In [37]:
```
import sympy as sp


t = sp.Symbol('t')
theta = sp.Function('theta')(t)
g, l = sp.symbols('g l')


equation = sp.Eq(theta.diff(t, t) + (g / l) * sp.sin(theta), 0)


equation
```

Out[37]: $\displaystyle \frac{g \sin{\left(\theta{\left(t \right)} \right)}}{l} + \frac{d^{2}}{d t^{2}} \theta{\left(t \right)} = 0$

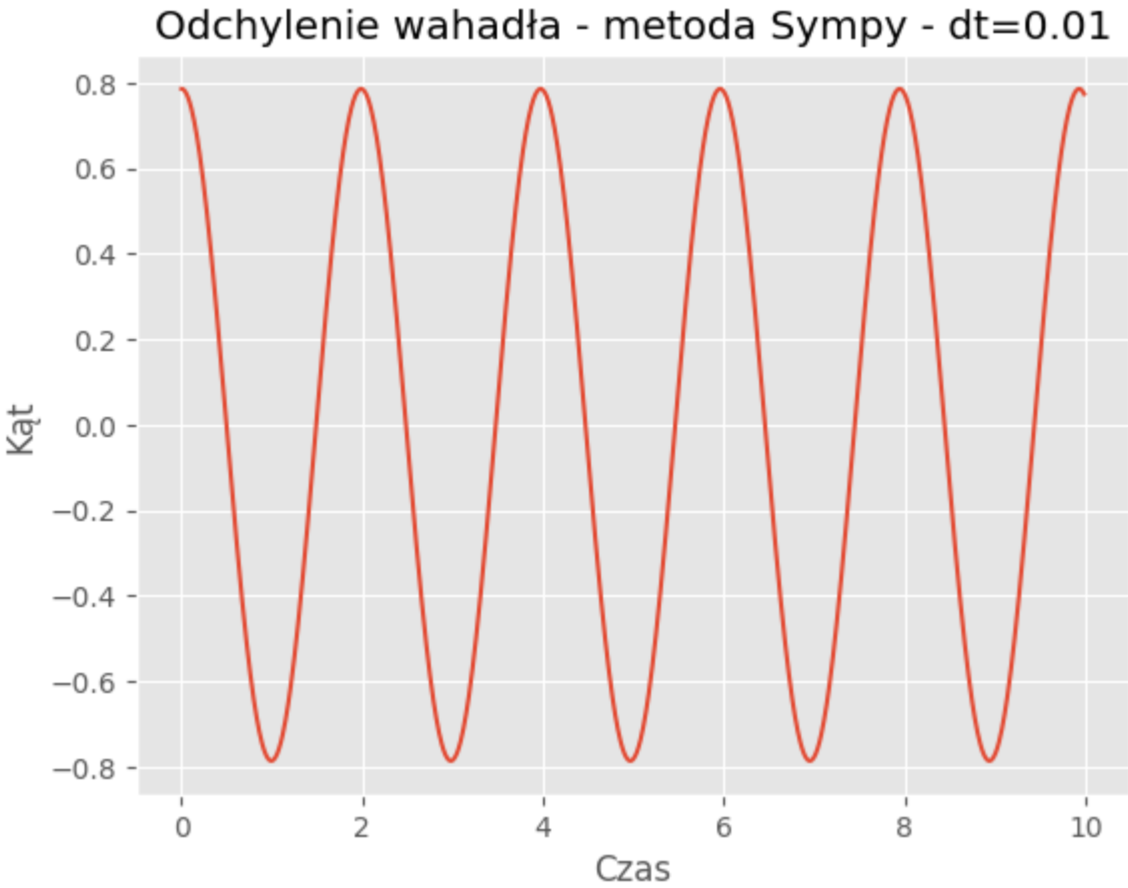
In [38]:
```
t = sp.Symbol('t')  # время
theta = sp.Function('theta')(t)
g, l = sp.symbols('g l')


linear_equation = sp.Eq(theta.diff(t, t) + (g / l) * theta, 0)

solution_linear = sp.dsolve(linear_equation, theta)

solution_linear
```

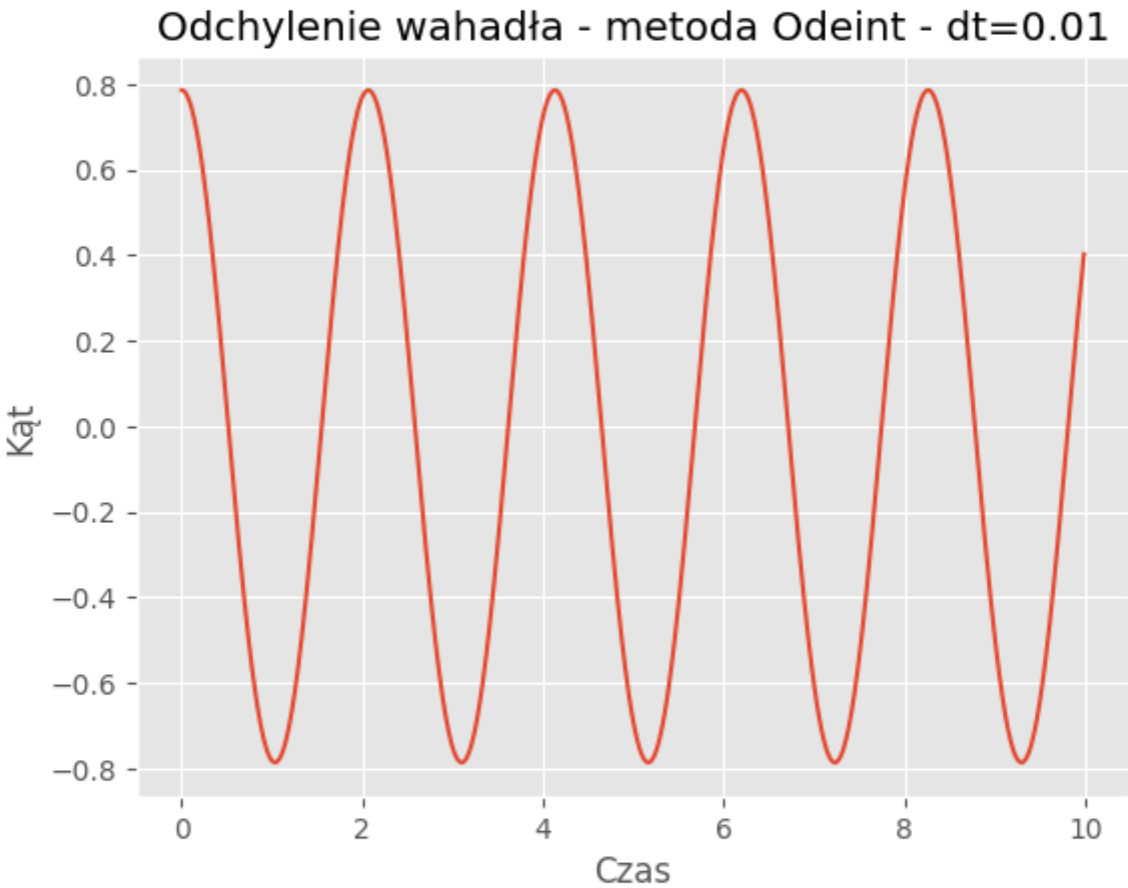
Out[38]: $\displaystyle \theta{\left(t \right)} = C_{1} e^{- t \sqrt{- \frac{g}{l}}} + C_{2} e^{t \sqrt{- \frac{g}{l}}}$

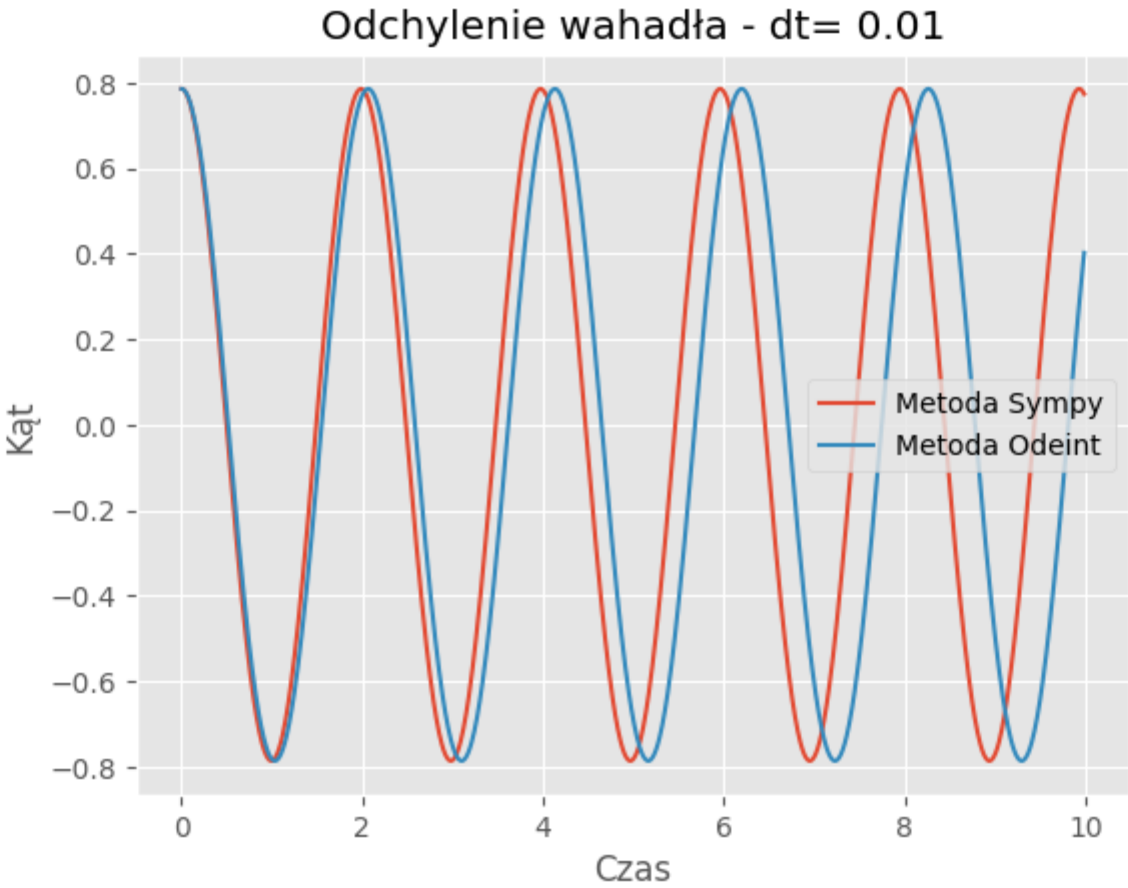
In [138…
```
import numpy as np
import matplotlib.pyplot as plt
from sympy import Function, dsolve, Eq, symbols, init_printing, lambdify
from scipy.integrate import odeint


plt.style.use('ggplot')
init_printing(use_latex=True)

# sympy
t = symbols('t')
x = Function('x')
k = 10
eq = Eq(x(t).diff(t, 2), -k * x(t))

sol = dsolve(eq, ics={x(0): np.pi / 4, x(t).diff(t).subs(t, 0): 0})

theta_numeric = lambdify(t, sol.rhs, modules=['numpy'])

#odeint
def rownaniew(theta, t, k):
        return [theta[1], -k * np.sin(theta[0])]

theta0 = [np.pi / 4, 0.0]
k = 10

t_dt = [0.01, 0.05, 0.1, 0.5, 1, 1.125, 1.25, 1.5, 1.75, 1.9, 2]
```

```
# Błędy


for dt in t_dt:
    t = np.arange(0, 10, dt)
    x = theta_numeric(t)

    #sympy
    plt.plot(t, x)
    plt.xlabel('Czas')
    plt.ylabel('Kąt')
    title = 'Odchylenie wahadła - metoda Sympy - dt=' + str(dt)
    plt.title(title)
    plt.grid(True)
    plt.show()

    # odeint
    sol = odeint(rownaniew, theta0, t, args=(k,))
    theta_vals = sol[:, 0]

    plt.plot(t, theta_vals)
    plt.xlabel('Czas')
    plt.ylabel('Kąt')
    title = 'Odchylenie wahadła - metoda Odeint - dt=' + str(dt)
    plt.title(title)
    plt.grid(True)
    plt.show()

    plt.plot(t, x, label='Metoda Sympy')
    plt.plot(t, theta_vals, label='Metoda Odeint')
    plt.xlabel('Czas')
    plt.ylabel('Kąt')
    title = 'Odchylenie wahadła - dt= ' + str(dt)
    plt.title(title)
    plt.grid(True)
    plt.legend()
    plt.show()
    mean_absolute_error = np.mean(np.abs(odeint_solution - sympy_solution))
    mean_squared_error = np.mean((odeint_solution - sympy_solution) ** 2)


    print(f"For dt = {dt}:")
    print(f"Mean Absolute Error: {mean_absolute_error:.4f}")
    print(f"Mean Squared Error: {mean_squared_error:.4f}")



    t = np.arange(0, 10, dt)
    sympy_solution = theta_numeric(t)

    sol = odeint(rownaniew, theta0, t, args=(k,))
    odeint_solution = sol[:, 0]


mean_absolute_error = []
mean_squared_error = []
t_dt_errors= np.arange(0.01, 1, 0.01)
for dt in t_dt_errors:
    t = np.arange(0, 10, dt)
```

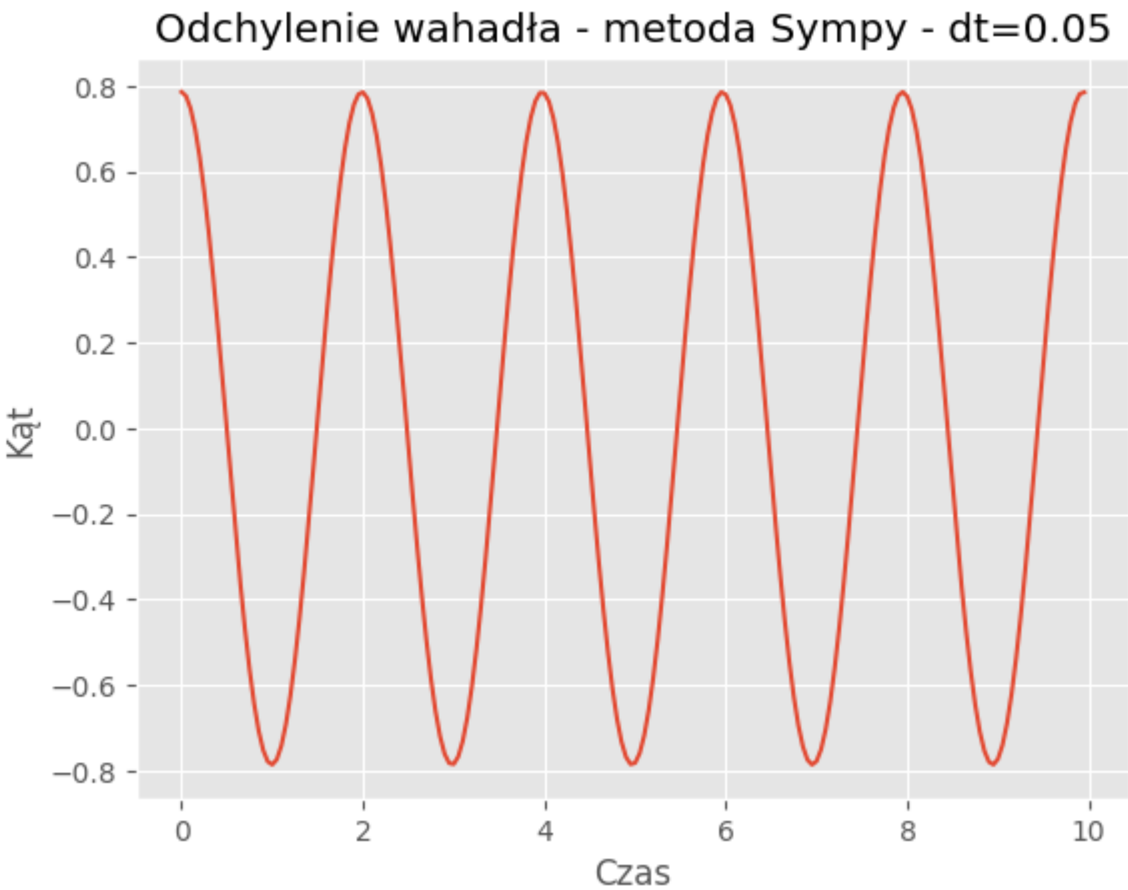
```
    sol = odeint(rownaniew, theta0, t, args=(k,))
    x = theta_numeric(t)


    abs_error = np.abs(sol[:, 0] - x)  # Calculate absolute error for the first
    sq_error = (sol[:, 0] - x) ** 2    # Calculate squared error for the first c

    mean_absolute_error.append(np.mean(abs_error))
    mean_squared_error.append(np.mean(sq_error))

plt.figure(figsize=(10, 6))
plt.plot(mean_absolute_error, label='MAE', marker='o', color='blue')
plt.xlabel('Epoch')
plt.ylabel('Mean Absolute Error')
plt.title('Mean Absolute Error')
plt.legend()
plt.grid(True)
plt.show()

# Создание графика для Mean Squared Error
plt.figure(figsize=(10, 6))
plt.plot(mean_squared_error, label='MSE', marker='s', color='green')
plt.xlabel('Epoch')
plt.ylabel('Mean Squared Error')
plt.title('Mean Squared Error')
plt.legend()
plt.grid(True)
plt.show()
```

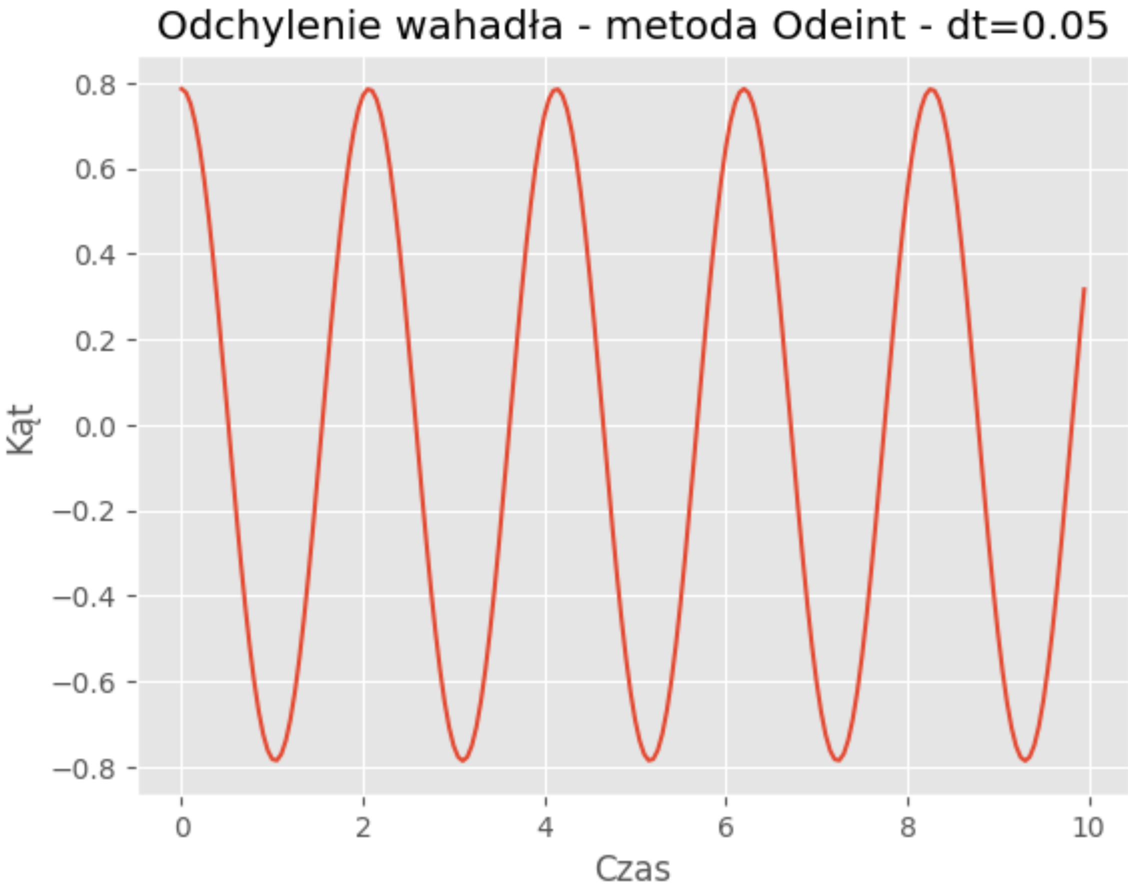

Odchylenie wahadła - metoda Sympy - dt=0.01
Kąt
Czas
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
2
4
6
8
10

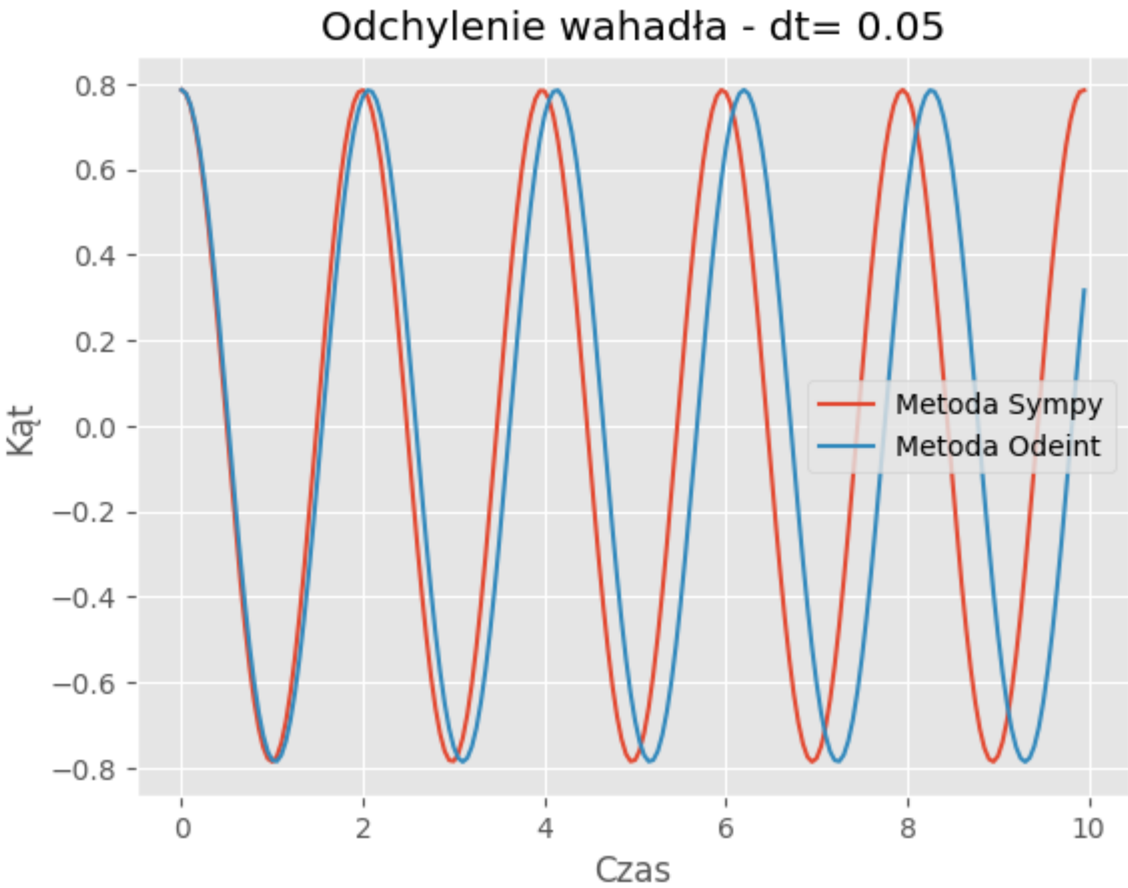

Odchylenie wahadła - metoda Odeint - dt=0.01
Kąt
Czas
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
2
4
6
8
10



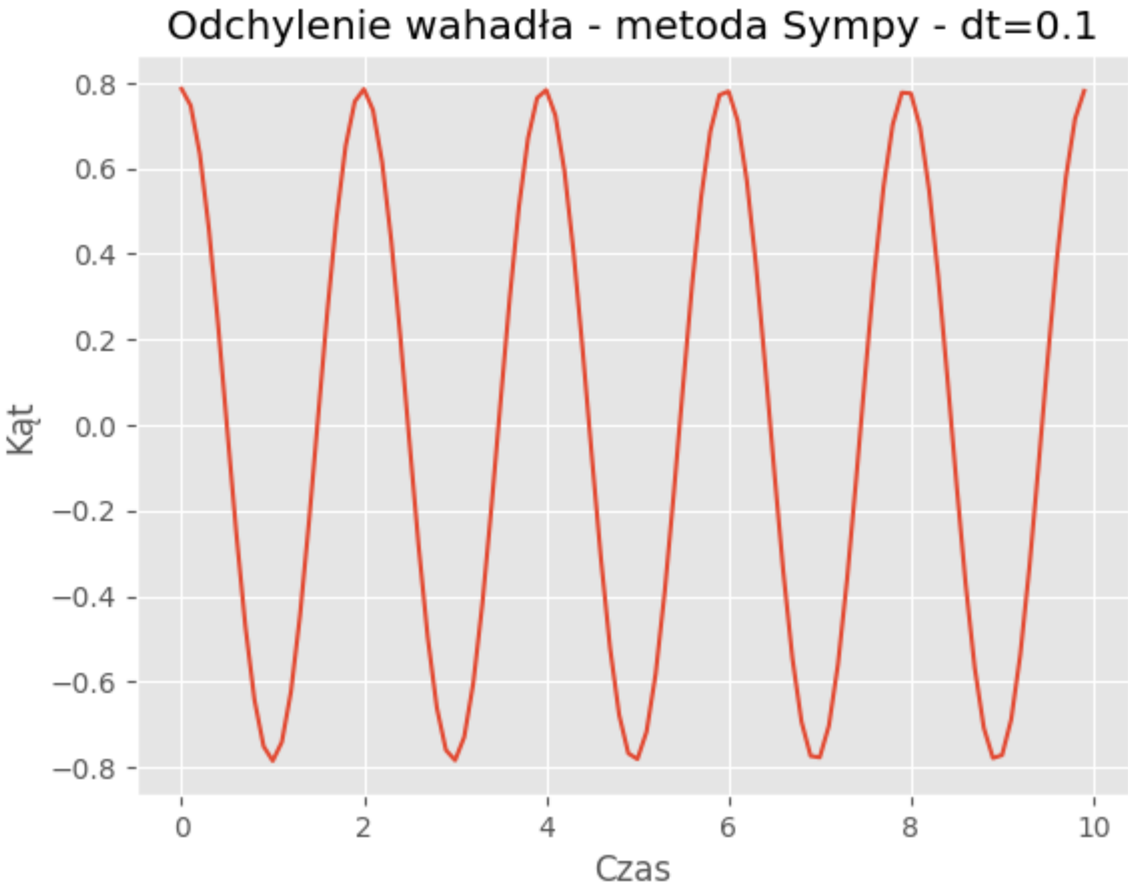

```
For dt = 0.01:
Mean Absolute Error: 0.0864
Mean Squared Error: 0.0145
```

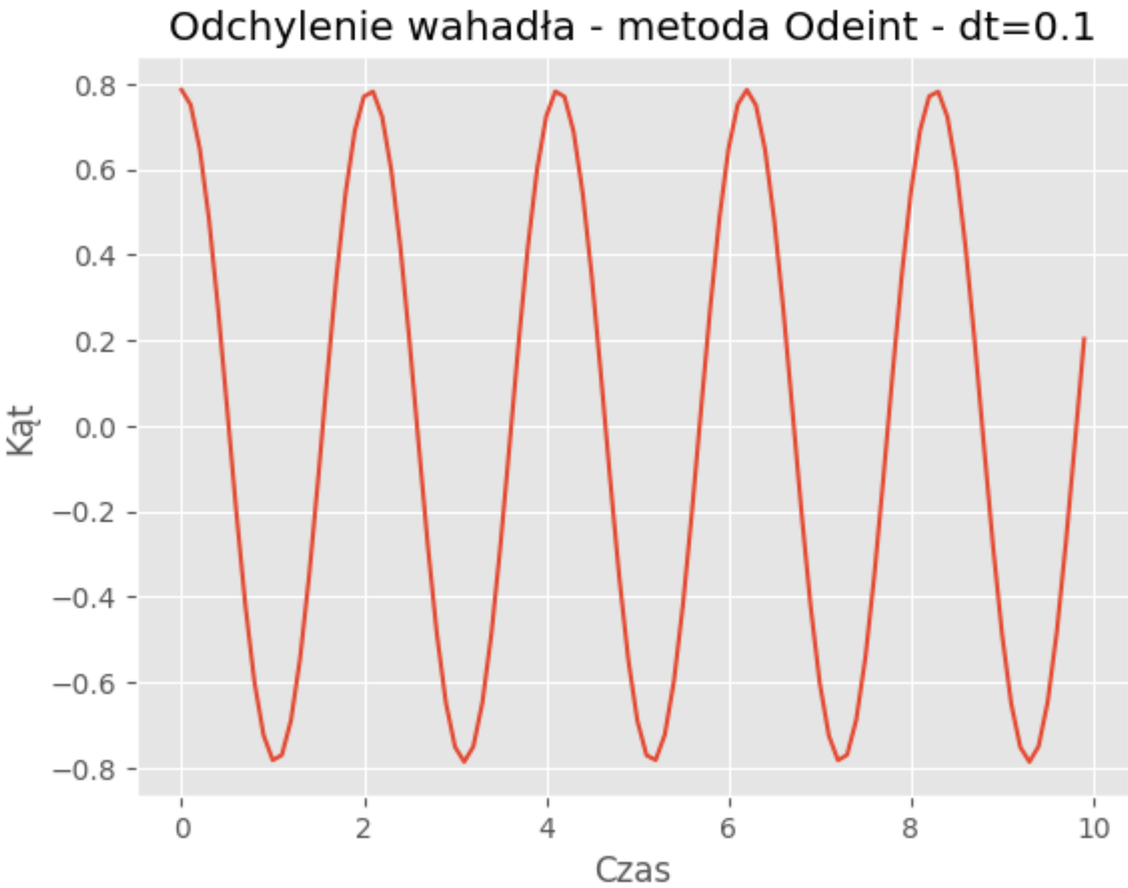


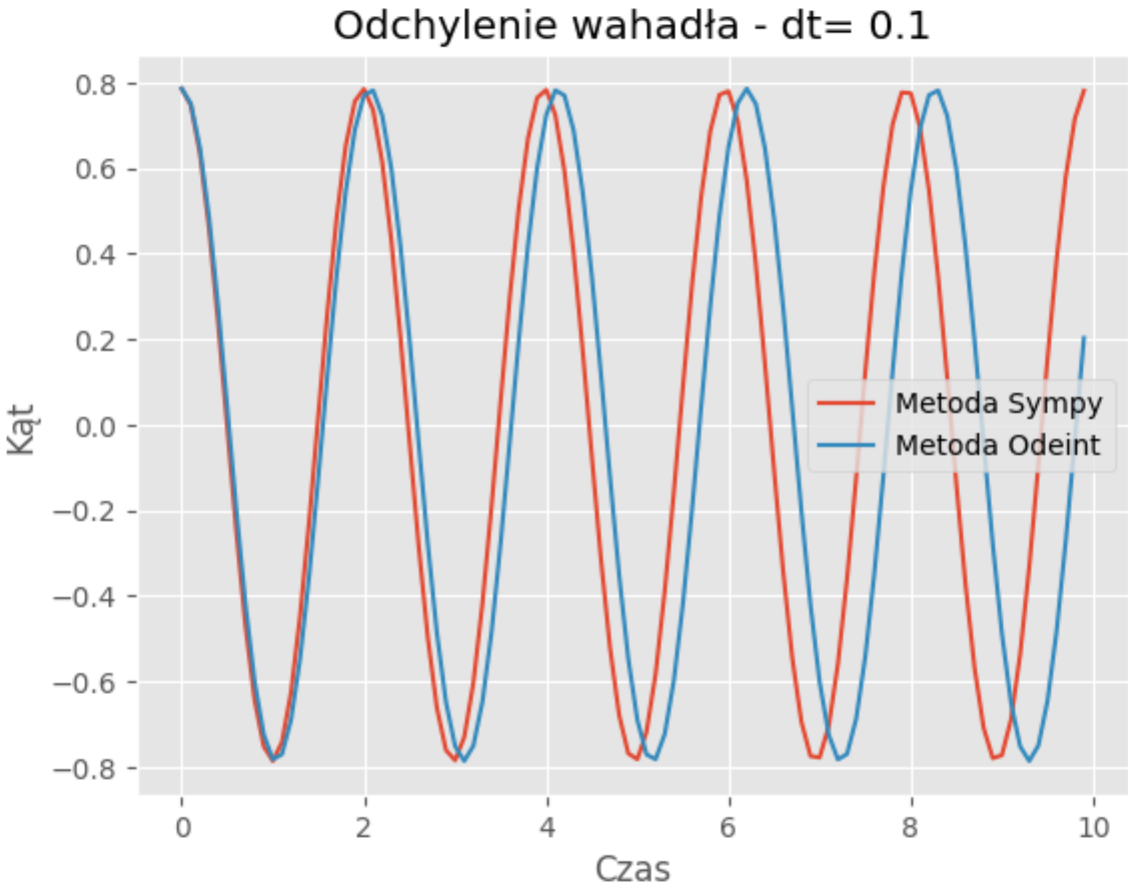




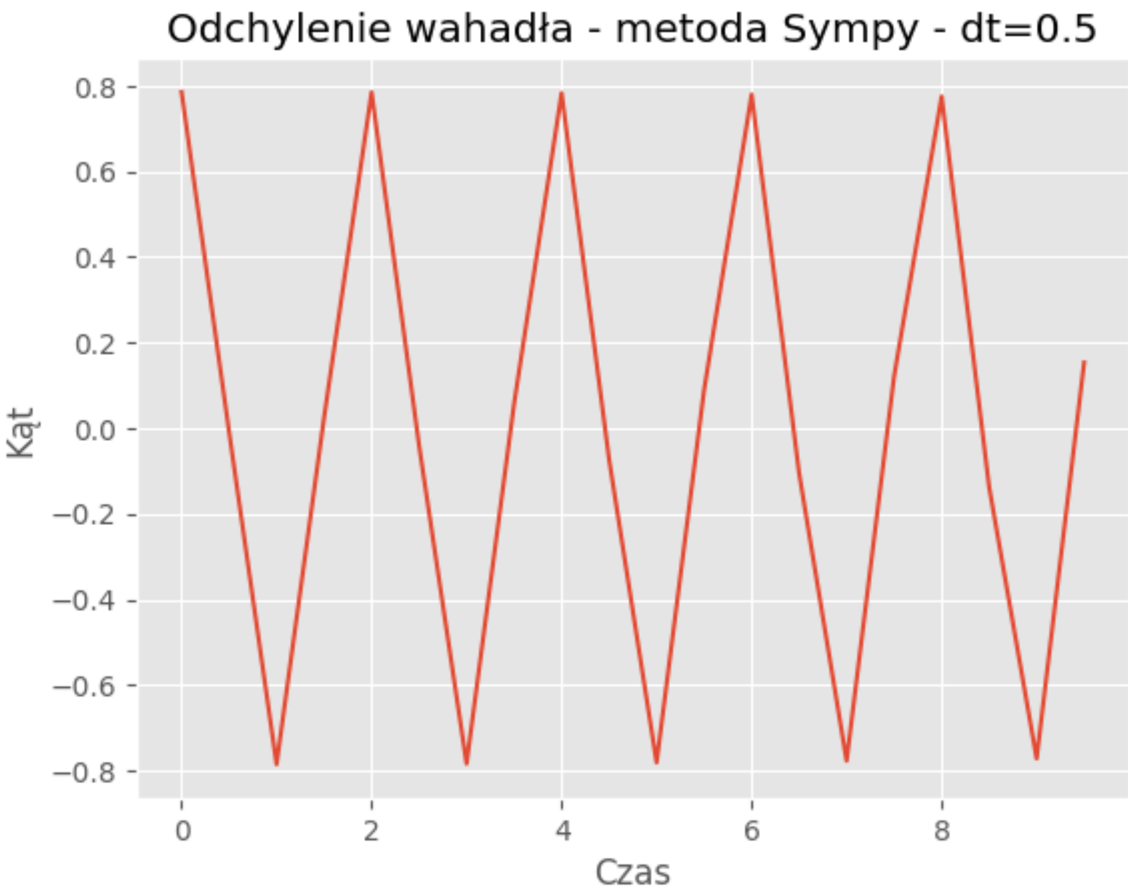

```
For dt = 0.05:
Mean Absolute Error: 0.2998
Mean Squared Error: 0.1460
```

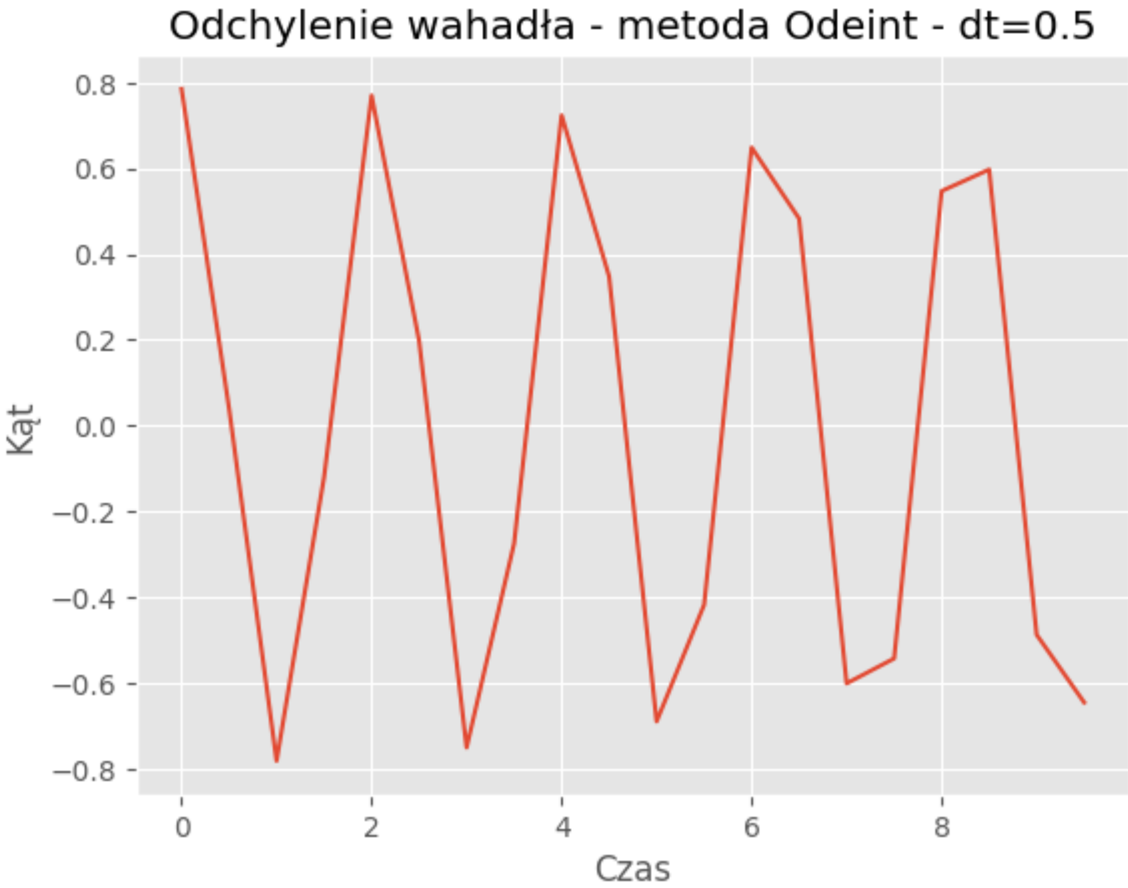

Odchylenie wahadła - metoda Sympy - dt=0.1
Kąt
Czas
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
2
4
6
8
10

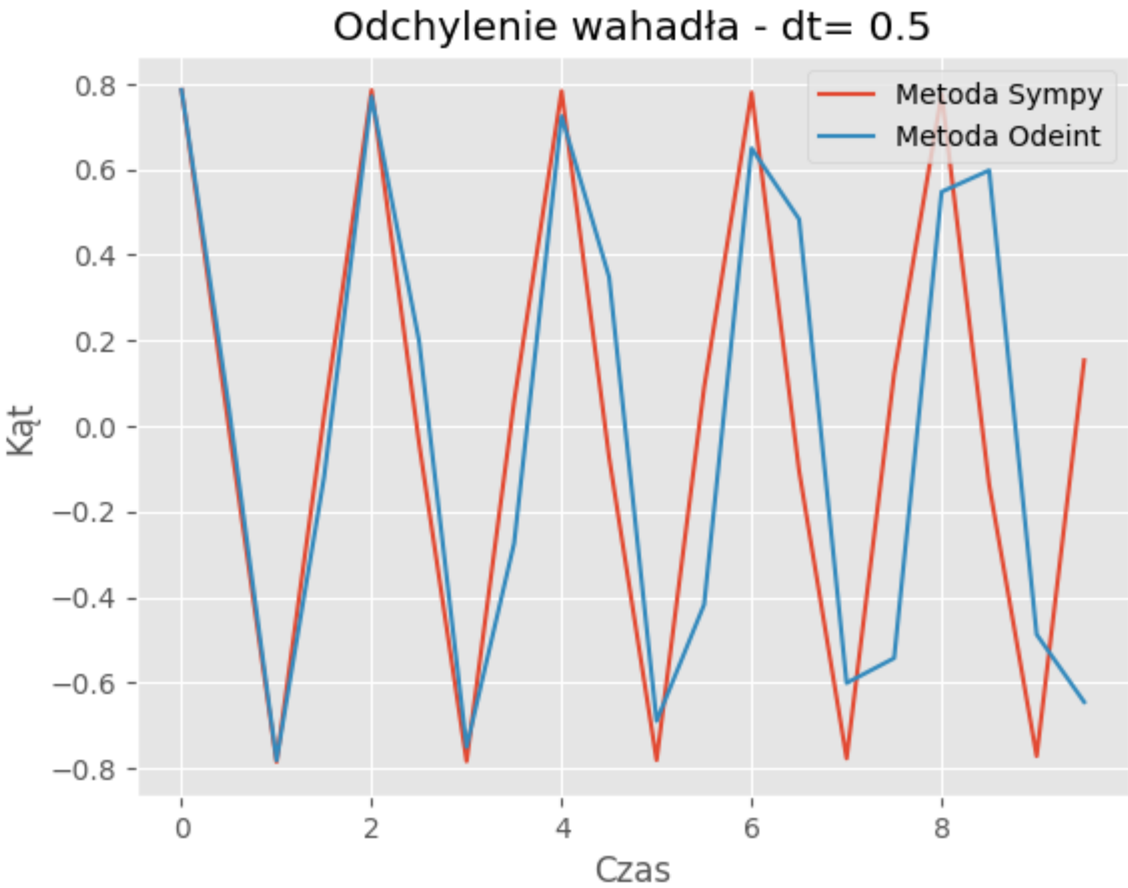

Odchylenie wahadła - metoda Odeint - dt=0.1
Kąt
Czas
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
2
4
6
8
10



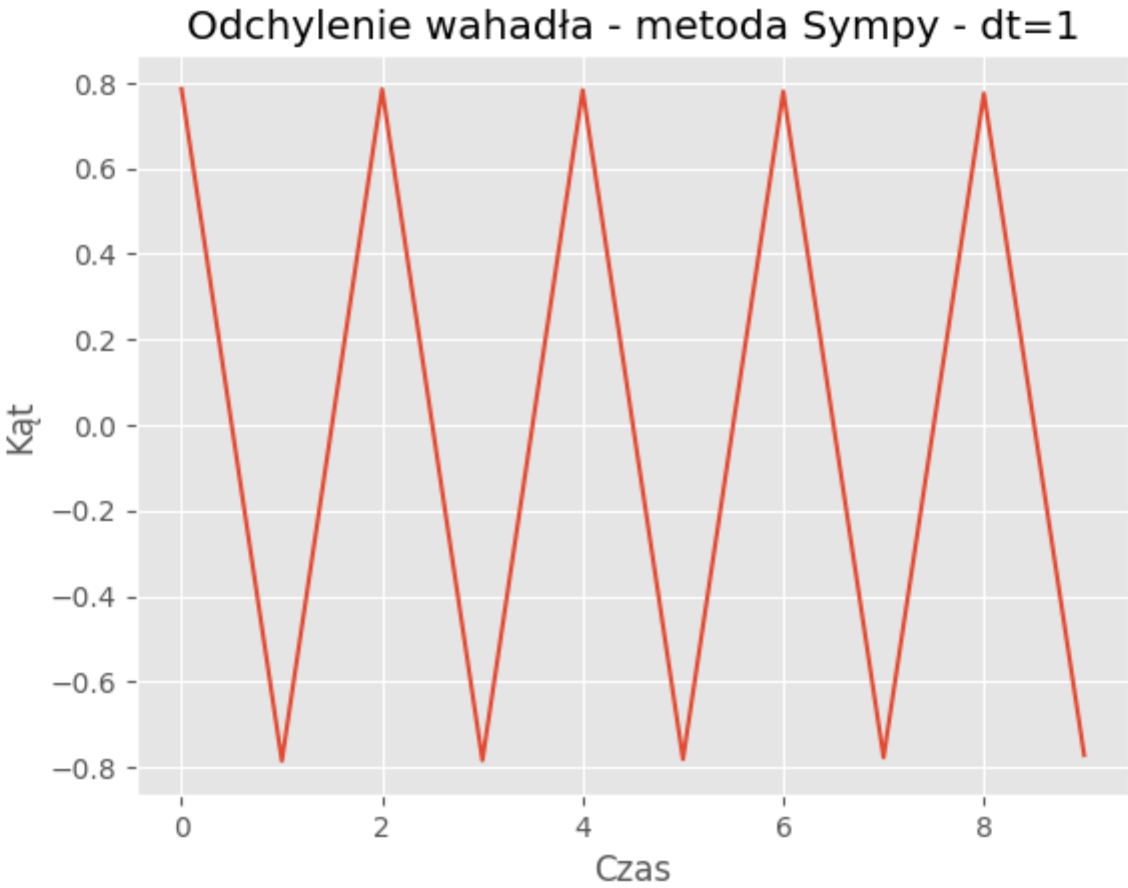

```
For dt = 0.1:
Mean Absolute Error: 0.2990
Mean Squared Error: 0.1457
```

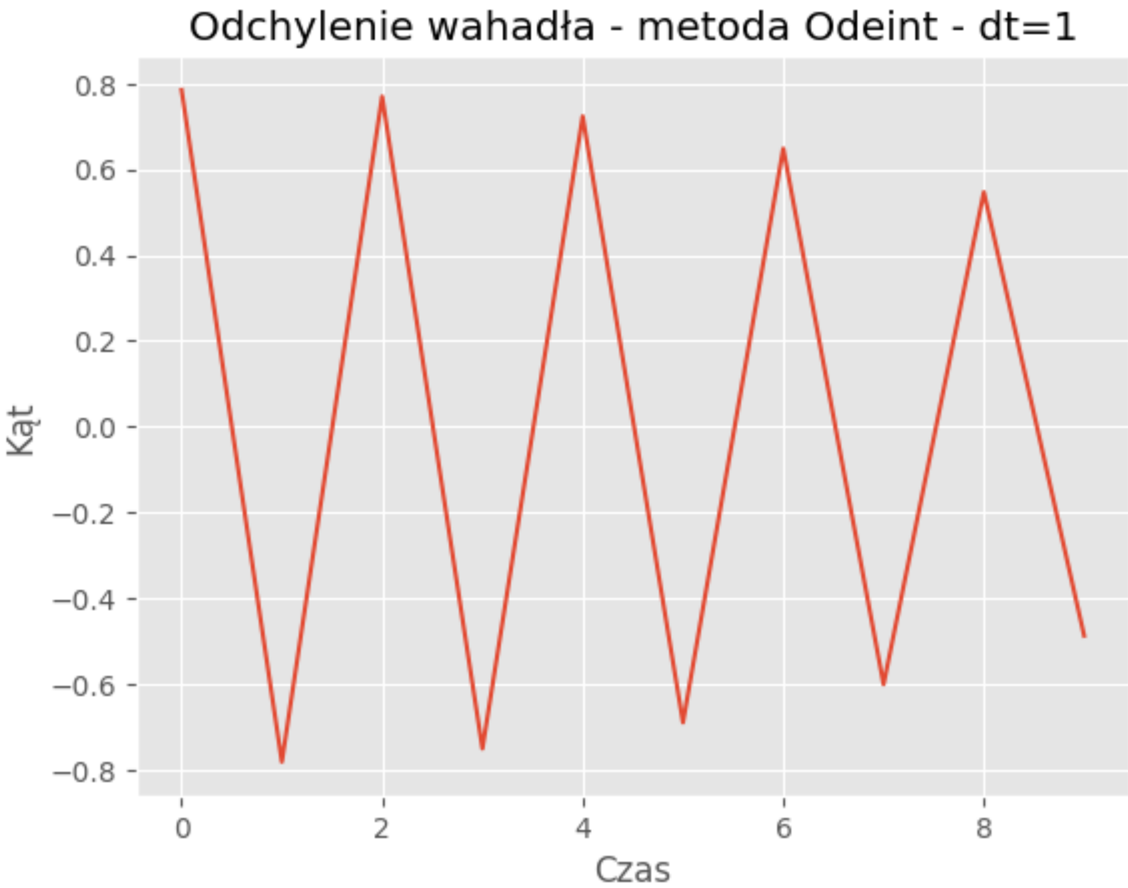


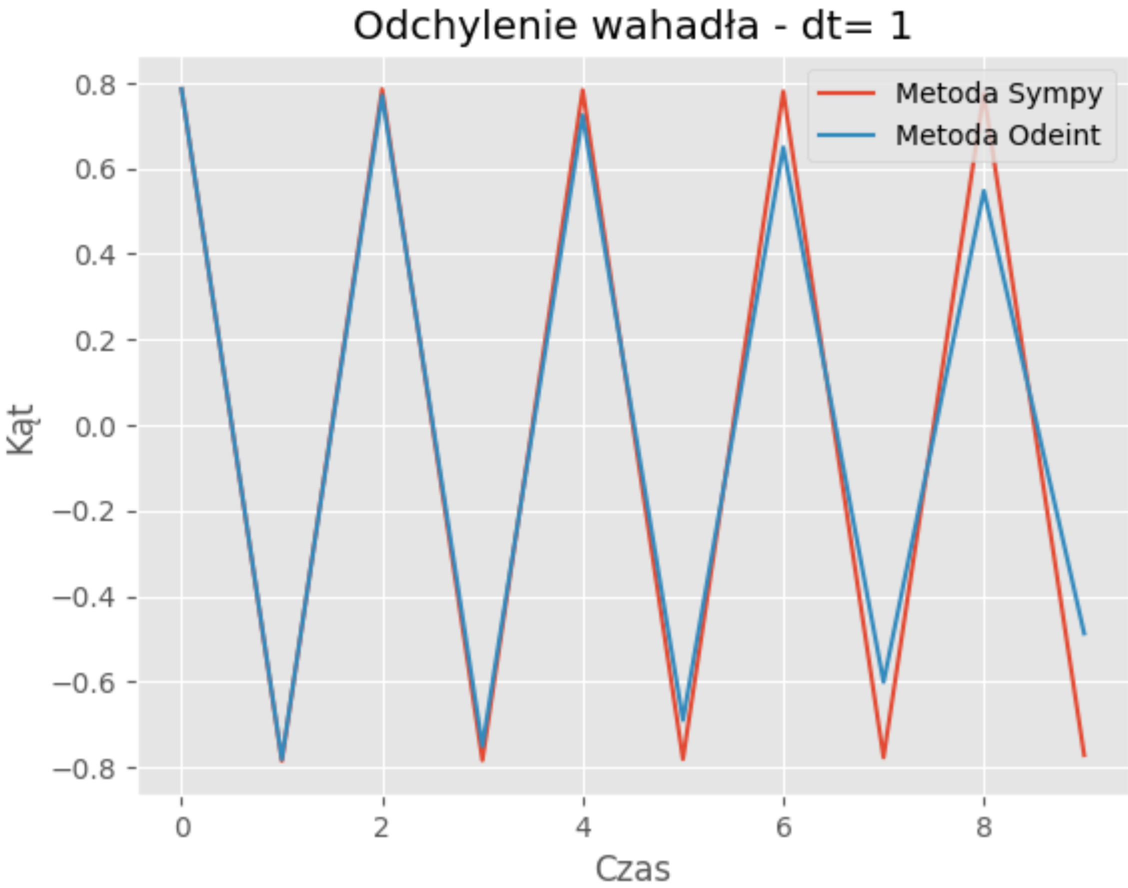




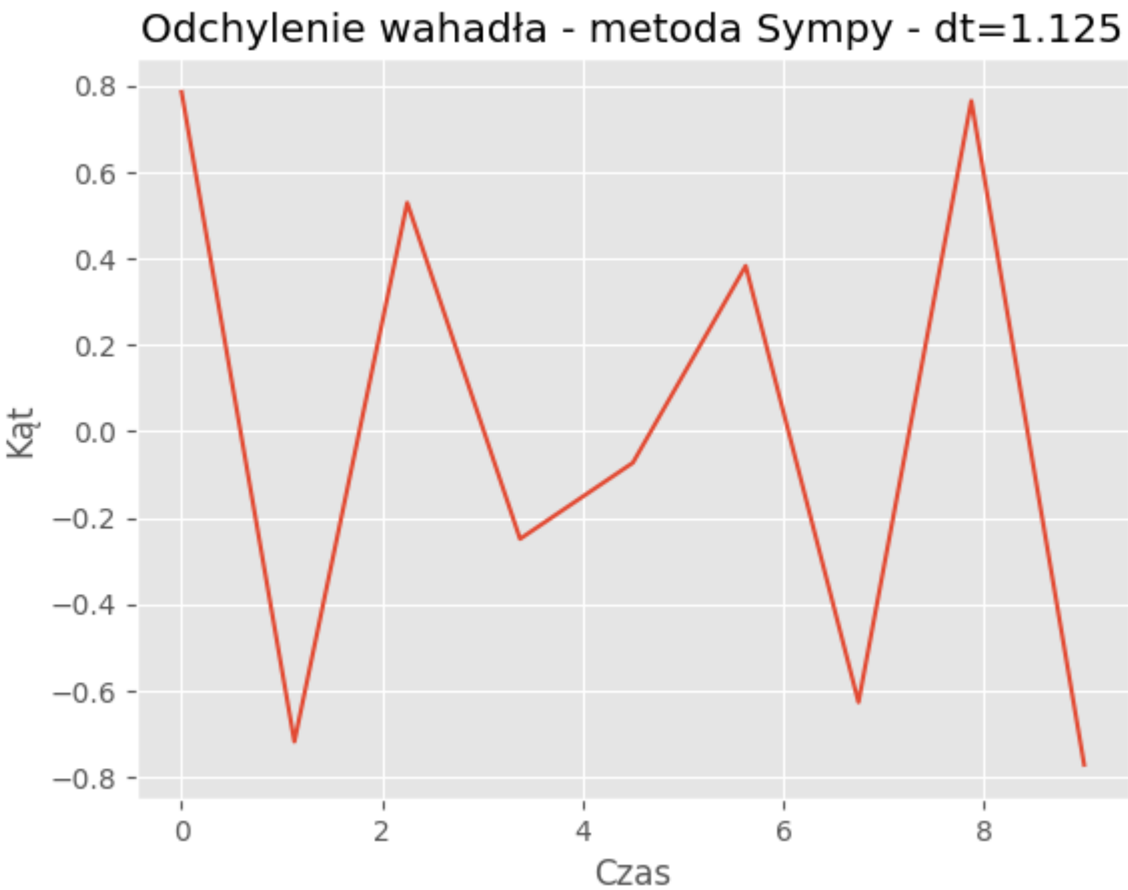

```
For dt = 0.5:
Mean Absolute Error: 0.2977
Mean Squared Error: 0.1453
```

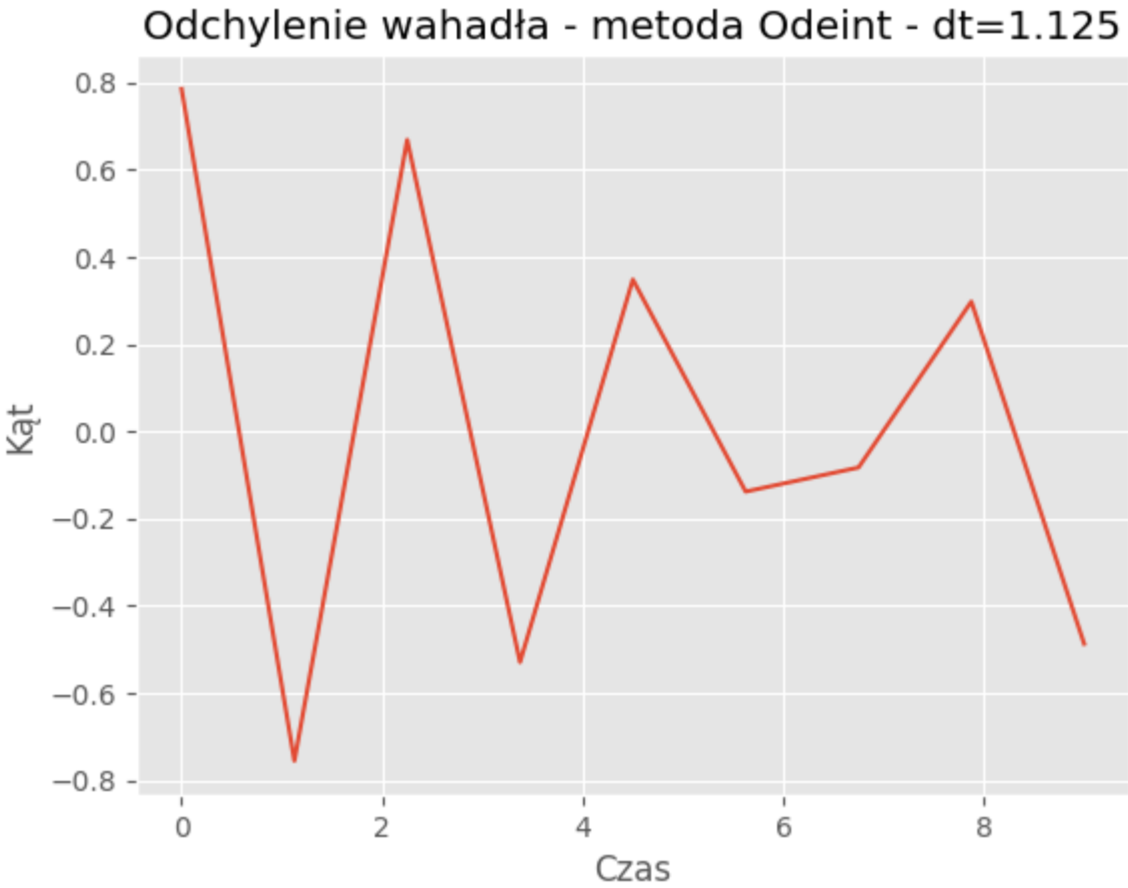

Odchylenie wahadła - metoda Sympy - dt=1
Kąt
Czas
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
2
4
6
8

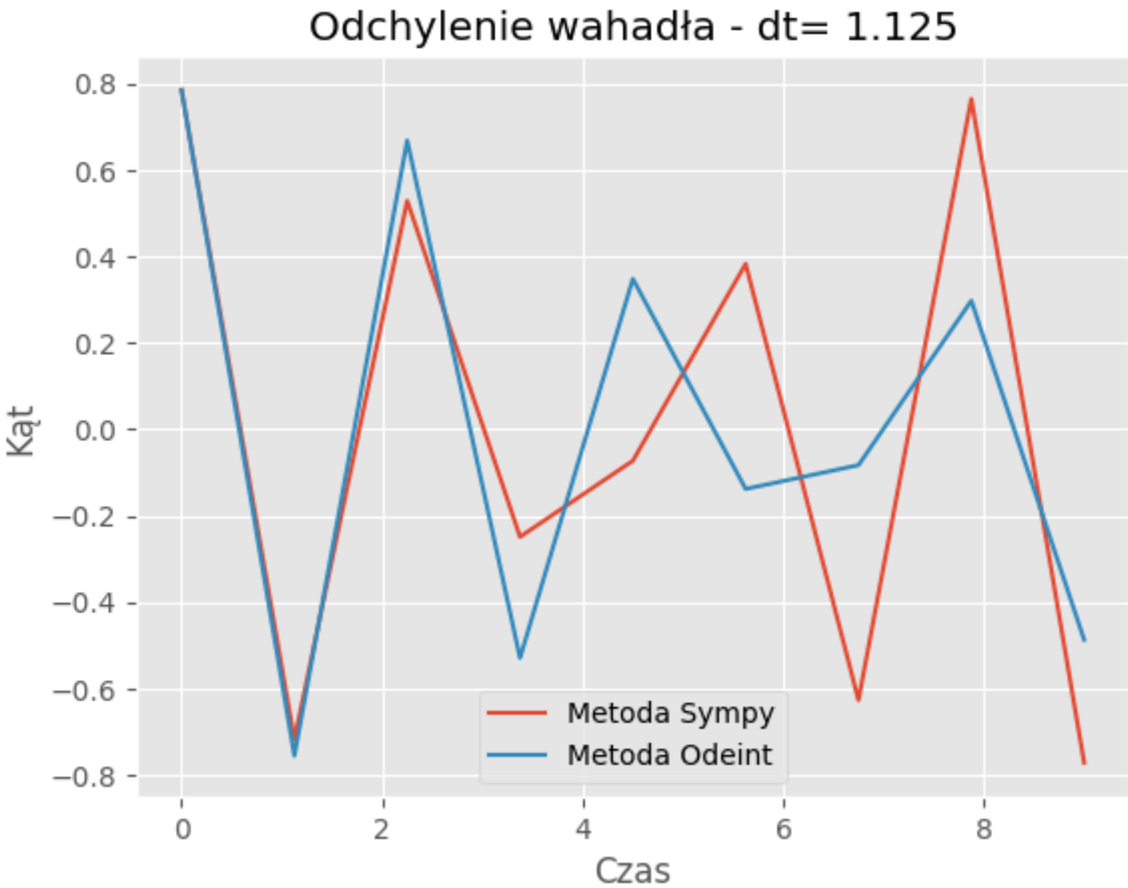

Odchylenie wahadła - metoda Odeint - dt=1
Kąt
Czas
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
2
4
6
8



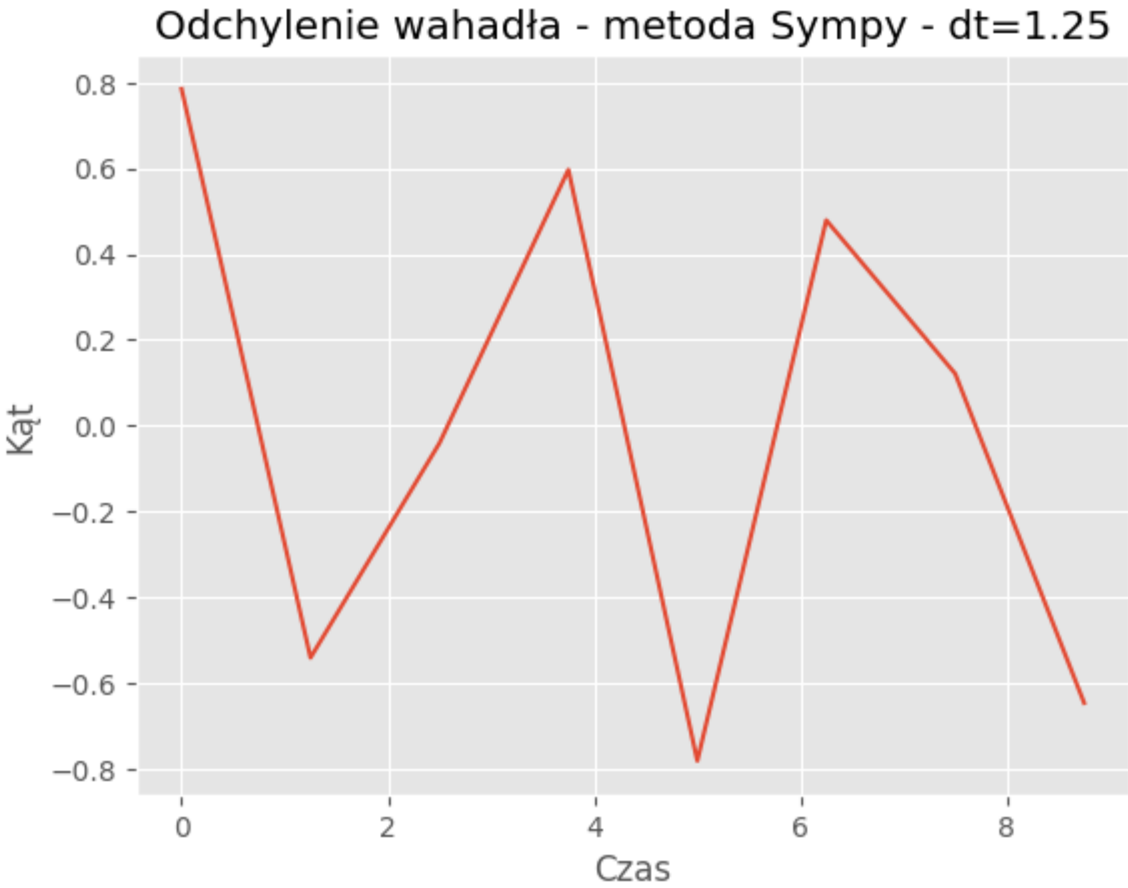

```
For dt = 1:
Mean Absolute Error: 0.2749
Mean Squared Error: 0.1392
```

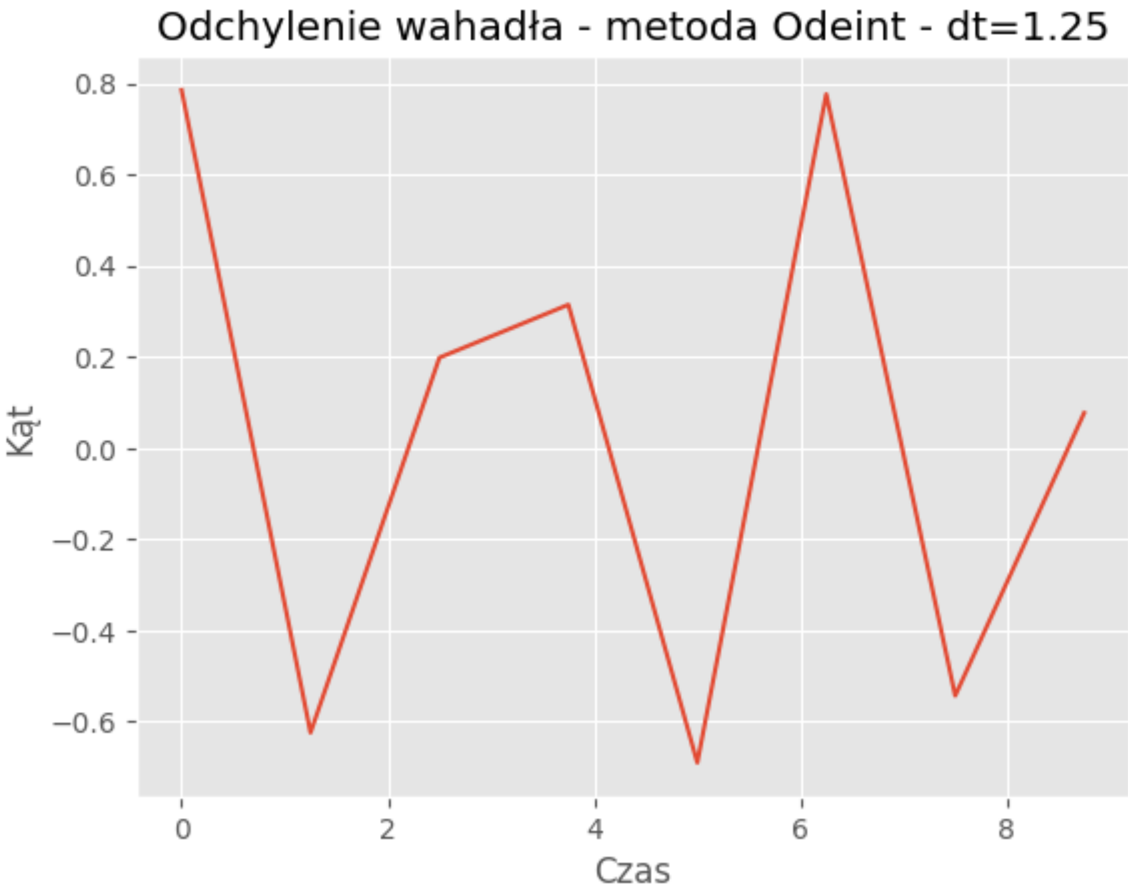


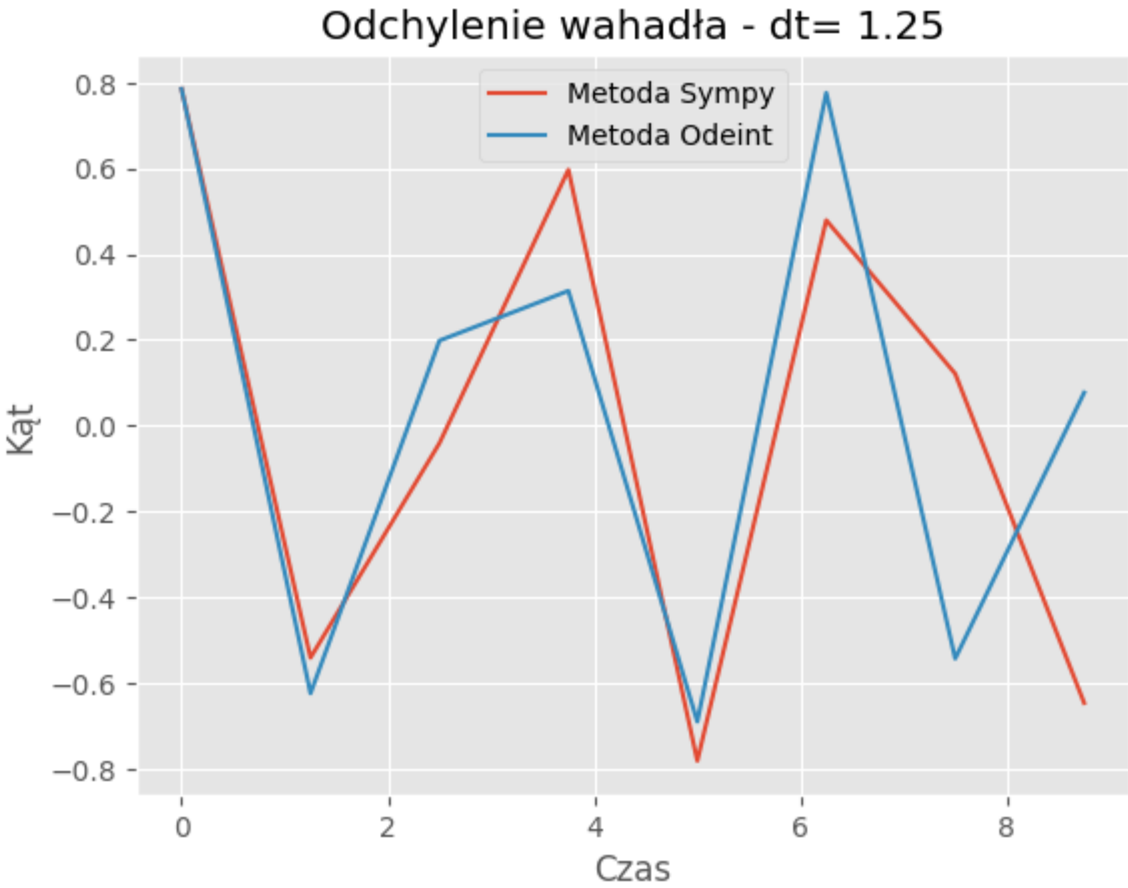




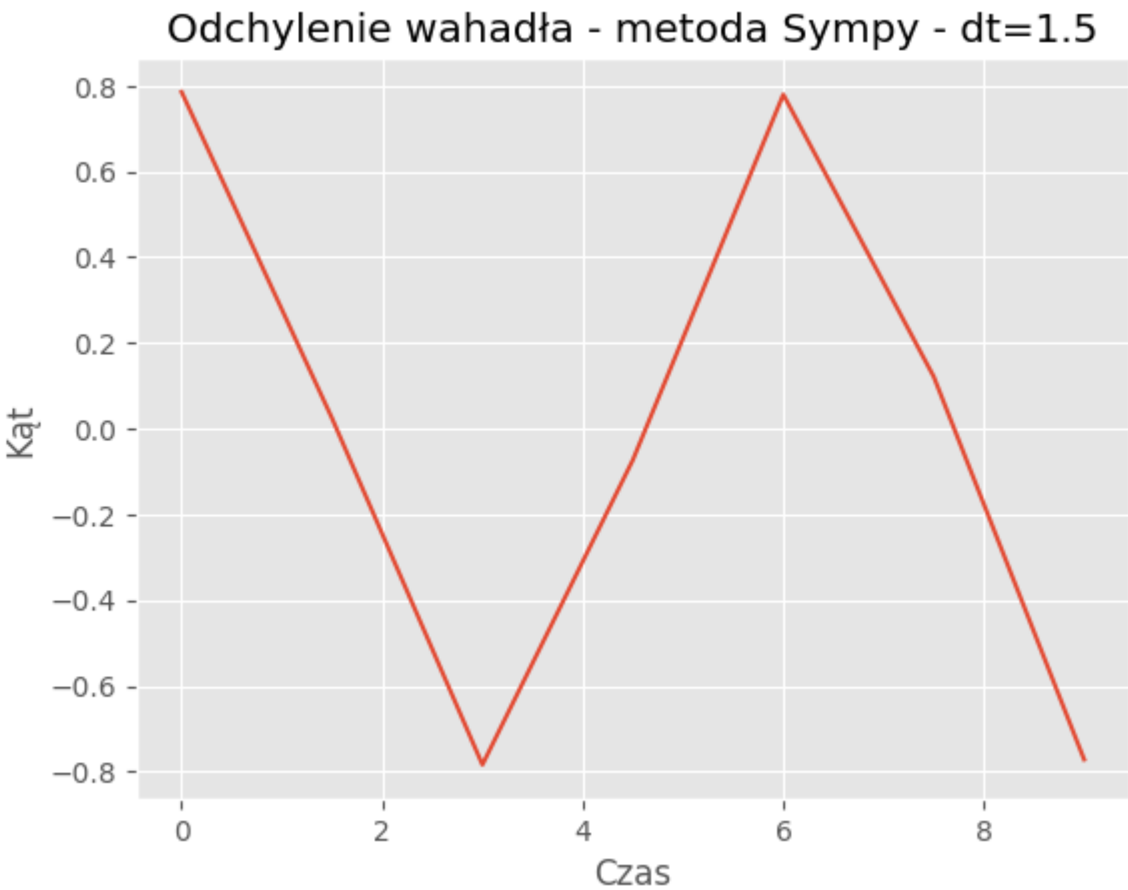

```
For dt = 1.125:
Mean Absolute Error: 0.1021
Mean Squared Error: 0.0194
```

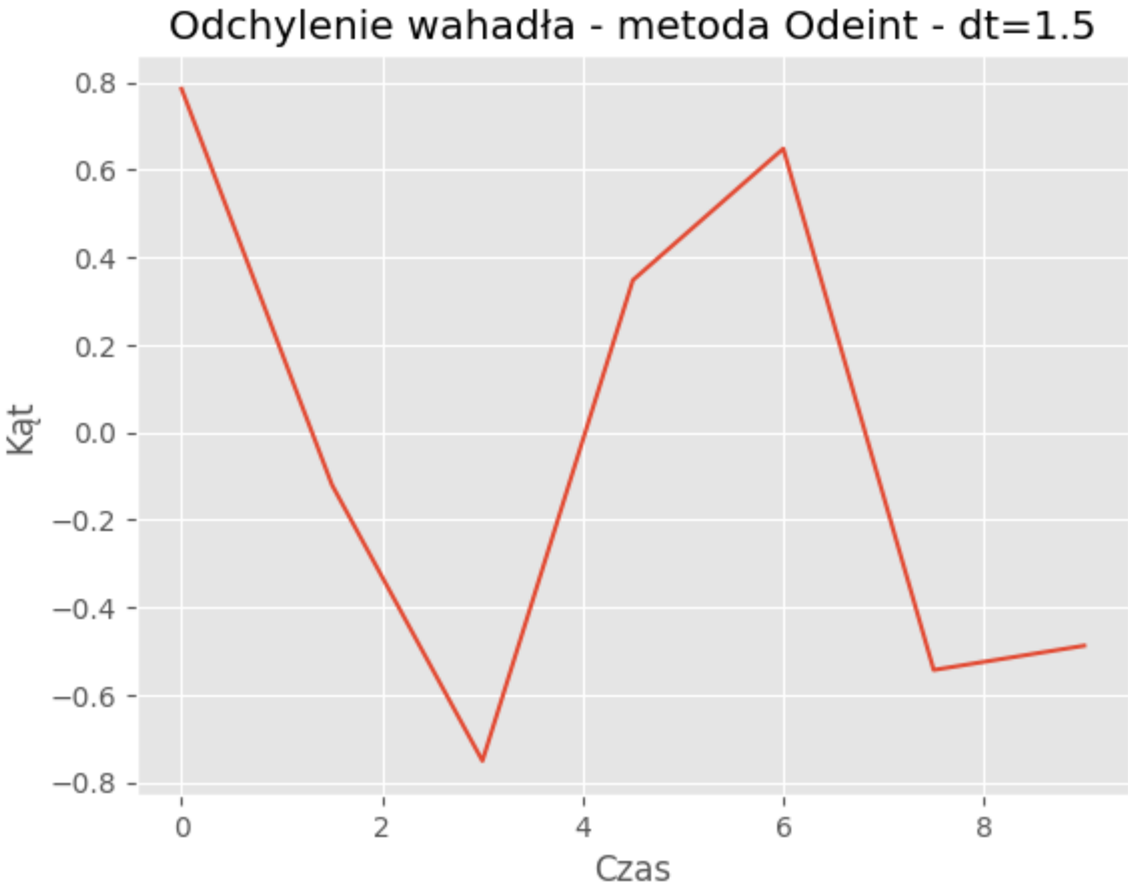

Odchylenie wahadła - metoda Sympy - dt=1.25
Kąt
Czas
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
2
4
6
8

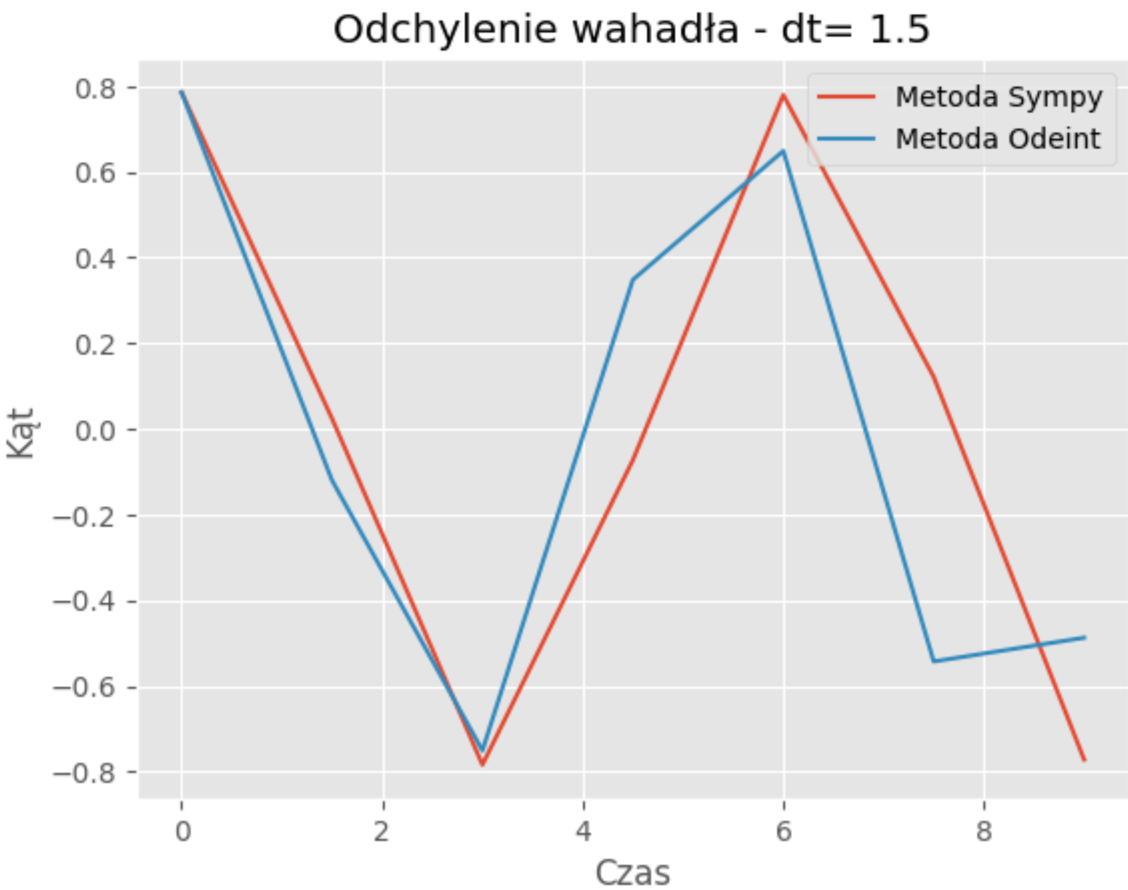

Odchylenie wahadła - metoda Odeint - dt=1.25
Kąt
Czas
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
2
4
6
8



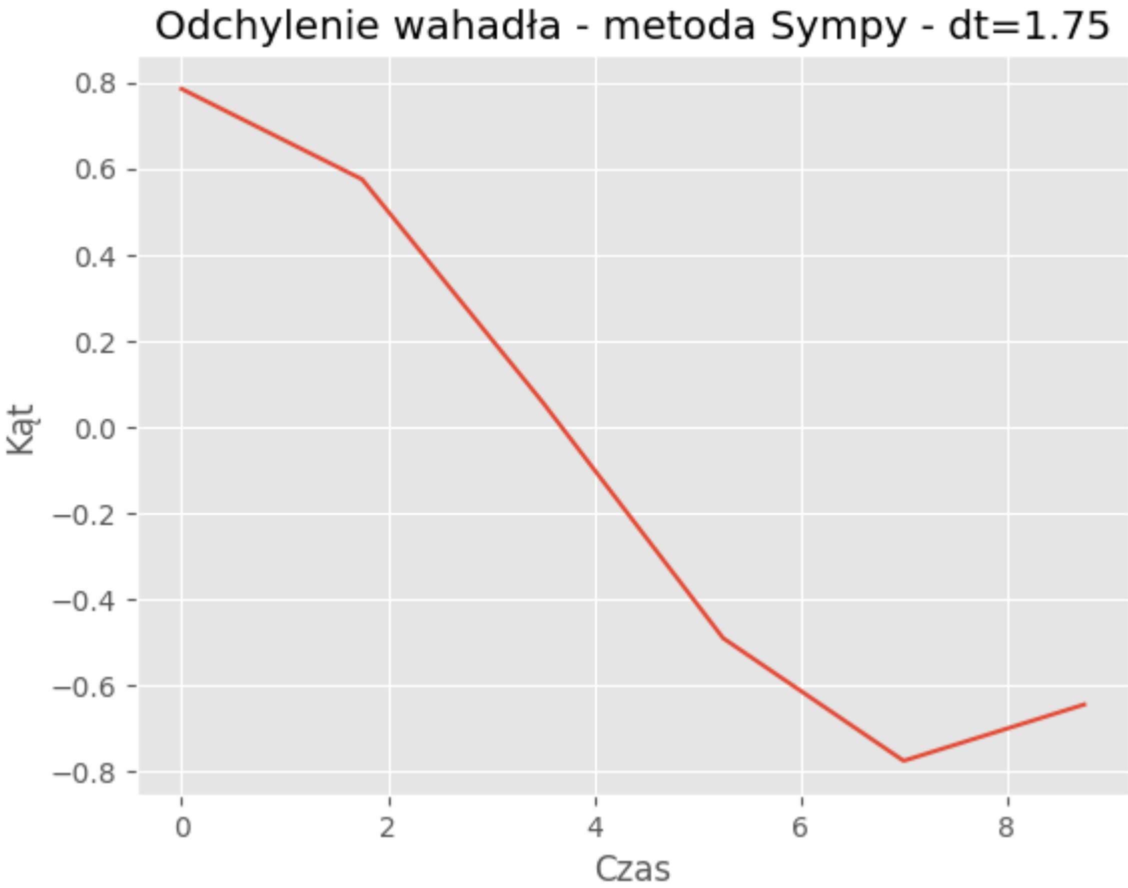

```
For dt = 1.25:
Mean Absolute Error: 0.2995
Mean Squared Error: 0.1271
```

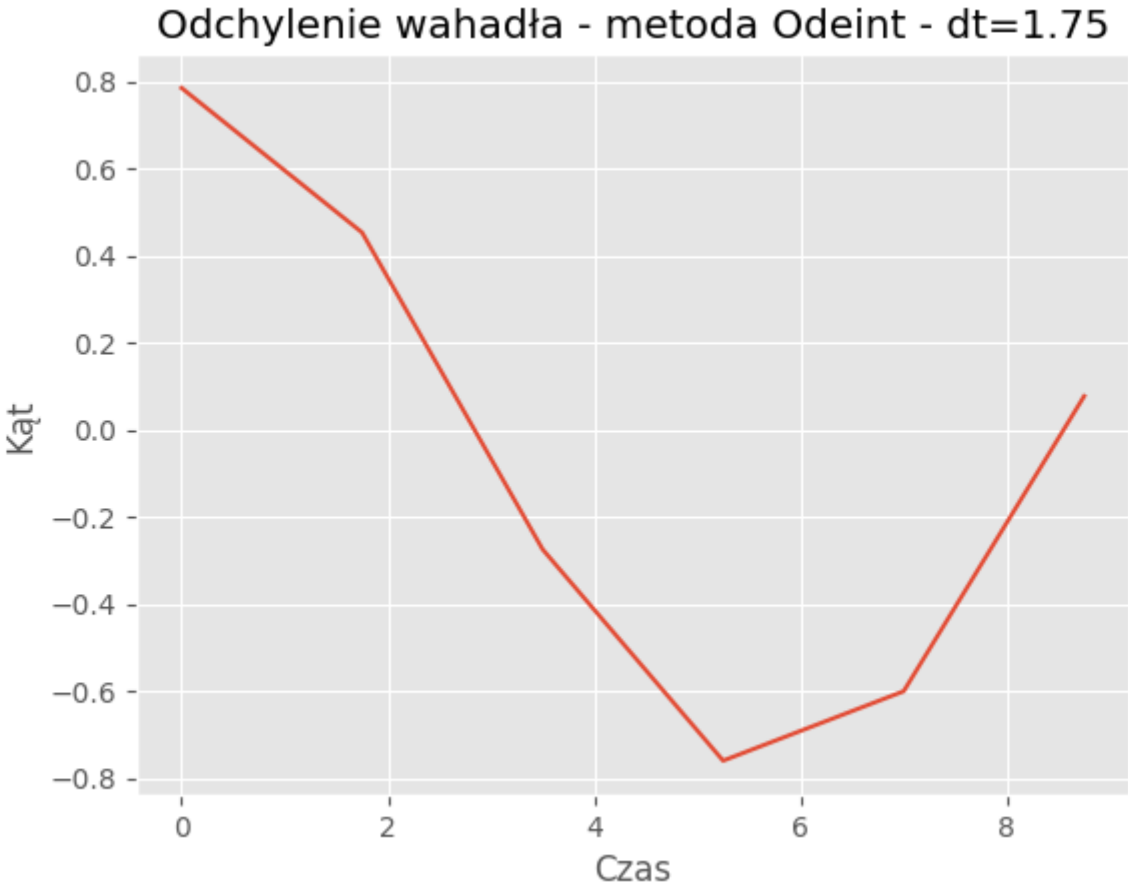


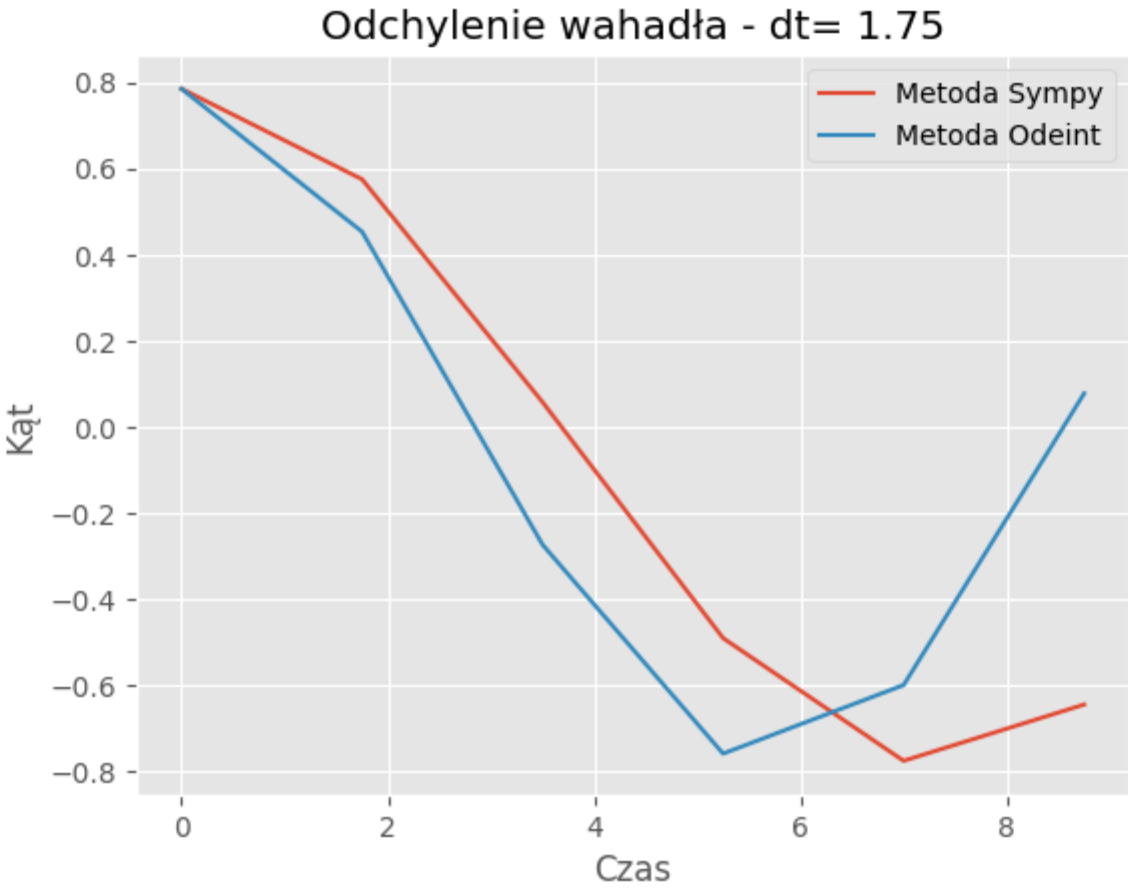




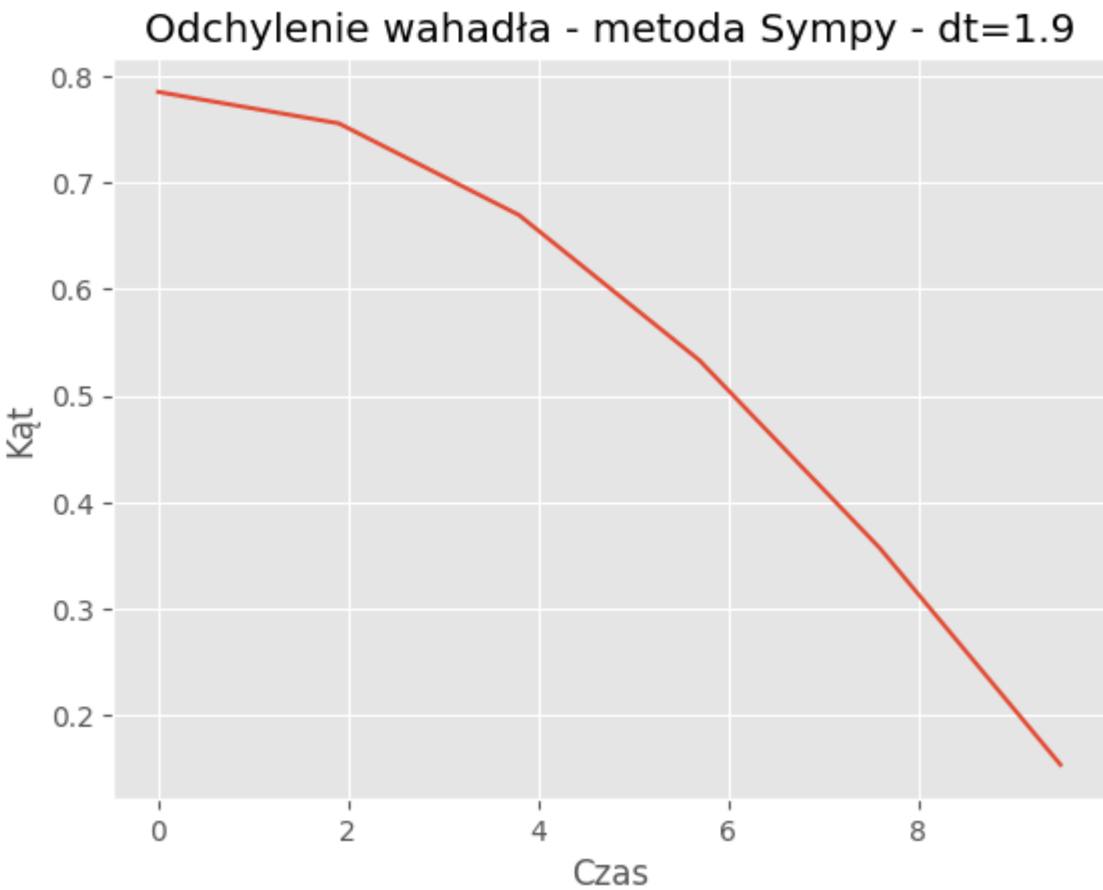

```
For dt = 1.5:
Mean Absolute Error: 0.2976
Mean Squared Error: 0.1506
```

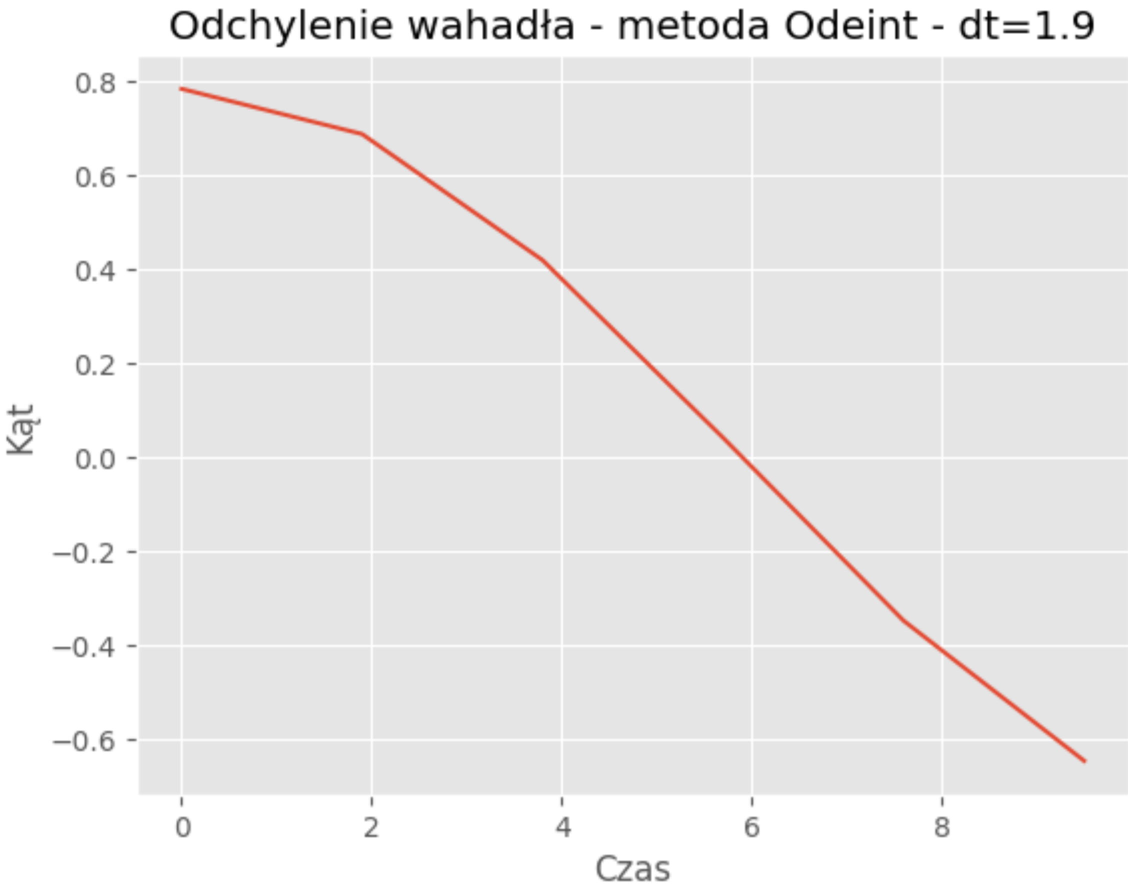

Odchylenie wahadła - metoda Sympy - dt=1.75
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
Kąt
0
2
4
6
8
Czas

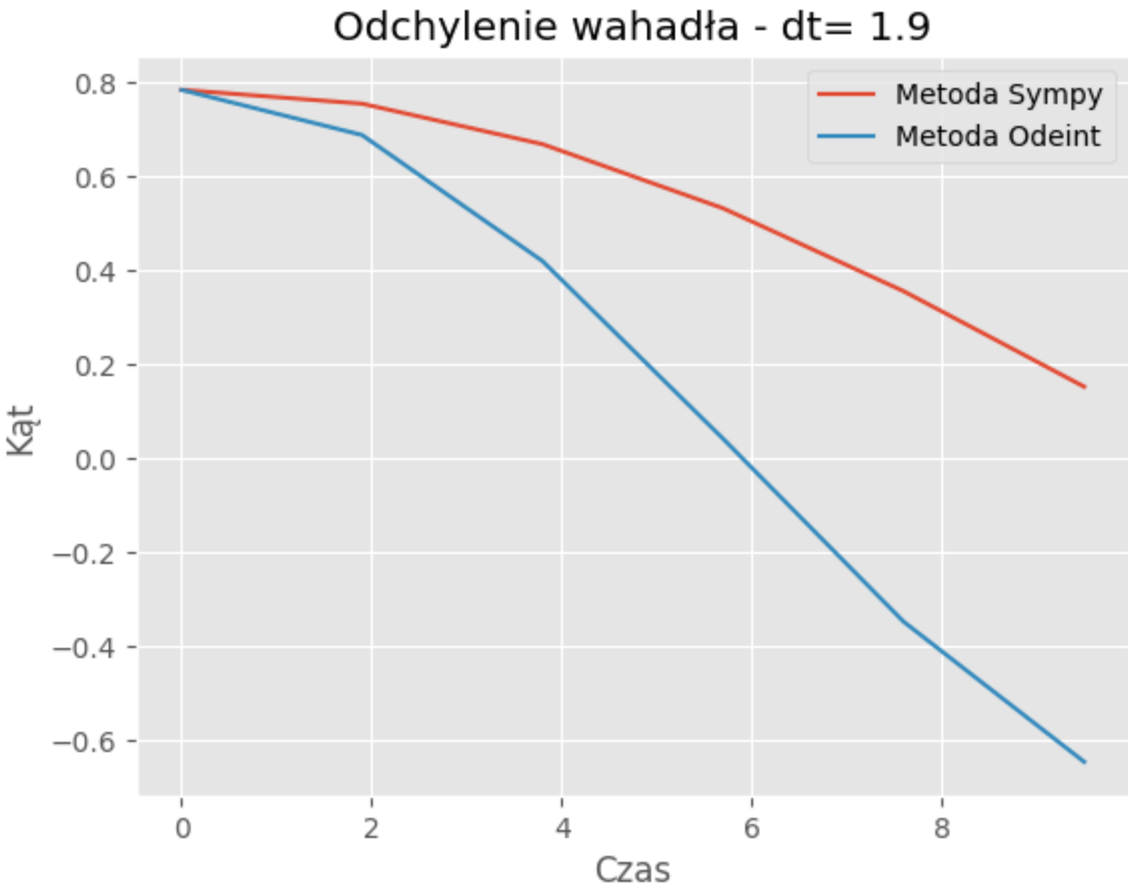

Odchylenie wahadła - metoda Odeint - dt=1.75
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
Kąt
0
2
4
6
8
Czas



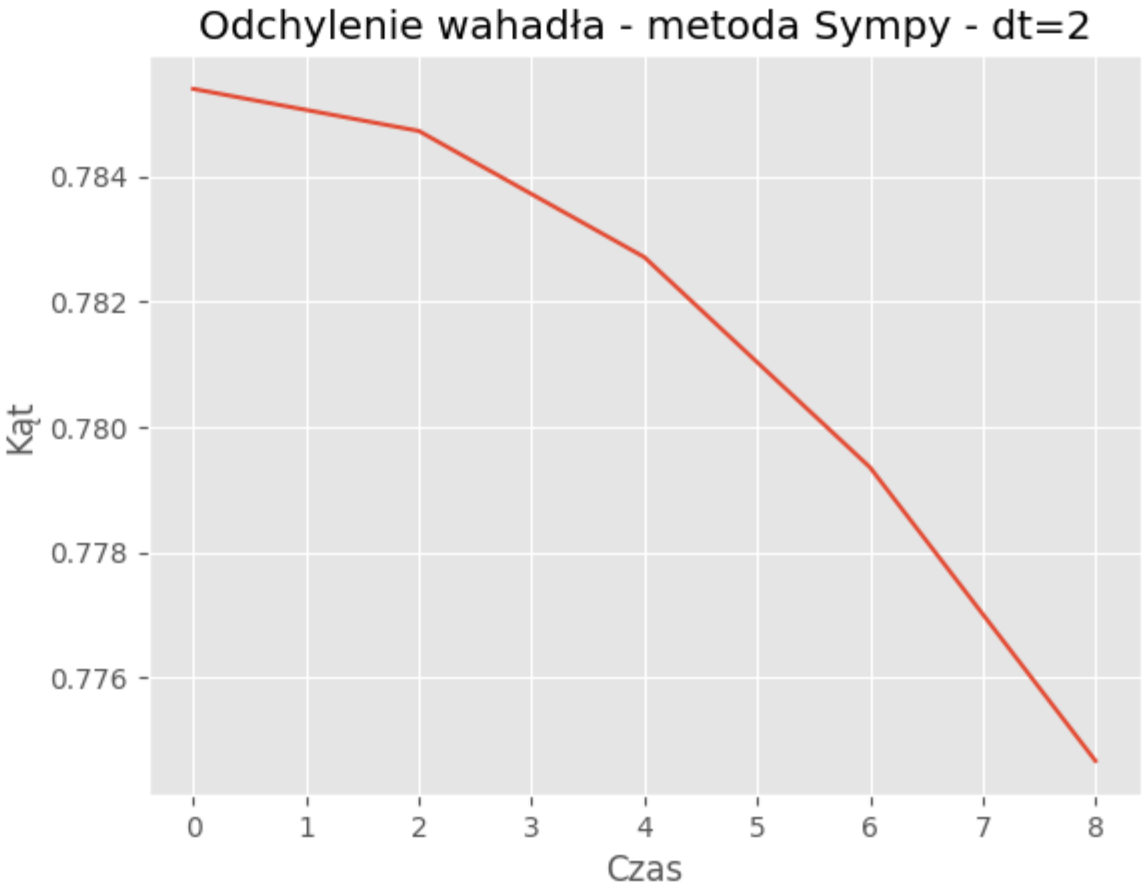

```
For dt = 1.75:
Mean Absolute Error: 0.2397
Mean Squared Error: 0.1055
```

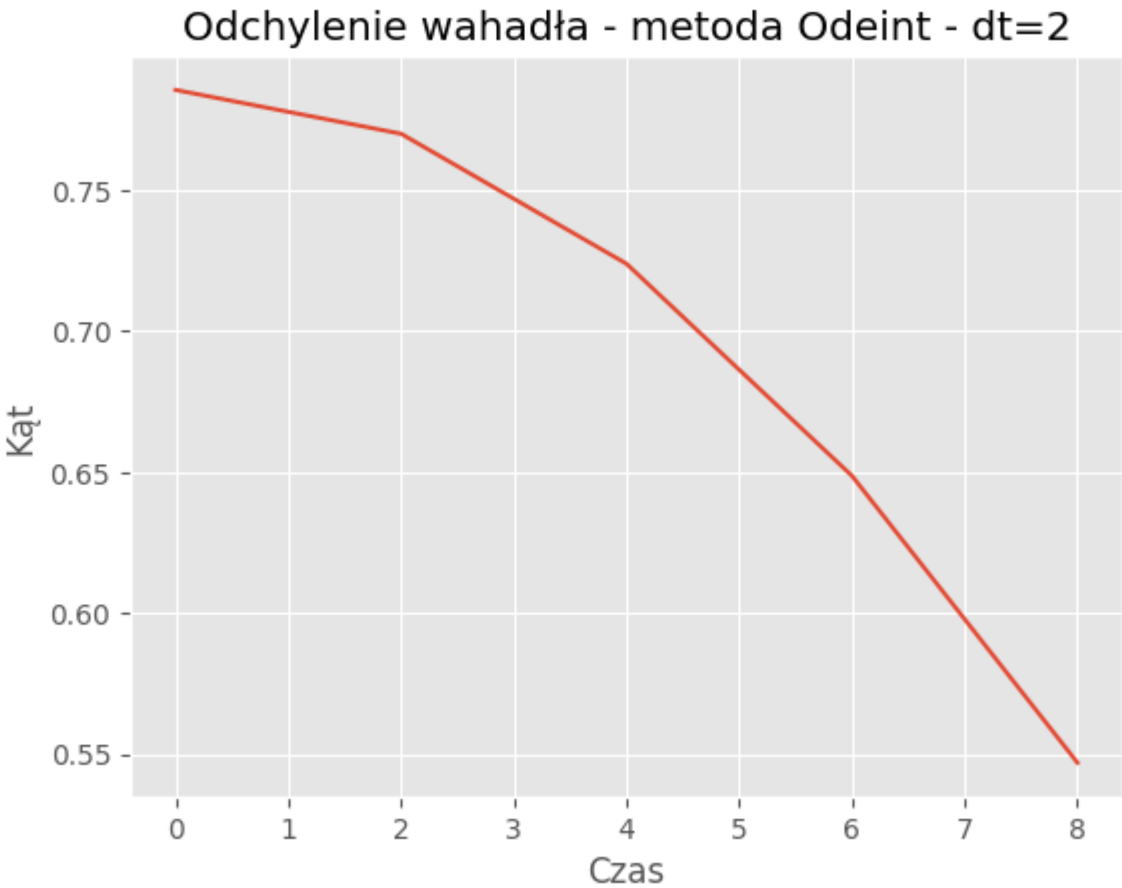


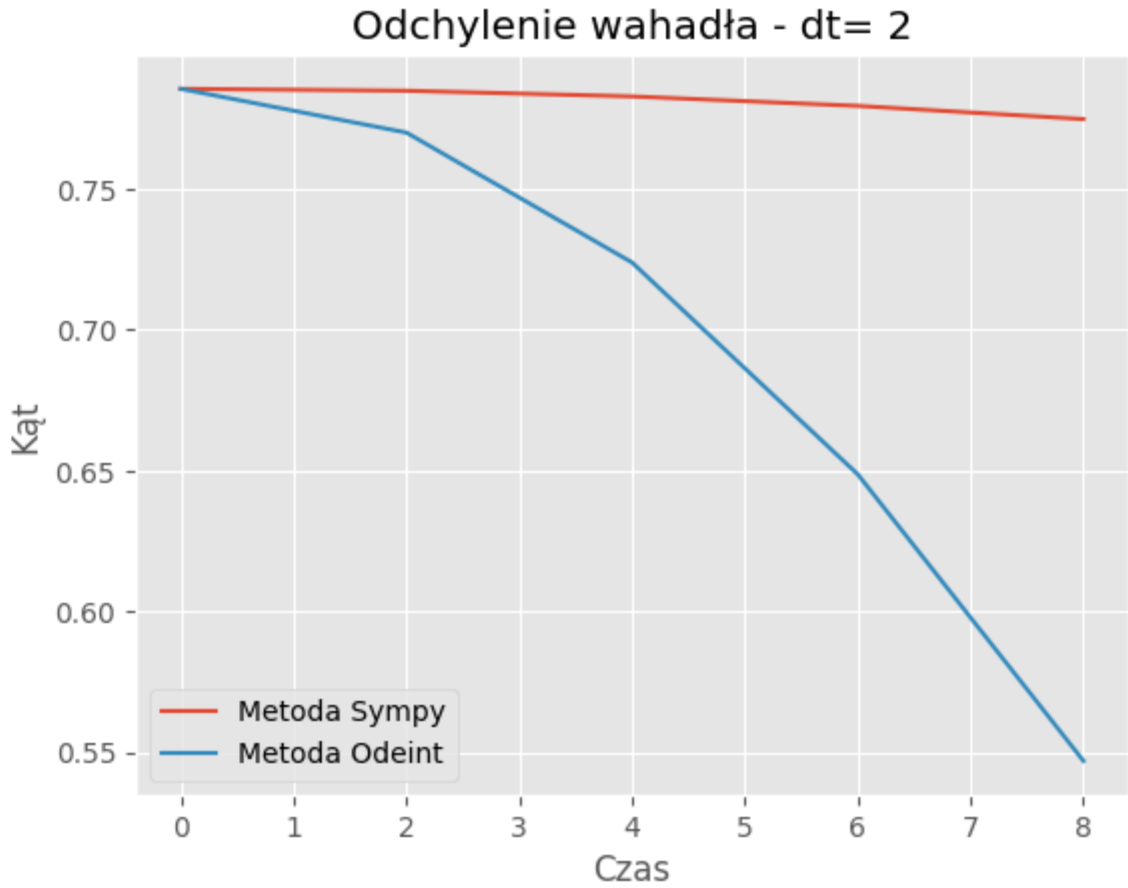




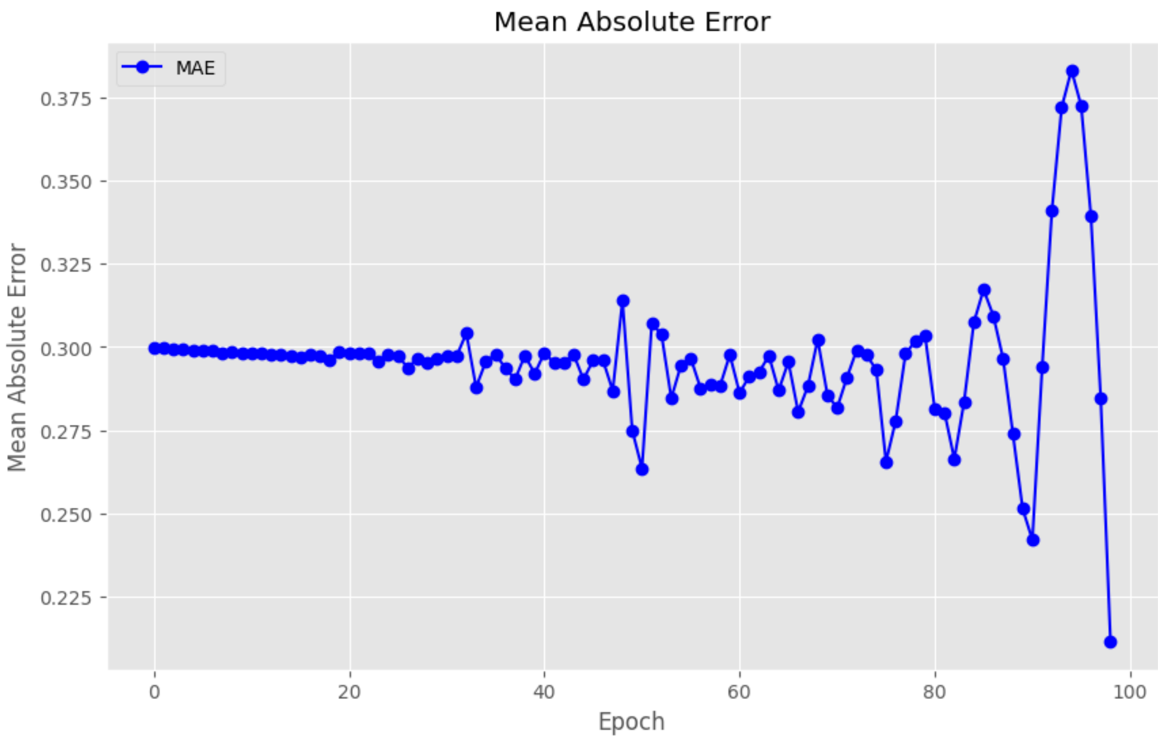

```
For dt = 1.9:
Mean Absolute Error: 0.2703
Mean Squared Error: 0.1252
```

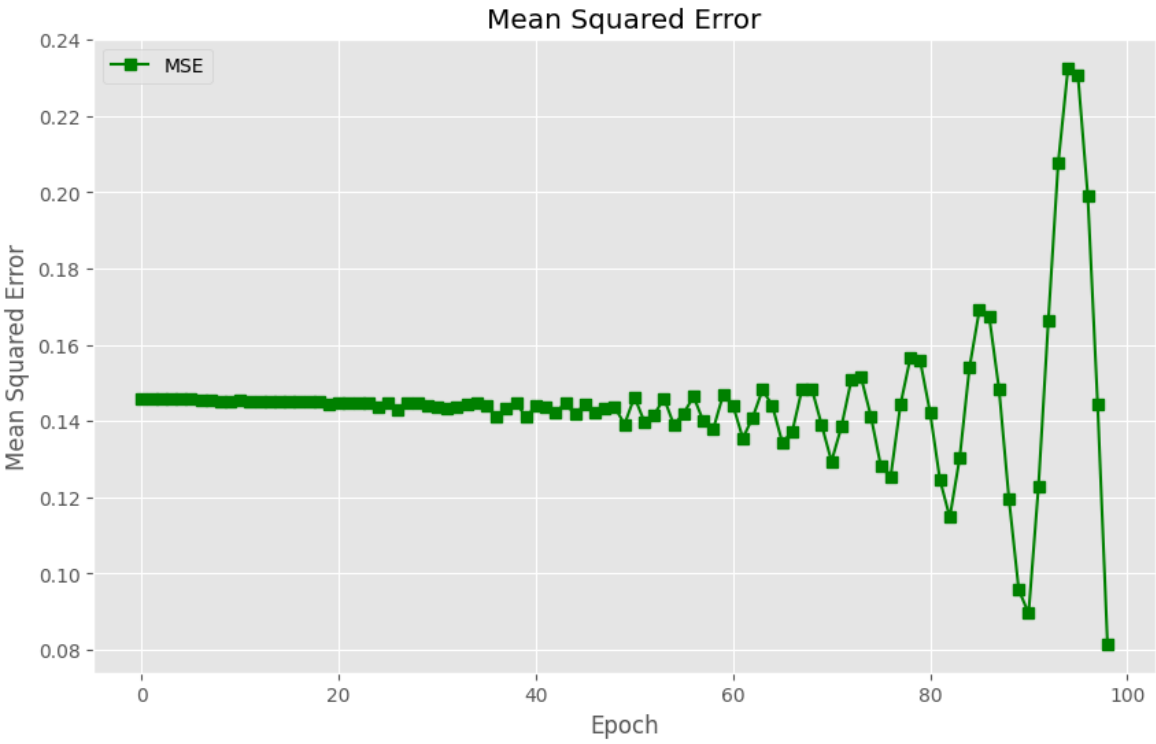

Odchylenie wahadła - metoda Sympy - dt=2
Kąt
0.784
0.782
0.780
0.778
0.776
0
1
2
3
4
5
6
7
8
Czas


Odchylenie wahadła - metoda Odeint - dt=2
Kąt
0.75
0.70
0.65
0.60
0.55
0
1
2
3
4
5
6
7
8
Czas




```
For dt = 2:
Mean Absolute Error: 0.3848
Mean Squared Error: 0.2401
```




Mean Squared Error
MSE
Mean Squared Error
Epoch
0.24
0.22
0.20
0.18
0.16
0.14
0.12
0.10
0.08
0
20
40
60
80
100

In [ ]: