# Lab 5A: Dynamical Systems

Juan Elenter, Ignacio Hounie,
Damian Owerko, and Alejandro Ribeiro*

April 9, 2023

## 1 Dynamical Systems

A dynamical system is a system that evolves in time. will consider here a *state* vector $\mathbf{x}(t)$ and a *control* action $\mathbf{u}(t)$. In a dynamical system the state at time $t+1$ is given by

$$\mathbf{x}(t+1) = f(\mathbf{x}(t), \mathbf{u}(t)). \tag{1}$$

The interpretation of (1) is that the state of the system $\mathbf{x}(t)$ and the control action $\mathbf{u}(t)$ determine the state of the system at time $t+1$. Our goal is to choose control actions $\mathbf{u}(t)$ to manipulate the state trajectory $\mathbf{x}(t)$ to satisfy some stated goal.

We point out that the system in (1) is time invariant, deterministic, and memoryless. Dynamical systems may also have memory, be stochastic, or time varying.

### 1.1 Circuit Navigation

An example of a time invariant, deterministic, and memoryless dynamical system is illustrated in Figure 1. Shown in this figure are trajectories

*In alphabetical order.

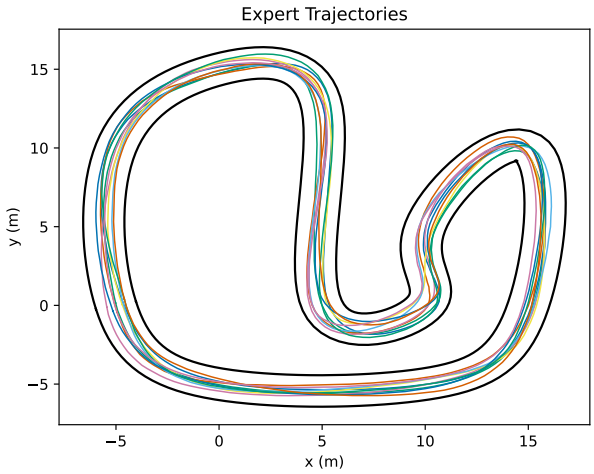


**Figure 1.** Circuit Navigation. We observe trajectories of a car that is driven along a circuit. Shown here are observed positions $\mathbf{p}_q(t)$ for some trajectories. For each trajectory we also observe velocities $\mathbf{v}_q(t)$ and accelerations $\mathbf{a}(t)$.

that we observe as a car is driven along a circuit. A simple characterization of a car is to model it as an ideal point mass whose acceleration we control. Thus, the state of the system at time $t$ includes the position $\mathbf{p}(t)$ and the velocity $\mathbf{v}$(t) of the car. Positions and velocities are both vectors with horizontal and vertical coordinates. We therefore write the state of the car as,

$$\mathbf{x}(t) \;=\; [\,\mathbf{p}(t);\, \mathbf{v}(t)\,] \;=\; [\,p_{\mathrm{H}}(t);\, p_{\mathrm{V}}(t);\, v_{\mathrm{H}}(t);\, v_{\mathrm{V}}(t)\,]. \tag{2}$$

The control action of the car is the acceleration $\mathbf{a}(t)$. As is the case of positions and velocities, the acceleration contains horizontal and vertical components. We therefore write the control action as

$$\mathbf{u}(t) \;=\; \mathbf{a}(t) \;=\; [\,a_{\mathrm{H}}(t);\, a_{\mathrm{V}}(t)\,]. \tag{3}$$

As we have assumed that the car is an ideal point mass, we can readily write the dynamical system using the laws of motion. To do so we just need to specify the sampling time $T_s$ that elapses between subsequent time instances and write,

$$\begin{aligned} \mathbf{p}(t+1) \;&=\; \mathbf{p}(t) + T_s\mathbf{v}(t) + \frac{T_s^2}{2}\mathbf{a}(t), \\ \mathbf{v}(t+1) \;&=\; \mathbf{v}(t) + T_s\mathbf{a}(t). \end{aligned} \tag{4}$$

This is true if the acceleration $\mathbf{a}(t)$ is maintained between times $tT_s$ and $(t+1)T_s$. It just follows from the definitions of velocity and acceleration.

In reality, the dynamics of a car are more complex than the dynamics of an ideal point mass. The laws of motion in (4) are therefore a *model* of the real dynamical system.

**Task 1** Load the data from this link.. This data contains $Q = 10$ trajectories, which correspond to different laps of the circuit shown in Figure 1. For each of these trajectories we have positions $\mathbf{p}_q(t)$, velocities $\mathbf{v}_q(t)$, and accelerations $\mathbf{a}_q(t)$ for times $t$ ranging from $t = 0$ to $t = T = 10s$. Positions are measured in meters ($m$), velocities in meters per second ($m/s$), and accelerations in meters per second squared ($m/s^2$). The sampling time is $T_s = 0.05s$.

Plot the positions $\mathbf{p}_q(t)$ for some chosen trajectories. ■

## 1.2 Model Predicted States

We expect the model in (4) to be a reasonable approximation of the actual car dynamics. Let us test this expectation.

As a first figure of merit we consider the state predictions that the model makes. For a given trajectory $q$ and a given point in time $t$ we consider the *observed* state of the car $\mathbf{x}_q(t)$ and the control action $\mathbf{u}_q(t)$. We then use the ideal point mass model in (4) to *predict* the state at time $t+1$,

$$\begin{aligned} \hat{\mathbf{p}}_q(t+1) &= \mathbf{p}_q(t) + T_s\mathbf{v}_q(t) + \frac{T_s^2}{2}\mathbf{a}_q(t), \\ \hat{\mathbf{v}}_q(t+1) &= \mathbf{v}_q(t) + T_s\mathbf{a}_q(t). \end{aligned} \tag{5}$$

We compare predicted positions $\hat{\mathbf{p}}_q(t+1)$ and predicted velocities $\hat{\mathbf{v}}_q(t+1)$ with observed positions $\mathbf{p}_q(t+1)$ and observed velocities $\mathbf{v}_q(t+1)$ to test the accuracy of the model in (4).

**Task 2** For a trajectory $\mathbf{x}_q(t)$ define the state estimation errors as

$$e_q(t) = \|\mathbf{p}_q(t) - \hat{\mathbf{p}}_q(t)\| + \|\mathbf{v}_q(t) - \hat{\mathbf{v}}_q(t)\|. \tag{6}$$

Plot trajectory errors $e_q(t)$ for some trajectories. Compute and plot the average error $\bar{e}_q(t) = (1/Q)\sum_{q=1}^{Q} e_q(t)$. Compute the average error over time $\bar{\bar{e}}_q(t) = (1/T)\sum_{t=1}^{T} \bar{e}_q(t)$. ■

## 1.3 Model Predicted Trajectories

The errors in Task 2 are small. This is an indication that the ideal point mass model is accurate. To further test the accuracy of this model we compare observed *trajectories* with model predicted trajectoriesThis entails selecting the actions $\mathbf{a}_q(t)$ of a particular trajectory to generate the model predicted trajectory as

$$\begin{aligned} \hat{\mathbf{p}}_q(t+1) &= \hat{\mathbf{p}}_q(t) + T_s\hat{\mathbf{v}}_q(t) + \frac{T_s^2}{2}\mathbf{a}_q(t), \\ \hat{\mathbf{v}}_q(t+1) &= \hat{\mathbf{v}}_q(t) + T_s\mathbf{a}_q(t). \end{aligned} \tag{7}$$

Observe how in this expression we use the *observed* accelerations $\mathbf{a}_q(t)$ as control actions to generate the trajectory $\hat{\mathbf{x}}(t) = [\hat{\mathbf{p}}(t); \hat{\mathbf{v}}(t)]$ predicted by the model in (4). This differs from (5) in which state predictions at time $t+1$ depend on the state that is observed at time $t$. In (7) we compound predictions.

Comparing $\hat{\mathbf{x}}(t)$ with $\mathbf{x}(t)$ gives an alternative measurement of the accuracy of the ideal point mass model in (4).

**Task 3** To compare trajectories $\mathbf{x}_q(t)$ with model predicted trajectories $\hat{\mathbf{x}}(t)$ we define the position errors

$$e_q(t) = \|\mathbf{p}_q(t) - \hat{\mathbf{p}}_q(t+1)\|. \tag{8}$$

Plot trajectory errors $e_q(t)$ for some trajectories. Compute and plot the average error $\bar{e}_q(t) = (1/Q)\sum_{q=1}^{Q} e_q(t)$.

Plot the positions $\mathbf{p}_q(t)$ along with the predicted positions $\hat{\mathbf{p}}(t)$ for some chosen trajectories.

These plots are likely unexpected. Comment. ■

The main conclusion of Task 3 is summarized in Figure 2. We see that the trajectories generated by the model deviate from the observed trajectories.

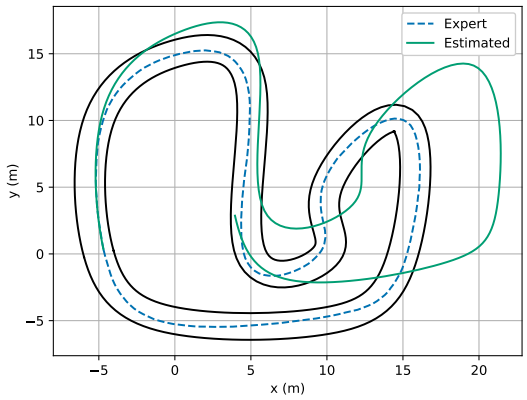


**Figure 2.** Trajectories and Model Predicted Trajectories. We compare observed trajectories with trajectories generated by the ideal point mass model in (4). As time progresses, model mismatch errors accumulate. This is a distinctive challenge of dynamical systems.

This is an indication that the model is inaccurate. It is also important to observe that the errors accumulate over time. We see that this is true because as time progresses the difference between observed trajectories and predicted trajectories grows.

## 2 Model Learning

To mitigate the limitations of the ideal point mass model we can use observed trajectories to learn a more accurate model. To that end we consider a linear parameterization to predict the state at time $t+1$ given observations of the state and control input at time $t$,

$$\hat{\mathbf{x}}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \tag{9}$$

In this equation the matrices $\mathbf{A}$ and $\mathbf{B}$ are parameters that we will learn. These parameters map the *observed* state $\mathbf{x}(t)$ and the *observed* control input $\mathbf{u}(t)$ into a *prediction* $\hat{\mathbf{x}}(t+1)$ of the next state.

The *prediction* $\hat{\mathbf{x}}(t+1)$ made by (9) is, in general, different from the actual

state $\mathbf{x}(t+1)$ *observed* at time $t+1$. We therefore seek parameters $\mathbf{A}$ and $\mathbf{B}$ that make predictions and reality as close as possible. For a loss function $\ell(\mathbf{x}(t+1), \hat{\mathbf{x}}(t+1))$ this gives the empirical risk minimization (ERM) problem

$$\begin{aligned}(\mathbf{A}^*, \mathbf{B}^*) &= \operatorname*{argmin}_{\mathbf{A},\mathbf{B}} \frac{1}{Q}\sum_{q=1}^{Q} \frac{1}{T}\sum_{t=1}^{T} \ell\Big(\mathbf{x}(t+1),\, \hat{\mathbf{x}}(t+1)\Big) \\ &= \operatorname*{argmin}_{\mathbf{A},\mathbf{B}} \frac{1}{Q}\sum_{q=1}^{Q} \frac{1}{T}\sum_{t=1}^{T} \ell\Big(\mathbf{x}(t+1),\, \mathbf{A}\mathbf{x}(t)+\mathbf{B}\mathbf{u}(t)\Big). \qquad (10)\end{aligned}$$

This problem differs from standard ERM in that we sum over trajectories and points in time. This is a minimal difference and the same techniques we use for solving standard ERM can be used to solve (10).

**Task 4** Create an object to represent the linear dynamical system model in (9). Instantiate this object to represent a car navigating a circuit. This requires a state $\mathbf{x}(t)$ with the car's position and velocity [cf. (2)] and a control action $\mathbf{u}(t)$ representing the car's acceleration. [cf. (3)].

Use the trajectories loaded in Task 1 to train a model for a car navigating a circuit. This training entails solving (10). Use an L1 loss and remember to save some trajectories for validation. We recommend that you save a single trajectory for this purpose.

Report the training loss and the test loss. ■

The test loss that we observe in Task 4 is smaller than the average error that we observed in Task 2. This indicates that the learned model is better than the ideal point mass model.

Observe that the loss computed in Task 4 compares the model's state predictions in (9) with observed states. As we did in Section 1.3 we may also consider predicted *trajectories.* This are defined as

$$\hat{\mathbf{x}}(t+1) = \mathbf{A}\hat{\mathbf{x}}(t) + \mathbf{B}\mathbf{u}(t). \qquad (11)$$

In both, (9) and (11), we apply *observed* actions $\mathbf{u}(t)$. The difference is that in (9) we start from the observed state $\mathbf{x}(t)$ whereas in (11) we start from the predicted state $\hat{\mathbf{x}}(t)$. Concatenating state predictions results in the accumulation of errors. We evaluate this next.

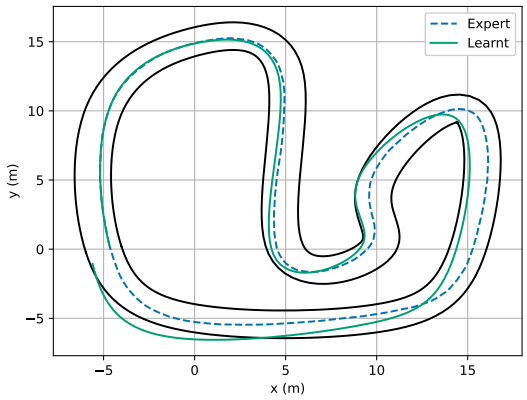


**Figure 3.** Trajectories Predicted by Learned Model. We compare observed trajectories with trajectories generated by the linear model with the parameters learned in Task 4. The model is more accurate than the ideal point mass model of Figure 2 but we still observe accumulation of errors.

**Task 5** As in Task 3 we compare trajectories $\mathbf{x}_q(t)$ with model predicted trajectories $\hat{\mathbf{x}}_q(t)$. We do that by evaluating the position error $e_q(t)$ defined in (8). Plot trajectory errors $e_q(t)$ for test trajectories. If you have more than one test trajectory, compute and plot the average error $\bar{e}_q(t) = (1/Q)\sum_{q=1}^{Q} e_q(t)$.

Plot the positions $\mathbf{p}_q(t)$ along with the predicted positions $\hat{\mathbf{p}}(t)$ for a test trajectory.

Compared the learned model with the ideal point mass model. ■

The main outcome of Task 5 is Figure 3. We see that the trajectories predicted by the learned model are more accurate. We still observe some deviation between the actual trajectory and the trajectory predicted by the model. As we said, this is a fundamental feature of dynamical systems. A more accurate model mitigates but does not eliminate the accumulation of errors. But in this case the deviation is much smaller.

This learned model is a good starting point for controlling the car. We explain how to do this in the next section.

# 3 Dynamical Systems Control

We are now given control of the car. This entails finding a sequence of actions $\mathbf{u}(t)$ that navigates the circuit. This means that our goal is to find a policy $\pi$ that maps states $\mathbf{x}(t)$ to control actions $\mathbf{u}(t)$,

$$\boldsymbol{\pi} : \mathbf{u}(t) = \pi(\mathbf{x}(t)), \tag{12}$$

such that the state trajectory $\mathbf{x}(t)$ of the dynamical system satisfies some given specifications.

The particular control problem we consider is to follow a reference trajectory $\mathbf{x}_{\mathrm{R}}(t)$. To formulate this problem mathematically consider a dynamical system in which the state at time $t = 0$ is given and set to $\mathbf{x}(0) = \mathbf{x}_0$. We say that $\mathbf{x}_0$ is the initial condition of the dynamical system. Starting from this initial condition we generate a trajectory $\mathbf{x}_\pi(t)$ by continuous execution of the policy $\pi$. In this trajectory the control action at time $t$ is $\mathbf{u}(t) = \pi(\mathbf{x}_\pi(t))$. Since the evolution of the dynamical system follows (1), the trajectory $\mathbf{x}_\pi(t)$ is generated by the recursion,

$$\mathbf{x}_\pi(t+1) \;=\; f\Big(\mathbf{x}_\pi(t), \pi\big(\mathbf{x}_\pi(t)\big)\Big). \tag{13}$$

To follow the reference trajectory $\mathbf{x}_{\mathrm{R}}(t)$ we introduce a loss $\ell$ to compare the reference trajectory $\mathbf{x}_{\mathrm{R}}(t)$ with trajectories $\mathbf{x}_\pi(t)$ generated by different policies $\pi$. We then search of the optimal policy,

$$\pi^* = \operatorname*{argmin}_{\pi} \frac{1}{T} \sum_{t=1}^{T} \ell\,\Big(\,\mathbf{x}_{\mathrm{R}}(t),\, \mathbf{x}_\pi(t)\,\Big). \tag{14}$$

This optimization problem appears similar to empirical risk minimization problems we encountered in earlier labs. It appears in particular, similar to (10). This similarity is misleading. The problems are, in fact, fundamentally different.

The fundamental difference between (14) and standard risk minimization is that the goal in (14) is to find policies $\pi(\mathbf{x}(t))$ that generate suitable *trajectories* and in a dynamical system the choice of action at time $t$ influences the trajectory of the system at *all* future times $s > t$. Conversely, the state of the system at time $t$ depends on *all* of the actions that were chosen at times $s < t$.

To explain this point better, start at time $t = 0$ and recall that the initial condition $\mathbf{x}_\pi(0) = \mathbf{x}_0$ is known. For this initial conidition and choosing actions under policy $\pi$, the control action at time $t = 0$ is $\mathbf{u}(0) = \pi(\mathbf{x}_\pi(0)) = \pi(\mathbf{x}_0)$. We now have the state $\mathbf{x}_\pi(0)$ given and the control policy $\mathbf{u}(0)$ chosen. Since the evolution of the dynamical system follows (1) the state $\mathbf{x}_\pi(1)$ at time $t = 1$ is

$$\mathbf{x}_\pi(1) \;=\; f\Big(\mathbf{x}(0), \pi\big(\mathbf{x}(0)\big)\Big) \;=\; f\Big(\mathbf{x}_0, \pi\big(\mathbf{x}_0\big)\Big). \tag{15}$$

We emphasize that In (15) the state as $\mathbf{x}_\pi(1)$ depends on the policy. Different policy choices $\pi$ result in different states $\mathbf{x}_\pi(1)$.

We now consider the next control action which is to be taken at time $t = 1$. Since the state of the system is $\mathbf{x}_\pi(1)$, the choice of action under policy $\pi$ is $\mathbf{u}(1) = \pi(\mathbf{x}_\pi(1))$. With state given and policy chosen, the dynamical system now moves to the state

$$\mathbf{x}_\pi(2) \;=\; f\Big(\mathbf{x}_\pi(1), \pi\big(\mathbf{x}_\pi(1)\big)\Big). \tag{16}$$

This state depends on the control action $\mathbf{u}(1) = \pi(\mathbf{x}_\pi(1))$. It also depends on the choice of control action $\mathbf{u}(0) = \pi(\mathbf{x}(0))$ because the state $\mathbf{x}_\pi(1)$ depends on this choice.

We repeat the procedure at time $t = 2$. The system is in state $\mathbf{x}_\pi(2)$, we choose action $\mathbf{u}(2) = \pi(\mathbf{x}_\pi(2))$ and the dynamical system transitions into state

$$\mathbf{x}_\pi(3) \;=\; f\Big(\mathbf{x}_\pi(2), \pi\big(\mathbf{x}_\pi(2)\big)\Big). \tag{17}$$

We see here that the state of the system at time $t = 3$ depends on all of the three actions chosen by the policy $\pi$ at times $s < 3$. Conversely, we see that the action chosen at time $t = 0$ affects all of the states observed at times $s > 0$. This propagation of dependencies between states and actions continues as we keep making control choices $\mathbf{u}(t) = \pi(\mathbf{x}_\pi(t))$ that depend on states $\mathbf{x}_\pi(t)$ that depend on all of the control choices $\mathbf{u}(s) = \pi(\mathbf{x}_\pi(s))$ taken at previous times $s < t$.

To emphasize that (14) is a problem in which policy choices $\pi(\mathbf{x}_\pi(t))$ are coupled across time we rewrite it to make the evolution of the dynamical

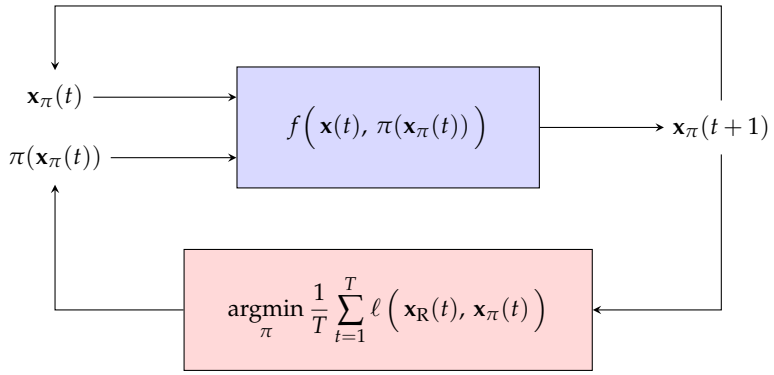


**Figure 4.** Optimal Control. The dynamical system is represented by the feedback loop at the top. Current states and control inputs are mapped to future states, which then become current states at the next time index. The controller is represented by the feedback loop at the bottom. We observe the trajectories generated by different policies and choose the one that solves (18). To execute this controller me must know the true behavior of the dynamical system.

system explicit,

$$\begin{aligned} \boldsymbol{\pi}^* \;=\; & \underset{\pi}{\text{argmin}}\; \frac{1}{T}\sum_{t=1}^{T} \ell\left(\mathbf{x}_{\text{R}}(t),\, \mathbf{x}_\pi(t)\right), \\ & \text{with} \quad \mathbf{x}_\pi(t+1) = f\Big(\mathbf{x}_\pi(t), \pi\big(\mathbf{x}_\pi(t)\big)\Big), \\ & \qquad\quad \mathbf{x}_\pi(0) = \mathbf{x}_0\,. \end{aligned} \tag{18}$$

This is the same problem in (14), where we write the evolution of the dynamical system and the initial condition explicitly. We say that $\boldsymbol{\pi}^*$ is the optimal control policy and we refer to (18) as the optimal control problem.

The optimal control strategy implied by (18) is illustrated in Figure 4. When we specify the policy $\pi$, the trajectory of the system is given. This is signified by the feedback loop at the top in which the state input $\mathbf{x}_\pi(t)$ and the control input $\pi(\mathbf{x}_\pi(t))$ produce the state output $\mathbf{x}_\pi(t+1)$. We say that this is a feedback loop, because the state output $\mathbf{x}_\pi(t+1)$ becomes an input to the dynamical system at time $t+1$.

The trajectory of the system is also an input to the block that chooses

the optimal control policy. This is signified by the feedback loop at the bottom. The controller observes the effect of different policies and chooses the one that solves (4). We say that this block is a controller because it chooses the control policy.

## 3.1 Model Based Optimization

To find the optimal control policy $\boldsymbol{\pi}^*$ we need to solve the optimization problem in (18). In this problem, control actions $\pi(\mathbf{x}_\pi(t_1))$ and $\pi(\mathbf{x}_\pi(t_2))$ chosen at different times are coupled. They both influence the state of the dynamical system at times $s > t_1$ and $s > t_2$. This coupling notwithstanding, we can still compute gradients of the loss to implement a descent algorithm to find the optimal control policy $\pi^*$.

Here we encounter a roadblock: The true behavior of the dynamical system is unknown. We circumvent this roadblock with the used of a model. This could be the model in (4) based on our understanding of the laws of motion or the model we learned in Section 2. We use the latter which turned out to be more accurate.

Consider then the learned dynamical system in (11) when controlled by the policy $\pi$. This dynamical system evolves according to

$$\hat{\mathbf{x}}_\pi(t+1) = \mathbf{A}\hat{\mathbf{x}}_\pi(t) + \mathbf{B}\pi\big(\hat{\mathbf{x}}_\pi(t)\big). \tag{19}$$

In (19) we use $\hat{\mathbf{x}}_\pi(t)$ to denote the state of the system. This notation emphasizes that the trajectory results from the execution of actions $\pi(\hat{\mathbf{x}}_\pi(t))$ on the *model* in (11).

We search now for policies that are optimal at controlling the model. They are given by the solution of the optimization problem,

$$\begin{aligned} \pi^* \;=\; \underset{\pi}{\operatorname{argmin}} \;& \frac{1}{T}\sum_{t=1}^{T} \ell\left(\mathbf{x}_{\mathrm{R}}(t),\, \hat{\mathbf{x}}_\pi(t)\right), \\ \text{with} \quad & \hat{\mathbf{x}}_\pi(t+1) = \mathbf{A}\hat{\mathbf{x}}_\pi(t) + \mathbf{B}\pi\big(\hat{\mathbf{x}}_\pi(t)\big), \\ & \hat{\mathbf{x}}_\pi(0) = \mathbf{x}_0\ . \end{aligned} \tag{20}$$

The problems in (18) and (20) are optimal control problems for different dynamical systems. The formulation in (18) is on the actual dynamical

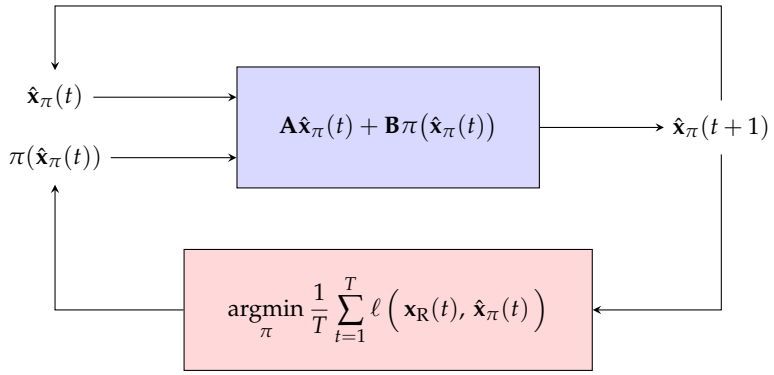


**Figure 5.** Model Based Optimization. As in Figure 4 we represent the dynamical system and the controller. Here, however, we work with the *model*, not the true system [cf. (20)]. This yields a policy optimization problem that we can solve, because the model is known. The price we pay is that the policy is optimal for the model, not the true system.

system that determines the car's motion. We can't solve this problem because we do not know the function $f$. The problem in (20) is an optimal control problem on the learned *model* of the dynamical system. We can solve this problem because we have access to $\mathbf{A}$ and $\mathbf{B}$.

A scheme describing (20) is shown in Figure 6. As in Figure 4 we have a feedback loop at the top representing the dynamical system and a feedback loop at the bottom representing the controller. The difference is that in Figure 6 the top feedback loop represents a model of the true dynamical system. The feedback loop at the bottom solves (20). This finds a policy that is optimal at controlling the *model*. Which need not be, indeed, it most likely is not, the same policy that is optimal at controlling the true dynamical system.

**Task 6** Solve the optimization problem in (20) for the dynamical system defined by the parameters $\mathbf{A}$ and $\mathbf{B}$ learned in Task 4. Use as a reference trajectory one of the trajectories that you used for validations in Task 4. Use MSE loss as figure of merit.

The solution of this optimization problem is a sequence of actions $\mathbf{u}(0 : T-1)$ with $\mathbf{u}(t) = \pi^*(\hat{\mathbf{x}}_\pi(t))$. ■

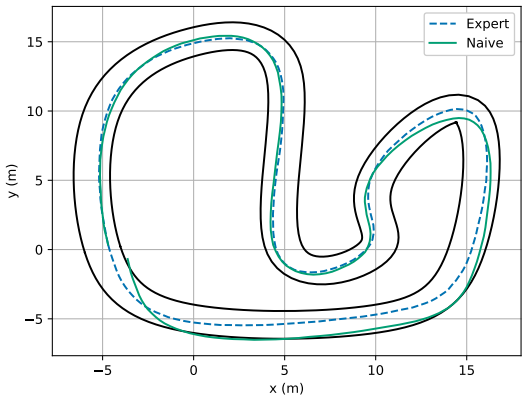


**Figure 6.** Model Based Optimization.

## 3.2 Evaluation of Control Policy

Solving optimal control on a model of a real system is not the same as solving optimal control on the real system. In the case of navigating a circuit, we saw in Task 5 that the learned model is accurate. We therefore expect that the solution of (20) is a good approximation to solving (18). We test this expectation here.

To test this expectation we execute the control actions $\mathbf{u}(0 : T-1)$ that we found as solutions of (20) in Task 6 in the actual dynamical system $f$. Since we do not have access to an actual car, we will test actions $\mathbf{u}(0 : T-1)$ in a car simulator.

**Task 7** Download the car simulator from this link. Execute the control actions $\mathbf{u}(0 : T-1)$ found as solutions of (20) in Task 6. Evaluate and plot the MSE loss between the generated trajectory and the reference trajectory. ■

The main conclusion of Task 7 is summarized in Figure 6. This conclusion is that model based optimization fails at controlling the car. The challenge is, as usual, that errors accumulate over time.

A successful controller requires the introduction of a mechanism to correct the accumulation of errors. This is closed loop control, which we study in Lab 5B.

# 4 Report

Do not take much time to prepare a lab report. We do not want you to report your code and we don't want you to report your work. Just give us answers to questions we ask. Specifically give us the following:

| **Question** | **Report deliverable** |
|---|---|
| Task 1 | Do not report |
| Task 2 | Plot of trajectory errors $e_q(t)$ [cf. (6)] |
| | Average error $\bar{\bar{e}}_q(t)$ |
| Task 3 | Plot of trajectory errors $e_q(t)$ [cf. (8)] |
| | Plot of an observed trajectory and the corresponding model predicted trajectory |
| | Paragraph explaining differences between observed and model predicted trajectories |
| Task 4 | Training loss and test loss |
| Task 5 | Plot of test trajectory errors $e_q(t)$ |
| | Plot of an observed trajectory and the corresponding learned model predicted trajectory |
| | Paragraph comparing ideal point mass model with learned model |
| Task 6 | Optimal loss |
| Task 7 | Plot of L1 loss as a function of time |

We will check that your answers are correct. If they are not, we will get back to you and ask you to correct them. As long as you submit responses, you get an A for the assignment. It counts for 10% of your lab grade.