TD 2: Meta-heuristic optimization methods

Exercise 01: PSO Step-by-Step (2 iterations)

Using PSO algorithm, we want to minimize the function:

$$f(x) = x^2$$

Given:

- Number of particles: **2**
- Dimension: **1D**
- Inertia weight: **w = 0.5**
- Cognitive coefficient: **c_1 = 1**
- Social coefficient: **c_2 = 1**
- Random values:
  - Iteration 1: $r_1 = 0.5$, $r_2 = 0.5$
  - Iteration 2: $r_1 = 0.3$, $r_2 = 0.7$

Initial positions and velocities:

Particle	x(0)	v(0)
P1	4	0
P2	-3	0

Exercise 02: 2D PSO with Full Iteration Tracking

Using PSO algorithm, Minimize:

$$f(x, y) = x^2 + y^2$$

Given:

- 2 particles, 2D
- $w = 0.7$, $c_1 = 1.5$, $c_2 = 1.5$
- $r_1 = 0.6$, $r_2 = 0.4$ (constant)

Initial:

Particle	Position (x,y)	Velocity
P1	(2, -1)	(0, 0)
P2	(-1, 3)	(0, 0)

Tasks:

1. Compute **pBest and gBest**
2. Perform **2 iterations**
3. Track all updates: velocity, position, fitness
4. Check convergence trend

Exercise 03: Task Offloading Optimization Using Genetic Algorithm

A system has **4 tasks** that must be executed either:

- **Locally (0)** → low delay, high energy
- **Offloaded to UAV (1)** → higher delay, lower energy

Each solution is encoded as a **binary chromosome of length 4**:

Example:

- 1010 → Task1 (UAV), Task2 (Local), Task3 (UAV), Task4 (Local)

- **Given Data**

Task	Local Delay	UAV Delay	Local Energy	UAV Energy
T1	2	5	10	4
T2	3	6	8	3
T3	4	7	12	5
T4	2	4	9	4

- **Objective Function**
  Minimize:

$$F=0.6\times\text{Total Delay}+0.4\times\text{Total Energy}$$

- **Initial Population**

- C1 = 0000
- C2 = 1111
- C3 = 1010
- C4 = 0101

Exercise 04: Knapsack Optimization Using Genetic Algorithm

A drone has a **limited payload capacity of 15 kg**.
You must select items to **maximize total value** without exceeding capacity.

Available Items

Item	Weight (kg)	Value
I1	5	10
I2	4	40

Item	Weight (kg)	Value
I3	6	30
I4	3	50

- **Encoding**

Each chromosome is a binary vector of length 4:

- 1 → item selected
- 0 → item not selected

Example:

- 1011 → I1, I3, I4 selected

- **Objective :** Maximize total value:

$$\text{Fitness}=\text{Total Value}$$

- **Constraint:** If total weight > 15 → **fitness = 0 (penalty)**
- **Initial Population**

  - C1 = 1010
  - C2 = 0111
  - C3 = 1100
  - C4 = 0011

Ant Colony Optimization (ACO) Algorithm

ACO is inspired by the behavior of real ants that find the shortest path between food and their nest using **pheromone trails**.

- ## Mathematical Model

The probability of an ant moving from node i to node j:

$$P_{ij} = \frac{(\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}}{\sum_{k \in N_i} (\tau_{ik})^{\alpha} \cdot (\eta_{ik})^{\beta}}$$

Where:

- τ_{ij}: pheromone on edge i, j
- $\eta_{ij} = \frac{1}{d_{ij}}$: heuristic (visibility)
- α: importance of pheromone
- β: importance of heuristic

- ## Pheromone Update

Evaporation:

$$\tau_{ij} = (1 - \rho)\,\tau_{ij}$$

Deposit:

$$\tau_{ij} = \tau_{ij} + \sum \Delta\tau_{ij}^{k}$$

With:

$$\Delta\tau_{ij}^{k} = \frac{Q}{L_k}$$

- L_k: length of path found by ant k
- Q: constant

- **ACO Algorithm (Step-by-Step)**

1. Initialize parameters:

- Number of ants m
- α, β (control parameters)
- ρ (evaporation rate)
- Initial pheromone τij

2. Repeat until stopping condition:

For each ant k:

- Place ant on a starting node
- Build a solution:

While solution not complete:

Select next node j using probability Pij

- Evaluate solution (path length, cost, etc.)

Update pheromones:

- Evaporate pheromones on all edges
- Add pheromone on edges used by ants

Memorize best solution found

3. Return best solution

Exercise 05:

A graph represents cities connected by distances:

Edge	Distance
A–B	2
A–C	4
B–C	1
B–D	7
C–D	3

Find the **shortest path from A to D** using one iteration of the ACO algorithm.

Parameters

- Initial pheromone on all edges:

$$\tau_{ij} = 1$$

- Heuristic information:

$$\eta_{ij} = \frac{1}{d_{ij}}$$

- Parameters:

$$\alpha = 1, \quad \beta = 2$$