MEC ENG 132: Dynamical Systems and Feedback Controls

Nonlinear Systems and Linearization

Summer 2025: Professor George Anwar
UC Berkeley, Department of Mechanical Engineering

Larry Hui[1]

Summary and Notes

This set of notes will cover everything you need to know about nonlinear systems and how to linearize them! This will be on the final exam and should be fair game as it is also covered in the textbook. This is not meant to be completely exhaustive but will cover the basics; there is a lot of literature online if you would like to learn more. We will look at IO differential equation models; state-space realizations; A magnetically suspended ball case study; Equilibrium points; Jacobians and the Taylor Theorem; Linearization about equilibrium points; Control using Jacobian Linearized Plant Models; Gain Scheduling; Feedback Linearization.

[1]University of California at Berkeley, College of Engineering, Department of Mechanical Engineering. Author to whom any correspondence should be addressed. email: larryhui7@berkeley.edu

Contents

1. Input-Output (IO) Differential Equation Models

So far, we have restricted ourselves to focus on linear and time-invariant (LTI) systems. But many models of physical processes that we are interest in controlling, if not most, are not linear! Fortunately, the study of LTI systems is a vital tool in learning how to control even nonlinear systems. Essentially, feedback control algorithms make small adjustments to the inputs based on measured outputs. For small deviations of the input about some nominal input trajectory, the output of a nonlinear system looks like a small deviation around some nominal output. The effects of the small input deviations on the output is well approximated by a linear (possibly time-varying) system.

Many nonlinear systems are modeled by nonlinear differential equations that relate the inputs u to the outputs y like:

$$\ddot{y} + 3\dot{y}y + y\cos(y) = 4u^2 + uy$$

When we have many inputs and many outputs, we will still have differential equations but there will be more of them. We will have one differential equation for each output. We cannot use transfer function methods for nonlinear systems. Transfer functions make sense only of linear time-invariant differential equations.

1.1. Linear Systems

Recall that a linear system has two properties.

1. Superposition (Additivity): the output response of a system to a sum of inputs is the sum of the responses to the individual inputs.
2. Homogeneity (Scaling): multiplying an input by a scalar will yield a response that is multiplied by the same scalar.
3. The condition to check both superposition and homogeneity is the following. Consider a system $\mathbf{G}$, it is linear if and only if
$$\mathbf{G}\{\alpha x_1[n] + \beta x_2[n]\} = \alpha\mathbf{G}\{x_1[n]\} + \beta\mathbf{G}\{x_2[n]\}$$

2. State-Space Realizations

We can rewrite an n^{th} order nonlinear differential equation model as a set of n coupled first-order differential equations. For example, consider the differential equation model above:

$$\ddot{y} + 3\dot{y}y + y\cos(y) = 4u^2 + uy$$

Define the *states* by

$$x_1 = y, x_2 = \dot{y}$$

We can re-write this differential equation model as

$$\begin{aligned}\dot{x}_1 &= \dot{y} = x_2\\ \dot{x}_2 &= \ddot{y} = -3\dot{y}y - 7\cos(y) + 4u^2 + uy = -3x_1x_2 - 7\cos(x_1) + 4u^2 + ux_1\\ y &= x_1\end{aligned}$$

We *cannot* write these equations in matrix form as we did for LTI differential equations. But not *form* of these equations:

$$\begin{aligned}\dot{x}_1 &= f_1(x_1, x_2, u) = x_2\\ \dot{x}_2 &= f_2(x_1, x_2, u) = -3x_1x_2 - 7\cos(x_1) + 4u^2 + ux_1\\ y &= h(x_1, x_2, u) = x_1\end{aligned}$$

More generally, for a m input, p output nonlinear differential equation model, we would get equations like

$$\begin{aligned}
\dot{x}_1 &= f_1(x_1, \cdots, x_n, u_1, \cdots, u_m) \\
\vdots \quad & \quad \vdots \\
\dot{x}_n &= f_n(x_1, \cdots, x_n, u_1, \cdots, u_m) \\
y_1 &= h_1(x_1, \cdots, x_n, u_1, \cdots, u_m) \\
\vdots \quad & \quad \vdots \\
y_p &= h_p(x_1, \cdots, x_n, u_1, \cdots, u_m)
\end{aligned}$$

Here each of the possible nonlinear functions $f_1, \cdots, f_n$ and $h_1, \cdots, h_p$ are scalar-valued. These equations are nasty to write down, so I'm just going to introduce the following notation:

$$x = (x_1, \cdots, x_n), \quad u = (u_1, \cdots, u_m), \quad y = (y_1, \cdots, y_p)$$

we can write these equations more compactly in the standard state-space form as follows

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

Finite-dimensional, time-invariant, causal nonlinear control systems, with input u and output y can typically be modeled using the set of differential equations above. Note that in this form, the nonlinear functions f and h are vector-valued. This means that there are actually many scalar functions contained in the symbols f and h. If either f AND/OR h are non-linear, then the dynamic system is called non-linear.

3. Example: Magnetically Suspended Ball

We will look at a common example in control systems—the magnetically suspended ball!This consists of a ball of mass m suspended by an electromagnet as shown below. Let y be the position of the ball, measured down from the base of the electromagnet. If a current u is injected into the coils of the electromagnet, it will produce a magnetic field which, in turn, exerts an upward force on the ball given by $F_{up} = -cu^2/y^2$. Note that the force decreases as the $1/y^2$ because the effect of the magnet weakens when the ball is further away, and that the force is proportional to u^2 which is representative of the power supplied to the magnet.

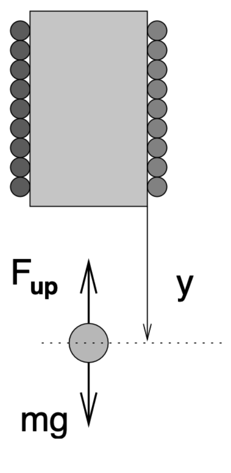

Figure 1: Magnetically Suspended Ball

Let us assume that we can measure the position y of the ball. This can be arranged optically using light-emitting diodes (LEDs) with photocells. We can then write a simple model for the motion of the ball from Newton's second law as

$$m\ddot{y} = mg - \frac{cu^2}{y^2}$$

Note that this is a SISO non-linear model. The input of the system is u, the current injected into the magnetic coils. The output of the system is the position of the ball. In the context of the numerical exercises proceeding this, we use the following constants: $m = 0.15$ kg, the mass of the ball, and $c = 0.01$ Kg-m^3/C^2, the magnetic coupling constant. We can convert this non-linear IO differential equation model to state-space by defining the states:

$$x_1 = y \quad \text{and} \quad x_2 = \dot{y}$$

We get the following set of coupled first-order differential equations:

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2, u) = x_2 \\ \dot{x}_2 &= f_2(x_1, x_2, u) = g - \frac{cu^2}{mx_1^2} = 10 - \left(\frac{u}{3.87x_1}\right)^2 \\ y &= h(x_1, x_2, u) = x_1 \end{aligned}$$

There are a number of sources of modeling error in our simple model above. The most significant of these is the leakage of the magnetic field. This is appreciable as y increases, and this manifests itself in a decrease in the exponent of u in F_{up}. The entire model breaks down if $y > 0.25$ m.

A second serious modeling omission is that we have treated the motion of the ball as rectilinear. In practice, unless the ball is confined to a vertical guide, it will have small motions in the horizontal plane also. These have significant effects on the dynamics of the ball, as the magnetic force reduces significantly off the magnet axis. We have also neglected resistive and inductive effects in the electromagnet coils, and approximated the ball as a point mass.

4. Equilibrium Points

Consider the general time-invariant nonlinear realization

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

An *equilibrium point* of the above realization is a pair (x^{eq}, u^{eq}) such that if we start with initial condition $x(0) = x^{eq}$ and apply the constant input $u(t) = u^{eq}$, $t \geq 0$, then the state trajectory remains at the equilibrium point, i.e.

$$x(0) = x^{eq}, \quad u(t) = u^{eq} \implies x(t) = x^{eq}$$

Notice that the output function $h(x, u)$ actually plays no role in finding equilibrium points. Also, we should stress that x^{eq} and u^{eq} are constants and NOT functions of time. The following results tells us how to find equilibrium points.

Theorem 1: Equilibrium Points
We call (x^{eq}, u^{eq}) an equilibrium point of the general time-invariant nonlinear system realization

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

if and only if $f(x^{eq}, u^{eq}) = 0$.

Proof. If we initialize the differential equation $\dot{x} = f(x, u)$ at $x(0) = x^{eq}$ and apply the constant inputs $u(t) = u^{eq}$, we see that

$$\dot{x} = f(x^{eq}, u^{eq}) = 0$$

This means that $x(t)$ does not change with time. So it stays at $x(t) = x^{eq}$. $\square$

For a nonlinear system, there can be many equilibrium points. To find all the equilibrium points of a nonlinear system, we simply set

$$f(x^{eq}, u^{eq}) = 0$$

and solve for x^{eq} and u^{eq}. If we start the system at x^{eq} and apply the constant input u^{eq}, then $x(t)$ stays at the equilibrium value x^{eq}. Corresponding to this equilibrium point, $y(t)$ stays at its equilibrium value

$$y^{eq} = h(x^{eq}, u^{eq})$$

Example 2: Magnetically Suspended Ball
Returning to this example, we calculate the equilibrium points as follows:

$$\begin{aligned} f(x^{eq}, u^{eq}) = 0 &\implies f_1(x_1^{eq}, x_2^{eq}, u^{eq}) = 0 \quad \text{and} \quad f_2(x_1^{eq}, x_2^{eq}, u^{eq}) = 0 \\ &\implies x_2^{eq} = 0 \quad \text{and} \quad 10 - \left(\frac{u^{eq}}{3.87 x_1^{eq}}\right) = 0 \\ &\implies x_2^{eq} = 0 \quad \text{and} \quad x_1^{eq} = 0.082 u^{eq} \end{aligned}$$

The equilibrium points are therefore:

$$x_1^{eq} = 0.082\zeta \quad \text{and} \quad x_2^{eq} = 0, \quad \text{and} \quad u^{eq} = \zeta$$

The resulting value of the output is

$$y^{\text{eq}} = h\big(x^{\text{eq}}, u^{\text{eq}}\big) = x_1^{\text{eq}} = 0.082\,\zeta.$$

Note that here we have many equilibrium points. In this case, we have parameterized these equilibrium points by the free variable $u^{\text{eq}} = \zeta$. We could also parameterize these equilibrium points by the free variable $y^{\text{eq}} = \xi$ as follows:

$$\begin{aligned} x_1^{\text{eq}} &= \xi, \\ x_2^{\text{eq}} &= 0, \\ u^{\text{eq}} &= 12.195\,\xi, \\ y^{\text{eq}} &= 0.082\,\zeta = \xi. \end{aligned}$$

This parametrization is sometimes more convenient when we study gain-scheduling to control nonlinear systems.

5. Jacobians and the Taylor Theorem

First consider a scalar-valued function $\phi(x_1, x_2, \ldots, x_n)$ of n variables. The Jacobian of ϕ is the row vector

$$\nabla\phi(x_1, \ldots, x_n) = \begin{bmatrix} \frac{\partial\phi}{\partial x_1} & \cdots & \frac{\partial\phi}{\partial x_n} \end{bmatrix}.$$

For example, for the function

$$\phi(x_1, x_2, x_3) = x_1 x_2^2 - \frac{1}{x_1} + \sin(x_3),$$

the Jacobian is

$$\nabla\phi = \begin{bmatrix} \frac{\partial\phi}{\partial x_1} & \frac{\partial\phi}{\partial x_2} & \frac{\partial\phi}{\partial x_3} \end{bmatrix} = \begin{bmatrix} x_2^2 + \frac{1}{x_1^2} & 2x_1x_2 & \cos(x_3) \end{bmatrix}.$$

This is a function of (x_1, x_2, x_3). We can evaluate it at a point $(1, 2, 0)$ to get

$$\nabla\phi(1, 2, 0) = \begin{bmatrix} 5 & 4 & 1 \end{bmatrix}.$$

Now consider a vector-valued function $\phi : \mathbb{R}^n \to \mathbb{R}^m$. This means we have m scalar functions of n variables, which we collect into

$$\phi(x) = \phi(x_1, \ldots, x_n) = \begin{bmatrix} \phi_1(x_1, \ldots, x_n) \\ \vdots \\ \phi_m(x_1, \ldots, x_n) \end{bmatrix}.$$

The Jacobian of ϕ is the $m \times n$ matrix

$$\nabla\phi = \begin{bmatrix} \frac{\partial\phi_1}{\partial x_1} & \cdots & \frac{\partial\phi_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial\phi_m}{\partial x_1} & \cdots & \frac{\partial\phi_m}{\partial x_n} \end{bmatrix}.$$

Fix a vector $\xi \in \mathbb{R}^n$. We can evaluate the Jacobian at ξ to get an $m \times n$ matrix of numbers. Our notation for this is

$$\nabla\phi(\xi) = \begin{bmatrix} \frac{\partial \phi_1}{\partial x_1} & \cdots & \frac{\partial \phi_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \phi_m}{\partial x_1} & \cdots & \frac{\partial \phi_m}{\partial x_n} \end{bmatrix}_\xi .$$

For example, consider the function

$$\phi(x_1, x_2, x_3) = \begin{bmatrix} \phi_1(x_1, x_2, x_3) \\ \phi_2(x_1, x_2, x_3) \end{bmatrix} = \begin{bmatrix} x_1 \sin(x_2) \\ x_2^2 \cos(x_3) \end{bmatrix}.$$

The Jacobian of $\phi(x)$ evaluated at $(1, \frac{\pi}{2}, 0)$ is

$$\nabla\phi(x_1, x_2, x_3) = \begin{bmatrix} \sin(x_2) & x_1\cos(x_2) & 0 \\ 0 & 2x_2\cos(x_3) & x_2^2 \sin(x_3) \end{bmatrix}_{(1, \frac{\pi}{2}, 0)} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & \pi & 0 \end{bmatrix}.$$

Analogous to the classical Taylor series expansion of a scalar function of one variable, we can write the Taylor series expansion of ϕ around ξ as

$$\phi(x) = \phi(\xi) + \nabla\phi(\xi)\,(x - \xi) + \text{higher order terms.}$$

The first two terms above are the linear approximation of ϕ around ξ. We can therefore approximate ϕ as

$$\boxed{\phi(x) \approx \phi(\xi) + \nabla\phi(\xi)\,(x - \xi)}$$

and this approximation will be good in some (possibly small) neighborhood of ξ.

6. Linearization about Equilibirum Points

We will now use the Taylor Theorem to approximate a nonlinear plant model by a LTI plant model. This approximation is called the Jacobian Linearization and is good as long as the applied input $u(t)$ is close to its equilibrium value u_{eq} and the resulting state trajectory $x(t)$ is close to the equilibrium value x_{eq}.

Theorem 2: Jacobian Linearization
Consider a non-linear time-invariant input-output differential equation model realized as

$$\dot{x} = f(x, u) \quad \text{and} \quad y = h(x, u)$$

Let $\xi = (x^{eq}, y^{eq})$ be an equilibrium point of this model, and define $y^{eq} = h(x^{eq}, u^{eq})$. We then define *deviation variables*

$$\delta x = x - x_{eq}$$
$$\delta u = u - u_{eq}$$
$$\delta y = y - y_{eq}$$

and compute the matrices

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}_{x=\xi}, \quad B = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_m} \end{bmatrix}_{x=\xi},$$

$$C = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_p}{\partial x_1} & \cdots & \frac{\partial h_p}{\partial x_n} \end{bmatrix}_{x=\xi}, \quad D = \begin{bmatrix} \frac{\partial h_1}{\partial u_1} & \cdots & \frac{\partial h_1}{\partial u_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_p}{\partial u_1} & \cdots & \frac{\partial h_p}{\partial u_m} \end{bmatrix}_{x=\xi}.$$

Then, in the vicinity of the equilibrium point, the nonlinear system can be approximated by the linear time-invariant state-space realization

$$\begin{aligned}\delta\dot{x}(t) &= A\,\delta x(t) + B\,\delta u(t),\\ \delta y(t) &= C\,\delta x(t) + D\,\delta u(t).\end{aligned}$$

Proof. The basic tool we use to prove this result is the Taylor theorem. We can approximate f in a neighborhood of ξ as

$$f(x,u) \;\approx\; f\big(x^{\mathrm{eq}},u^{\mathrm{eq}}\big) \;+\; \nabla f(\xi)\left(\begin{bmatrix}x\\u\end{bmatrix}-\begin{bmatrix}x^{\mathrm{eq}}\\u^{\mathrm{eq}}\end{bmatrix}\right).$$

Writing out the two blocks of the Jacobian gives

$$f(x,u) \;\approx\; \underbrace{\begin{bmatrix}\frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n}\\ \vdots & \ddots & \vdots\\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n}\end{bmatrix}_{(x^{\mathrm{eq}},u^{\mathrm{eq}})}}_{A}\big(x-x^{\mathrm{eq}}\big) \;+\; \underbrace{\begin{bmatrix}\frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_m}\\ \vdots & \ddots & \vdots\\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_m}\end{bmatrix}_{(x^{\mathrm{eq}},u^{\mathrm{eq}})}}_{B}\big(u-u^{\mathrm{eq}}\big)$$

$$=\; A\,(x-x^{\mathrm{eq}}) \;+\; B\,(u-u^{\mathrm{eq}})$$

In a similar fashion we approximate h in a neighborhood of ξ as

$$h(x,u) \;\approx\; h\big(x^{\mathrm{eq}},u^{\mathrm{eq}}\big) \;+\; \nabla h(\xi)\left(\begin{bmatrix}x\\u\end{bmatrix}-\begin{bmatrix}x^{\mathrm{eq}}\\u^{\mathrm{eq}}\end{bmatrix}\right),$$

which expands to

$$h(x,u) \;\approx\; \underbrace{\begin{bmatrix}\frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_n}\\ \vdots & \ddots & \vdots\\ \frac{\partial h_p}{\partial x_1} & \cdots & \frac{\partial h_p}{\partial x_n}\end{bmatrix}_{(x^{\mathrm{eq}},u^{\mathrm{eq}})}}_{C}\big(x-x^{\mathrm{eq}}\big) \;+\; \underbrace{\begin{bmatrix}\frac{\partial h_1}{\partial u_1} & \cdots & \frac{\partial h_1}{\partial u_m}\\ \vdots & \ddots & \vdots\\ \frac{\partial h_p}{\partial u_1} & \cdots & \frac{\partial h_p}{\partial u_m}\end{bmatrix}_{(x^{\mathrm{eq}},u^{\mathrm{eq}})}}_{D}\big(u-u^{\mathrm{eq}}\big) \;+\; y^{\mathrm{eq}},$$

or equivalently

$$h(x,u) \approx y^{\mathrm{eq}} + C\,(x-x^{\mathrm{eq}}) + D\,(u-u^{\mathrm{eq}}).$$

Now define the deviation variables

$$\delta x = x - x^{\mathrm{eq}}, \quad \delta u = u - u^{\mathrm{eq}}, \quad \delta y = y - y^{\mathrm{eq}}.$$

Observe that

$$\delta\dot{x} = \dot{x} = f(x,u) \;\approx\; A\,\delta x + B\,\delta u, \qquad \delta y = h(x,u) - y^{\mathrm{eq}} \;\approx\; C\,\delta x + D\,\delta u,$$

Thus, the original non-linear system can be approximated by a LTI system. This approximation is good provided we remain sufficiently close to the equilibrium point $\xi = (x^{eq}, u^{eq})$. □

The original non-linear plant and its Jacobian linearization are illustrated below

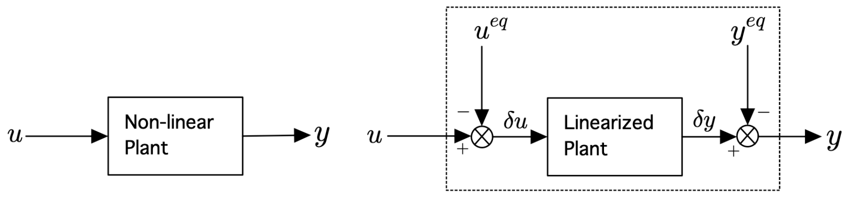

Figure 2: Nonlinear Plant and its Linearized Approximation

Example 3: Magnetically Suspended Ball
Consider again the magnetically suspended ball. We linearize the dynamics about the equilibrium point

$$x^{\text{eq}} = \begin{bmatrix} \xi \\ 0 \end{bmatrix}, \qquad u^{\text{eq}} = 12.195\,\xi, \qquad y^{\text{eq}} = \xi.$$

Define the deviation variables

$$\tilde{x} = x - x^{\text{eq}}, \quad \tilde{u} = u - u^{\text{eq}}, \quad \tilde{y} = y - y^{\text{eq}}.$$

We calculate the linearization matrices at the equilibrium:

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}_{(x^{\text{eq}},u^{\text{eq}})} = \begin{bmatrix} 0 & 1 \\ \frac{76.86}{\xi} & 0 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \frac{\partial f_2}{\partial u} \end{bmatrix}_{(x^{\text{eq}},u^{\text{eq}})} = \begin{bmatrix} 0 \\ \frac{6.31}{\xi} \end{bmatrix},$$

$$C = \begin{bmatrix} \frac{\partial h}{\partial x_1} & \frac{\partial h}{\partial x_2} \end{bmatrix}_{x^{\text{eq}}} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} \frac{\partial h}{\partial u} \end{bmatrix}_{(x^{\text{eq}},u^{\text{eq}})} = 0.$$

Then the nonlinear system can be approximated by the LTI state-space realization

$$\begin{aligned} \delta\dot{x}(t) &= A\,\delta x(t) + B\,\delta u(t), \\ \delta y(t) &= C\,\delta x(t) + D\,\delta u(t). \end{aligned}$$

Notice that the eigenvalues of A are

$$\lambda = \pm\frac{8.77}{\sqrt{\xi}}.$$

Since one eigenvalue lies in the right-half plane, the linearized approximation is unstable. Consequently, the nonlinear system is locally unstable: small deviations from the equilibrium will cause the ball to move away from its equilibrium position.

7. Control using Jacobian Linearized Plat Models

Let us begin with a nonlinear plant model P. We could choose an equilibrium point of this plant model, and find the Jacobian linearization $H(s)$ of the plant about this equilibrium point. Now we have a LTI "plant" model, and we can use the state-space methods or PID methods to design a controller $K(s)$ for the linearized plant model $H(s)$. This feedback system is shown in Figure 3. The command for this feedback system is $\delta r(t)$. This signal is what we want the linearized plant output $\delta y(t)$ to track, and this is what the controller $K(s)$ is designed to do.

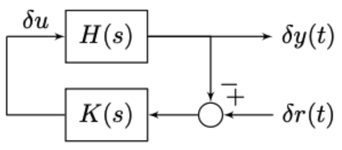

Figure 3: Control System for Linear Plant Model $H(s)$

Let us figure out what $\delta r(t)$ should be. We want the output of the nonlinear plant $y(t)$ to track the command $r(t)$. This signal $r(t)$ will be generated externally (by a human operator, a computer, or another control system). Since $\tilde{y}(t) = y(t) - y^{\text{eq}}$, this implies that we want $\tilde{y}(t)$ to track $r(t) - y^{\text{eq}}$. Therefore, we should choose

$$\tilde{r}(t) \;=\; r(t) - y^{\text{eq}}.$$

The control system for the linearized plant model is shown in the figure below.

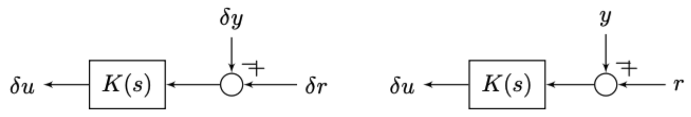

Figure 4: Control System for Linear Plant Model

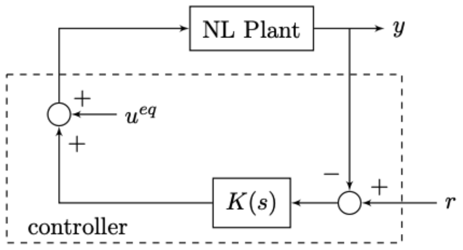

Figure 5: Final Implementation of Controller based on Jacobian Linearization

But the linearized model $H(s)$ is not a physical plant; it is just a construct of our imagination. So we now need to figure out how we actually *implement* the controller $K(s)$ for the nonlinear physical plant P. If we examine the controller $K(s)$ in isolation, its input is $r(t) - y^{\text{eq}}$. The output of the controller is $\tilde{u}(t)$. Notice that $\delta u(t) = u(t) - u^{\text{eq}}$, so we can generate the input to the nonlinear physical plant easily as

$$u(t) = \delta u(t) + u^{\text{eq}}.$$

The final implementation of the control system for the nonlinear plant is shown in Figure 5.

This controller is based on a linearized model of the nonlinear plant about the equilibrium point $\left(u^{\text{eq}}, x^{\text{eq}}\right)$. The linearized model is a good approximation of the plant as long as we stay close to the equilibrium point. This means that the controller will work well as long as the input applied to the plant is close to u^{eq}, the initial condition of the plant is close to x^{eq}, and the subsequent state trajectory $x(t)$ of the plant remains close to x^{eq}.

8. Gain Scheduling

In many situations, we cannot be assured that we will stay close to a particular equilibrium point of a nonlinear plant during control maneuvers. To effectively control the nonlinear plant we will need to design controllers at many equilibrium points (also called *operating points*) and switch between these controller in an intelligent way. This control strategy is called *gain scheduling*.

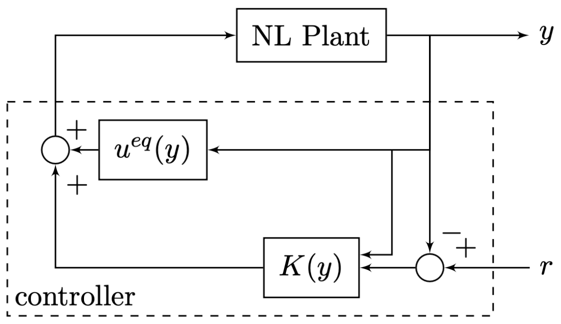

Figure 6: Gain Scheduling Schematic

The strategy for Gain Scheduling is illustrated in Figure 6. The idea is to first parameterize equilibrium points of the nonlinear system to be controlled using the free variable $y^{\text{eq}} = \xi$. So the equilibrium points look like:

$$x^{\text{eq}} = a(\xi), \quad u^{\text{eq}} = b(\xi), \quad y^{\text{eq}} = \xi,$$

where a and b are functions of the variable ξ. We will refer to the equilibrium point above simply as ξ. Next, we can linearize the nonlinear plant about the equilibrium point ξ. We will get a linearized model

$$\begin{aligned}\delta\dot{x}(t) &= A(\xi)\,\delta x(t) + B(\xi)\,\delta u(t),\\ \delta y(t) &= C(\xi)\,\delta x(t) + D(\xi)\,\delta u(t).\end{aligned}$$

The matrices A, B, C, D clearly depend on the equilibrium point and so they are functions of ξ. Now we can design controllers based on these linearized models. This is just what we did in the previous section, only this time we have a continuous family of equilibrium points parameterized by ξ. So the controller designed at the equilibrium point ξ might have a state-space realization

$$K(s) \;\sim\; \left[\begin{array}{c|c} F^{\xi} & G^{\xi} \\ \hline H^{\xi} & 0 \end{array}\right].$$

Clearly the controller state-space matrices F, G, H depend on ξ as indicated by the superscripts. Remember that the final implementation of the controller (see previous section) is

$$u(t) = \delta u(t) + u^{\text{eq}}(\xi), \quad \text{where} \quad \delta u(t) = \big[K^{\xi}(s)\big]\big(r(t) - y(t)\big).$$

This implementation of the controller $K(s)$ for the nonlinear plant is shown in Figure 5. Now let us write this more carefully. Let $z(t)$ denote the state of the controller. We can rewrite the equation above as

$$\dot{z}(t) = F^{\xi}\,z(t) \;+\; G^{\xi}\,\big(r(t) - y(t)\big), \qquad u(t) = H^{\xi}\,z(t) \;+\; u^{\text{eq}}(\xi).$$

Again, we stress that the matrices F, G, and H depend on the scheduling variable ξ. A natural choice for ξ is to set

$$\xi \;=\; y(t),$$

the current output of the nonlinear plant, since this is the closest equilibrium point to the plant's present state. Substituting $\xi = y(t)$ into the above yields our final controller implementation:

$$\dot{z}(t) = F^{y}\,z(t) \;+\; G^{y}\,\big(r(t) - y(t)\big), \quad u(t) = H^{y}\,z(t) \;+\; u^{\text{eq}}\big(y(t)\big).$$

This discussion is summarized in Figure 6, which shows the final form of the gain-scheduled controller.

A very important point is that it is often necessary to "slow down" or low-pass filter the command signal for gain scheduling to work properly. For example, in the magnetically suspended ball system, if we command the ball position to jump abruptly from 0.25 m to 0.5 m, the required speed would take the system far from any equilibrium. Since at each equilibrium the ball speed is zero, such a rapid command cannot be well approximated by a sequence of nearby linear models. By slowing down the command, we allow the ball to move gradually between equilibria, keeping the system within the validity region of the local linearizations.

9. Feedback Linearization

A powerful technique for treating nonlinear systems in the context of control problems is *feedback linearization* or *dynamic inversion*. The essential idea here is to "cancel" the nonlinearities. You have seen an example of this already on your midterm, question 6 of the multiple choice!

Consider a physical system modeled by the nonlinear realization

$$\dot{x} = f(x, u), \qquad y = h(x, u).$$

In many cases it is possible to construct a feedback function

$$u = \phi(v, y)$$

such that the dynamics from the new inputs v to the outputs y are linear and time-invariant. We can then design a controller for the linearized system using standard techniques that we have studied earlier. These ideas are illustrated in Figure 7.

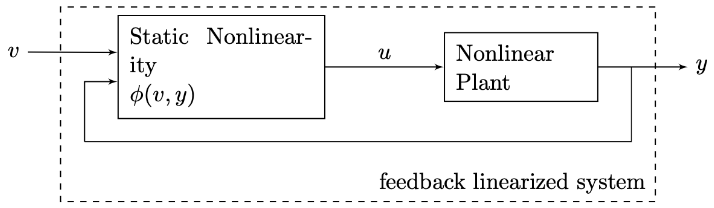

Figure 7: Feedback Linearization

By setting $u = \phi(v, y)$, we are essentially building an "input-redefining box" that has new inputs v and generates the plant inputs u based on measurements of the plant outputs y. *This box is part of the controller that we would implement for the nonlinear plant.*

Example 4: Magnetically Suspended Ball
Consider again the magnetically suspended ball. If we set

$$u = \phi(y, v) = y\sqrt{\frac{m(g+v)}{c}}$$

We obtain the linear time-invariant model

$$\dot{x}_1 = x_2, \qquad \dot{x}_2 = v, \qquad y = x_1.$$

Many nonlinear systems are not output-feedback linearizable. Sometimes, if we have the states (or good estimates) available, we can perform state-feedback linearization. It is important to note that this technique is based on a model of some physical system. If this model is inaccurate, the nonlinearity will not "cancel" exactly. It is therefore critical to understand the *robustness* of this technique with respect to plant modeling errors.