Homework 5: Ergodic Control

Due: 11:59PM, May 30, 2025

Software

Python packages *NumPy*, *SciPy* and *Matplotlib* are sufficient for the calculation used this assignment. *SciPy* contains the function "solve_bvp", which can be used to solve the two-point boundary value problem in this homework. The computation of this assignment could take a while to finish on your computer, but it should not take more than *a few minutes* per iLQR iteration. Feel free to use Jupyter Notebook (or Google Colab) as the programming environment.

Preliminaries

In this assignment, you will need to implement ergodic control using the iterative linear quadratic regulator (iLQR) you implemented in the previous assignment. You should re-use your iLQR code from the previous assignment as much as you can. To help you get started on the ergodic metric and its numerical implementation, this notebook demonstrate how to configure the Fourier basis function and compute the ergodic metric between a given spatial distribution and and a trajectory: `https://drive.google.com/file/d/1DCdPa3d9tYNY4clxriuu8hD2Z2XZ8_JM/view?usp=sharing`.

You will still use the "solve_bvp" function from SciPy to solve the optimal control problem. In the iLQR template provided for the previous assignment, the only place the ergodic metric will change is how you compute the variable "a_list". You will need to implement ergodic control for three different dynamic systems. The changes in the system dynamics will affect how you compute "A_list" and "B_list".

Important notes

Even though we define the Fourier basis functions over a bounded rectangular space, the Fourier basis functions can still be evaluated outside of the bounded space. Furthermore, because the Fourier basis functions are inherently periodic, you will actually get very similar evaluations outside of the bounds. This introduces an issue in practice. During line search, if the candidate trajectory goes out of the bound, it may never go back!

In practice, there are two fixes to this issue. The more robust solution is to implement a barrier function as part of the objective function to explicit prevent the robot to travel out of bounds. A barrier function may look like this:

$$barr(x) = \min\{0, x - \text{lower bound}\}^2 + \max\{0, x - \text{upper bound}\}^2. \tag{14}$$

The other solution, which I recommend for this assignment, is to pick a small enough initial step, so the line search never generates candidate trajectory that is out of bounds.

Lastly, the magnitude of the parameters in this assignment (e.g., entries of the matrix Q and R) should be around 0.01 to 0.001.

Problems

1. (40 pts) We have a 1m-by-1m search space and the target spatial distribution as a Gaussian

mixture distribution specified as follow:

$$w_1 = 0.5, w_2 = 0.2, w_3 = 0.3,$$
$$\mu_1 = [0.35, 0.38]^\top, \mu_2 = [0.68, 0.25]^\top, \mu_3 = [0.56, 0.64]^\top,$$
$$\Sigma_1 = \begin{bmatrix} 0.01 & 0.004 \\ 0.004 & 0.01 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 0.005 & -0.003 \\ -0.003 & 0.005 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 0.008 & 0.0 \\ 0.0 & 0.004 \end{bmatrix}. \tag{15}$$

The system is a 2D first-order dynamic system:

$$\dot{x}(t) = f(x(t), u(t)) = u(t) \tag{16}$$

with the initial condition $x_0 = [0.3, 0.3]$. Generate an ergodic trajectory with a time horizon of 10 seconds with $dt = 0.1s$.

Turn in: A figure with 3 plots showing: (1) The converged ergodic trajectory; (2) Optimal control signals; (3) The evolution of objective value over iterations. An example is shown below.

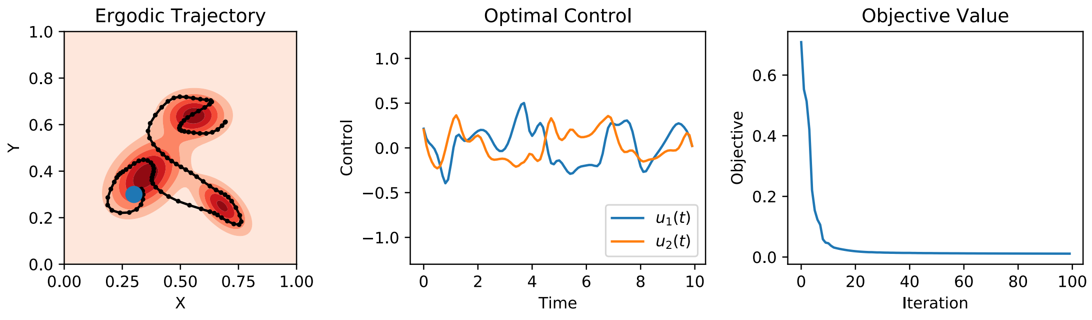

Figure 8: Example visualization of the ergodic trajectory with a first order system.

2. (30 pts) Generate an ergodic trajectory with the same specifications from the last problem, but with the system dynamics being a second-order system :

$$\ddot{x}(t) = f(x(t), u(t)) = u(t) \tag{17}$$

Turn in: A figure with 3 plots showing: (1) The converged ergodic trajectory; (2) Optimal control signals; (3) The evolution of objective value over iterations. An example is shown below.

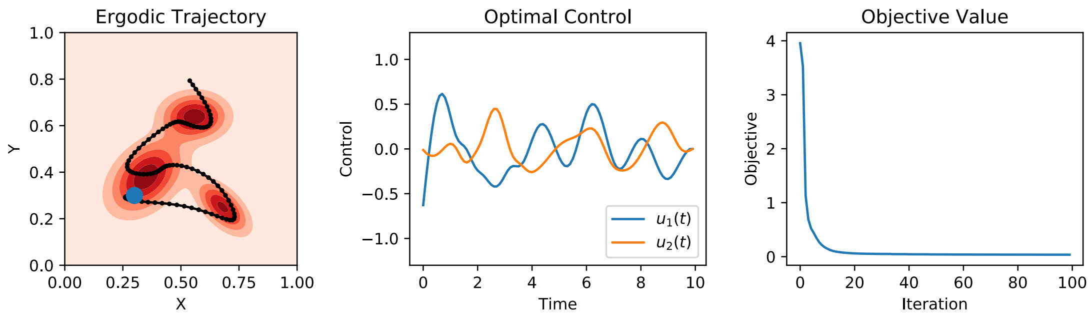

Figure 9: Example visualization of the ergodic trajectory with a second order system.

3. (30 pts) Generate an ergodic trajectory With the same specifications from the last problem, but with the system being a differential-drive vehicle and initial condition being $[0.3, 0.3, \frac{\pi}{2}]$. Note that you only need to evaluate the ergodic metric over the x, y position of the robot.

   Turn in: A figure with 3 plots showing: (1) The converged ergodic trajectory; (2) Optimal control signals; (3) The evolution of objective value over iterations. An example is shown below.

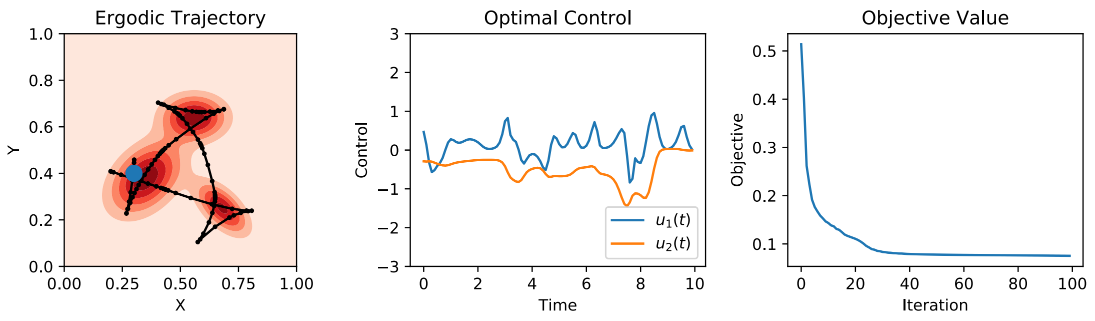

Figure 10: Example visualization of the ergodic trajectory with a differential-drive robot.