Metaheuristic algorithms

Lab 5: Ant Colony Optimization. Particle Swarm Optimization. Differential Evolution

---

1. **Ant Colony Optimization (ACO)**

ACO is a metaheuristic inspired by the behavior of the ant colonies. It is especially used in solving combinatorial optimization problems (e.g. routing, scheduling, assignment) It uses a population of artificial ants (agents) which is changed during an iterative process. At each iteration each ant constructs, component by component, a potential solution. The values for the solution components are chosen randomly based on a probability distribution. The probability distribution is computed by using both local information (what the ant can collect from its neighbourhood) and global information (obtained by using the indirect communication process between ants based on pheromone trails).

Solving TSP using ACO. The input data consists of the graph describing the direct connections between towns and their costs. A population of ants is initially placed on random nodes (or all of them in the first node). At each iteration, each ant visits *n* distinct nodes, constructing a tour. The ants have a local memory where the list of visited nodes is stored in order to avoid visiting twice the same node. The transition of an ant *k* from the node *i* to the node *j* at step *t* is based on the following probability:

$$P_t^k(i,j) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{l\in N(k)}\tau_{il}^{\alpha}\eta_{il}^{\beta}} & j \text{ was not visited by ant k} \\ 0 & j \text{ was visited by ant k} \end{cases}$$

The factors appearing in the computation of the probability are:

- τ_{ij}: quantifies the pheromone concentration released by the ants on edge *(i,j);* the pheromone concentration is randomly initialized with small positive values. Each ant which visits an edge *(i,j)* can release some pheromone on it contributing to the update of the pheromone concentration:

  $$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\sum_k Q_{ij}(k)/\text{cost}(T_k)$$

  ρ is a constant less than *1* which controls the evaporation process, $Q_{ij}(k)$ is *0* if *(i,j)* does not belong to the tour constructed by ant *k*. $Cost(T_k)$ denotes the cost of the tour constructed by the ant *k*.
- η_{ij}: models the local information concerning the quality of the edge; the simplest variant is when it is *1/cost(i,j).*
- α and β are parameters which control the relative importance of those two types of information: the global information provided by the pheromone concentration and the local one provided by the cost of the edge.
- *N(k)* denotes the neighborhood of node *i* and contains the nodes which can be reached from node i and have not been visited yet.

Application 1. Implement an ACO algorithm for TSP.
Hint. See function ACO_TSP.sci

Exercise. Change the previous implementation such that when the pheromone matrix elements are updated, the tours visited by all ants are taken into account.
Hint. The updating terms are cumulated after each tour construction.

2. Particle Swarm Optimization (PSO)

PSO is a metaheuristic used for continuous function optimization inspired by the behavior of bird swarms. It uses a population of *m* "particles", each particle *i* being characterized by its position (x_i) and its velocity (v_i). Moreover, each particle memorizes the best position it visited up to the current moment ($xbest_i$). There is also another variable which contains the best position found up to the current iteration by the entire swarm (*xbest*). The evolutionary process consists in the change, at each generation *t*, of the position of all particles in the population according to the following rules:

$$v_i(t+1) = \gamma(v_i(t) + r_1 rand(0,1)(xbest_i - x_i(t)) + r_2 rand(0,1)(xbest - x_i(t)))$$
$$x_i(t+1) = x_i(t) + v_i(t+1)$$

where:

- gamma is a constriction factor (a typical value for gamma is 0.7)
- r_1 and r_2 are two constant values (e.g. $r_1=r_2=2.05$)

Besides this variant, where *xbest* is the global best element from the swarm, there is also another variant where for each particle *i*, *xbest(i)* is selected as the best element from the neighborhood of the particle *i*. The neighborhood of a particle can be defined by various topologies, one of the most used is the ring topology (in this case the neighborhood of size K of particle i is represented by the particles having the indices *{i-K,i-K+1,...,i-1,i,i+1,...,i+K-1,i+K}*).

Application 2. Implement a PSO algorithm (using the above eqs.) and test its behavior for a unimodal function (e.g. sphere) and for a multimodal function (e.g. Griewank).

Hint. See function PSO.m

Exercise. Change PSO.m such that it implements the "local best" variant using a ring topology to define the neighbourhood.

3. Differential Evolution (DE).

DE is a popular optimization technique based on a simple rule of constructing new candidates by using differences between elements of the current population. The basic idea is to construct for each population element x(i) a new trial element following the steps:

- Construct a "mutant" vector, y, by combining several elements of the population. Two of the most used approaches are:
  - DE/rand/1/bin: y=x(r1)+F*(x(r2)-x(r3)) where r1,r2,r3 are distinct random indices
  - DE/best/1/bin: y=x(ibest)+F*(x(r2)-x(r3)) where x(ibest) is the best element of the population

  Rmk: in both cases, F is a scale factor taking values in (0,2).

- Construct a trial element, z, by crossing over the components of the mutant y with those of the current element, x(i), by following the rule (known as binomial crossover):
  - z(j)=y(j) with probability CR
  - z(j)=x(i,j) with probability 1-CR

  Rmk: CR (with values in (0,1)) is a crossover probability

Application 3. Implement the DE algorithm (both variants) and analyze its performance in comparison with that of an evolution strategy (see lab 3). Hint: see DE.sci