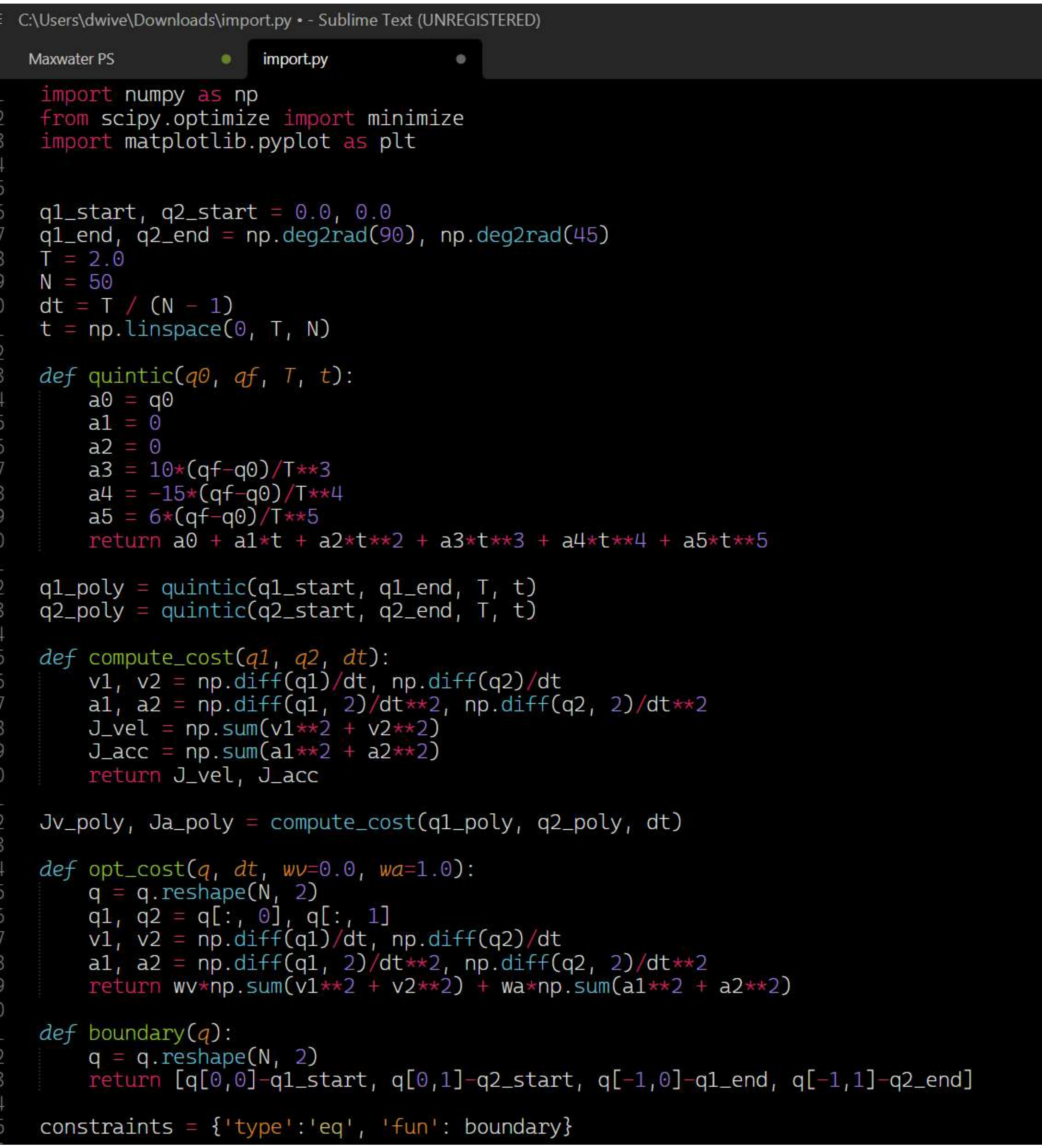
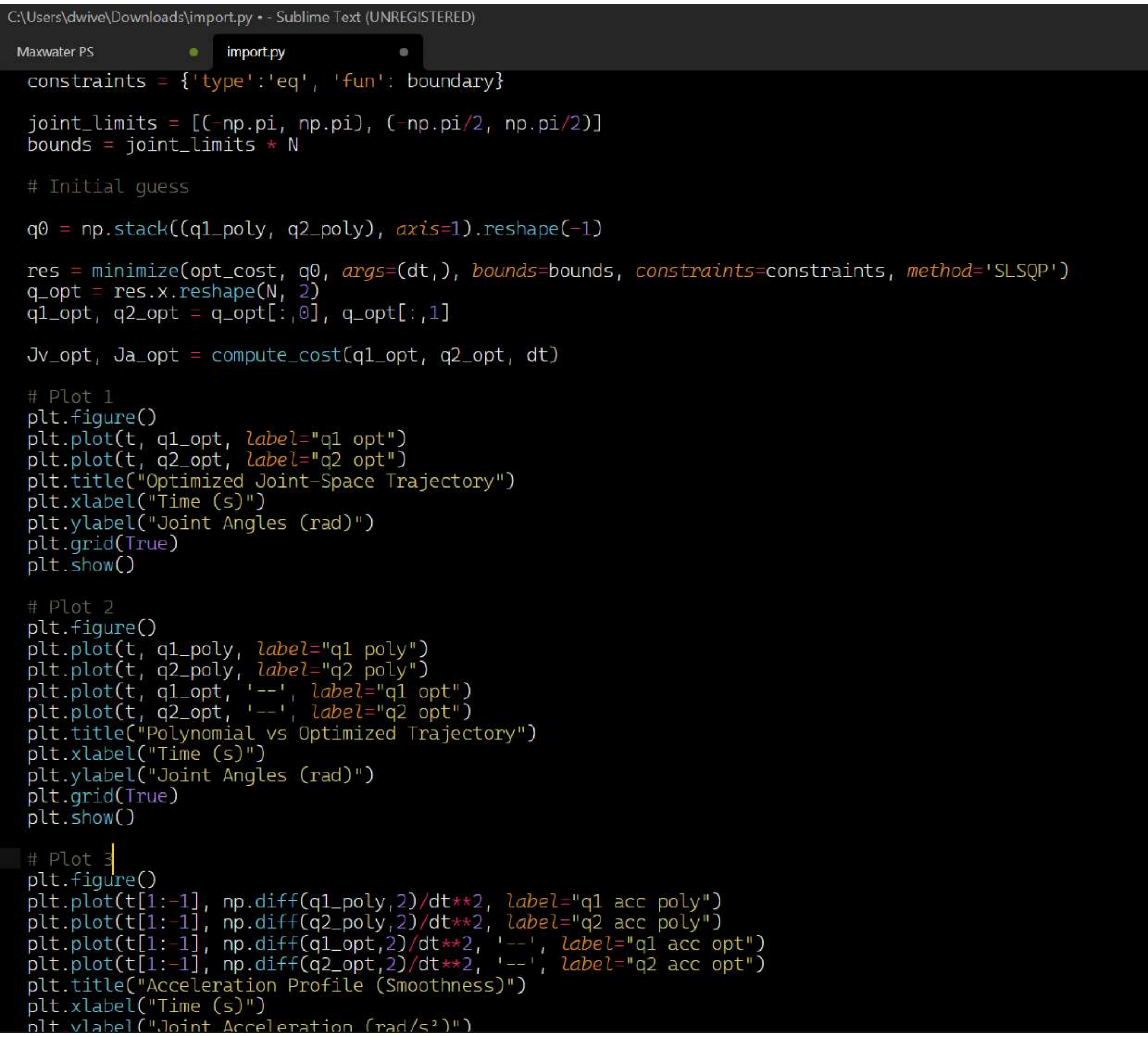
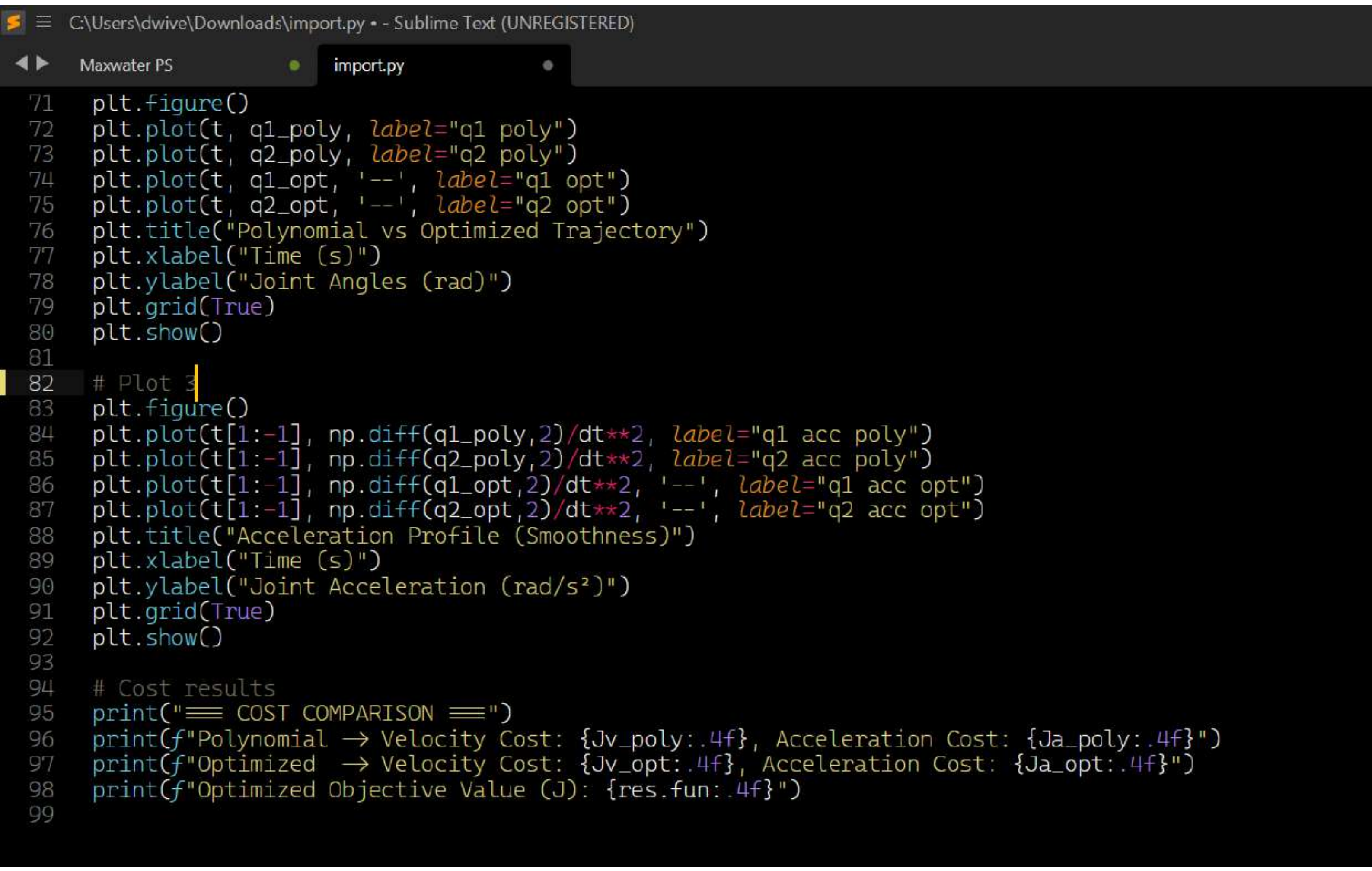

C:\Users\dwive\Downloads\import.py • - Sublime Text (UNREGISTERED)
Maxwater PS
import.py

```python
import numpy as np
from scipy.optimize import minimize
import matplotlib.pyplot as plt


q1_start, q2_start = 0.0, 0.0
q1_end, q2_end = np.deg2rad(90), np.deg2rad(45)
T = 2.0
N = 50
dt = T / (N - 1)
t = np.linspace(0, T, N)

def quintic(q0, qf, T, t):
    a0 = q0
    a1 = 0
    a2 = 0
    a3 = 10*(qf-q0)/T**3
    a4 = -15*(qf-q0)/T**4
    a5 = 6*(qf-q0)/T**5
    return a0 + a1*t + a2*t**2 + a3*t**3 + a4*t**4 + a5*t**5

q1_poly = quintic(q1_start, q1_end, T, t)
q2_poly = quintic(q2_start, q2_end, T, t)

def compute_cost(q1, q2, dt):
    v1, v2 = np.diff(q1)/dt, np.diff(q2)/dt
    a1, a2 = np.diff(q1, 2)/dt**2, np.diff(q2, 2)/dt**2
    J_vel = np.sum(v1**2 + v2**2)
    J_acc = np.sum(a1**2 + a2**2)
    return J_vel, J_acc

Jv_poly, Ja_poly = compute_cost(q1_poly, q2_poly, dt)

def opt_cost(q, dt, wv=0.0, wa=1.0):
    q = q.reshape(N, 2)
    q1, q2 = q[:, 0], q[:, 1]
    v1, v2 = np.diff(q1)/dt, np.diff(q2)/dt
    a1, a2 = np.diff(q1, 2)/dt**2, np.diff(q2, 2)/dt**2
    return wv*np.sum(v1**2 + v2**2) + wa*np.sum(a1**2 + a2**2)

def boundary(q):
    q = q.reshape(N, 2)
    return [q[0,0]-q1_start, q[0,1]-q2_start, q[-1,0]-q1_end, q[-1,1]-q2_end]

constraints = {'type':'eq', 'fun': boundary}
```

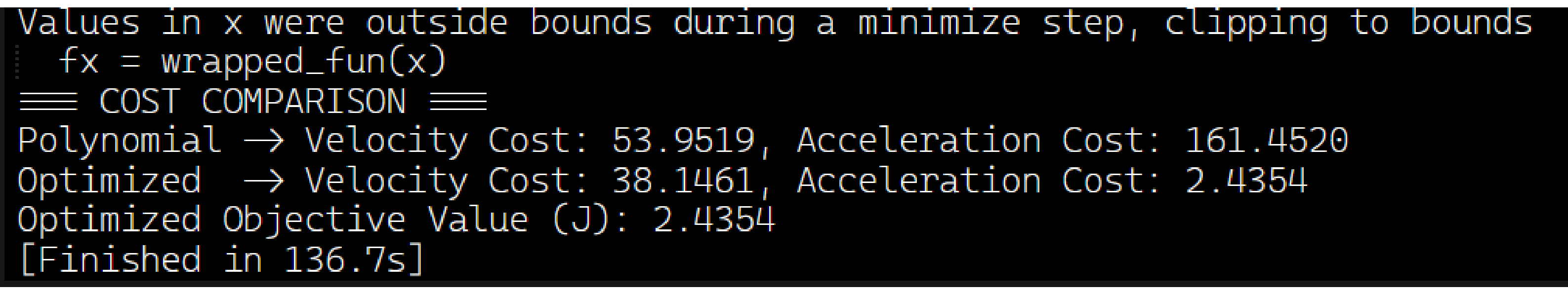

C:\Users\dwive\Downloads\import.py • - Sublime Text (UNREGISTERED)

Maxwater PS | import.py

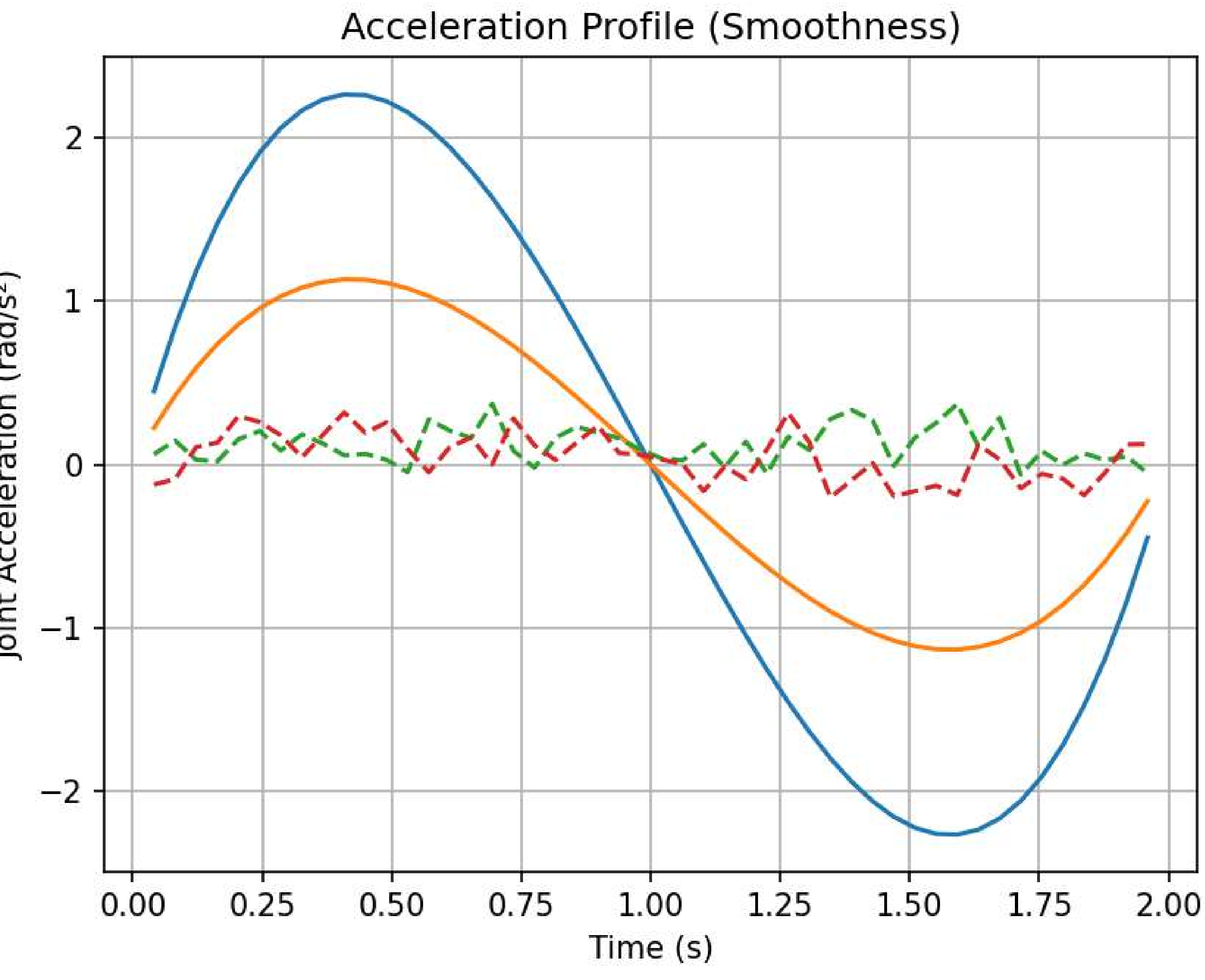
```
constraints = {'type':'eq', 'fun': boundary}

joint_limits = [(-np.pi, np.pi), (-np.pi/2, np.pi/2)]
bounds = joint_limits * N

# Initial guess

q0 = np.stack((q1_poly, q2_poly), axis=1).reshape(-1)

res = minimize(opt_cost, q0, args=(dt,), bounds=bounds, constraints=constraints, method='SLSQP')
q_opt = res.x.reshape(N, 2)
q1_opt, q2_opt = q_opt[:,0], q_opt[:,1]

Jv_opt, Ja_opt = compute_cost(q1_opt, q2_opt, dt)

# Plot 1
plt.figure()
plt.plot(t, q1_opt, label="q1 opt")
plt.plot(t, q2_opt, label="q2 opt")
plt.title("Optimized Joint-Space Trajectory")
plt.xlabel("Time (s)")
plt.ylabel("Joint Angles (rad)")
plt.grid(True)
plt.show()

# Plot 2
plt.figure()
plt.plot(t, q1_poly, label="q1 poly")
plt.plot(t, q2_poly, label="q2 poly")
plt.plot(t, q1_opt, '--', label="q1 opt")
plt.plot(t, q2_opt, '--', label="q2 opt")
plt.title("Polynomial vs Optimized Trajectory")
plt.xlabel("Time (s)")
plt.ylabel("Joint Angles (rad)")
plt.grid(True)
plt.show()

# Plot 3
plt.figure()
plt.plot(t[1:-1], np.diff(q1_poly,2)/dt**2, label="q1 acc poly")
plt.plot(t[1:-1], np.diff(q2_poly,2)/dt**2, label="q2 acc poly")
plt.plot(t[1:-1], np.diff(q1_opt,2)/dt**2, '--', label="q1 acc opt")
plt.plot(t[1:-1], np.diff(q2_opt,2)/dt**2, '--', label="q2 acc opt")
plt.title("Acceleration Profile (Smoothness)")
plt.xlabel("Time (s)")
plt.ylabel("Joint Acceleration (rad/s²)")
```


C:\Users\dwive\Downloads\import.py • - Sublime Text (UNREGISTERED)
Maxwater PS
import.py

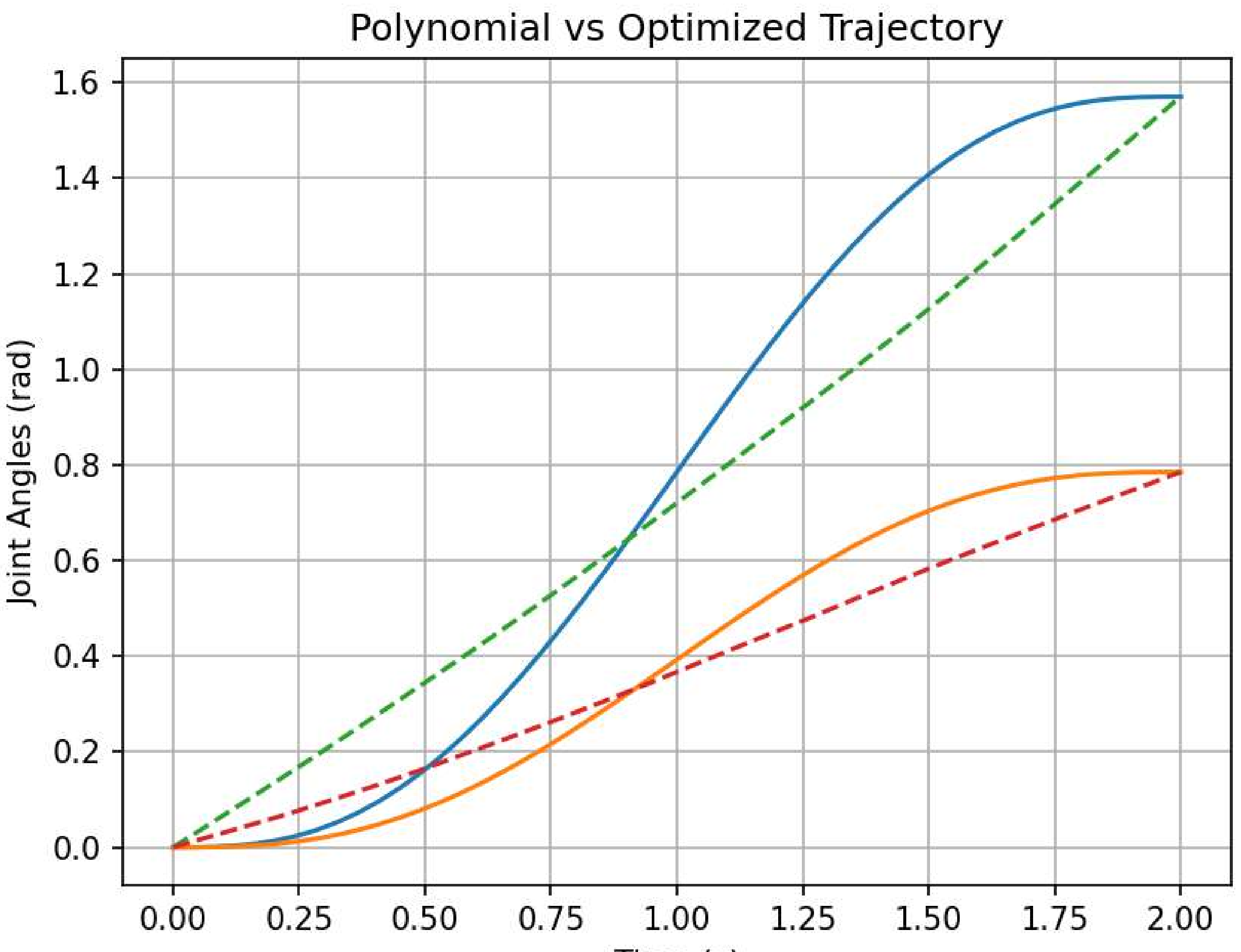
```python
plt.figure()
plt.plot(t, q1_poly, label="q1 poly")
plt.plot(t, q2_poly, label="q2 poly")
plt.plot(t, q1_opt, '--', label="q1 opt")
plt.plot(t, q2_opt, '--', label="q2 opt")
plt.title("Polynomial vs Optimized Trajectory")
plt.xlabel("Time (s)")
plt.ylabel("Joint Angles (rad)")
plt.grid(True)
plt.show()

# Plot 3
plt.figure()
plt.plot(t[1:-1], np.diff(q1_poly,2)/dt**2, label="q1 acc poly")
plt.plot(t[1:-1], np.diff(q2_poly,2)/dt**2, label="q2 acc poly")
plt.plot(t[1:-1], np.diff(q1_opt,2)/dt**2, '--', label="q1 acc opt")
plt.plot(t[1:-1], np.diff(q2_opt,2)/dt**2, '--', label="q2 acc opt")
plt.title("Acceleration Profile (Smoothness)")
plt.xlabel("Time (s)")
plt.ylabel("Joint Acceleration (rad/s²)")
plt.grid(True)
plt.show()

# Cost results
print("=== COST COMPARISON ===")
print(f"Polynomial -> Velocity Cost: {Jv_poly:.4f}, Acceleration Cost: {Ja_poly:.4f}")
print(f"Optimized  -> Velocity Cost: {Jv_opt:.4f}, Acceleration Cost: {Ja_opt:.4f}")
print(f"Optimized Objective Value (J): {res.fun:.4f}")
```

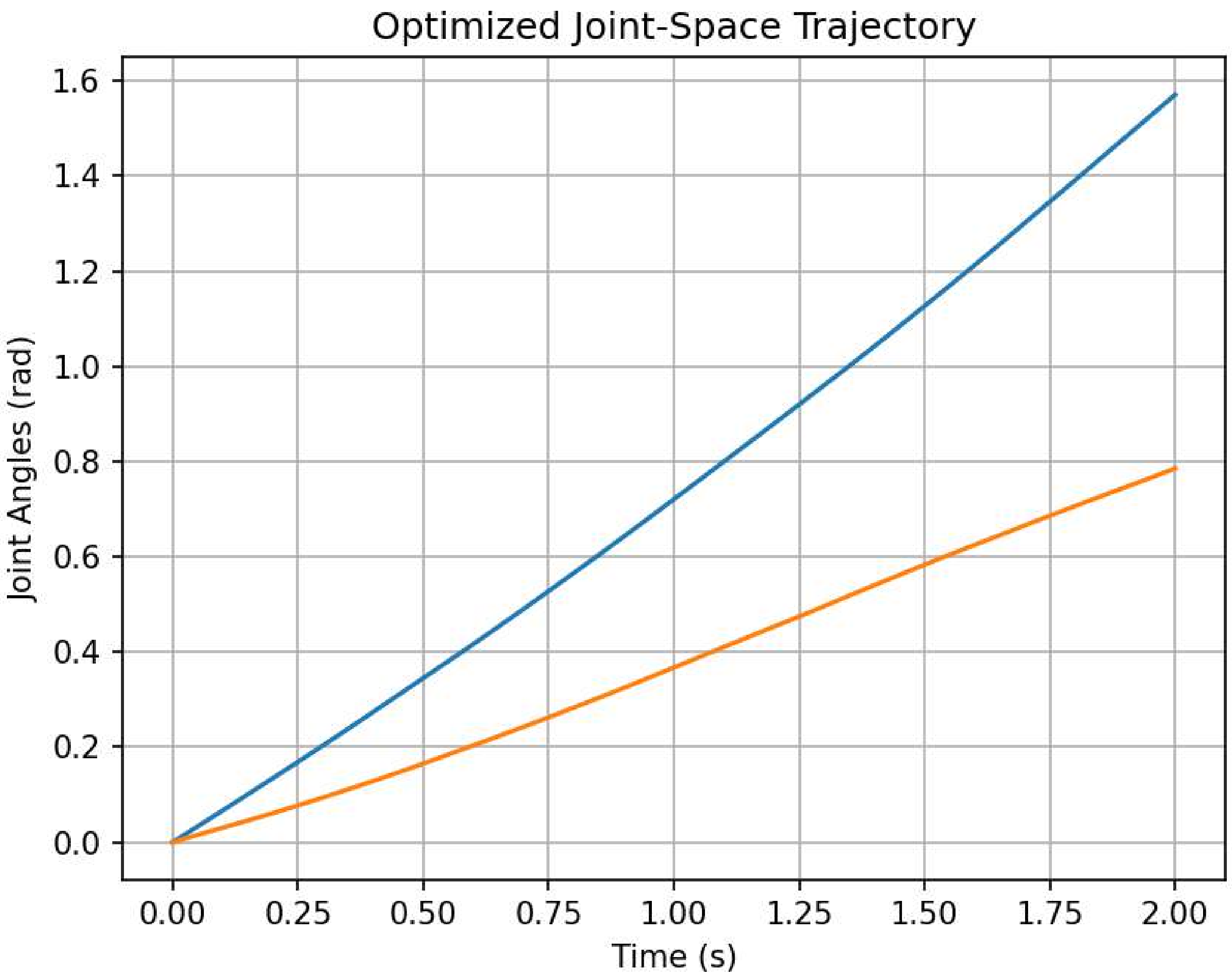


```
Values in x were outside bounds during a minimize step, clipping to bounds
  fx = wrapped_fun(x)
=== COST COMPARISON ===
Polynomial → Velocity Cost: 53.9519, Acceleration Cost: 161.4520
Optimized  → Velocity Cost: 38.1461, Acceleration Cost: 2.4354
Optimized Objective Value (J): 2.4354
[Finished in 136.7s]
```



Acceleration Profile (Smoothness)
Joint Acceleration (rad/s²)
Time (s)
2
1
0
−1
−2
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00


Polynomial vs Optimized Trajectory
Joint Angles (rad)
Time (s)
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00


Optimized Joint-Space Trajectory
Joint Angles (rad)
Time (s)
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00