hw0

January 23, 2023

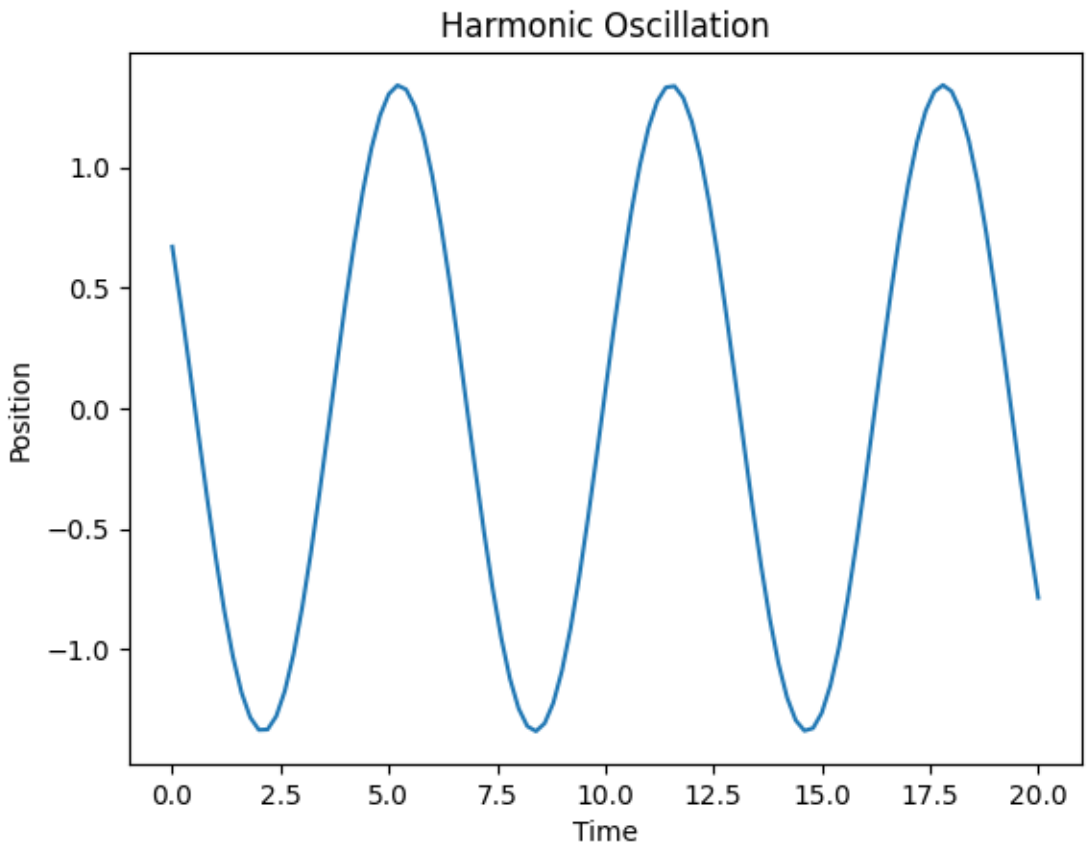
Tory Smith

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
```

1) Analytic Solution

[ ]:
```
times = np.linspace(0, 20, 101)
A = 1.34
phi =  np.pi/3.0
kmratio = 1.0
x_t = A*np.cos(np.sqrt(kmratio)*times + phi)
plt.plot(times, x_t)
plt.ylabel('Position')
plt.xlabel('Time')
plt.title('Harmonic Oscillation')
```

[ ]:
```
Text(0.5, 1.0, 'Harmonic Oscillation')
```



2) Numeric Solution and Error

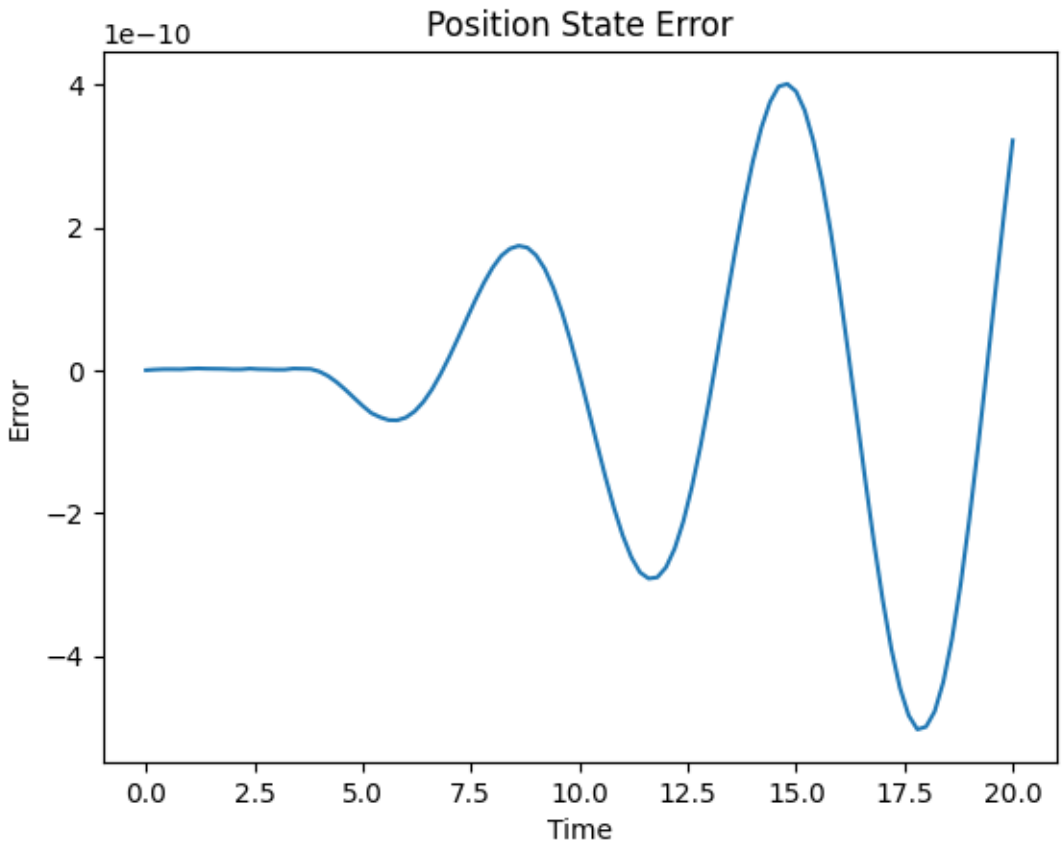
[ ]:
```
def harmonic_oscillator(x, t, kmratio):
    dx = np.zeros(2)
    dx[0] = x[1]
    dx[1] = -kmratio*x[0]
    return dx

y0 = [A*np.cos(phi), -A*np.sqrt(kmratio)*np.sin(phi)]

sol = odeint(harmonic_oscillator, y0, times, args=(kmratio,), rtol=1e-12,␣
 ↪atol=1e-20)
y = sol[:, 0]-x_t
print("Maximum Error:", np.max(y))
plt.plot(times, y)
plt.title("Position State Error")
plt.ylabel('Error')
plt.xlabel("Time")
```

```
Maximum Error: 4.005180631594385e-10
```

[ ]: Text(0.5, 0, 'Time')



Error occurs because the solution is truncated due to the discretization of the solver. Integration is done over an infinite number of steps, but because we are limited to a finite number of steps in computation some information is lost between those steps.