturally leads to the following adaptation law with $\gamma > 0$ a chosen constant

$$\dot{\phi} = \begin{bmatrix}\dot{\tilde{h}}\\ \dot{\tilde{a}}_1\\ \vdots\\ \dot{\tilde{a}}_n\end{bmatrix} = -\text{sgn}(h)\gamma s \begin{bmatrix}q_r^{(n)}\\ f_1(x,t)\\ \vdots\\ f_n(x,t)\end{bmatrix}. \tag{6.32}$$

Tracking and parameter convergence. With the following Lyapunov function

$$V(s,\phi) = |h|s^2 + \frac{1}{\gamma}\phi^T\phi, \quad \dot{V}(s,\phi) - 2|k|s^2, \tag{6.33}$$

the global convergence of s to zero can be easily shown. For parameter convergence, it is easy to see that when v satisfies the persistent excitation condition, we have that ϕ converges to zero. (However, the relationship between the reference trajectory $q_d(t)$ and the persistent excitation of v becomes nontrivial due to the nonlinearities f_i.)

6.1.3 Robotic Manipulator

So far our focus has been on systems with a single input ($u \in \mathbb{R}$). In the following, we will show that similar techniques can be applied to adapative control of systems with multiple inputs, particularly, trajectory control of a robotic manipulator.

Let $q \in \mathbb{R}^n$ be the joint angles of a multi-link robotic arm, and $\dot{q} \in \mathbb{R}^n$ be the joint velocities. The dynamics of a robotic manipulator reads

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau, \tag{6.34}$$

where $H(q) \in \mathbb{S}^n_{++}$ is the manipulator inertia matrix (that is positive definite), $C(q,\dot{q})\dot{q}$ is a vector of centripetal and Coriolis torques (with $C(q,\dot{q}) \in \mathbb{R}^{n\times n}$), and $g(q)$ denotes gravitational torques.

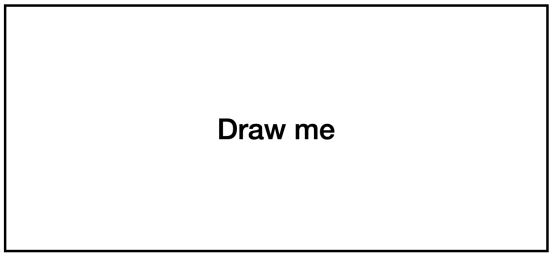

Figure 6.1: Planar two-link manipulator

Example 6.1 (Planar Two-link Manipulator)**.** The dynamics of a planar two-link manipulator in Fig. 6.1 is

$$\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} -h\dot{q}_2 & -h(\dot{q}_1 + \dot{q}_2) \\ h\dot{q}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}, \tag{6.35}$$

where

$$H_{11} = a_1 + 2a_3 \cos q_2 + 2a_4 \sin q_2 \tag{6.36}$$
$$H_{12} = H_{21} = a_2 + a_3 \cos q_2 + a_4 \sin q_2 \tag{6.37}$$
$$H_{22} = a_2 \tag{6.38}$$
$$h = a_3 \sin q_2 - a_4 \cos q_2 \tag{6.39}$$

with

$$a_1 = I_1 + m_1 l_{c1}^2 + I_e + m_e l_{ce}^2 + m_e l_1^2 \tag{6.40}$$
$$a_2 = I_e + m_e l_{ce}^2 \tag{6.41}$$
$$a_3 = m_e l_1 l_{ce} \cos \delta_e \tag{6.42}$$
$$a_4 = m_e l_1 l_{ce} \sin \delta_e. \tag{6.43}$$

As seen from the above example, the parameters a (which are nonlinear functions of the physical parameters such as mass and length) enter linearly in H and C ($g(q)$ is ignored because the manipulator is on a horizontal plane).

The goal of the control design is to have the manipulator track a desired trajectory $q_d(t)$.

Known parameters. When the parameters are known, we follow the sliding control design framework. Let $\tilde{q} = q(t) - q_d(t)$ be the tracking error, and define the combined error

$$s = \dot{\tilde{q}} + \Lambda \tilde{q} = \dot{q} - \underbrace{(\dot{q}_d - \Lambda \tilde{q})}_{\dot{q}_r}$$

where $\Lambda \in \mathbb{S}^n_{++}$ is a user-chosen positive definite matrix (in general we want $-\Lambda$ to be Hurwitz). In this case, $s \to 0$ implies $\tilde{q} \to 0$ as $t \to \infty$. Choosing the

control law (coming from feedback linearization Appendix D)

$$\tau = H\ddot{q}_r - K_D s + C\dot{q} + g(q) \tag{6.44}$$

with $K_D \in \mathbb{S}^n_{++}$ positive definite leads to the closed-loop dynamics

$$H\dot{s} + K_D s = 0 \iff \dot{s} = -H^{-1}K_D s.$$

Because the matrix $H^{-1}K_D$ is the product of two positive definite matrices (recall H is positive definite and so is H^{-1}), it has strictly positive real eigenvalues.[1] Hence, $-H^{-1}K_D$ is Hurwitz and s is guaranteed to converge to zero.

Control law. A closer look at the controller (6.44) allows us to write it in the following form

$$\begin{aligned} \tau &= H\ddot{q}_r + C(s + \dot{q}_r) + g(q) - K_D s && (6.45)\\ &= H\ddot{q}_r + C\dot{q}_r + g(q) + (C - K_D)s && (6.46)\\ &= Y(q, \dot{q}, \dot{q}_r, \ddot{q}_r)a + (C - K_D)s && (6.47) \end{aligned}$$

where $a \in \mathbb{R}^m$ contains all the parameters and $Y \in \mathbb{R}^{n\times m}$ is the matrix that collects all the coeffients of a in $H\ddot{q}_r + C\dot{q}_r + g(q)$. As a result, we design the adapative control law to be

$$\tau = Y\hat{a} - K_D s, \tag{6.48}$$

with $\hat{a}$ the time-varying parameter that we wish to adapt. Note that here we have done something strange: the adapative control law does not exactly follow the controller (6.44) in the known-parameter case.[2] We first separated s from $\dot{q}$ and wrote $Ya = H\ddot{q}_r + C\dot{q}_r + g$ instead of $Ya = H\ddot{q}_r + C\dot{q} + g$; then we dropped the "$C$" matrix in front of s in the adapative control law. The reason for doing this will soon become clear when we analyze the tracking convergence.

Adaptation law and tracking convergence. Recall that the key of adapative control is to design a control law and an adaptation law such that global converge of the tracking error s can be guaranteed by a Lyapunov function. Looking at the previous Lyapunov functions in (6.18) and (6.33), we see that they both contain a positive definite term in the tracking error s (or e if in first-order systems) and another positive definite term in the parameter error $\tilde{a}$. This hints us that we may try a Lyapunov candidate function of the following form

$$V = \frac{1}{2}\left(s^T H s + \tilde{a}\Gamma^{-1}\tilde{a}\right), \tag{6.49}$$

[1] Consider two positive definite matrices A and B, let $B = B^{1/2}B^{1/2}$. The product AB can be written as $AB = AB^{1/2}B^{1/2} = B^{-1/2}(B^{1/2}AB^{1/2})B^{1/2}$. Therefore AB is similar to $B^{1/2}AB^{1/2}$ and is positive definite.

[2] In fact, one can show that the controller (6.48) with known parameters, i.e., $\tau = Ya - K_D s$, also guarantees the convergence of s towards zero, though it is different from the feedback linearization controller (6.44). Try proving the convergence with a Lyapunov candidate $V = \frac{1}{2}s^T H s$.

where $\Gamma \in \mathbb{S}^m_{++}$ is a constant positive definite matrix, and $\tilde{a} = \hat{a} - a$ is the parameter error.

The next step would be to derive the time derivative of V, which, as we can expect, will contain a term that involves $\dot{H}$ and complicates our analysis. Fortunately, the following lemma will help us.

Lemma 6.2. *For the manipulator dynamics* (6.34)*, there exists a way to define* C *such that* $\dot{H} - 2C$ *is skew-symmetric.*

Proof. See Section 9.1, page 399-402 in (Slotine et al., 1991). You should also check if this is true for the planar two-link manipulator dynamics in Example 6.1. □

With Lemma 6.2, the time derivative of V in (6.49) reads

$$\begin{aligned}
\dot{V} &= s^T H \dot{s} + \frac{1}{2} s^T \dot{H} s + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}} && (6.50)\\
&= s^T (H\ddot{q} - H\ddot{q}_r) + \frac{1}{2} s^T \dot{H} s + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}} && (6.51)\\
&= s^T (\tau - C\dot{q} - g - H\ddot{q}_r) + \frac{1}{2} s^T \dot{H} s + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}} && (6.52)\\
&= s^T (\tau - H\ddot{q}_r - C(s + \dot{q}_r) - g) + \frac{1}{2} s^T \dot{H} s + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}} && (6.53)\\
&= s^T (\tau - H\ddot{q}_r - C\dot{q}_r - g) + \frac{1}{2} s^T (\dot{H} - 2C) s + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}} && (6.54)\\
&= s^T (\tau - H\ddot{q}_r - C\dot{q}_r - g) + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}} && (6.55)\\
&= s^T (Y\hat{a} - K_D s - Ya) + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}} && (6.56)\\
&= s^T Y \tilde{a} + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}} - s^T K_D s, && (6.57)
\end{aligned}$$

where we used the manipulator dynamics (6.34) to rewrite $H\ddot{q}$ in (6.52), used $\dot{H} - 2C$ is skew-symmetric in (6.54), invoked the adapative control law (6.48) and reused $Ya = H\ddot{q}_r + C\dot{q}_r + g(q)$ in (6.56). The derivation above explains why the choice of the control law in (6.48) did not exactly follow its counterpart when the parameters are known: we need to use $s^T C s$ to cancel $\frac{1}{2} s^T \dot{H} s$ in (6.54).

We then wonder if we can design $\dot{\tilde{a}}$ such that $\dot{V}$ in (6.57) is negative semidefinie? This turns out to be straightforward with the adaptation law

$$\dot{\tilde{a}} = -\Gamma Y^T s, \tag{6.58}$$

to make $s^T Y \tilde{a} + \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}}$ vanish and so

$$\dot{V} = -s^T K_D s \leq 0.$$

We are not done yet. To show s converges to zero (which is implied by $\dot{V}$ converges to zero), by Barbalat's stability certificate 4.6, it suffices to show

$$\ddot{V} = -2s^T K_D \dot{s}$$

is bounded. We already know s and $\tilde{a}$ are bounded, due to the fact that V in (6.49) is bounded. Therefore, we only need to show $\dot{s}$ is bounded. To do so, we plug the adapative control law (6.48) into the manipulator dynamics (6.34) and obtain

$$H\dot{s} + (C + K_D)s = Y\tilde{a},$$

which implies the boundedness of $\dot{s}$ (note that H is uniformly positive definite, i.e., $H \succeq \alpha I$ for some $\alpha > 0$). This concludes the analysis of the tracking convergence $s \to 0$ as $t \to \infty$.

6.2 Certainty-Equivalent Adaptive Control

Appendix A

Linear System Theory

A.1 Stability

Check this note.

Theorem A.1 (Lyapunov Equation)**.** *The following is equivalent for a linear time-invariant system $\dot{x} = Ax$*

1. *The system is globally asymptotically stable, i.e., A is Hurwitz and $\lim_{t\to\infty} x(t) = 0$ regardless of the initial condition;*
2. *For any positive definite matrix Q, the unique solution P to the Lyapunov equation*
$$A^T P + PA = -Q \tag{A.1}$$
is positive definite.

Proof. (a): $2 \Rightarrow 1$. Suppose we are given two positive definie matrices $P, Q \succ 0$ that satisfies the Lyapunov equation (A.1). Define a scalar function

$$V(x) = x^T P x.$$

It is clear that $V > 0$ for any $x \neq 0$ and $V(x) = 0$ (i.e., $V(x)$ is positive definie). We also see $V(x)$ is radially unbounded because:

$$V(x) \geq \lambda_{\min}(P)\|x\|^2 \Rightarrow \lim_{x\to\infty} V(x) \to \infty.$$

The time derivative of V reads

$$\dot{V} = 2x^T P\dot{x} = x^T(A^T P + PA)x = -x^T Q x.$$

Clearly, $\dot{V} < 0$ for any $x \neq 0$ and $\dot{V}(0) = 0$. According to Lyapunov's global stability theorem 4.3, we conclude the linear system $\dot{x} = Ax$ is globally asymptotically stable at $x = 0$.

(b): $1 \Rightarrow 2$. Suppose A is Hurwitz, we want to show that, for any $Q \succ 0$, there exists a unique $P \succ 0$ satisfying the Lyapunov equation (A.1). In fact, consider the matrix

$$P = \int_{t=0}^{\infty} e^{A^T t} Q e^{At} dt.$$

Because A is Hurwitz, the integral exists, and clearly $P \succ 0$ due to $Q \succ 0$. To show this choice of P satisfies the Lyapunov equation, we write

$$A^T P + PA = \int_{t=0}^{\infty} \left(A^T e^{A^T t} Q e^{At} + e^{A^T t} Q e^{At} A \right) dt \tag{A.2}$$

$$= \int_{t=0}^{\infty} d\left(e^{A^T t} Q e^{At} \right) \tag{A.3}$$

$$= e^{A^T t} Q e^{At}|_{t=\infty} - e^{A^T t} Q e^{At}|_{t=0} = -Q, \tag{A.4}$$

where the last equality holds because $e^{A\infty} = 0$ (recall A is Hurwitz).

To show the uniqueness of P, we assume that there exists another matrix P' that also satisfies the Lyapunov equation. Therefore,

$$P' = e^{A^T t} P' e^{At}|_{t=0} - e^{A^T t} P' e^{At}|_{t=\infty} \tag{A.5}$$

$$= -\int_{t=0}^{\infty} d\left(e^{A^T t} P' e^{At} \right) \tag{A.6}$$

$$= -\int_{t=0}^{\infty} e^{A^T t} \left(A^T P' + P' A \right) e^{At} dt \tag{A.7}$$

$$= \int_{t=0}^{\infty} e^{A^T t} Q e^{At} dt = P, \tag{A.8}$$

leading to $P' = P$. Hence, the solution is unique. □

Convergence rate estimation. We now show that Theorem A.1 can allow us to quantify the convergence rate of a (stable) linear system towards zero.

For a Hurwitz linear system $\dot{x} = Ax$, let us pick a positive definite matrix Q. Theorem A.1 tells us we can find a unique $P \succ 0$ satisfying the Lyapunov equation (A.1). In this case, we can upper bound the scalar function $V = x^T P x$ as

$$V \leq \lambda_{\max}(P) \|x\|^2.$$

The time derivative of V is $\dot{V} = -x^T Q x$, which can be upper bounded by

$$\dot{V} \leq -\lambda_{\min}(Q) \|x\|^2 \tag{A.9}$$

$$= -\frac{\lambda_{\min}(Q)}{\lambda_{\max}(P)} \underbrace{\left(\lambda_{\max}(P) \|x\|^2 \right)}_{\geq V} \tag{A.10}$$

$$\leq -\frac{\lambda_{\min}(Q)}{\lambda_{\max}(P)} V. \tag{A.11}$$

Denoting $\gamma(Q) = \frac{\lambda_{\min}(Q)}{\lambda_{\max}(P)}$, the above inequality implies

$$V(0)e^{-\gamma(Q)t} \geq V(t) = x^T P x \geq \lambda_{\min}(P)\|x\|^2.$$

As a result, $\|x\|^2$ converges to zero exponentially with a rate at least $\gamma(Q)$, and $\|x\|$ converges to zero exponentially with a rate at least $\gamma(Q)/2$.

Best convergence rate estimation. I have used $\gamma(Q)$ to make it explict that the rate γ depends on the choice of Q, because P is computed from the Lyapunov equation as an implicit function of Q. Naturally, choosing different Q will lead to different $\gamma(Q)$. So what is the choice of Q that maximizes the convergence rate estimation?

Corollary A.1 (Maximum Convergence Rate Estimation)**.** *$Q = I$ maximizes the convergence rate estimation.*

Proof. let us denote P_0 as the solution to the Lyapunov equation with $Q = I$

$$A^T P_0 + P_0 A = -I.$$

Let P be the solution corresponding to a different choice of Q

$$A^T P + P A = -Q.$$

Without loss of generality, we can assume $\lambda_{\min}(Q) = 1$, because rescaling Q will recale P by the same factor, which does not affect $\gamma(Q)$. Subtracting the two Lyapunov equations above we get

$$A^T(P - P_0) + (P - P_0)A = -(Q - I).$$

Since $Q - I \succeq 0$ (due to $\lambda_{\min}(Q) = 1$), we know $P - P_0 \succeq 0$ and $\lambda_{\max}(P) \geq \lambda_{\max}(P_0)$. As a result,

$$\gamma(Q) = \frac{\lambda_{\min}(Q)}{\lambda_{\max}(P)} = \frac{\lambda_{\min}(I)}{\lambda_{\max}(P)} \leq \frac{\lambda_{\min}(I)}{\lambda_{\max}(P_0)} = \gamma(I),$$

and $Q = I$ maximizes the convergence rate estimation. □

Consider the linear time-invariant (LTI) system

$$\dot{x} = Ax + Bu, \quad y = Cx + Du, \tag{A.12}$$

where $x \in \mathbb{R}^n$ the state, $u \in \mathbb{R}^m$ the control, and A, B, C, D are constant matrices with proper sizes.

Theorem A.2 (Observability)**.** *The following are equivalent for the LTI system* (A.12)*:*

1. *(A, C) is observable;*
2. *The observability grammian*

$$W_o(t) = \int_0^t e^{A^*\tau} C^* C e^{A\tau} d\tau$$

is positive definite for any $t > 0$;

3. *The observability matrix*

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

has full column rank;

4. *The matrix $\begin{bmatrix} A - \lambda I \\ C \end{bmatrix}$ has full column rank for all $\lambda \in \mathbb{C}$;*
5. *For all λ and x such that $Ax = \lambda x$, $Cx \neq 0$;*
6. *The eigenvalues of $A + LC$ can be freely assigned (with the restriction that complex eigenvalues are in conjugate pairs);*
7. *(A^*, C^*) is controllable.*

Theorem A.3 (Detectability)**.** *The following are equivalent for the LTI system* (A.12)*:*

1. *(A, C) is detectable;*
2. *The matrix $\begin{bmatrix} A - \lambda I \\ C \end{bmatrix}$ has full column rank for all $\lambda \in \mathbb{C}$ such that $\mathrm{Re}(\lambda) \geq 0$;*
3. *For all λ and x such that $Ax = \lambda x$ and $\mathrm{Re}(\lambda) \geq 0$, $Cx \neq 0$;*
4. *There exists a matrix L such that $A + LC$ is Hurwitz;*
5. *(A^*, C^*) is stabilizable.*

Appendix B

Convex Analysis and Optimization

Appendix C

The Kalman-Yakubovich Lemma

Lemma C.1 (Kalman-Yakubovich)**.** *Consider a controllable linear time-invariant system*

$$\dot{x} = Ax + buy = c^T x.$$

The transfer function

$$h(p) = c^T(pI - A)^{-1}b$$

is strictly positive real (SPR) if and only if there exist positive definite matrices P and Q such that

$$A^T P + PA = -QPb = c.$$

Appendix D

Feedback Linearization

Appendix E

Sliding Control

Bibliography

Andrieu, V. and Praly, L. (2006). On the existence of a kazantzis–kravaris/luenberger observer. *SIAM Journal on Control and Optimization*, 45(2):432–456.

Astolfi, A. and Ortega, R. (2003). Immersion and invariance: A new tool for stabilization and adaptive control of nonlinear systems. *IEEE Transactions on Automatic control*, 48(4):590–606.

Bernard, P. (2019). *Observer design for nonlinear systems*, volume 479. Springer.

Bernard, P., Andrieu, V., and Astolfi, D. (2022). Observer design for continuous-time dynamical systems. *Annual Reviews in Control*.

Bertsekas, D. (2012). *Dynamic programming and optimal control: Volume I*, volume 1. Athena scientific.

Besançon, G., Bornard, G., and Hammouri, H. (1996). Observer synthesis for a class of nonlinear control systems. *European Journal of control*, 2(3):176–192.

Blekherman, G., Parrilo, P. A., and Thomas, R. R. (2012). *Semidefinite optimization and convex algebraic geometry*. SIAM.

Dawson, C., Gao, S., and Fan, C. (2023). Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics*.

Ebenbauer, C., Renz, J., and Allgower, F. (2005). Polynomial feedback and observer design using nonquadratic lyapunov functions. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 7587–7592. IEEE.

Hammouri, H. and de Leon Morales, J. (1990). Observer synthesis for state-affine systems. In *29th IEEE Conference on Decision and Control*, pages 784–785. IEEE.

Janny, S., Andrieu, V., Nadri, M., and Wolf, C. (2021). Deep kkl: Data-driven output prediction for non-linear systems. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 4376–4381. IEEE.

Kalman, R. E. and Bucy, R. S. (1961). New results in linear filtering and prediction theory.

Karagiannis, D. and Astolfi, A. (2005). Nonlinear observer design using invariant manifolds and applications. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 7775–7780. IEEE.

Kazantzis, N. and Kravaris, C. (1998). Nonlinear observer design using lyapunov's auxiliary theorem. *Systems & Control Letters*, 34(5):241–247.

Lasserre, J. B. (2001). Global optimization with polynomials and the problem of moments. *SIAM Journal on optimization*, 11(3):796–817.

Lasserre, J. B. (2009). *Moments, positive polynomials and their applications*, volume 1. World Scientific.

Luenberger, D. G. (1964). Observing the state of a linear system. *IEEE transactions on military electronics*, 8(2):74–80.

Magron, V. and Wang, J. (2023). *Sparse polynomial optimization: theory and practice*. World Scientific.

Miao, K. and Gatsis, K. (2023). Learning robust state observers using neural odes. In *Learning for Dynamics and Control Conference*, pages 208–219. PMLR.

Murray, R., Chandrasekaran, V., and Wierman, A. (2021). Signomial and polynomial optimization via relative entropy and partial dualization. *Mathematical Programming Computation*, 13:257–295.

Niazi, M. U. B., Cao, J., Sun, X., Das, A., and Johansson, K. H. (2023). Learning-based design of luenberger observers for autonomous nonlinear systems. In *2023 American Control Conference (ACC)*, pages 3048–3055. IEEE.

Nie, J. (2023). *Moment and Polynomial Optimization*. SIAM.

Slotine, J.-J. E., Li, W., et al. (1991). *Applied nonlinear control*, volume 199. Prentice hall Englewood Cliffs, NJ.

Vinograd, R. È. (1957). Inapplicability of the method of characteristic exponents to the study of non-linear differential equations. *Matematicheskii Sbornik*, 83(4):431–438.

Wang, J. (2022). Nonnegative polynomials and circuit polynomials. *SIAM Journal on Applied Algebra and Geometry*, 6(2):111–133.

Yang, H. and Carlone, L. (2022). Certifiably optimal outlier-robust geometric perception: Semidefinite relaxations and scalable global optimization. *IEEE transactions on pattern analysis and machine intelligence*, 45(3):2816–2834.

Yang, H., Liang, L., Carlone, L., and Toh, K.-C. (2022). An inexact projected gradient method with rounding and lifting by nonlinear programming for

solving rank-one semidefinite relaxation of polynomial optimization. *Mathematical Programming*, pages 1–64.