# Math 291H Computational Lab #2

Rajeev Atla

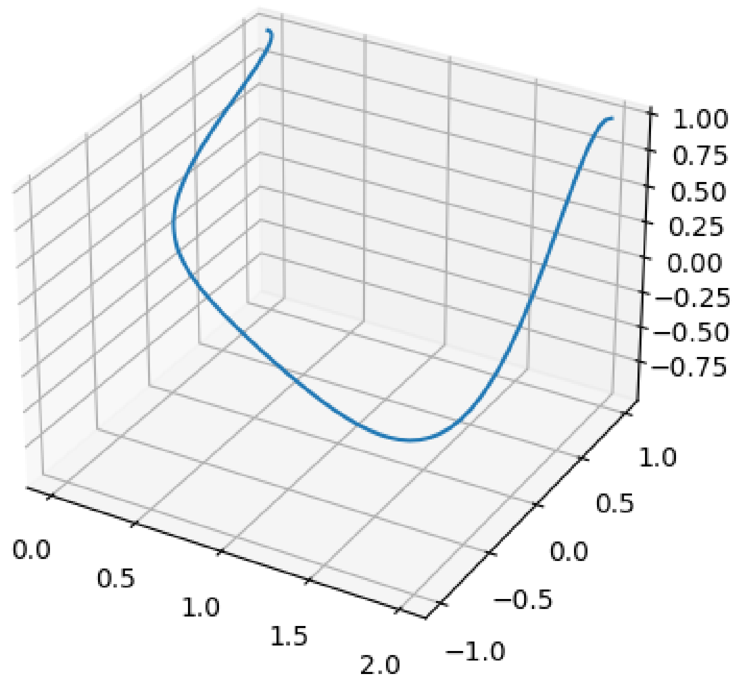
# Problem 1

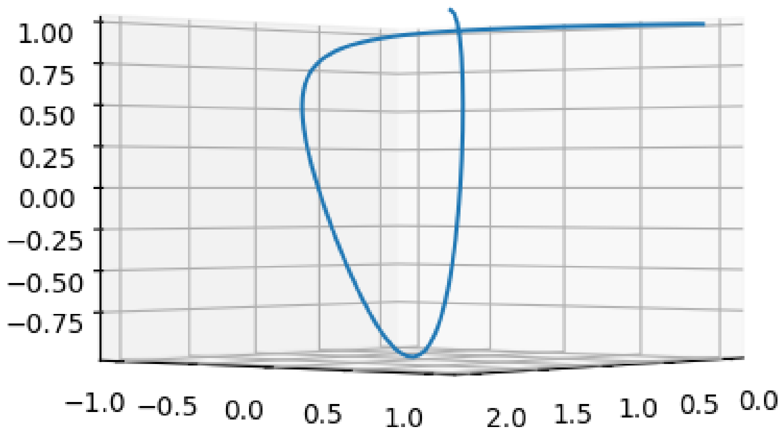
We use Python 3 for this lab, especially the SymPy package.

```python
from sympy.vector import CoordSys3D
from sympy import Symbol, pi
from sympy import *

t = Symbol("t")

x1 = t + t ** 2
x2 = cos(2 * pi * t)
x3 = cos(2 * pi * (t ** 3))

N = CoordSys3D("N")
x = N.locate_new("M", x1 * N.i + x2 * N.j + x3 * N.k)

Tf = 1
Tf_sym = Integer(Tf)
```

# Problem 2

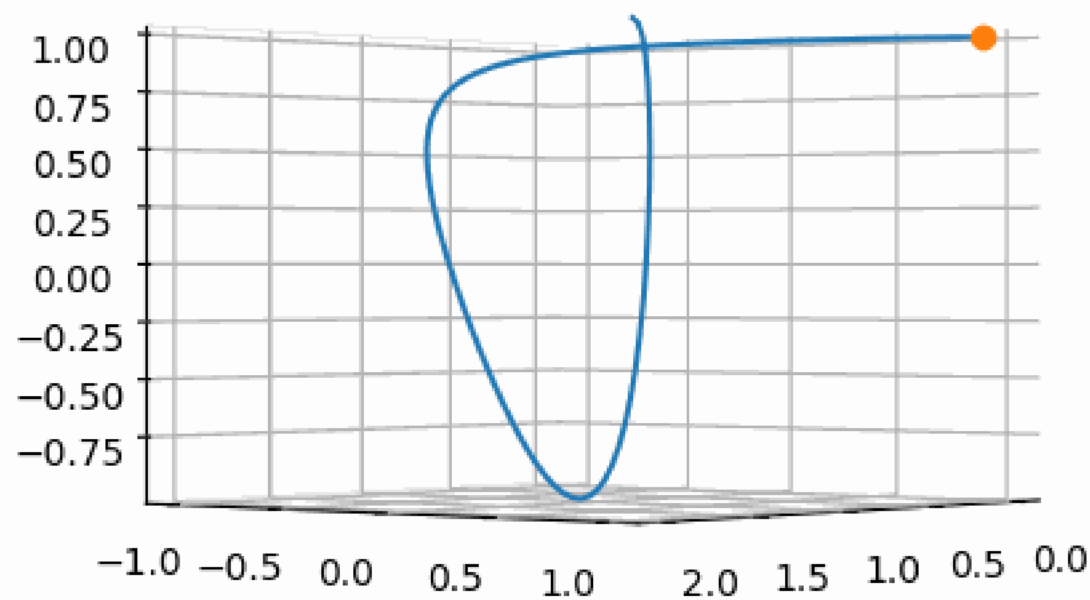
```
from mpl_toolkits import mplot3d
import numpy as np
import matplotlib.pyplot as plt


fig = plt.figure()
ax = plt.axes(projection="3d")


t_range = np.linspace(0, Tf, 100)


x1_lambda = lambdify(t, x1, "numpy")
x2_lambda = lambdify(t, x2, "numpy")
x3_lambda = lambdify(t, x3, "numpy")

x1_range = x1_lambda(t_range)
x2_range = x2_lambda(t_range)
x3_range = x3_lambda(t_range)

x_range = np.array([x1_range, x2_range, x3_range])


plt.plot(x1_range, x2_range, x3_range)


plt.savefig("fig1.png")

ax.view_init(0, 40)
plt.savefig("fig2.png")
```


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
1.0
0.5
0.0
−0.5
−1.0
0.0
0.5
1.0
1.5
2.0


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.0 −0.5 0.0 0.5 1.0
2.0 1.5 1.0 0.5 0.0

# Problem 3

```
from matplotlib import animation


def curve_numpy(t):
    return np.array([x1_lambda(t), x2_lambda(t), x3_lambda(t)])


(point,) = plt.plot(x1_lambda(0), x2_lambda(0), x3_lambda(0), marker="o")


def update(t):
    x, y, z = curve_numpy(t)
    point.set_data(x, y)
    point.set_3d_properties(z, "z")
    return (point,)


N = 50

ani = animation.FuncAnimation(
    fig, update, interval=N, blit=True, repeat=True, frames=t_range
)
ani.save("fig3.gif")
```


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.0 −0.5 0.0 0.5 1.0
2.0 1.5 1.0 0.5 0.0

# Problem 4

```
v1 = simplify(diff(x1, t))
v2 = simplify(diff(x2, t))
v3 = simplify(diff(x3, t))


v_scalar = simplify(sqrt(v1 ** 2 + v2 ** 2 + v3 ** 2))
path_length = integrate(v_scalar, (t, 0, Tf_sym))
print(path_length.evalf())
```

The length of the curve is $\approx 6.913$.

# Problem 5

```
from sympy.physics.vector import *

N = ReferenceFrame("N")

x = x1 * N.x + x2 * N.y + x3 * N.z
v = x.diff(t, N)
T = v.normalize()

print("Tangent vector:")
print(T.subs(t, 0))
print(T.subs(t, Tf_sym / 2).evalf())
print(T.subs(t, Tf_sym))
```

$$\begin{aligned} \mathbf{T}(0) &= \mathbf{e}_1 \\ \mathbf{T}\left(\frac{T_f}{2}\right) &\approx 0.515\mathbf{e}_1 - 0.857\mathbf{e}_2 \\ \mathbf{T}(T_f) &= \mathbf{e}_1 \end{aligned}$$

# Problem 6

```
T_diff = T.diff(t, N)

a_perp = T_diff * v_scalar
normal = a_perp.normalize()

print("Normal vector:")
print(normal.subs(t, 0))
print(normal.subs(t, Tf_sym / 2).evalf())
print(normal.subs(t, Tf_sym))
```

$$\mathbf{N}(0) = -\mathbf{e}_2$$
$$\mathbf{N}\left(\frac{T_f}{2}\right) \approx -0.272\mathbf{e}_1 + 0.948\mathbf{e}_2 - 0.164\mathbf{e}_3$$
$$\mathbf{N}(T_f) = -\frac{\sqrt{82}}{82}\mathbf{e}_2 - \frac{9\sqrt{82}}{82}\mathbf{e}_3$$

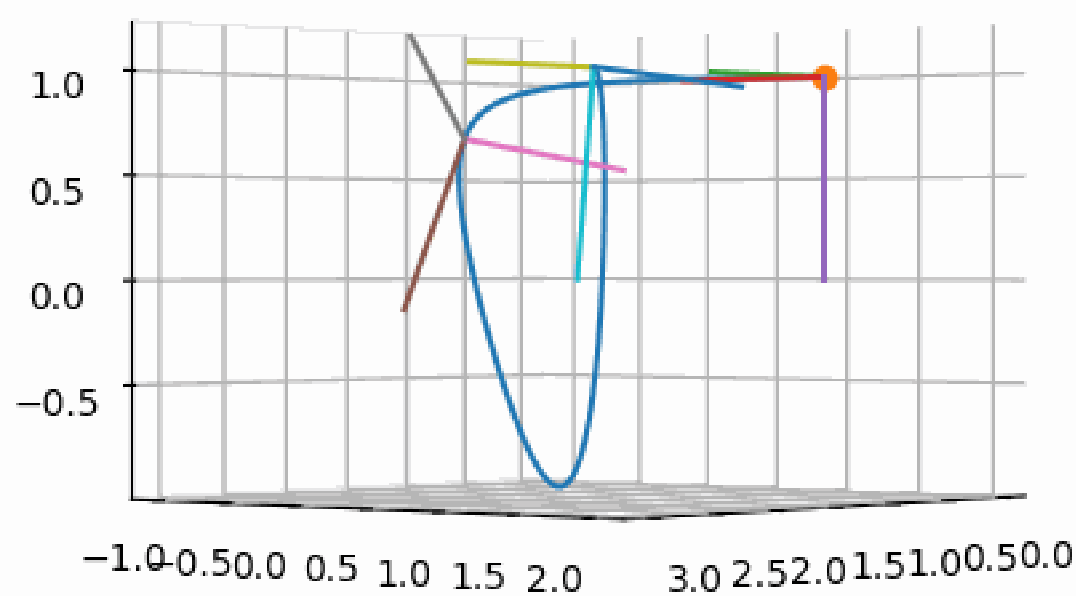
# Problem 7

```
T_diff = T.diff(t, N)

a_perp = T_diff * v_scalar
normal = a_perp.normalize()

print("Normal vector:")
print(normal.subs(t, 0))
print(normal.subs(t, Tf_sym / 2).evalf())
print(normal.subs(t, Tf_sym))
```

$$\begin{aligned}
\mathbf{B}\left(0\right) &= -\mathbf{e}_3 \\
\mathbf{B}\left(\frac{T_f}{2}\right) &= 0.813\mathbf{e}_1 + 0.318\mathbf{e}_2 + 0.488\mathbf{e}_3 \\
\mathbf{B}\left(T_f\right) &\approx \frac{9\sqrt{82}}{82}\mathbf{e}_2 - \frac{\sqrt{82}}{82}\mathbf{e}_3
\end{aligned}$$

# Problem 8

```
t_vals = [0, Tf_sym / 2, Tf_sym]
vectors = [T, normal, B]

for i in t_vals:
    for v in vectors:
        vx = dot(v.subs(t, i), N.x).evalf()
        vy = dot(v.subs(t, i), N.y).evalf()
        vz = dot(v.subs(t, i), N.z).evalf()

        original_point = np.array(
            [x1.subs(t, i).evalf(), x2.subs(t, i).evalf(), x3.subs(t, i).evalf()]
        )
        final_point = np.array(
            [original_point[0] + vx, original_point[1] + vy, original_point[2] + vz]
        )

        x_vals = np.array([original_point[0], final_point[0]])
        y_vals = np.array([original_point[1], final_point[1]])
        z_vals = np.array([original_point[2], final_point[2]])

        plt.plot(x_vals, y_vals, z_vals)

ani.save("fig4.gif")
```


1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
3.0
2.5
2.0
1.5
1.0
0.5
0.0

# Problem 9

```
curvature = (a_perp.magnitude()) / (v_scalar ** 2)
print(curvature.subs(t, 0))
print(curvature.subs(t, Tf_sym / 2).evalf())
print(curvature.subs(t, Tf))
```

$$\kappa\left(0\right) = 4\pi^2$$
$$\kappa\left(\frac{T_f}{2}\right) \approx 2.757$$
$$\kappa\left(T_f\right) = \frac{4\pi^2\sqrt{82}}{9}$$