Reading: Class notes.

Problems:

1. The transfer function of a discrete-time (DT) system is given by

$$H(z) = \frac{1}{z + \frac{1}{2}}$$

with initial conditions $y[-1] = 1$.

1. What is the zero-input response?
2. What is the zero-state response with input $u[n] = u_0[n]$ where $u_0[n]$ denotes the unit step.
3. What is the total response – including both zero-input and zero-state response?
4. Draw the pole-zero diagram. Is the system BIBO stable?
5. What is the steady-state response? [Hint: $1 = \cos(0n)$]. Compare with the answer from part 3.
6. Now consider the input
$$u(t) = (1 + \sin(\frac{\pi}{6}n) + \cos(\frac{\pi}{4}n))u_0[n]$$
Obtain the steady-state response.

2. The transfer function of a continuous-time (CT) system is

$$H(s) = \frac{s+1}{s^2 + \frac{5}{6}s + \frac{1}{6}}$$

1. Write the associated ODE with output $y(t)$ and input $u(t)$.
2. Obtain the expression for the impulse response $h(t)$.
3. Consider the input signal $u(t) = u_0(t)$, where $u_0(t)$ denotes the unit step. Assuming zero initial conditions, obtain the output $y(t)$ by using
   (a) convolution, and
   (b) the method of Laplace transform.
4. Draw the pole-zero diagram. Is the system BIBO stable?
5. What is the steady-state response? Compare with the answer from part 3.
6. Now consider the input
$$u(t) = (1 + \sin(t) + \cos(t) + \sin(2t) + \sin(10t) + \sin(100t))u_0(t)$$
Obtain the steady-state response.

3. Coding question: In this problem, you will implement a numerical algorithm to solve an ODE

$$\dot{y}(t) + ay(t) = u(t), \quad y(0-) = b$$

where a and b are known constants (you may use $a = 0.1$ and $b = 0$).

Because time in computer is discrete-time, we use sampling to approximate the solution. Fix sampling time T_s to appropriate small value (for example, $T_s = 0.2$) and define

$$y_s[n] = y(nT_s), \quad u_s[n] = u(nT_s)$$

The derivative is approximated as a finite-difference as follows:

$$\dot{y}(nT_s) = \frac{y((n+1)T_s) - y(nT_s)}{T_s} = \frac{y_s[n+1] - y_s[n]}{T_s}$$

(why does this make sense?) Using all of this, we write the differential equation as a difference equation

$$\frac{y_s[n+1] - y_s[n]}{T_s} + ay_s[n] = u_s[n], \quad y_s[-1] = b$$

which is simplified to

$$y_s[n+1] = (1 - aT_s)y_s[n] + T_s u_s[n], \quad y_s[-1] = b$$

This is a type of a difference equation that we studied in the class. The overall algorithm is described in Algorithm 1. In the books concerned with numerical approximation, the notation Δ is used to denote the sampling time T_s. (Δ is the time elapsed between sucessive discrete-times). Using this notation,

$$y_s[n+1] = (1 - a\Delta)y_s[n] + \Delta u_s[n], \quad y_s[0] = b$$

A code file is uploaded on GitHub which a demo code for the system. You need to edit the code file to correctly implement the system given. Do not submit anything for this problem. The results of this Problem 3 will be used in the submission of the Problem 4.

Algorithm 1 Finite-difference approximation of an ODE

Require: T, Δ, a, b and u

1: Calculate $H = T/\Delta$
2: Let $y_s[0] = b$
3: **for** $n = 0, 1, \ldots, H-1$ **do**
4: $\quad u_s[n] = u(n\Delta)$
5: $\quad y_s[n+1] = (1 - a\Delta)y_s[n] + \Delta u_s[n]$
6: **end for**
7: **return** $\{y_s[n]\}_{n=1}^{H+1}$

---

4. In this problem, we use a convolution method to solve the difference equation. Such a method is closer to the learning algorithms that we learned in this course. Recall the convolution solution for the ODE

$$y(t) = \int_0^t h(t-\tau)u(\tau)\,\mathrm{d}\tau$$

Once again, we sample the continuous-time signal as discrete-time signal

$$y_s[n] = y(n\Delta), \quad u_s[m] = u(m\Delta), \quad h_s[n-m] = h(n\Delta - m\Delta)$$

In terms of these the integral is approximated as

$$\int_0^{n\Delta} h(t-\tau)u(\tau)\,\mathrm{d}\tau \approx \sum_{m=0}^{n} h[n-m]u[m]\Delta$$

(Such an approximation is referred to as the Riemannian sum). The algorithm is described in Algorithm 2.

Algorithm 2 Convolution method for approximation of an ODE

Require: T, Δ, a, b and u

Calculate $H = T/\Delta$
for $n = 0, 1, \ldots, H$ **do**
    Calculate $u_s[n] = u(n\Delta)$
    Calculate $h_s[n] = h(n\Delta)$
end for
Calculate $H = T/\Delta$
for $n = 0, 1, \ldots, H$ **do**
    $c = 0$
    for $m = 0, 1, \ldots, n$ **do**
        $c \leftarrow c + \Delta h_s[n-m]u_s[m]$
    end for
    $y_s[n] = c$
end for
return $\{y_s[n]\}_{n=1}^{H+1}$

Task for you. A Python code implementing the two algorithms has been uploaded on GitHub. Use the code to numerically approximate the solution for the following choice:

$$a = 0.1, \quad b = 0, \quad u(t) = \sin(t)u_0(t), \quad T = 30, \quad \Delta = 0.2$$

You need to submit a single plot comparing the output $y(t)$ obtained using three methods:

1. Analytically using the method of z-transform.
2. Numerically using the finite-difference approximation.
3. Numerically using the convolution algorithm.