Project 4: Modeling Car Suspensions

Wafik Aboualim

March 12, 2022

Problem Statement :

We have a car with tires of mass M with suspension springs that has a spring constant k, with a damping constant c. The car starts oscillating at position $y = y_0$ with velocity $y' = 0$. Using the simple oscillator model we learned in class, the position $y(t)$ can be modeled by the I.V.P :

$$My'' + cy' + ky = 0,\ y(0) = y_0,\ y'(0) = 0 \tag{1}$$

Determine Damping :

We have a particular case of the I.V.P, where the mass of the of the Wheel of the car is $M = 390\ kg$ and spring has a spring constant $k = 44000\ N/m$ with initial conditions $y(0) = 0.2$ and $y'(0) = 0$, plugging the values in our O.D.E:

$$390y'' + cy' + 44000y = 0 \tag{2}$$

Given that the system is critically damped:

$$c^2 - 4Mk = 0 \implies c^2 = 4 \times 390 \times 44000 = 68640000 \implies c \approx \pm 8285 \tag{3}$$

Now we have our I.V.P:

$$390y'' \pm 8285y' + 44000y = 0,\ y(0) = 0.2,\ y'(0) = 0 \tag{4}$$

Solving the I.V.P :

Looking at the O.D.E, we easily compute a general solution to $y(t)$, first we start by solving the corresponding characteristic equation, we take $c = +8285$ for convention :

$$390r^2 + 8285r + 44000 = 0 \implies r_1 = -\frac{275}{26},\ r_2 = -\frac{32}{3} \tag{5}$$

We have two distinct real eigen values that satisfy the characteristic equation, thus the fundamental set of solutions to the O.D.E:

$$y = \{e^{-\frac{275}{26}t}, e^{-\frac{32}{3}t}\} \tag{6}$$

from which we have our general solution:

$$y = C_1 e^{-\frac{275}{26}t} + C_2 e^{-\frac{32}{3}t} \tag{7}$$

Using our initial conditions $y(0) = 0.2,\ y'(0) = 0$ to solve for C_1, C_2:

$$y(0) = 0.2 \implies C_1 + C_2 = 0.2$$

$$y'(0) = 0 \implies \frac{275}{26}C_1 + \frac{32}{3}C_2 = 0$$

Solving the two equations, we have :

$$C_1 = \frac{832}{35},\ C_2 = -\frac{165}{7}$$

Thus, the solution to I.V.P:

$$y(t) = \frac{832}{35}e^{-\frac{275}{26}t} - \frac{165}{7}e^{-\frac{32}{3}t} \tag{8}$$

Plotting our solution :

```
[47]: import numpy as np
      import math
      import matplotlib.pyplot as plt

[48]: y = lambda x:  math.exp(-(32*x)/3)*((23.7714*math.exp((7*x)/78) - 23.5714))

[49]: t_vals = np.arange(0,10,0.1,dtype=float)
      y_vals = [y(t) for t in t_vals]

[50]: plt.plot(t_vals,y_vals)
      plt.plot(t_vals,[0.1 for t in t_vals])

[50]: [<matplotlib.lines.Line2D at 0x262788f0340>]
```

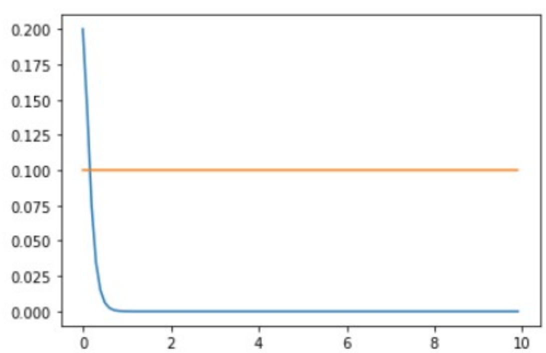


Figure 1: plotting $y(t),\ 0 < t < 10$

Looking closely at the plot we can see that the system reaches a tolerance of $y = 0.1$ after $t \approx 0.2$

Changing Ride Quality :

In this case, we have $M = 450, k = 16000, c = 1600$. With initial conditions $y(0) = 0.2,\ y'(0) = 0$.

Plugging our values into our simple oscillator model :

$$450y'' + 1600y' + 16000y = 0 \tag{9}$$

Solving the I.V.P as in the first case, the solution to the I.V.P :

$$y(t) = 0.0624695e^{-\frac{16}{9}t}\sin(\frac{8\sqrt{41}}{9}) + 0.2e^{-\frac{16}{9}t}\cos(\frac{8\sqrt{41}}{9}) \tag{10}$$

Plotting our solution:

```
[88]: import numpy as np
      import math
      import matplotlib.pyplot as plt
```

```
[89]: y = lambda t: math.exp(-(16*t)/9)*((0.0624695*math.sin((8*math.sqrt(41)*t)/9) + 0.2*math.cos((8*math.sqrt(41)*t)/9)))
```

```
[94]: t_vals = np.arange(0,10,0.1,dtype=float)
      y_vals = [y(t) for t in t_vals]
```

```
[96]: plt.plot(t_vals,y_vals)
      plt.plot(t_vals,[0.1 for t in t_vals])
```

```
[96]: [<matplotlib.lines.Line2D at 0x2b6057aa4d0>]
```

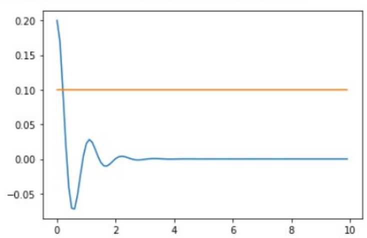


Figure 2: plotting $y(t),\ 0 < t < 10$

-We can see that the graph of the system shows a an oscillating behavior which implies a bumpy ride.
-Looking at the quantities $c^2,\ 4Mk$:

$$c^2 = 1600^2,\ 4Mk = 4 \times 450 \times 16000$$

we can see that $c^2 < 4Mk$, which implies that the system has imaginary roots and thus underdamped.

-The second model follows a similar behavior around $y = 0.1$, the model reaches a tolerance of 0.2 after $t \approx 0.2$

Changing the mass on the wheels:

Adding an additional $1000lbs$ to the car will increase a load of $250lbs$ on each wheel, given an evenly distributed weight. to compute and graph the solutions, we repeat the same process but with the new mass $M = M_0 + 250$ in both cases, we get the two I.V.Ps :

$$640y'' + 8285y' + 44000y = 0, \ y(0) = 0.2, \ y'(0) = 0 \tag{11}$$

$$700y'' + 1600y' + 16000y = 0 \ y(0) = 0.2, \ y'(0) = 0 \tag{12}$$

Similarly, We solve the two I.V.Ps and graph the solutions:

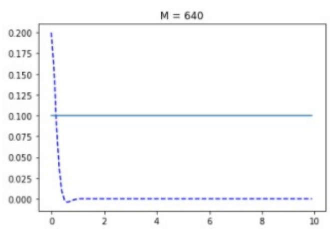
```
[115]: import numpy as np
       import math
       import matplotlib.pyplot as plt

[130]: y = lambda t: math.exp(-(1657*t)/256)*((0.249806*math.sin((math.sqrt(1759951)*t)/256) + 0.2*math.cos((math.sqrt(1759951)*t)/256)))
       g = lambda t: math.exp(-(1657*t)/280)*((0.224331*math.sin((math.sqrt(2182351)*t)/280) + 0.2*math.cos((math.sqrt(2182351)*t)/280)))

[146]: t_vals = np.arange(0,10,0.1,dtype=float)
       y_vals = [y(t) for t in t_vals]
       g_vals = [g(t) for t in t_vals]

[152]: plt.plot(t_vals,y_vals,'b--')
       plt.plot(t_vals,[0.1 for t in t_vals])
       plt.title('M = 640')

[152]: Text(0.5, 1.0, 'M = 640')
```




```
[153]: plt.plot(t_vals,g_vals,'r--')
       plt.plot(t_vals,[0.1 for t in t_vals])
       plt.title('M = 700')

[153]: Text(0.5, 1.0, 'M = 700')
```

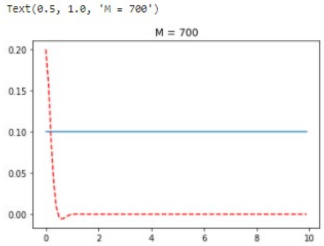


Figure 3: plotting $y(t), \ 0 < t < 10$ at $M = 640, \ M = 700$

Looking at the first I.V.P (M=640), the new mass of the car caused the system to be underdamped. The second I.V.P behavior did not change as the system still underdamped. However, the oscillations occurred at a faster rate and had lower amplitude which caused the system to reach equilibrium faster. Overall, the two systems ($M = 640, \ M = 700$) show approximately the same behavior.

Commentary :

In this project I split my work into three main stages :

1- Constructing the model :
I located the given information such as the mass of the wheels and the damping coefficient and the spring constant. I plugged the values in the simple oscillator model given in the documentation to develop a model for the system in each of the cases.

2- Solving the model :
In that stage, I used the methods we took in class to find a general solution to the system and then solve the I.V.P in each of the cases.

3- Graphical solutions :
In this stage, I used IPython and matplotlib and numpy, to graph and compare the solutions.

Given that this model represents the actual dynamics of the suspension. Most cars are engineered to be slightly overdamped for few reasons :

-Most cars are overdamped to adapt with bumps at high speeds. However, overdamping when done more than required will affect the traction system of the car because it will increase the time it takes for the suspension springs to reach equilibrium which will make it more volatile to loss of control.

- Underdamping comes with huge risks. Although in the underdamped case the car will be extremely stable at lower speeds. In higher speeds, the slightest oscillation will cause the car to lose control.

- Critically damping a car is not realistically achievable, as the slightest change in mass or other factors will cause the car to be either underdamped or overdamped.

Overall this project made me understand the car dynamics a lot better and how they relate to differential equations. I learned about the suspension system and how it is composed of 4 springs in each tire. I learned why cars are designed to be overdamped and why underdamping a car might be dangerous. I also applied my knowledge of D.E to solve real world systems and used the graphical approach and compared several models and made inference on them.

Problems I encountered :

Overall, the project was straightforward, but I ran into some minor problems :
- It was tedious and took me a few trials write solution function correctly using Python in order to graph the solutions.
- I had to redo some of the graphs because the aspect ratios and had to settle for $0 < t < 10$ because the graphs were condensed and not clear.
- I had problem calculating the required time for the model to reach 0.1m mark and used WolframeAlpha to save time and to ensure correctness.