Lab Assignment: Particle Swarm Optimization (PSO) and Multi-objective Optimization using NSGA-II

Course: Advanced Algorithms
Instructor: Dr. Akash Yadav, Assistant Professor, Dept. of CSE, RGIPT

Learning Objectives

After completing this laboratory, you will be able to:

- Formulate real-world optimization problems for both single and multi-objective cases.
- Understand and implement the Particle Swarm Optimization (PSO) algorithm.
- Understand the working of the NSGA-II algorithm and apply it to a discrete selection problem.
- Analyze and visualize Pareto fronts and solution trade-offs.
- Identify suitable problem domains for PSO and NSGA-II and recognize their limitations.

1 Problem Context: City Logistics and EV Planning

A logistics company aims to optimize its urban infrastructure. Two related planning tasks are considered:

1. **Task A (PSO):** Identify the best location for a single micro-fulfillment center (MFC) that minimizes the weighted average customer distance and the land cost.
2. **Task B (NSGA-II):** Select three candidate sites for EV charging stations that balance between minimizing the total installation cost and minimizing customer travel distance to the nearest charger.

These two tasks allow students to apply both *single-objective continuous optimization* (PSO) and *multi-objective discrete optimization* (NSGA-II).

City Data (Common for Both Tasks)

Demand Nodes (15):

Node	X (km)	Y (km)	Demand (units/day)
1	2.0	3.0	30
2	5.1	8.2	45
3	7.0	2.1	20
4	1.5	10.0	10
5	4.0	5.0	35
6	8.5	6.0	25
7	9.0	1.0	15
8	3.0	7.5	40
9	6.0	9.0	50
10	2.5	1.0	12
11	5.5	3.5	22
12	7.5	8.0	18
13	0.5	5.0	8
14	10.0	5.0	28
15	4.5	0.5	14

Candidate Sites for EV Chargers (10):

Site	X (km)	Y (km)	Acquisition Cost (k)	Equipment Cost (k)
A1	1.0	2.5	800	200
A2	3.5	6.0	950	180
A3	6.0	2.0	700	210
A4	8.0	7.0	1200	190
A5	9.0	0.8	900	220
A6	2.0	10.0	850	170
A7	5.0	4.5	880	185
A8	7.8	9.2	1100	180
A9	0.8	6.0	600	200
A10	4.6	1.2	760	205

The coordinate space is bounded by $x \in [0, 11]$, $y \in [0, 11]$.

2 Task A: Single-objective Optimization using PSO

2.1 Objective Function

Determine the optimal location (x, y) for a micro-fulfillment center that minimizes:

$$F(x, y) = \alpha \cdot D(x, y) + (1 - \alpha) \cdot P_{\text{norm}}(x, y)$$

where

$$D(x, y) = \frac{\sum_{i=1}^{N} w_i \, d_i(x, y)}{\sum_i w_i}, \quad d_i(x, y) = \sqrt{(X_i - x)^2 + (Y_i - y)^2}$$

and

$$P_{\text{norm}}(x, y) = \frac{\text{LandPrice}(x, y)}{P_{\max}}, \quad \text{LandPrice}(x, y) = 500 + 100 \sin(0.3x) + 80 \cos(0.2y).$$

Here $\alpha \in [0, 1]$ balances service quality and land cost. Use $\alpha = 0.7$ as the default value.

2.2 PSO Algorithm Overview

Particle Swarm Optimization is a population-based metaheuristic inspired by bird flocking. Each "particle" represents a possible solution (x, y), moving through the search space according to its own best-known position and the global best-known position.

Velocity and Position Updates

$$v_p(t+1) = w\, v_p(t) + c_1 r_1 [pbest_p - x_p(t)] + c_2 r_2 [gbest - x_p(t)]$$

$$x_p(t+1) = x_p(t) + v_p(t+1)$$

where:

- w: inertia weight (linearly decreases from 0.9 to 0.4)
- c_1, c_2: cognitive and social learning coefficients (typically 1.5)
- r_1, r_2: random values in $[0, 1]$

Parameter Settings

- Number of particles: 40
- Iterations: 200–300
- Velocity limits: $v_{\max} = 2.0$
- Bounds: $x, y \in [0, 11]$

2.3 Implementation Steps

1. Load demand node coordinates and weights.
2. Define $F(x, y)$ using the provided expressions.
3. Initialize particles uniformly within bounds and assign small random velocities.
4. For each iteration:
   - Evaluate $F(x_p)$ for each particle.
   - Update *pbest* and *gbest*.
   - Update velocities and positions using the above equations.
5. Plot convergence (fitness vs. iterations) and mark the best location on a city map.

2.4 Experiments and Discussion

- Run PSO for $\alpha = 0.9, 0.7, 0.5$ and observe the effect on location.
- Analyze how increasing the number of particles affects convergence.
- Discuss how normalization prevents one component from dominating.

2.5 Expected Output

- Optimal (x^*, y^*) coordinate.
- Breakdown of weighted distance and land cost contributions.
- Convergence plot of global best fitness.
- Map visualization showing demand nodes and final MFC location.

3 Task B: Multi-objective Optimization using NSGA-II

3.1 Problem Formulation

Select exactly $K = 3$ sites from 10 candidates (A_1–A_{10}) to minimize:

$$f_1(S) = \frac{\sum_{i=1}^{N} w_i \, \min_{s \in S} d(i,s)}{\sum_i w_i} \quad \text{and} \quad f_2(S) = \sum_{s \in S} (\text{acq\_cost}_s + \text{equip\_cost}_s)$$

Optional third objective (for extension): maximize coverage within radius R by minimizing

$$f_3(S) = -\text{covered\_demand}(S)$$

where covered_demand counts all demand nodes within distance R of any chosen site.

3.2 Representation

Each solution S is encoded as a binary vector of length 10, where bit $j = 1$ indicates that site A_j is selected. The constraint $\sum_j S_j = 3$ must be enforced (repair if necessary).

3.3 NSGA-II Parameters

- Population size: 100
- Generations: 150–200
- Crossover probability: 0.9
- Mutation probability: 0.1
- Tournament selection based on (rank, crowding distance)

3.4 Algorithm Outline

1. Initialize feasible population (each vector has exactly three 1s).
2. Evaluate f_1 and f_2 for each solution.
3. Perform non-dominated sorting to assign ranks.
4. Apply crossover and mutation, ensuring feasibility via repair.

5. Combine parent and offspring populations and select the next generation using crowding distance.
6. Repeat for all generations.

3.5 Analysis and Visualization

- Plot the Pareto front (f_2 vs. f_1).
- Identify "knee" points representing good trade-offs.
- For 3 representative solutions:
  - List chosen sites (e.g., $\{A3, A7, A10\}$).
  - Plot demand nodes and selected sites.
  - Report cost and average distance.

3.6 Comparative Study

- Compare NSGA-II with a weighted-sum approach (using PSO) and discuss missing non-convex solutions.
- Test algorithm stability by running multiple times and comparing Pareto fronts.

4 Where PSO and NSGA-II Apply

4.1 PSO

Effective for:

- Continuous, real-valued problems (e.g., facility location, parameter tuning).
- Smooth or moderately noisy objective landscapes.

Not suitable for:

- Highly discrete or combinatorial problems (without binary adaptation).
- Problems with complex feasibility constraints.

4.2 NSGA-II

Effective for:

- Multi-objective continuous or combinatorial problems.
- Non-convex Pareto fronts and conflicting objectives.

Not suitable for:

- Problems requiring strict optimality or deterministic guarantees.
- Extremely high-dimensional search spaces without domain heuristics.

5 Implementation Tips

- Normalize objectives to ensure comparable scaling.
- Use random seeds for reproducibility and run multiple trials.
- Always visualize intermediate progress (PSO swarm or Pareto front evolution).
- Apply repair operators for constraints (e.g., enforce exactly K selected sites).
- Use available libraries for convenience:
  - `pyswarms` for PSO (https://pyswarms.readthedocs.io)
  - `pymoo` or `DEAP` for NSGA-II (https://pymoo.org)

6 Expected Results

- PSO identifies an optimal MFC location near the central demand cluster, balancing service and land price.
- NSGA-II produces a diverse Pareto front showing trade-offs between installation cost and service quality.
- Students should interpret and justify their selected trade-off solution.

7 Deliverables

1. Source code files (`.py` or `.m`) with a README.
2. A short report (5–6 pages) containing:
   - Problem formulation and objective equations.
   - Algorithm parameters and implementation details.
   - Plots: PSO convergence, Pareto front, and city maps.
   - Comparative discussion of results and limitations.

8 Recommended References

- Kennedy, J., Eberhart, R. (1995). *Particle Swarm Optimization*, Proc. IEEE Int. Conf. Neural Networks.
- Deb, K., Pratap, A., Agarwal, S., Meyarivan, T. (2002). *A Fast and Elitist Multi-objective Genetic Algorithm: NSGA-II*, IEEE Trans. Evol. Computation.
- Pymoo Library Documentation: https://pymoo.org/