LAB-06 GREY WOLF OPTIMIZER

import numpy as np

def fitness(angle):

return np.sin(np.radians(angle)) # max at 90 degrees

# Input parameters

N = int(input("Enter number of wolves: "))

MaxIter = int(input("Enter number of iterations: "))

lower\_bound = 0

upper\_bound = 90

# Input initial wolves

wolves = []

print(f"Enter {N} initial wolf positions (angles between {lower\_bound}-{upper\_bound}):")

for i in range(N):

while True:

x = float(input(f"Wolf {i+1}: "))

if lower\_bound <= x <= upper\_bound:

wolves.append(x)

break

else:

print(f"Enter a value between {lower\_bound} and {upper\_bound}!")

wolves = np.array(wolves)

# Evaluate initial fitness

fitness\_values = np.array([fitness(x) for x in wolves])

sorted\_idx = np.argsort(fitness\_values)[::-1] # descending order

alpha = wolves[sorted\_idx[0]]

beta = wolves[sorted\_idx[1]]

delta = wolves[sorted\_idx[2]]

print("\nInitial wolves:", wolves)

print(f"Alpha = {alpha}, Beta = {beta}, Delta = {delta}\n")

# Main GWO loop

for iter in range(1, MaxIter+1):

a = 2 - 2 \* (iter / MaxIter) # decreases from 2 → 0

for i in range(N):

r1 = np.random.rand()

r2 = np.random.rand()

r3 = np.random.rand()

A1, A2, A3 = 2\*a\*r1 - a, 2\*a\*r2 - a, 2\*a\*r3 - a

C1, C2, C3 = 2\*r1, 2\*r2, 2\*r3

# Distance to leaders

D\_alpha = abs(C1 \* alpha - wolves[i])

D\_beta = abs(C2 \* beta - wolves[i])

D\_delta = abs(C3 \* delta - wolves[i])

# Position update

X1 = alpha - A1 \* D\_alpha

X2 = beta - A2 \* D\_beta

X3 = delta - A3 \* D\_delta

wolves[i] = (X1 + X2 + X3) / 3

# Keep within bounds

wolves[i] = np.clip(wolves[i], lower\_bound, upper\_bound)

# Update fitness

fitness\_values = np.array([fitness(x) for x in wolves])

# Update leaders

sorted\_idx = np.argsort(fitness\_values)[::-1]

alpha = wolves[sorted\_idx[0]]

beta = wolves[sorted\_idx[1]]

delta = wolves[sorted\_idx[2]]

# Display iteration info

print(f"Iteration {iter}: Wolves positions = {np.round(wolves,2)}")

print(f"Alpha = {alpha:.2f}, Beta = {beta:.2f}, Delta = {delta:.2f}\n")

# Final output

print(f"Best tilt angle = {alpha:.2f} degrees")

print(f"Maximum solar energy captured = {fitness(alpha):.4f}")

