EECS/ME/BioE 106B Homework 2: Controls

Spring 2023

Problem 1: Linear Lyapunov Stability

In this question, we'll perform an exploratory analysis of the concept of Lyapunov stability through the lens of linear systems. Note that in our analysis, we'll assume the equilibrium point of interest is $x_e = 0$. If x_e is nonzero, we can perform a simple change of coordinates $x' = x - x_e$ to make the equilibrium point at $x_e = 0$.

Definition 1 ***Lyapunov Stability***
The equilibrium point $x_e = 0 \in \mathbb{R}^n$ of the system $\dot{x} = f(x,t),\ x(t_0) = x_0,\ x \in \mathbb{R}^n$ is stable in the sense of Lyapunov (SISL) if for all $\varepsilon > 0$, there exists a $\delta > 0$ such that if:

$$||x_0|| < \delta \tag{1}$$

It is guaranteed that for all $t > t_0$:

$$||x(t)|| < \varepsilon \tag{2}$$

In words, this definition tells us that an equilibrium point is stable if *starting* close to the equilibrium point (some distance δ away) means that we'll *stay* close to the equilibrium point (within some distance ε) for all time. If a point is stable, we'll *always* be able to find a δ for every ε to satisfy this condition.
Although this definition is important for understanding what stability means in a mathematical sense, it's challenging to apply directly to study whether equilibrium points are stable or not. To see if an equilibrium point is Lyapunov stable, we commonly use the basic theorem of Lyapunov.

Theorem 1 ***Basic Theorem of Lyapunov***
If there exists a locally positive definite function $V(x,t)$ such that $\dot{V}(x,t) \le 0$ locally in x and for all t along the trajectories of the system, the origin of the system $\dot{x} = f(x,t)$ is locally stable in the sense of Lyapunov.

This powerful theorem states that if we can find a function $V(x,t)$, known as a *Lyapunov function*, that is positive definite for some region around the origin and decreases for all time within that region, the origin of the system is locally stable. Why is this?
If V is locally positive definite around the origin, its only minimum is at the origin, $x = 0$. If $V(x,t)$ decreases as the system evolves, x will never move away from the origin. Thus, x will *remain close* to the origin even though it didn't necessarily start there!

Questions

In this question, we'll work towards finding conditions for the Lyapunov stability of:

$$\dot{x} = Ax, \ x \in \mathbb{R}^n \tag{3}$$

1. First, we need to identify a Lyapunov function $V(x,t)$. Consider the function:

   $$V(x) = x^T Px, \ x \in \mathbb{R}^n, P \in \mathbb{R}^{n\times n} \tag{4}$$

   Where P is a positive definite matrix, denoted $P \succ 0$. If $P \succ 0$, we know a few important things: P is symmetric, its eigenvalues are all > 0, and its eigenvectors are all orthogonal. Let T be a matrix whose columns are the eigenvectors of $P \succ 0$. If $z = T^{-1}x$, show that:

   $$z^T Dz = x^T Px \tag{5}$$

   Where D is a diagonal matrix with the eigenvalues of P along the diagonal. *Hint: If the columns of T are orthogonal, then $T^{-1} = T^T$.*

2. Now, we need to show our candidate Lyapunov function $V(x) = x^T Px$ is positive definite! Formally, a function $V(x)$ is positive definite if there exists a strictly increasing scalar function $\alpha : \mathbb{R}^+ \to \mathbb{R}$, $\alpha(0) = 0$, $\lim_{p\to\infty} \alpha(p) = \infty$, such that:

   $$V(0) = 0 \text{ and } V(x) \geq \alpha(||x||) \tag{6}$$

   Prove that for a matrix $P \succ 0$, the function:

   $$V(x) = x^T Px \tag{7}$$

   Is positive definite. *Hint: use your answer to part 1.*

3. Prove that along *any* trajectory of the system $\dot{x} = Ax$, the time derivative of the Lyapunov function $V(x) = x^T Px$ for positive definite $P \succ 0$ is equal to the following:

   $$\dot{V}(x) = x^T(A^T P + PA)x \tag{8}$$

   Hint: To take the derivative along a trajectory, substitute the constraint $\dot{x} = Ax$ for $\dot{x}$.

4. Now that we have an expression for the derivative of the Lyapunov function, we have to ensure that our P matrix is chosen such that $\dot{V}$ is always negative!
   Prove that if we can find a positive definite matrix P that solves the equation:

   $$A^T P + PA = -Q \tag{9}$$

   Where Q is *any* positive definite matrix, we can ensure the origin of the system is stable in the sense of Lyapunov. This equation is known as the Lyapunov equation. *Hints: How can we use $x^T Qx$ to our advantage?*

5. In the above, we only proved a weak form of Lyapunov stability. Does a stronger form of stability hold? An equilibrium point $x_e = 0$ of $\dot{x} = f(x,t)$ is *globally exponentially stable* if there exists a $V(x,t)$ such that for all $x \in \mathbb{R}^n$, there exist $\alpha_i > 0 \in \mathbb{R}$ such that:

   $$\alpha_1||x||^2 \leq V(x,t) \leq \alpha_2||x||^2 \tag{10}$$

   $$\dot{V} \leq -\alpha_3||x||^2 \tag{11}$$

   $$\left|\left|\frac{\partial V}{\partial x}(x,t)\right|\right| \leq \alpha_4||x|| \tag{12}$$

   Prove that if we can find a matrix $P \succ 0$ that satisfies the Lyapunov equation, $x_e = 0$ is a globally exponentially stable equilibrium point of $\dot{x} = Ax$. *Hint: the gradient of $x^T Px$ is calculated $\partial(x^T Px)/\partial x = (P^T + P)x$. How can you use an induced norm?*

Problem 2: The Indirect Method of Lyapunov

Finding Lyapunov functions for general nonlinear systems is a notoriously difficult process! If we *aren't* dealing with a physical system where we can use energy as a potential Lyapunov function, what can we do? Instead of directly finding a Lyapunov function and taking its time derivative, we can use the *indirect method of Lyapunov*, stated below:

Theorem 2 ***Indirect Method of Lyapunov (Stability by Linearization)***
If the Jacobian linearization around the origin of the system $\dot{x} = f(x,t)$, defined:

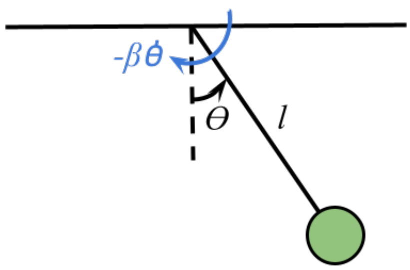
$$\dot{x} = A(t)x, \; A(t) = \frac{\partial f}{\partial x}\Big|_{x=0} \tag{13}$$

Exists, is well-behaved around the origin, and has bounded $A(t)$, $A(t)$ may be used to determine the local stability of the origin. If all eigenvalues of $A(t)$ have negative real components, $x_e = 0$ is a locally uniformly asymptotically stable equilibrium point of $\dot{x} = f(x,t)$.

Thus, if the Jacobian linearization of a system exists and is well behaved,[1] we can use the eigenvalues of the *linearized* system to make conclusions about the stability of the *nonlinear* system! The indirect method of Lyapunov thus allows us to forgo the need for a Lyapunov function and directly check for stability using eigenvalues.

Questions

1. A simple pendulum with mass m, length l, and angle θ has a frictional force $-\beta\dot{\theta}$, where $\beta > 0 \in \mathbb{R}$, applied to it at its point of rotation.



The dynamics of this pendulum may be written in state space as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin x_1 - \beta x_2 \end{bmatrix}, \; x = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} \tag{14}$$

The Jacobian linearization of the system about its equilibrium point $x = [0,0]^T$ is:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} \approx \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & -\beta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{15}$$

Show that the eigenvalues of the Jacobian linearization have $Re(\lambda_i) < 0$, and conclude that $x = [0,0]^T$ is a locally uniformly asymptotically stable equilibrium point using the indirect method of Lyapunov.

2. Why does the indirect method of Lyapunov only allow us to conclude local stability? Provide a brief worded response.

[1] Well-behaved refers to the *uniform convergence* of the Jacobian linearization to $f(x,t)$ as $x \to x_e$. We won't need to worry about this condition here!

Problem 3: Control Lyapunov Functions

In this question, we'll explore a fascinating application of Lyapunov functions in the development of optimal feedback controllers for nonlinear systems. Let's first review some basic concepts from optimization.

In an optimization problem, we typically seek to minimize a *cost function* subject to some *optimization constraints*. For example, consider the problem below:

$$c = \min_{x \in \mathbb{R}^n} f(x) \text{ (cost function)} \tag{16}$$

$$\text{s.t. } Ax \leq b \text{ (optimization constraint)} \tag{17}$$

This notation specifies that we want to find the minimum, c, of a function $f(x)$, where x is a vector in $\mathbb{R}^n$ subject to the constraint that $Ax \leq b$. In this problem, x, the variable that we optimize over, is called a *decision variable*. It is an unknown in the optimization problem.

$$x^* = \arg\min_{x \in \mathbb{R}^n} f(x) \text{ (cost function)} \tag{18}$$

$$\text{s.t. } Ax \leq b \text{ (optimization constraint)} \tag{19}$$

If we were to write "arg min" instead of simply "min," instead of solving for the smallest values of f, we would seek x^*, the value of our decision variable x that minimizes f. In other words, we search for the "argument" that minimizes f.

What types of optimization problems are there? Do certain types of problems have simpler solutions than others? Consider the following optimization problem:

$$x^* = \arg\min_{x \in \mathbb{R}^n} x^T Q x + c^T x \tag{20}$$

$$\text{s.t. } Ax \leq b \tag{21}$$

Where $Q \in \mathbb{R}^{n \times n}$ is a positive semidefinite matrix (symmetric, all eigenvalues ≥ 0) and $c \in \mathbb{R}^n$. This type of optimization problem, known as a *quadratic program* (QP), has a global minimum and may be solved efficiently by a computer! Quadratic programs are commonly used in optimal control, the study of applying optimization techniques to find the best control input to a system.

Imagine that we have a control-affine nonlinear system, of the form:

$$\dot{x} = f(x) + g(x)u, \ x \in \mathbb{R}^n, \ u \in \mathbb{R}^m \tag{22}$$

Suppose this system has a Lyapunov function $V(x)$ that satisfies some special conditions we'll discuss shortly. Consider the optimization problem:

$$u^* = \arg\min_{u \in \mathbb{R}^n} u^T Q u \tag{23}$$

$$\text{s.t. } L_f V + L_g V u \leq -\gamma(V(x)); \tag{24}$$

Where $\gamma : \mathbb{R}^+ \to \mathbb{R}$ is a strictly increasing function with $\gamma(0) = 0$, Q is a positive semidefinite matrix, and $L_f V$ and $L_g V$ are the Lie derivatives of the system, calculated:

$$L_f V = \frac{\partial V}{\partial x} f(x), \ L_g V = \frac{\partial V}{\partial x} g(x) \tag{25}$$

Given the right conditions, this optimization problem, which gives a controller known as a control Lyapunov function quadratic program (CLF-QP), allows for the asymptotic stabilization of a system! Let's learn about its properties![2]

[2] CLF-QPs may also be used for trajectory tracking, not only stabilization! Your Lyapunov function should simply be a function of tracking error instead of state in this case.

Questions

1. Let's start out by analyzing the setup of the optimization problem. Identify the decision variables, the cost function, and the constraints of the optimization problem. Then, show that the CLF-QP is a quadratic program by rewriting it in standard QP form:

$$x^* = \arg\min_{x \in \mathbb{R}^n} x^T Q x + c^T x \tag{26}$$

$$\text{s.t. } Ax \leq b \tag{27}$$

2. Let's take a moment to think about the cost function of the CLF-QP controller:

$$\text{cost} = u^T Q u \tag{28}$$

Why is this cost something we want to minimize? Provide a brief worded response with mathematical justification where necessary.

3. Show that the time derivative of the Lyapunov function $V(x)$ along the trajectories of the control-affine system $\dot{x} = f(x) + g(x)u$ may be expressed:

$$\dot{V}(x, u) = L_f V + L_g V u \tag{29}$$

4. Prove that if the optimization constraint:

$$L_f V + L_g V u \leq -\gamma(V(x)) \tag{30}$$

Is always satisfied, where $\gamma(y) : \mathbb{R}^+ \to \mathbb{R},\ y \in \mathbb{R}$ is an always-increasing scalar function such that $\gamma(0) = 0$,[3] the CLF-QP controller will make the equilibrium point of the system stable in the sense of Lyapunov.[4] *Hint: Consider the conditions required of a Lyapunov function for a stable equilibrium point.*

5. Thus far, we haven't discussed the *feasibility* of this optimization problem. Is this optimization problem something we can always solve given any control-affine system with a valid Lyapunov function? Not quite! One of the conditions we require is that $V(x)$ be a valid *control Lyapunov function.*
A control Lyapunov function is a positive definite function $V(x)$ such that:

$$\inf_{u \in U} \{L_f V + L_g V u\} \leq -\gamma(V(x)) \tag{31}$$

Where U is the set of possible inputs for the system and $\inf_{u \in U}$ is the largest lower bound of the set $\{L_f V + L_g V u\}$ over all possible values of u. Provide a brief worded response explaining why this is a necessary condition for the CLF-QP controller to stabilize a system. *Hint: Think about what happens if we can't satisfy this condition - what might go wrong with the convergence of our Lyapunov function?*

6. Let's think about the effect of $\gamma(y)$ on the convergence of the system to its equilibrium point. Consider a linear scalar function $\gamma(y)$ of the form:

$$\gamma(y) = \beta y,\ \beta \in \mathbb{R} \tag{32}$$

Assuming the optimization constraint is always satisfied and that $\dot{V}(x, u)$, is a smooth function of x and u, find a function that is an upper bound on the value of $V(t)$ given an initial condition $x(t_0) = x_0$. Comment on the effect of β on the tightness of the upper bound. *Hint: use the optimization constraint to form a bound. Can you solve the ODE?*

[3] γ is known as a class-$\mathcal{K}$ function.

[4] Note: The CLF-QP controller ensures a stronger form of Lyapunov stability than what we ask you to show here - for the sake of simplicity, we only ask you to prove a weaker form of Lyapunov stability.

Problem 4: MIMO Feedback Linearization & Dynamic Extension

Let's discuss how feedback linearization generalizes to the case of multi input multi output, or MIMO, systems. In particular, we'll focus on the a class of MIMO systems known as "square" systems, where the number of inputs is the same as the number of outputs. These systems are of the form:

$$\dot{x} = f(x) + g(x)u, \; x \in \mathbb{R}^n, u \in \mathbb{R}^p \tag{33}$$

$$y = h(x), \; y \in \mathbb{R}^p \tag{34}$$

Note that instead of being a single quantity, the output, y, is a vector! This output vector could be filled with variables of interest from the state vector, for example the (x, y) coordinates of the center of mass of a car!
As it happens, feedback linearization for many[5] MIMO systems is *quite similar* to that for SISO systems. Let's begin by taking the time derivative of the j^{th} output, y_j:

$$\dot{y}_j = L_f h_j(x) + \sum_{i=1}^{p} L_{g_i} h_j(x) u_i \tag{35}$$

Since we now have p inputs, we must sum over the lie derivatives that are associated with each input u_i to the system. As with before, we continue taking derivatives of y_j until at least one of the $L_{g_i} h_j(x) \neq 0$ - this allows for the presence of an input variable! The *smallest* derivative at which this happens is called γ_j, the relative degree of output y_j. We can calculate the γ_j derivative of y_j as follows:

$$y_j^{(\gamma_j)} = \frac{d^{\gamma_j} y_j}{dt^{\gamma_j}} = L_f^{\gamma_j} h_j(x) + \sum_{i=1}^{p} L_{g_i} L_f^{\gamma_j - 1} h_j(x) u_i \tag{36}$$

If the relative degree of each y_j is well defined, we can write the derivatives of each output as follows:

$$\begin{bmatrix} y_1^{(\gamma_1)} \\ \vdots \\ y_p^{(\gamma_p)} \end{bmatrix} = \begin{bmatrix} L_f^{\gamma_1} h_1(x) \\ \vdots \\ L_f^{\gamma_p} h_p(x) \end{bmatrix} + \begin{bmatrix} L_{g_1} L_f^{\gamma_1 - 1} h_1(x) & \dots & L_{g_p} L_f^{\gamma_1 - 1} h_1(x) \\ \vdots & \ddots & \vdots \\ L_{g_1} L_f^{\gamma_p - 1} h_p(x) & \dots & L_{g_p} L_f^{\gamma_1 - 1} h_p(x) \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} \tag{37}$$

Where the $p \times p$ matrix in the expression above is referred to as $A(x)$. If $A(x)$ is invertible, we find that under certain conditions, we can find an input u that input-output linearizes the system!
In this question, we'll discuss a technique called *dynamic extension*, which is used when A is non-invertible, and apply it to develop a controller for a turtlebot!

Questions

The dynamics of a turtlebot are described by the following equations:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} \cos x_3 & 0 \\ \sin x_3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \; y = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{38}$$

Where x_1 is the x coordinate of the turtlebot, y_1 is the y coordinate of the turtlebot, x_3 is the orientation ϕ of the turtlebot, and u_1 and u_2 are speed and turning rate respectively. This model is known as the *unicycle model.*

[5] Here, we focus on the class of MIMO systems linearizable by static state feedback! The conditions for linearizability are out of scope of this course, but are discussed in ME C237/EE C222, Nonlinear Systems.

1. Recall that γ_j is the first time derivative of y where at least one input variable appears. For the turtlebot system described above, show that $\gamma_1 = \gamma_2 = 1$, and that the derivatives of y are calculated:

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \end{bmatrix} = \begin{bmatrix} \cos x_3 & 0 \\ \sin x_3 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = A(x)u \tag{39}$$

2. To find a feedback linearizing input, we'd like to take the inverse of $A(x)$, but $A(x)$ is not invertible! To get around this, we'll use a technique called dynamic extension. First, instead of directly controlling the input u_1, we'll control its derivative, $w_1 = \dot{u}_1$. Thus, we form a new input vector, $w = [w_1, w_2]^T = [\dot{u}_1, u_2]^T$.
Now that we're using $w_1 = \dot{u}_1$, we need some way to keep track of the value of u_1 itself! We can do this by adding u_1 as a *virtual state* to the system, and form a new state vector:

$$\tilde{x} = \begin{bmatrix} x_1 & x_2 & x_3 & u_1 \end{bmatrix}^T \tag{40}$$

Using the extended state vector $\tilde{x}$ and input vector w, show that the dynamics of the system may be rewritten:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{u}_1 \end{bmatrix} = \begin{bmatrix} u_1 \cos x_3 \\ u_1 \sin x_3 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = f(\tilde{x}) + g(\tilde{x})w \tag{41}$$

3. Now that we have an extended system, let's see if we can come up with a feedback linearizing control law! If the outputs are $x_1 = y_1, x_2 = y_2$, show that for the extended system, $\gamma_1 = \gamma_2 = 2$. Then, show that the relationship:

$$\begin{bmatrix} \ddot{y}_1 \\ \ddot{y}_2 \end{bmatrix} = \begin{bmatrix} \cos x_3 & -u_1 \sin x_3 \\ \sin x_3 & u_1 \cos x_3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = A'(\tilde{x})w \tag{42}$$

Holds between the input w and output y. For what values of u_1 will $A'(\tilde{x})$ not be invertible?

4. Assuming $A'(\tilde{x})$ is invertible, find an expression for w such that when w is applied to the nonlinear system above, the following linear system governs the dynamics:

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \ddot{y}_1 \\ \ddot{y}_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \dot{y}_1 \\ \dot{y}_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} v \tag{43}$$

Where $v \in \mathbb{R}^2$ is an arbitrary vector. We've now successfully linearized the dynamics of the turtlebot! *Hint: start by finding a control law in terms of the second derivatives of y.*

5. Let $z = [y_1, y_2, \dot{y}_1, \dot{y}_2]^T$ be the state vector of the linearized system above. Show that $k_{ij} \in \mathbb{R}$ may be chosen such that the control law:

$$v = -Kz, \ K = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \end{bmatrix} \in \mathbb{R}^{2\times 4} \tag{44}$$

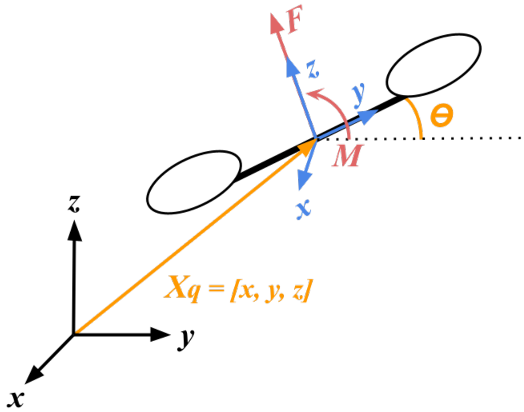
Stabilizes the linearized system above around the origin, $z = 0$. This choice of input is known as *state feedback*. *Hint: Substitute in v and reduce the system to the form $\dot{z} = Pz, \ P \in \mathbb{R}^{n\times n}$. Can we choose K to control the eigenvalues of P?*

Problem 5: Implementing Control Lyapunov Functions

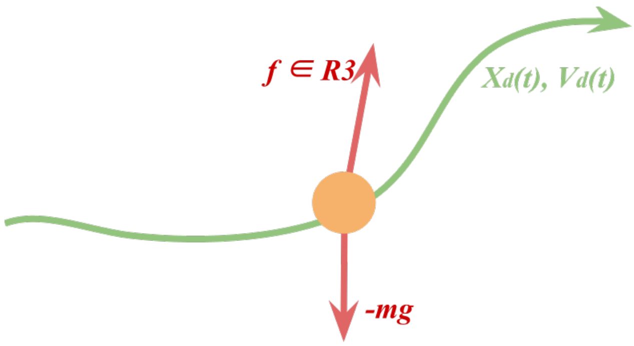
Note: This question requires the Python optimization library CasADi. If you're unable to install CasADi, please use the provided Google Colab notebook.

In this question, we'll implement a type of control Lyapunov function known as an *exponentially stabilizing control Lyapunov function* (ES-CLF). Note that you should complete problem 3 on the theory of control Lyapunov functions before attempting this question - problem 3 provides much of the theory we'll need here!

In this question, we'll extend the theory of control Lyapunov functions from a controller that stabilizes a system about its equilibrium point to a controller that helps a system *track a trajectory*. The system we'll use is the planar quadrotor, which is constrained to travel in the $y - z$ plane.



To design a controller for this system, we may treat the rotational and translational dynamics separately. The first step in the design of a tracking controller for a planar quadrotor is to *abstract away* the rotational dynamics & control of the system, and to treat the quadrotor as a point mass with a force vector input that travels in the $y - z$ plane.



For now, we'll ignore the effects of gravity on the point mass to simplify our Lyapunov function. With this in mind, the simple point mass system has dynamics described by the equation:

$$m \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \tag{45}$$

$$m\ddot{q} = f \tag{46}$$

Where m is the mass of the quadrotor. Note that since the system is planar, we set $f_1 = 0$.

To track a desired trajectory $(q_d(t), \dot{q}_d(t))$, we can use a Lyapunov function defined around the *error* of the system! Following Wu and Sreenath's approach (2016)[6], we define:

$$V = \frac{1}{2}m||\dot{q} - \dot{q}_d||^2 + \frac{\alpha}{2}||q - q_d||^2 + \varepsilon(q - q_d)^T(\dot{q} - \dot{q}_d) \tag{47}$$

Where $\alpha > 0$ and $\varepsilon > 0$ are constants set to make V quadratic in tracking error. Instead of driving the state of the system to an equilibrium point, like an ordinary Lyapunov function might, this Lyapunov function will drive the *error* of the system to zero!

In this question, you'll implement a CLF-QP controller in Python to enable a quadrotor to stably track trajectories. We're going to use a special type of CLF known as an exponentially stabilizing control Lyapunov function (ES-CLF). The ES-CLF optimization problem is formulated:

$$u^* = \arg\min_{u \in \mathbb{R}^n} u^T Q u \tag{48}$$

$$\text{s.t. } \dot{V} \leq -\gamma V \tag{49}$$

Where $Q \succeq 0$ is a positive semidefinite matrix. An exponentially stabilizing CLF-QP controller is one that uses a *linear* function $g(x) = \gamma x$, where $\gamma \in \mathbb{R}$ is a positive constant, for its strictly increasing function in the optimization constraint. Note that in the above, we simply opted to write the time derivative of V without using Lie derivative notation. Let's get started on writing our controller!

Questions

1. As can be observed in the optimization problem above, we must solve for the time derivative of the Lyapunov function. If we define position error $e_x = q - q_d$ and velocity error $e_v = \dot{q} - \dot{q}_d$, where $q_d, \dot{q}_d$ are desired position and velocity, the Lyapunov function is:

   $$V([q, \dot{q}]) = \frac{1}{2}m e_v^T e_v + \frac{\alpha}{2} e_x^T e_x + \varepsilon e_x^T e_v \tag{50}$$

   Recalling that the dynamics of the simple particle system are:

   $$m\ddot{q} = f \tag{51}$$

   Show that the time derivative, $\dot{V}(x)$ along the trajectories of the simple particle system is expressed:

   $$\dot{V} = (\frac{1}{m}f - \ddot{q}_d)^T(m e_v + \varepsilon e_x) + \alpha e_v^T e_x + \varepsilon e_v^T e_v \tag{52}$$

2. Provide a brief worded explanation as to why the final force vector input to the quadrotor should be:

   $$f_{in} = u^* + mge_3 \tag{53}$$

   Where m is the mass of the quadrotor, g is the gravitational constant, $e_3 = [0, 0, 1]^T$, and u^* is the solution to the CLF-QP optimization problem for the system above.

3. Go to line 101 in the file **lyapunov.py**, and fill in the functions *evalLyapunov()* and *evalLyapunovDerivs()*.

4. Go to line 30 in the file **run_simulation.py** and select the value of *gamma* so your quadrotor smoothly tracks the desired trajectory - this will take a little bit of tuning!

5. Test your controller by running the file **run_simulation.py**. The optimization code has been implemented for you using a library called CasADi, which we will introduce in homework 3. Attach the plots titled "Evolution of States in Time" and "Evolution of Inputs in Time" to your solution. *Note: The drone has sensors with zero mean Gaussian noise, so you should see some noisy inputs!*

[6] *Safety-Critical Control of a 3D Quadrotor with Range-Limited Sensing, DSCC 2016*