6) PSO
CODE:

```
import numpy as np
def objective_function(a):
    return 1 + 2*a[0] + (3*a[1] - 1) + 3*a[2] + 2*a[3]**2 + (5*a[4] + 2)

n_particles = 20
n_dimensions = 5
max_iter = 50
w_max = 0.9
w_min = 0.3
c1 = 1
c2 = 1

bounds = [(10, 60), (15, 30), (27, 75), (10, 30), (10, 50)]
positions = np.random.uniform(low=np.array([bound[0] for bound in bounds]),
                              high=np.array([bound[1] for bound in bounds]),
                              size=(n_particles, n_dimensions))
velocities = np.random.uniform(low=-1, high=1, size=(n_particles, n_dimensions))

# Initialize best positions and global best
best_positions = positions.copy()
best_values = np.array([objective_function(pos) for pos in positions])
global_best_index = np.argmax(best_values)
global_position = positions[global_best_index].copy()
global_best_value = best_values[global_best_index]

# Perform PSO iterations
for _ in range(max_iter):
    w = np.random.uniform(low=w_min, high=w_max)
    #w = w_max - (_ / max_iter) * (w_max - w_min)
    r1 = np.random.uniform(size=(n_particles, n_dimensions))
    r2 = np.random.uniform(size=(n_particles, n_dimensions))
    velocities = w * velocities + c1 * r1 * (best_positions - positions) + c2 * r2 * (global_position -
positions)
    positions += velocities

    # Apply bounds
    positions = np.clip(positions, np.array([bound[0] for bound in bounds]), np.array([bound[1] for
bound in bounds]))

    # Update best positions
    current_values = np.array([objective_function(pos) for pos in positions])
```

```
    for i in range(n_particles):
        if current_values[i] > best_values[i]:
            best_positions[i] = positions[i].copy()
            best_values[i] = current_values[i]

    new_global_best_index = np.argmax(best_values)
    if best_values[new_global_best_index] > global_best_value:
        global_best_index = new_global_best_index
        global_position = best_positions[global_best_index].copy()
        global_best_value = best_values[global_best_index]

print("Optimal solution:")
print("a1 =", global_position[0])
print("a2 =", global_position[1])
print("a3 =", global_position[2])
print("a4 =", global_position[3])
print("a5 =", global_position[4])
print("Maximum value of f(a1,a2,a3,a4,a5) =", global_best_value)
```

OUTPUT:

```
Optimal solution:
a1 = 60.0
a2 = 30.0
a3 = 75.0
a4 = 30.0
a5 = 50.0
Maximum value of f(a1,a2,a3,a4,a5) = 2487.0
```