```
import numpy as np
from scipy import integrate


from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.colors import cnames
from matplotlib import animation

# Note: t0 is required for the odeint function, though it's not used here.
def lorentz_deriv( y, t0, sigma=10., b=8./3, r=28.0):
    """Compute the time-derivative of a Lorenz system."""
    return [sigma * (y[1] - y[0] ), y[0] * (r - y[2]) - y[1], y[0] * y[1] - b * y[2]]

N_trajectories = 3

x0 = np.array([[0, 2,  0],  # starting vectors
               [0, -2, 0],
               [0, 2.01, 0]])

# Solve for the trajectories
t = np.linspace(0, 100, 10000)
x_t = np.asarray([integrate.odeint(lorentz_deriv, x0i, t)
                  for x0i in x0])

# Set up figure & 3D axis for animation
fig = plt.figure()
ax = fig.gca(projection='3d')

plt.plot(x_t[0,:,0], x_t[0,:,1], x_t[0,:,2])
plt.plot(x_t[1,:,0], x_t[1,:,1], x_t[1,:,2])
plt.plot(x_t[2,:,0], x_t[2,:,1], x_t[2,:,2])

plt.show()
```