1 AE332: Modelling and Analysis Lab

1.1 Session 1: Solving Ordinary Differential Equations (7^{th} August 2023)

Name: Gaurav Gupta ; **SC-Code:** SC21B026

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.integrate as scpy
```

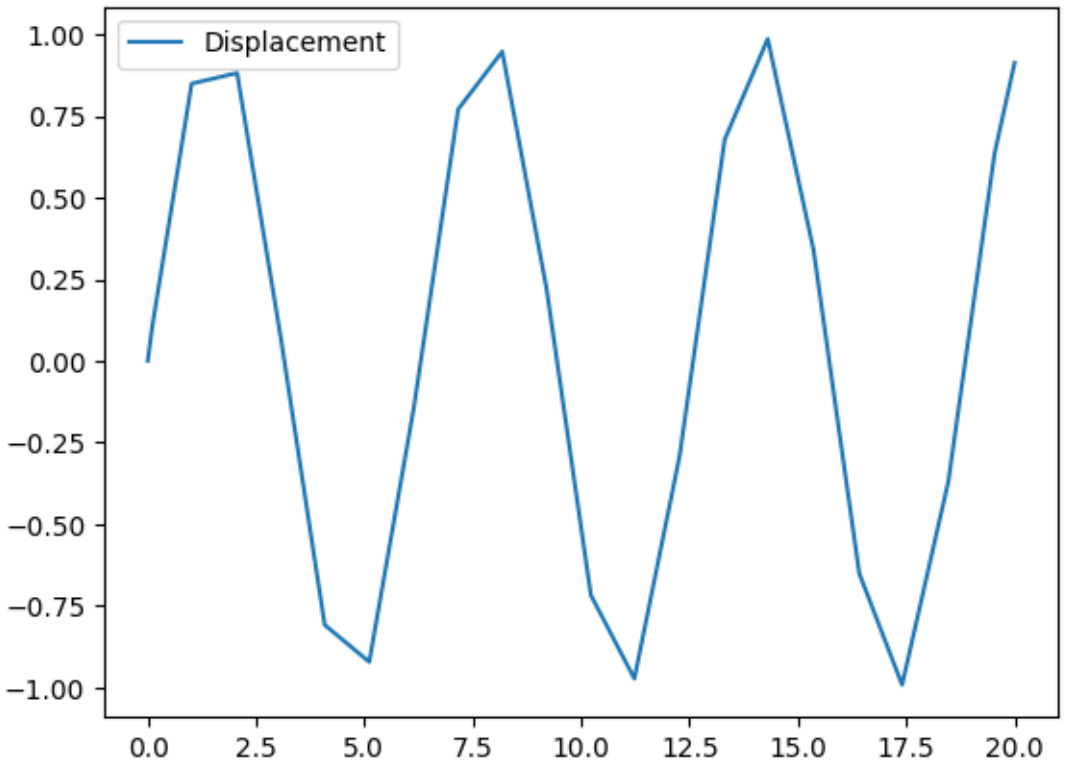
1.2 Problem 1: Solving a First Order ODE

1.2.1 Solution with default values of `atol`=1e-5, `rtol`=1e-5

[2]:
```
y0 = np.array([0])
t= [0,20]
atol = np.array([0.001, 0.0001, 0.00001, 1e-5, 1e-6, 1e-7, 1e-8, 1e-9, 1e-10,␣
↪1e-11, 1e-12])
rtol = atol
def yprime(t,y):
    return np.cos(t)
```

[3]:
```
sol = scpy.solve_ivp(yprime, t, y0, atol=1e-5, rtol=1e-5)
plt.plot(sol.t, sol.y[0], label='Displacement')
plt.legend()
```

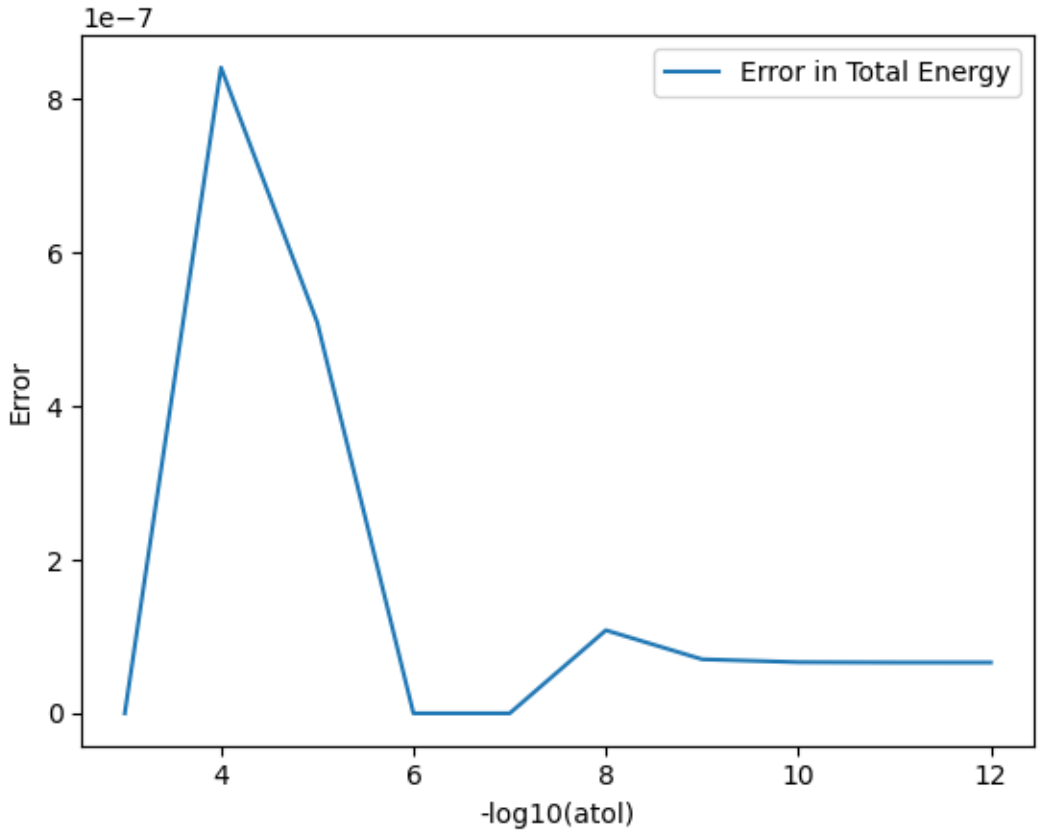
[3]:
```
<matplotlib.legend.Legend at 0x2adec0f5cd0>
```



1.2.2 Variation of `rtol` keeping `atol` constant

[4]:
```
error1 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, atol=atol[i], rtol=1e-5)
    error1[i] = np.max(np.sin(sol.t) - sol.y)
plt.plot(-np.log10(atol), error1, label='Error in Total Energy')
plt.xlabel("-log10(atol)")
plt.ylabel('Error')
plt.legend()
```

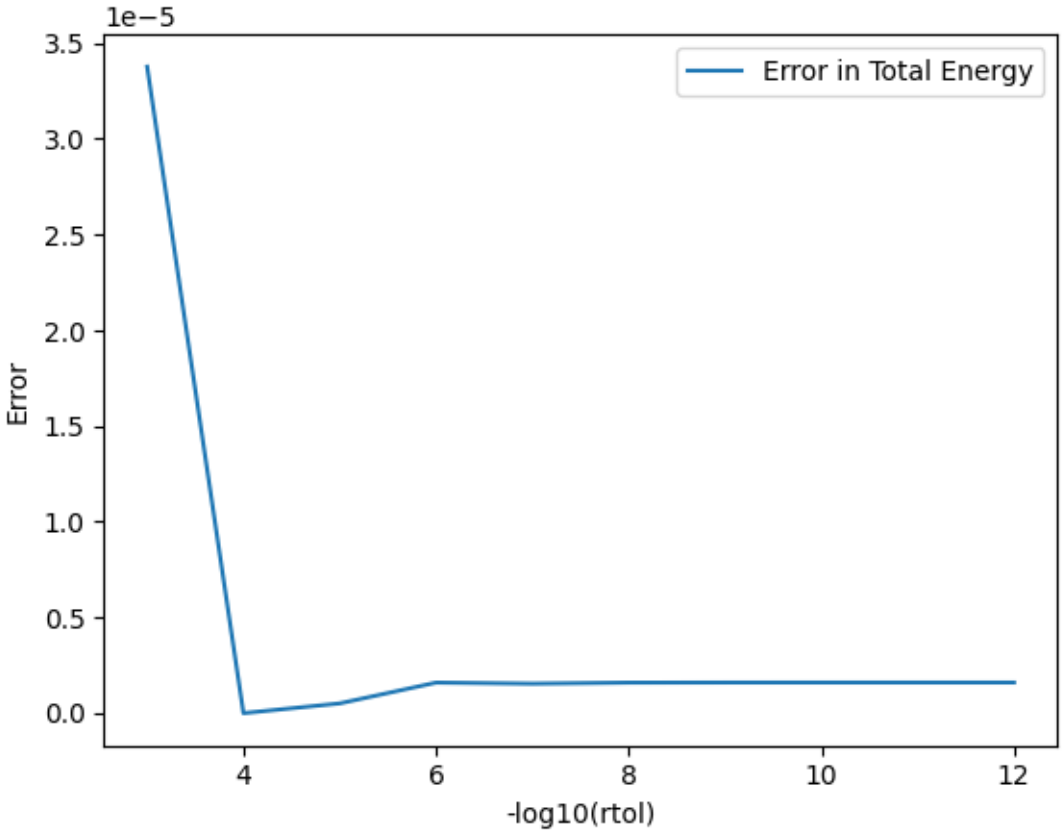
[4]:
```
<matplotlib.legend.Legend at 0x2adec2c3290>
```



1.2.3 Variation of `rtol` keeping `rtol` constant

[5]:
```
error1 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, rtol=rtol[i], atol=1e-5)
    error1[i] = np.max(np.sin(sol.t) - sol.y)
plt.plot(-np.log10(atol), error1, label='Error in Total Energy')
plt.xlabel("-log10(rtol)")
plt.ylabel('Error')
plt.legend()
```

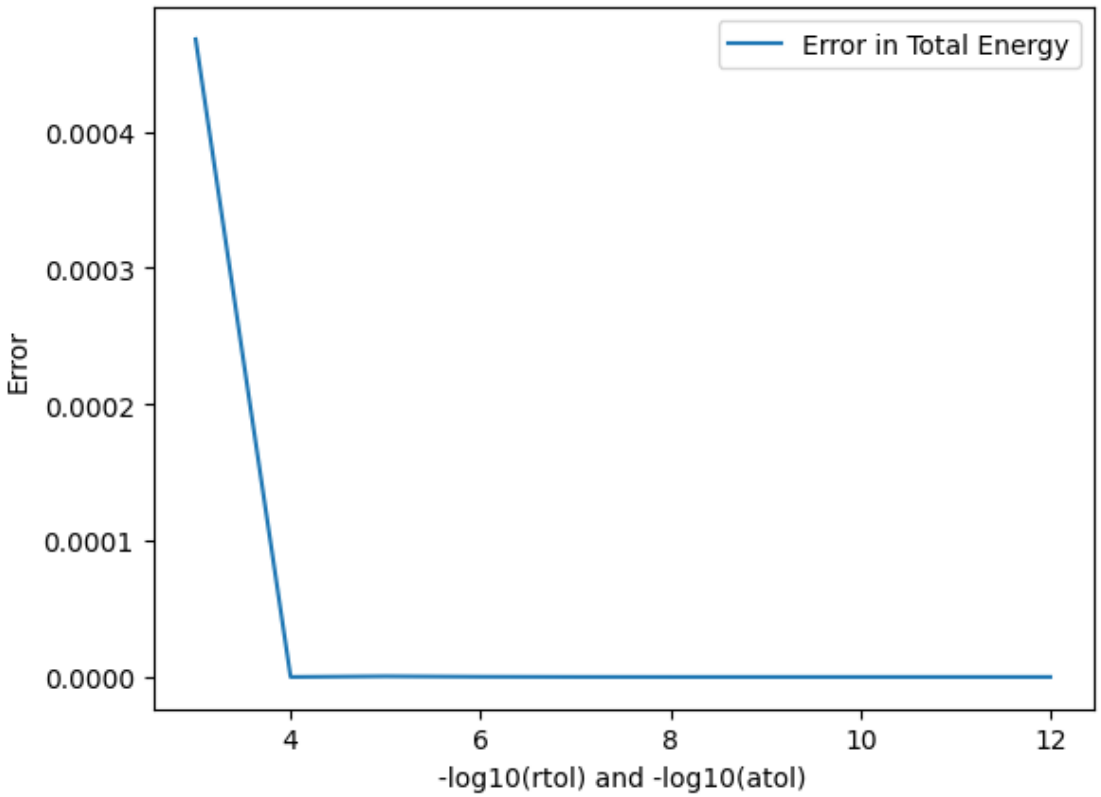
[5]:
```
<matplotlib.legend.Legend at 0x2adec37b510>
```



1.2.4 Variation of both `atol` and `rtol`

[6]:
```
error1 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, rtol=rtol[i], atol=atol[i])
    error1[i] = np.max(np.sin(sol.t) - sol.y)
plt.plot(-np.log10(atol), error1, label='Error in Total Energy')
plt.xlabel("-log10(rtol) and -log10(atol)")
plt.ylabel('Error')
plt.legend()
```

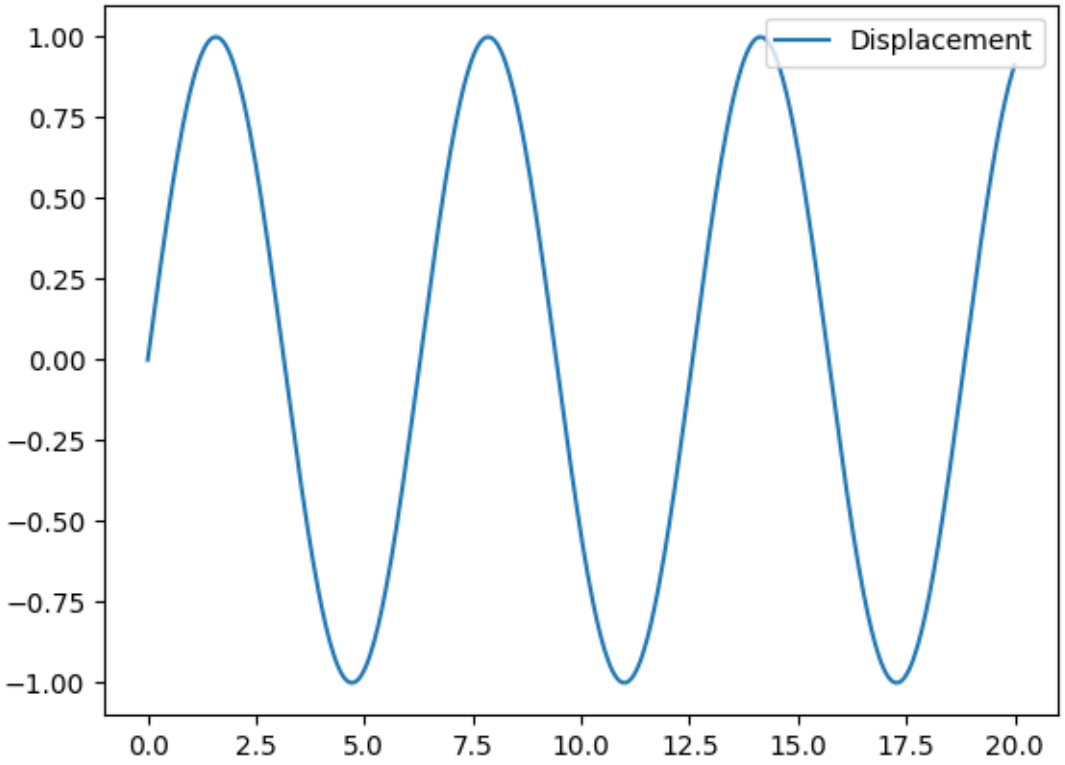
[6]:
```
<matplotlib.legend.Legend at 0x2adec2ef450>
```



[7]:
```
### Solution with the values of `atol`=1e-12, `rtol`=1e-12
```

[8]:
```
sol = scpy.solve_ivp(yprime, t, y0, atol=1e-12, rtol=1e-12)
plt.plot(sol.t, sol.y[0], label='Displacement')
plt.legend()
```

[8]:
```
<matplotlib.legend.Legend at 0x2adec43d010>
```



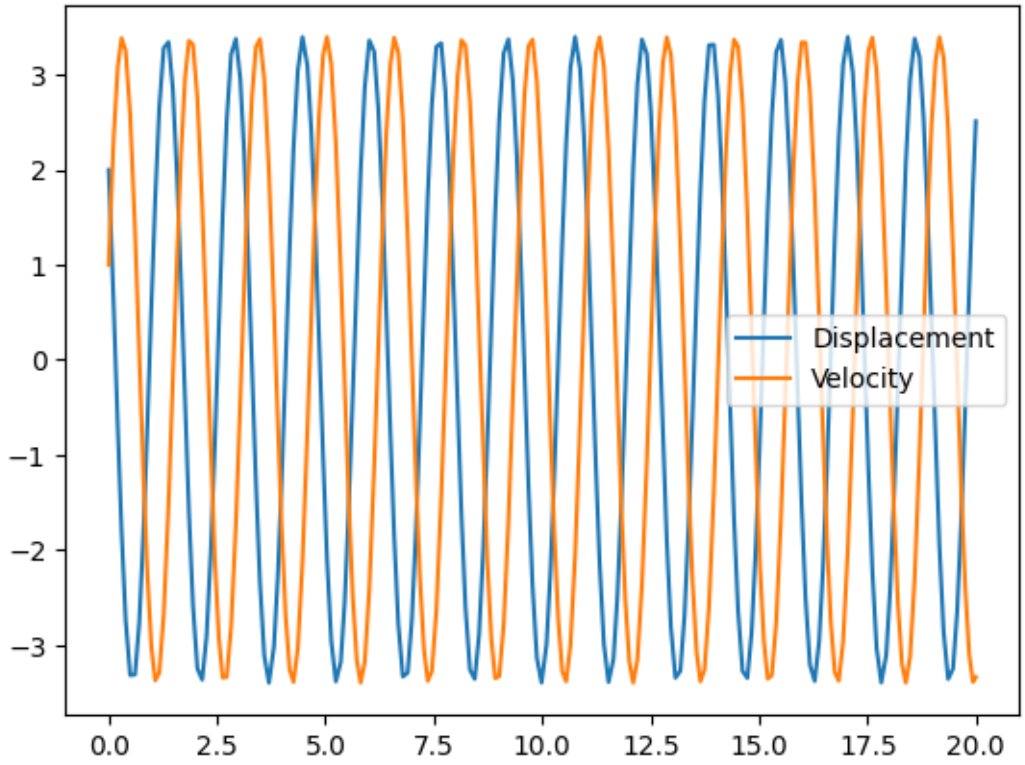
1.3 Problem 2: Solving a coupled ODE

1.3.1 Solution with default values of `atol`=1e-5, `rtol`=1e-5

[9]:
```
y0 = np.array([2, 1])
t=[0,20]
def yprime(t,y):
    return np.dot(y,np.array([[-3, 5], [-5, 3]])) # [[a11, a21], [a12, a22]]
```

[10]:
```
sol = scpy.solve_ivp(yprime, t, y0, atol=1e-5, rtol=1e-5)
plt.plot(sol.t, sol.y[0], label='Displacement')
plt.plot(sol.t, sol.y[1], label='Velocity')
plt.legend()
```

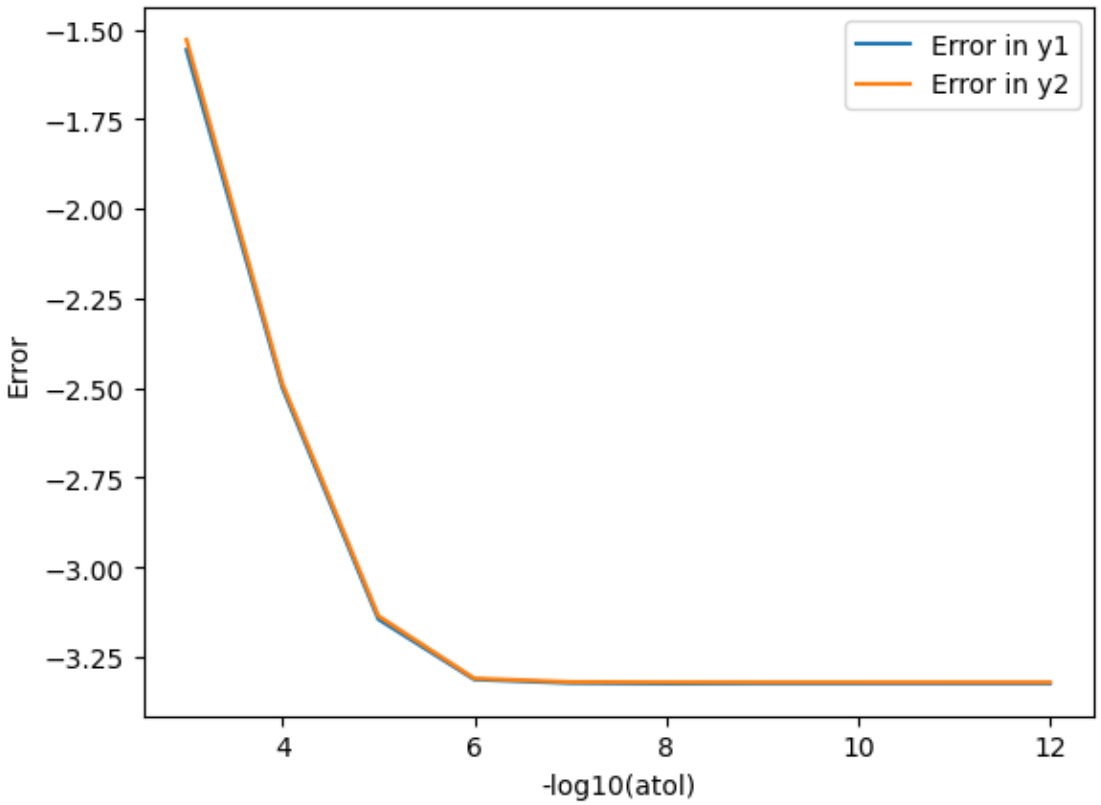
[10]:
```
<matplotlib.legend.Legend at 0x2adec2a3d90>
```



1.3.2 Variation of `atol` while keeping `rtol` constant

[11]:
```
error1 = np.zeros_like(atol)
error2 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, atol=atol[i], rtol=1e-5)
    y1 = 2*np.cos(4*sol.t) - 2.75*np.sin(4*sol.t)
    y2 = 3.25*np.sin(4*sol.t) + np.cos(4*sol.t)
    error1[i] = np.max(y1 - sol.y[0])
    error2[i] = np.max(y2 - sol.y[1])
plt.plot(-np.log10(atol), np.log10(error1), label='Error in y1')
plt.plot(-np.log10(atol), np.log10(error2), label='Error in y2')
plt.xlabel("-log10(atol)")
plt.ylabel('Error')
plt.legend()
```

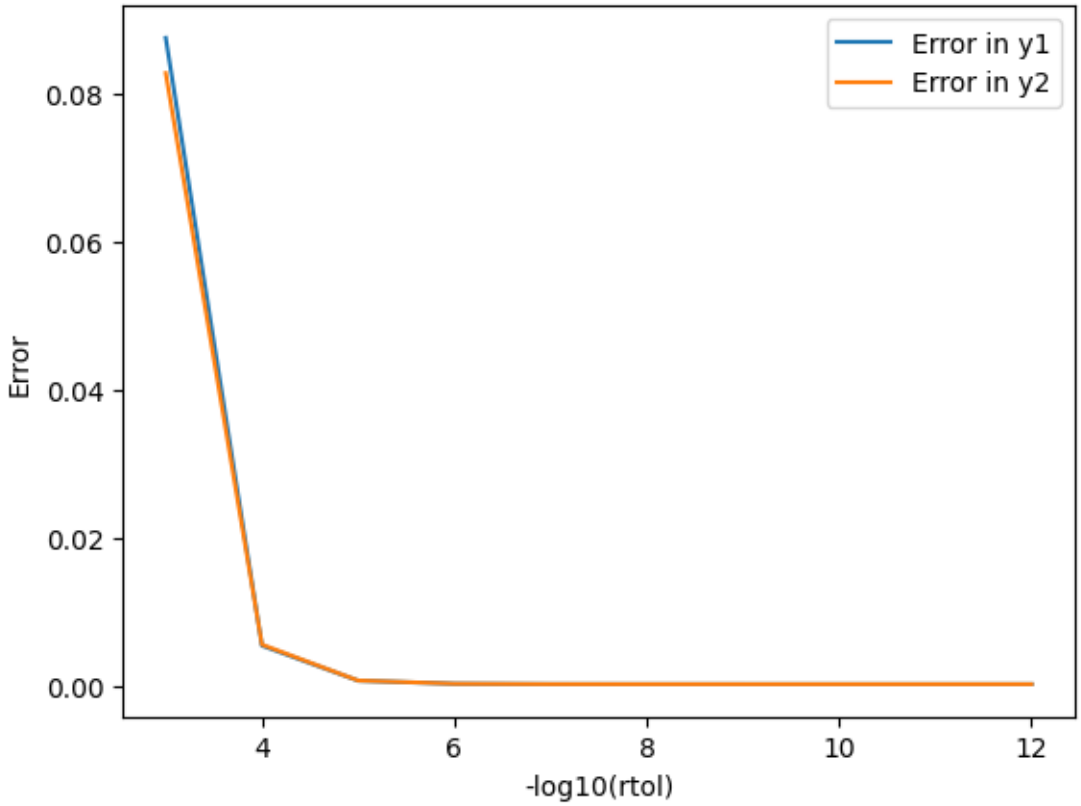
[11]:
```
<matplotlib.legend.Legend at 0x2adee698450>
```



1.3.3 Variation of `rtol` while keeping `atol` constant

[12]:
```
error1 = np.zeros_like(atol)
error2 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, rtol=rtol[i], atol=1e-5)
    y1 = 2*np.cos(4*sol.t) - 2.75*np.sin(4*sol.t)
    y2 = 3.25*np.sin(4*sol.t) + np.cos(4*sol.t)
    error1[i] = np.max(y1 - sol.y[0])
    error2[i] = np.max(y2 - sol.y[1])
plt.plot(-np.log10(atol), error1, label='Error in y1')
plt.plot(-np.log10(atol), error2, label='Error in y2')
plt.xlabel("-log10(rtol)")
plt.ylabel('Error')
plt.legend()
```

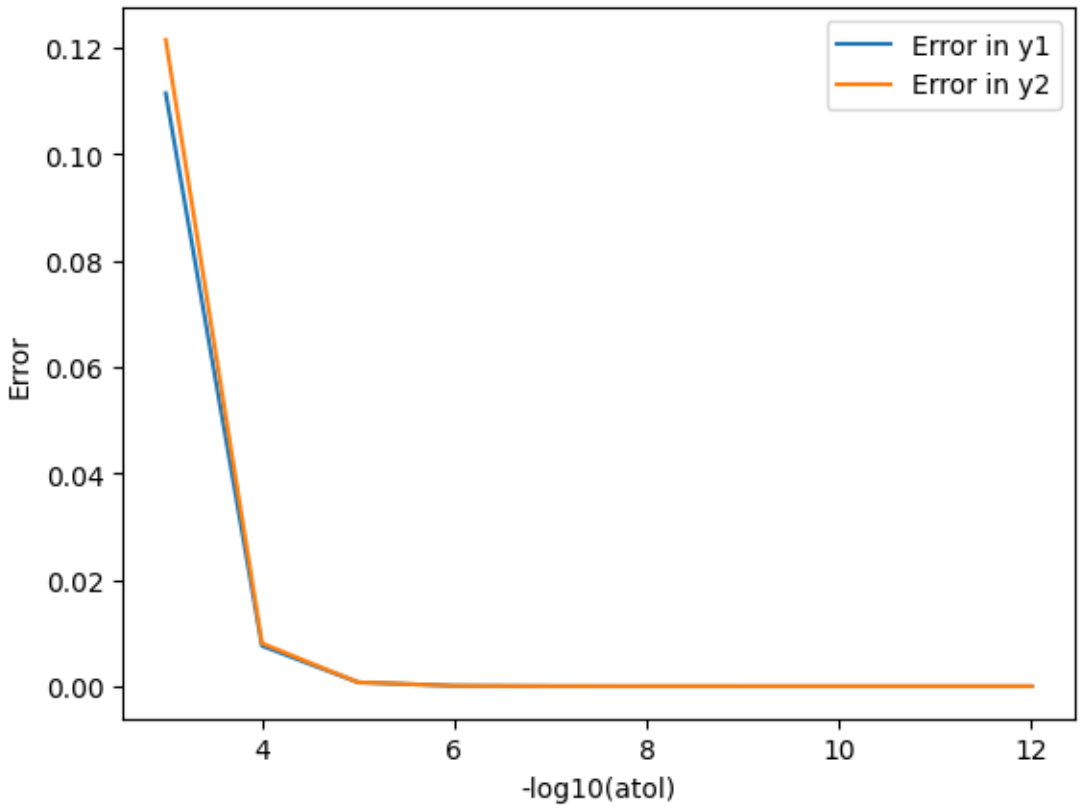
[12]:
```
<matplotlib.legend.Legend at 0x2adee793510>
```



1.3.4 Variation of both `atol` and `rtol`

[13]:
```
error1 = np.zeros_like(atol)
error2 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, atol=atol[i], rtol=rtol[i])
    y1 = 2*np.cos(4*sol.t) - 2.75*np.sin(4*sol.t)
    y2 = 3.25*np.sin(4*sol.t) + np.cos(4*sol.t)
    error1[i] = np.max(y1 - sol.y[0])
    error2[i] = np.max(y2 - sol.y[1])
plt.plot(-np.log10(atol), error1, label='Error in y1')
plt.plot(-np.log10(atol), error2, label='Error in y2')
plt.xlabel("-log10(atol)")
plt.ylabel('Error')
plt.legend()
```

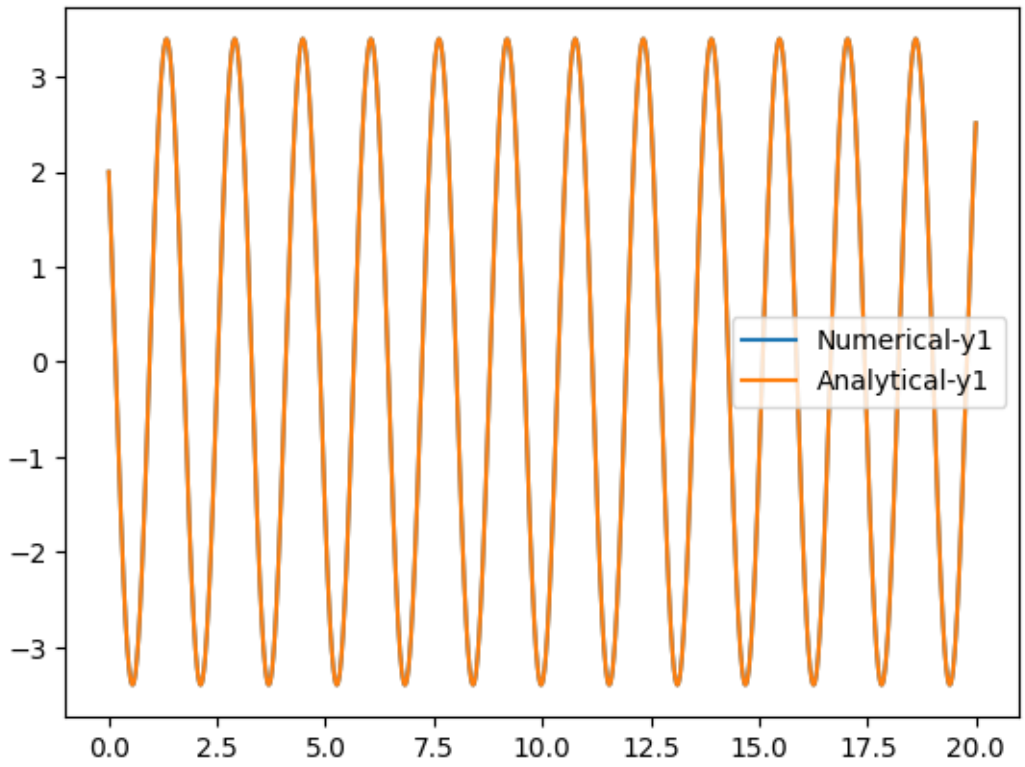
[13]:
```
<matplotlib.legend.Legend at 0x2adee4a3d90>
```



[14]:
```
### Solution with the values of `atol`=1e-12, `rtol`=1e-12
```

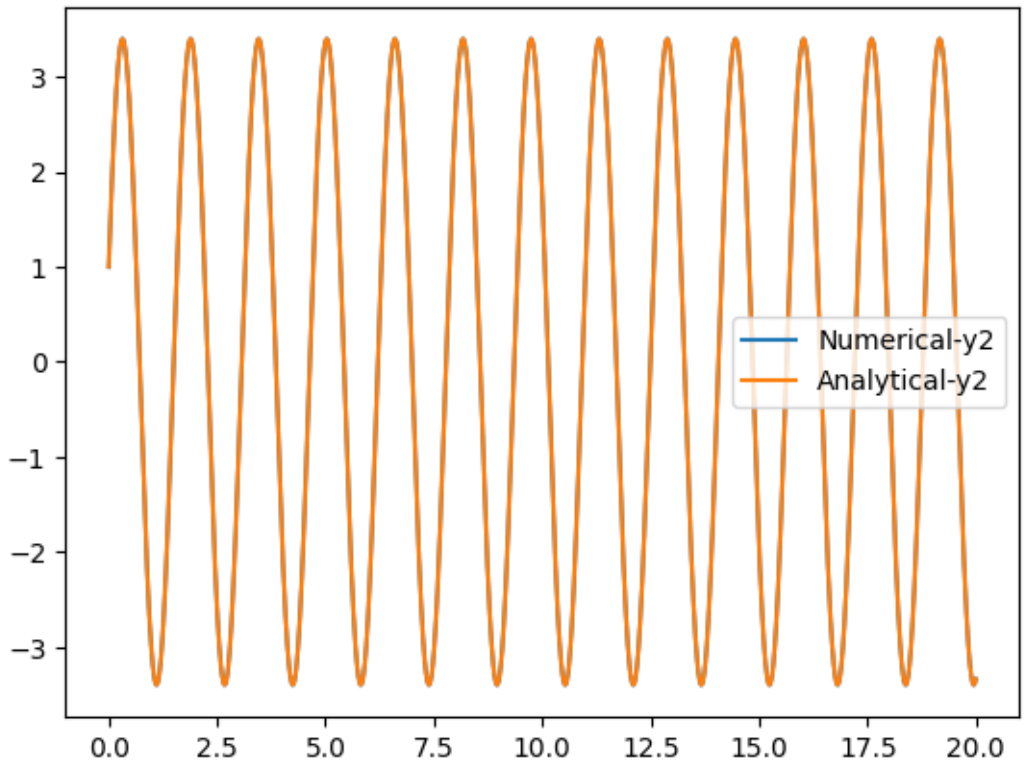
[15]:
```
sol = scpy.solve_ivp(yprime, t, y0, atol=1e-12, rtol=1e-12)
plt.plot(sol.t, sol.y[0], label='Numerical-y1')
plt.plot(sol.t, y1, label='Analytical-y1')
plt.legend()
```

[15]:
```
<matplotlib.legend.Legend at 0x2adee552410>
```



[16]:
```
plt.plot(sol.t, sol.y[1], label='Numerical-y2')
plt.plot(sol.t, y2, label='Analytical-y2')
plt.legend()
```

[16]: <matplotlib.legend.Legend at 0x2adee673dd0>



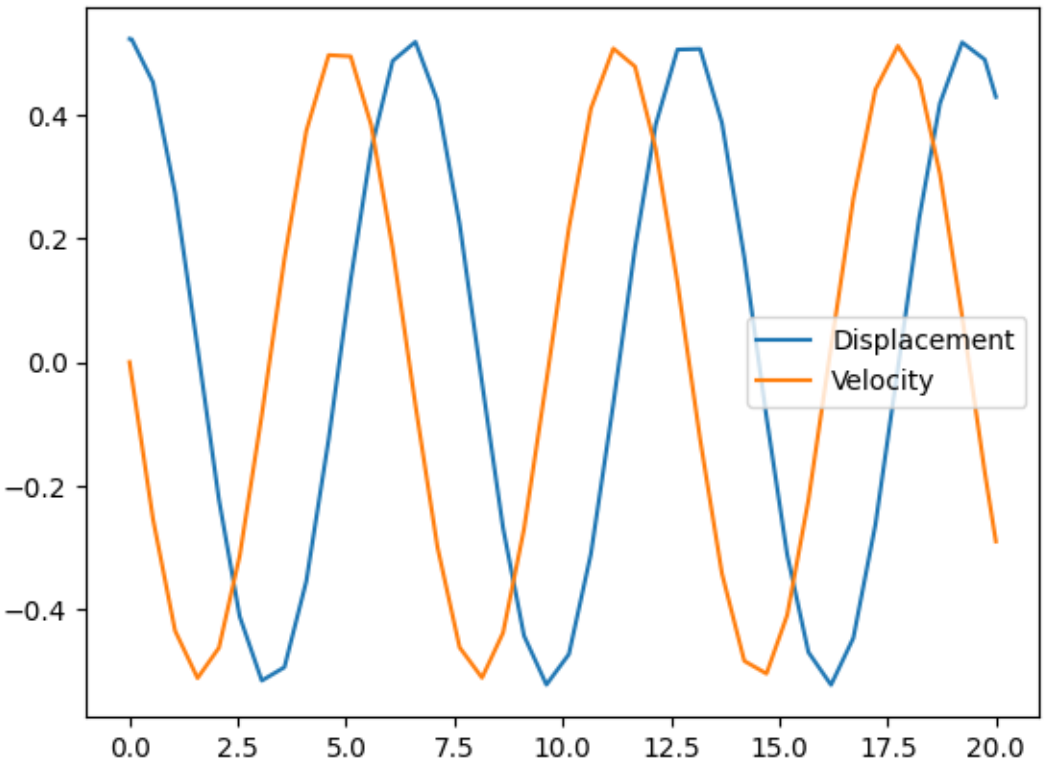
1.4 Problem 3: Simple Pendulum

[17]:
```
y0 = np.array([np.pi/6,0]) # stable position theta = 0
t = [0,20]
m, l, g, = 1, 10, 9.8
def yprime(t,y):
    return np.array([y[1], -g/l*np.sin(y[0])])
```

1.4.1 Solution with default values of atol=1e-5, rtol=1e-5

[18]:
```
sol = scpy.solve_ivp(yprime, t, y0, atol=1e-5, rtol=1e-5)
plt.plot(sol.t, sol.y[0], label='Displacement')
plt.plot(sol.t, sol.y[1], label='Velocity')
plt.legend()
```

[18]:
```
<matplotlib.legend.Legend at 0x2adee7e61d0>
```

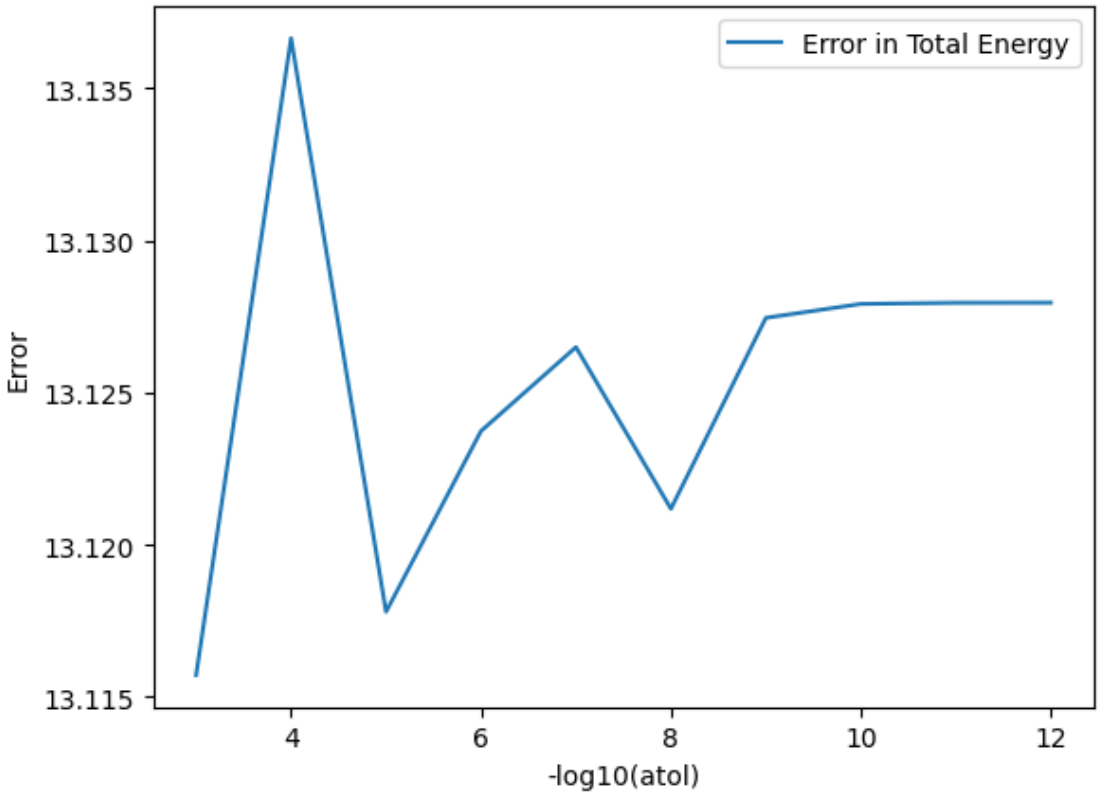


1.4.2 Variation of error with atol keeping rtol constant

[19]:
```
error1 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, atol=atol[i], rtol=1e-5)
    E = m*l*l*np.square(sol.y[1]) + m*g*l*(np.ones_like(sol.y[0]) - np.cos(sol.
 ↪y[0]))
```

```
    error1[i] = np.max(E)-np.min(E)
plt.plot(-np.log10(atol), error1, label='Error in Total Energy')
plt.xlabel("-log10(atol)")
plt.ylabel('Error')
plt.legend()
```

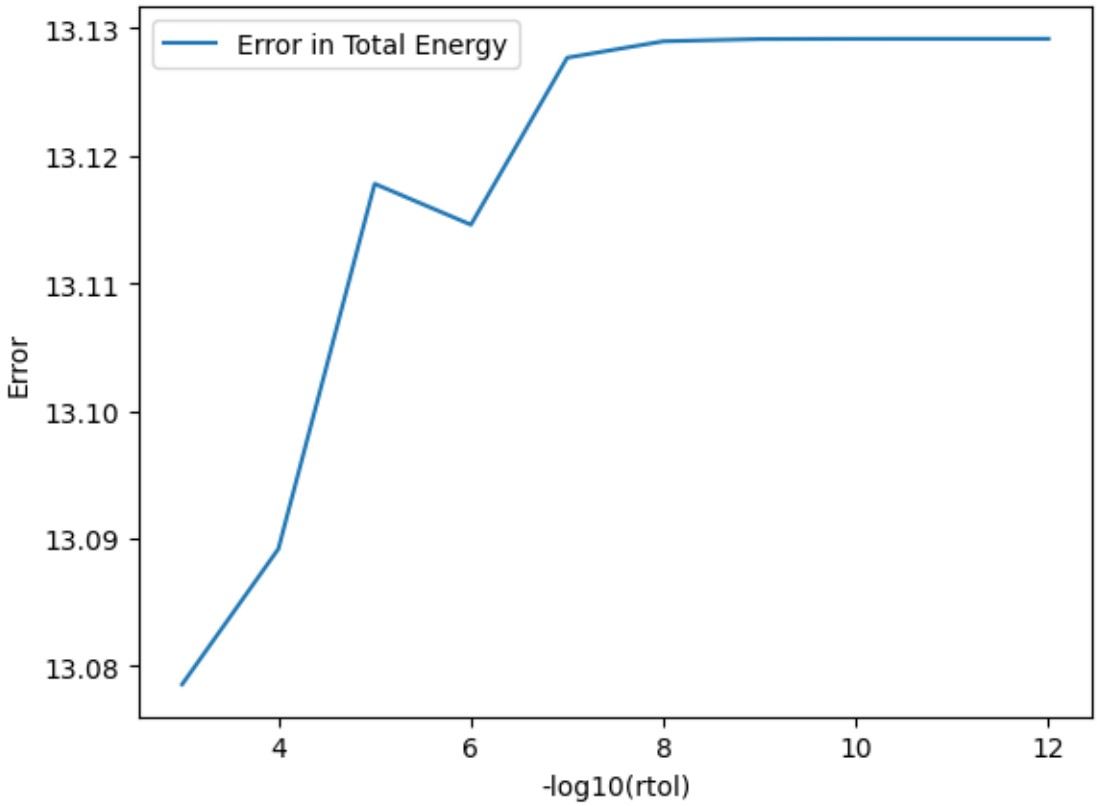
[19]: <matplotlib.legend.Legend at 0x2adef957210>



1.4.3 Variation of error with `rtol` keeping `atol` constant

[20]:
```
error1 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, rtol=rtol[i], atol=1e-5)
    E = m*l*l*np.square(sol.y[1]) + m*g*l*(np.ones_like(sol.y[0]) - np.cos(sol.
 ↪y[0]))
    error1[i] = np.max(E)-np.min(E)
plt.plot(-np.log10(atol), error1, label='Error in Total Energy')
plt.xlabel("-log10(rtol)")
plt.ylabel('Error')
plt.legend()
```

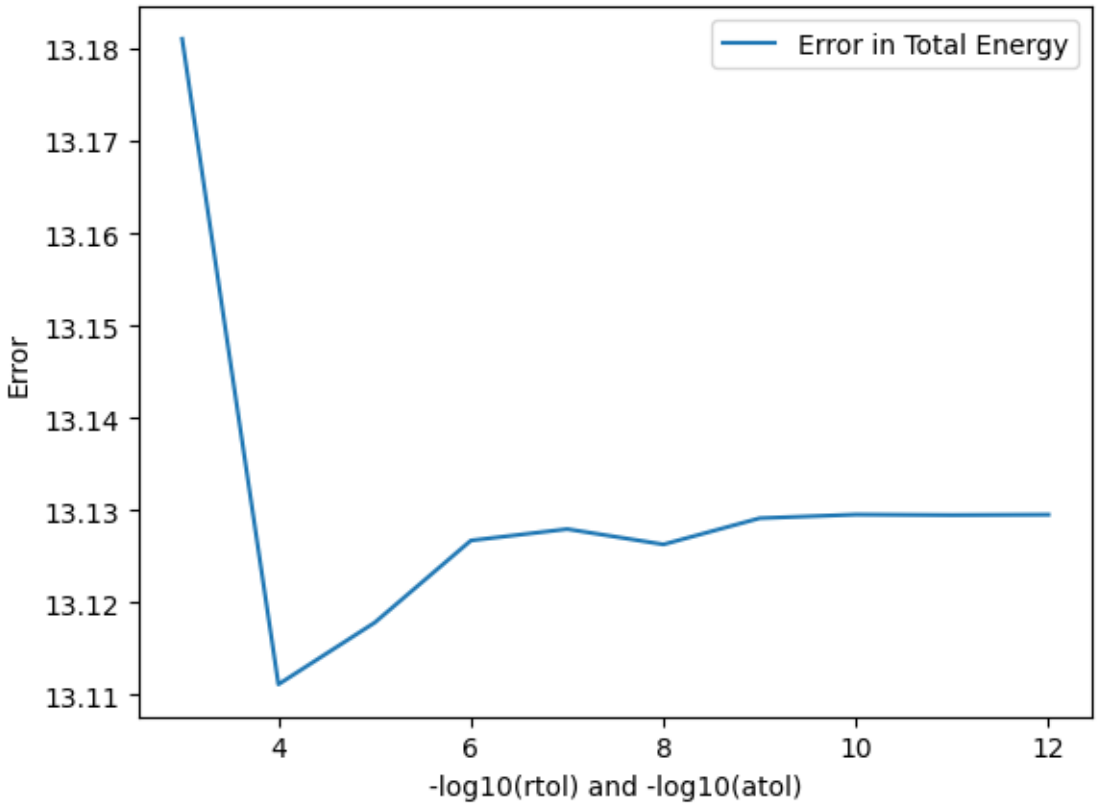
[20]: <matplotlib.legend.Legend at 0x2adee88bd90>



1.4.4 Variation of error with both `atol` and `rtol`

[21]:
```
error1 = np.zeros_like(atol)
for i in range(np.size(atol)):
    sol = scpy.solve_ivp(yprime, t, y0, rtol=rtol[i], atol=atol[i])
    E = m*l*l*np.square(sol.y[1]) + m*g*l*(np.ones_like(sol.y[0]) - np.cos(sol.
 ↪y[0]))
    error1[i] = np.max(E)-np.min(E)
plt.plot(-np.log10(atol), error1, label='Error in Total Energy')
plt.xlabel("-log10(rtol) and -log10(atol)")
plt.ylabel('Error')
plt.legend()
```

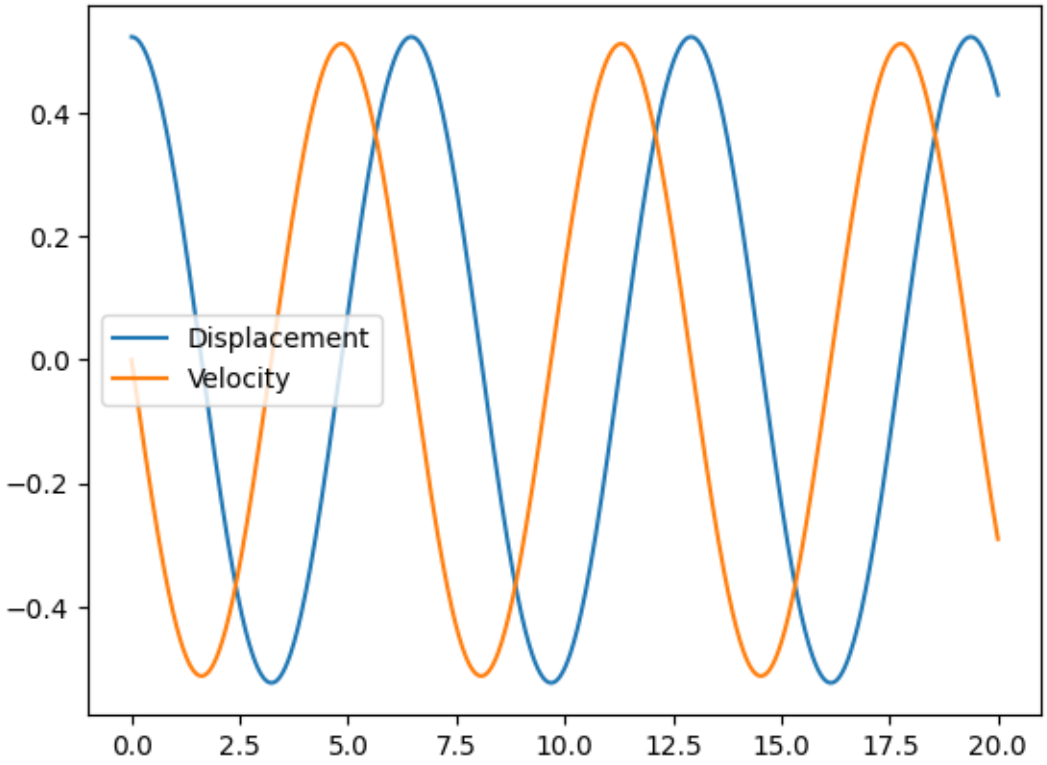
[21]:
```
<matplotlib.legend.Legend at 0x2adefa275d0>
```



1.4.5 Solution with default values of `atol`=1e-12, `rtol`=1e-12

[22]:
```
sol = scpy.solve_ivp(yprime, t, y0, atol=1e-12, rtol=1e-12)
plt.plot(sol.t, sol.y[0], label='Displacement')
plt.plot(sol.t, sol.y[1], label='Velocity')
plt.legend()
```

[22]:
```
<matplotlib.legend.Legend at 0x2adefa5ba10>
```


Displacement
Velocity
0.4
0.2
0.0
−0.2
−0.4
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0