ME 193B / 292B: Feedback Control of Legged Robots
HW #3

Problem 1. **Vertical Spring-Mass Hopper**

$$\ddot{y} = \begin{cases} \text{Flight Dynamics, if } y > l_0, \\ \text{Stance Dynamics, if } y \leq l_0. \end{cases} \tag{1}$$

In this problem we will analyze a simple model of a vertical hopper and design our first controller for vertical hopping. Consider a spring-mass-damper system as shown in Figure 1. The free length of the spring is l_0. The mass enters into flight mode when its vertical position is greater than the free length and enters stance mode otherwise, i.e. the dynamics can be written as

During stance, a linear spring (Spring Force, $F_s = -k\,(y - l_0)$) and a linear damper (Damping Force, $F_d = -c\dot{y}$) along with a control input force $u(t)$ acts on the mass. Upon impact, assume an identity impact map (i.e. assume the post-impact position and velocity of the mass to be equal to the pre-impact positions and velocities). Furthermore, unless otherwise mentioned, use the parameters given in Table 1.

(a) Draw the free body diagrams in flight and in stance and write down the Flight and Stance Dynamics using the state $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \end{bmatrix}$.

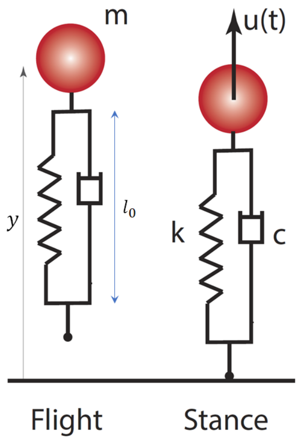

Figure 1: The spring mass hopper.

Parameter	Symbol	Value
Spring Constant	k	20kN/m
Mass	m	80kg
Free length of spring	l_0	1m
Damping Coefficient	c	5kN s/m
Initial height	x_0	5m

Table 1: Parameters for the spring mass hopper.

(b) We will define the Poincaré section $\mathcal{S}$ at the apex of the flight phase, i.e.

$$\mathcal{S} := \left\{ \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \end{bmatrix} \;\mid\; y > l_0, \dot{y} = 0 \right\}. \tag{2}$$

Let $P : \mathcal{S} \to \mathcal{S}$ represent the Poincaré map. Consider a state $\mathbf{x} \in \mathcal{S}$ and analytically compute $P(x)$ assuming zero control input force, i.e., $u(t) \equiv 0$, and zero damping, i.e., $c = 0$.

(c) What is a fixed point for this Poincaré map? Compute the linear approximation of the Poincaré map about the fixed point. Determine if the fixed point is stable (to do this, compute the eigenvalues of the Jacobian of the Poincaré map. Note that you will get one eigenvalue equal to zero.)

(d) Now, use your *intuition* to design a controller $u(t)$ such that the height at the apex converges to $y_d = 2m$. Choose **5** different initial conditions and provide plots of the vertical height $y(t)$ of the mass as a function of time t to illustrate convergence of the apex height of the hopping to y_d.

Note: For the above plot, you can either solve for the position $y(t)$ analytically, or by using a numerical solver such as `ode45` in MATLAB.

Problem 2. Van der Pol Oscillator

In the previous problem, we looked at deriving the Poincaré map analytically. This relied on computing the solution of the hybrid system in (1). However, for legged systems with nonlinear dynamics, it is almost impossible to analytically compute the solution and hence the Poincaré map. For such systems, the Poincaré map can be computed *numerically.* In this problem, we will look at a 2-dimensional nonlinear system and obtain a Poincaré map numerically.

Consider the Van der Pol Oscillator with state $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ with dynamics given by,

$$\dot{x}_1 = x_2, \tag{3}$$

$$\dot{x}_2 = \mu \left(1 - x^2\right) x_2 - x_1. \tag{4}$$

For this problem, assume $\mu = 1$. We will define the Poincaré section to be

$$\mathcal{S} := \{\mathbf{x} \mid x_1 = 0, x_2 > 0\}, \tag{5}$$

and the Poincaré map $P : \mathcal{S} \to \mathcal{S}$ as $P(\mathbf{x})$.

(a) Simulate the system (using `ode45` in MATLAB or any numerical solver of your choice) for 10 different initial conditions with $x_1, x_2 \in [-5, 5]$ for 50 seconds and provide a plot of x_1 vs. x_2 (also known as the *phase portrait*). Observe the periodic orbit that the solutions converge to.

(b) We will now numerically determine a fixed point x^* of the Poincaré map such that $P(x^*) = x^*$. We will do this through the following steps:

  i. Write a MATLAB function `x1 = VanderPolPoincare(x0)` that takes in a point `x0` on the Poincaré section and returns the point on the Poincaré section after one complete cycle (i.e. at the next intersection of the solution with the Poincaré section). Inside the function, you should change one of the coordinates of $\mathbf{x}_0$ to ensure $\mathbf{x}_0 \in \mathcal{S}$.

  ii. Pick any initial condition $\mathbf{x}_0 \in \mathcal{S}$ (i.e. pick $x_1 = 0$ and any $x_2 > 0$). Apply the above function to obtain $\mathbf{x}_1 = P(\mathbf{x}_0)$.

  iii. Repeat the above step to obtain a sequence $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_n$, (for a large n) where $\mathbf{x}_{k+1} = P(\mathbf{x}_k)$, i.e., $\mathbf{x}_k = \mathbf{x}(t_k)$ where t_k is the time of the k^{th} intersection of the system solution starting at initial condition $\mathbf{x}_0$.

  iv. Plot $x_2(t_k)$ vs k (Note that $x_1(t_k) \equiv 0$ since $\mathbf{x}_k \in \mathcal{S}$). Does this value of $x_2(t_k)$ settle to any particular value? If it does, then this value of x_2 along with $x_1 = 0$ will be a fixed point of the Poincaré map.

(c) We will next numerically determine the linear approximation of the Poincaré map about the fixed point you found. We have,

$$\mathbf{x}_{k+1} = P\left(\mathbf{x}_k\right). \tag{6}$$

Taking the Taylor series expansion around the fixed point $\mathbf{x}^*$ and ignoring the higher order terms, we get,

$$\mathbf{x}_{k+1} \approx P(\mathbf{x}^*) + \frac{\partial P}{\partial \mathbf{x}}(\mathbf{x}^*)\left(\mathbf{x}_k - \mathbf{x}^*\right), \tag{7}$$

$$\mathbf{x}_{k+1} - \mathbf{x}* \approx \frac{\partial P}{\partial \mathbf{x}}(\mathbf{x}^*)\left(\mathbf{x}_k - \mathbf{x}^*\right), \tag{8}$$

$$\Delta\mathbf{x}_{k+1} \approx A\Delta\mathbf{x}_k, \tag{9}$$

where $\Delta\mathbf{x}_k := (\mathbf{x}_k - \mathbf{x}^*)$ and $A := \frac{\partial P}{\partial \mathbf{x}}(\mathbf{x}^*)$. Thus, (9) is the linear approximation of the Poincaré map locally about the fixed point $\mathbf{x}^*$. This is a discrete-time system and we can assess the (local) stability of the fixed point $\mathbf{x}^*$ by analyzing the eigenvalues of A, i.e. $\mathbf{x}^*$ is locally stable if the magnitude of the eigenvalues of A are less than 1. (Remember that there will be one eigenvalue equal to zero that you will ignore - this arises due to the fact that $P(\mathbf{x}) \in \mathcal{S}$.)

Your task is to compute A numerically. The Euler approximation for computing the derivative of P around y^* is given by the following symmetric difference

$$A_j = \frac{\partial P}{\partial \mathbf{x}_j}(\mathbf{x}^*) \approx \frac{P\left(\mathbf{x}^* + \delta\mathbf{e}_j\right) - P\left(\mathbf{x}^* - \delta\mathbf{e}_j\right)}{2\delta}, \tag{10}$$

where A_j is the j^{th} column of A, δ is a small scalar denoting perturbation to $\mathbf{x}^*$ along the direction $\mathbf{e}_j$. Here, $\mathbf{e}_j$ is a vector with the j^{th} element being one and the rest zeros.

Using the `vanderPolPoincare(x0)` function and taking $\delta = 0.01$, compute A and find its eigenvalues (Note that one of the eigenvalues will be zero.) Is the limit cycle of the Van der Pol Oscillator stable?

Instructions

1. You may submit either a typeset or handwritten solution. In either case, submit a **PDF** version of your solutions on bCourses, with the naming convention: `firstName_lastName_HW.pdf`.

2. Start each problem on a separate page.
3. You may choose to use a symbolic math package such as the `Symbolic Math Toolbox` (`https://www.mathworks.com/help/symbolic/index.html`) in `MATLAB` or `Mathematica`.
4. Do include all your code, if any.
5. Please submit a single pdf of your HW. (If typset on a computer, please save to pdf. If handwritten, please scan to pdf.)
6. **Honor Code.** You are to do your own work. Discussing the homework with a friend is fine. Sharing results or MATLAB code is not.