

# Report 2

## MODELING OF CONTINUOUS SYSTEMS

Fernando Gabriel  
Student ID: 22M12329

April 18, 2022

### 1 Problem 1

#### Code

```
1  clc, clear all;
2  %% Set symbolic variables
3  syms f real % input
4
5  syms x th real % outputs
6  syms dx dth real % first derivatives
7  syms ddx ddth real % second derivatives
8
9  syms M m g l real % systems constants
10
11  %% Define the system
12  % States of systme
13  q = [x; th];
14  dq = [dx; dth];
15  ddq = [ddx; ddth];
16
17  q_dq = [q; dq];
18  dq_ddq = [dq; ddq];
19
20  % Postion of mass M
21  x_M = [x, 0];
22  % Postion of mass m
23  x_m = [x + sin(th) * l, cos(th) * l];
24
25  %% Velocity of each element of the system
26  dx_Mdt = jacobian(x_M, q) * dq;
27  dx_mdt = jacobian(x_m, q) * dq;
28
```

```

29 %% Kynetic Energy of the system
30 % Sum of both kynectic energies
31 K = 1/2 * M * (dx_Mdt' * dx_Mdt) + 1/2 * m * (dx_mdt' * dx_mdt);
32
33 %% Potential Energy with respect th = 0
34 U = g * M * x_M(2) + g * m * x_m(2);
35
36 %% Lagangian
37 L = K - U;
38
39 % Lagrangian calculation, equation without inputs
40 eq = jacobian(jacobian(L, dq), q_ddq) * dq_ddq - jacobian(L, q)';
41 eq = simplify(eq);
42
43 % Get matrix A
44 A = jacobian(eq, ddq);
45 % Get constants
46 B = simplify(eq - A * ddq);
47
48 %% Assemble system in matrix form with inputs
49 Eq_left = A * ddq;
50 Eq_right = [f; 0] - B;
51
52 Eq = Eq_left == Eq_right;
53 pretty(Eq)
54
55 %% Getting the solution by using the inverse matrix
56 sol_ddq = simplify(A \ Eq_right);
57 pretty(sol_ddq)
58
59 %% Equilibrium conditions
60 eq_q = [0; 0];
61 eq_dq = [0; 0];
62 eq_f = 0;
63
64 %% Linearization
65 linearized_eq = subs(jacobian(sol_ddq, q), [q_dq; f], [eq_q; eq_dq; eq_f]) * dq + ...
66 subs(jacobian(sol_ddq, f), [q_dq; f], [eq_q; eq_dq; eq_f]) * f;

```

## Results

Non linear systems after applying Lagrangian

$$\begin{pmatrix} \ddot{x} (M + m) + \ddot{\theta} l m \cos(\theta) = l m \sin(\theta) \dot{\theta}^2 + f \\ \ddot{\theta} m l^2 + \ddot{x} m \cos(\theta) l = g l m \sin(\theta) \end{pmatrix} \quad (1)$$

Matrix A and B are:

$$A = \begin{pmatrix} M + m & l m \cos(\theta) \\ l m \cos(\theta) & l^2 m \end{pmatrix}$$

$$B = \begin{pmatrix} -\dot{\theta}^2 l m \sin(\theta) \\ -g l m \sin(\theta) \end{pmatrix}$$

We use them to get the matrix form of the system.

Solution of the system is:

$$\begin{pmatrix} \ddot{x} \\ \ddot{\theta} \end{pmatrix} = \begin{pmatrix} \frac{l m \sin(\theta) \dot{\theta}^2 + f - g m \cos(\theta) \sin(\theta)}{-m \cos(\theta)^2 + M + m} \\ -\frac{l m \cos(\theta) \sin(\theta) \dot{\theta}^2 + f \cos(\theta) - g m \sin(\theta) - M g \sin(\theta)}{l (-m \cos(\theta)^2 + M + m)} \end{pmatrix}$$

Linearized system for equilibrium conditions  $x_0 = 0$   $\theta_0 = 0$

$$\begin{pmatrix} \Delta \ddot{x} \\ \Delta \ddot{\theta} \end{pmatrix} = \begin{pmatrix} \frac{f}{M} - \frac{\Delta \theta g m}{M} \\ \frac{\Delta \theta (M g + g m)}{M l} - \frac{f}{M l} \end{pmatrix}$$

To verify the derivation for the state equation of the inverted pendulum, we followed the following steps:

1. Define the position of the mass M and m
2. Calculate the velocity of each mass
3. Calculate the total kinetic energy of the system
4. Calculate the potential energy of the system
5. Get the Lagrangian based on the kinetic and potential energy
6. Apply Lagrange's equations to get the differential equations of the system
7. Transforming it into state space equation
8. Solve the values of the second time derivative of the states of the system,  
 $\ddot{x} \quad \ddot{\theta}$

To do most of the above steps, we used symbolic variable and the build in function **jacobian**.

## 2 Problem 2

---

```
1  clc, clear all, close all;
2  M = [0, 1, 2, 10];
3
4  % Range of x
5  x = -pi:0.1:pi;
6  figure()
7  % Plot original curve
8  plot(x, sin(x), '-x', 'LineWidth', 2);
9  hold on
10 grid on
11
12 % Plot expanded curves for different values of m
13 for i = 1:length(M)
14     m = M(i);
15     plot(x, mySine(x, m), 'LineWidth', 2)
16 end
17
18 legend('sin', 'm=0', 'm=1', 'm=2', 'm=10')
19
20 title('Sine function expansion');
21
22 %% Function to calculate the expanded sine function
23 function fx = mySine(x, m)
24     fx = 0;
25
26     for k = 0:m
27         fx = fx + (-1)^k * myFactorial(2 * k + 1) * x.^(2 * k + 1);
28     end
29
30 end
31 %% Function to calculate the inverse factorial
32 function x = myFactorial(k)
33     x = 1 / factorial(k);
34 end
35 % function X = myFactorial2(k)
36 % X = 1;
37
38 % for i = 1:k
39 %     X = X * i;
40 % end
41
42 % X = 1 / X;
43
44 % end
```

---

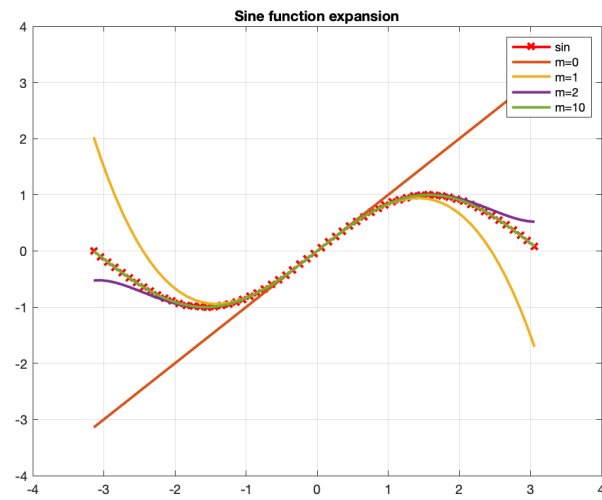


Figure 1: Comparison of plot for different  $m$

We can observe a tendency to get closer to the real curve when using more expansion terms