ISE 5113 Advanced Analytics and Metaheuristics
Final Exam

Instructor: Charles Nicholson

Due: See course website for due date

Requirement details

1. To be completed in teams of one or two. If team members disagree on an answer, you can record solutions corresponding to each member (please clearly mark which solution belongs to which team member).
2. Your primary submission will be a single Word or PDF document and must be of professional quality: clean, clear, concise, yet thoroughly responsive.
3. Any code (e.g., Python) must also be submitted separately. Your code MUST be well-documented. Failure to submit the files will result in a penalty. For this problem you may submit separate code for each problem $2, \ldots, 9$, or submit a single file with multiple functions associated with each coding problem.
4. You cannot use preexisting Python packages for heuristics or metaheuristics. In fact, other than `numpy, copy, random`, and other basic utilities, you should seek specific permission from the instructor if you have any doubt. That is, *you* are responsible for creating the logic yourself.
5. There are 15 points of extra-credit included in this assignment.

You will develop Python code to implement a Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) algorithm for the Schwefel minimization problem.

The Schwefel problem is a generalizable benchmark problem than can be formulated in $n > 0$ dimensions.

Figure 1 presents the formula and a 2D representation of the solution landscape. The feasible region is an n-dimensional hybercube centered at the origin with possible values ranging from -500 to 500 for each dimension. The optimal objective and solution is known.

$$f(x) = 418.982887272433n - \sum_{i=1}^{n} x_i \sin(\sqrt{|x_i|})$$

Dimensions:	n
Domain:	$-500.0 \leq x_i \leq 500.0$
Global Optimum:	$f(x) \approx 0.0$ at $x = (420.9687, 420.9687, \ldots, 420.9687)$

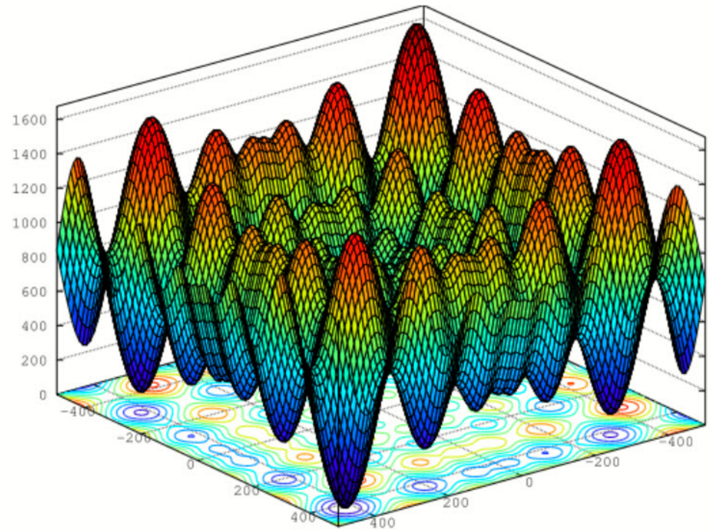

Figure 1: Schwefel function in 2D

Question 1: GENETIC ALGORITHM IMPLEMENTATION (47 points)

Some basic Python code is available to help you begin complete a GA implementation for the Schwefel minimization problem. The code however is incomplete and requires multiple improvements.

(a) (24 points) Finalize the code.

i. (6 points) Create code to generate chromosomes in the initial population.

ii. (6 points) Create code to mutate chromosomes.

iii. (6 points) Create logic to implement crossover rate.

iv. (6 points) Implement some type of elitism in the insertion step.

(b) (13 points) Solve the 2D Schwefel.

i. (7 points) Create a small population of 6 chromosomes and depict their locations on a graph for the initial population and after the first and second generations.

ii. (6 points) Using the 6 chromosome population, solve the problem as best as possible. Provide information on the quality of the best solution and the total number of solutions evaluated (e.g., population size $\times$ iterations)

(c) (10 points) Solve the 200D Schwefel.

i. (10 points) Solve the 200D problem as best as possible (you may choose whatever population size you wish). Report your best solution and the total number of solutions evaluated (e.g., population size $\times$ iterations)

ii. (5 points) EXTRA CREDIT: Produce a chart that tracks the population quality – both average objective value of the population and the variance of the objectives across generations.

Question 2: Particle Swarm Optimization (53 points)

Some basic Python code is available to help you begin a PSO implementation for the Schwefel minimization problem.

(a) (30 points) Complete the original PSO implementation based on "particle best" and "global best". Make sure to address the following elements:

  i. (7 points) velocity update function
  ii. (2 points) velocity max limitations
  iii. (7 points) position updates
  iv. (7 points) dealing with infeasible space
  v. (7 points) main loop, stopping criterion, etc.

  Note: inertial weights and/or constriction factors are not required.

(b) (13 points) Solve the 2D Schwefel with the original PSO.

  i. (7 points) Create a swarm of size 5 and depict their locations on a graph for the initial random set and the first generation.
  ii. (6 points) Solve the problem as best as possible and provide information on the quality of the best solution and the performance of the approach overall.

(c) (10 points) Solve the 200D Schwefel problem as best as possible (you may choose whatever swarm size you wish). Report your best solution and the total number of solutions evaluated.

(d) (15 points) EXTRA CREDIT: Implement a PSO algorithm that uses the "local best" in place of the global best. You may also add inertia weighting or constriction if you like. Implement one of the following neighborhood topologies: ring, star, or von Neumann. Use this to solve the 200D Schewefel problem and compare the results with the original PSO implementation.