BIS

Index

	Date	Title	Sign
1)	24/10/24	Genetic Algo	
2)	7/11/24	Particle swarm Algo	
3)	14/11/24	Ant colony opt	
4)	21/11/24	Cuckoo search	
5)	28/11/24	Grey wolf opt	
6)	18/12/24	Parallel cellular	
7)	18/12/24	Optimization via gene expression	

24/10/24

Genetic Algorithm for Optimization

Steps -

*) Initilization - Randomly initilize a population of candidate solution
*) Selection - Select indivisuals based on their ~~fac es~~ fitness.
*) Crossover - combine pairs of individuals to form new offsprings.
*) Mutation - Randomly tweak offspring to maintain diversity in the population.
*) Replacement - Replace part of the old population with the new offsprings.
*) Termination - Repeat the process until a stopping condition is met.

Pseudocode

```
Function GeneticAlgo.        // initilization (Parameter).
  SET population_size = 100
  SET mutation_rate = 0.01
  SET num_generation = 50
  SET x_bound = (-10, 10).

//Initilization.
population = Randomly_Generate population_size
                within x-bounds.

For generation from 1 to num_generations.
    fitness_values = Evaluate (population).
        //fitness evaluation.

//selection.
selected_parents = select (population, fitness_values)
```

f(x) = x²

// Crossover and Mutation
new_population = []
for each pair of selected parents
(child1, child2) = crossover(parent1, parent2)
child1 = Mutate(child1)
child2 = Mutate(child2)
Append child1 and child2 to new_population

//Replacement
population = new_population

// Return solution
Return solution(population)

Main points
*) Initialize a population of candidate solutions
*) Evaluate their fitness
*) Select the fittest individuals to create new offspring
*) Apply crossover & mutation to produce the next generation.
*) Repeat the process for a specified no. of generations
*) output - best solution.

Write all steps completely

Op - Best_soln - X = -9.99137...
Best_fitness = 99.8275...

24/09/24

7/11/24

1

Particle Swarm Optimization

PSO algo inspired by the social behavior of birds flocking or fish schooling.
*) It is used to find approximate solution to complex optimization prob by simulating the movement of particles through a multidimensional search space.

Algo
*) define function which is needed to optimize.
*) ~~Calculations of pso~~
*) Initialize parameter:
P -> no of particles
D -> no. of dimension
T -> no. of iterations
c_1 -> (cognitive coefficient) -> own influence / Pbest
c_2 -> social coeff
c_3 -> inertial weight. -> controls momentum

*) initialize ~~pso~~ particle:-
particle position -> randomly initilize
particle velocity -> randomly initialize in range

*) Determine global best
gbest - particle with best fitness
set gbest to the position of the particles with best fitness.

*) in loop from 1 to T
-> update velocity: generate r_1, r_2
$c = c_1 * r_1 * (Pbest - x_i)$
$s = c_2 * r_2 * (gbest - x_i)$
$v_i = w * v_1 + c$
update pos: $x_i = x_i + v_i$

calculate fitness of new pos x_i
Update Pbest if new fitness is greate & also gbest

*) Print the optimal solution

Summary
*) PSO is useful for solving complex optimization problems.
*) It works by initializing a group of particles with random pos & velocity.
*) Each particle evaluates its own position & tracks pbest
*) Swarm tracks -> gbest.
*) particles update their pos based on Pbest & gbest
*) Influenced by inertia (momentum) & cognetive (individual learning) coeffs.
*) Over multiple iterations the swarm converge towards the optimal solution.

Write pseudocode properly
10/11/24

Op:- Best pos = {1.9408, -1.1133}
Best fitness = 5.0055.

11/11/24

LAB-3

Ant Colony optimization for Travelling Salesman Problem

Algo

1) Represent the cities as nodes in a graph & construct a distance matrix.

2) Initial parameters:
m -> no of ants
α -> importance of pheromone
β -> heuristic information
ρ -> pheromone evaporation rate
Q -> " deposit constant.

3) for each ant:
*) Randomely select start city
*) Build a complete tour by iteratively selecting next city
↳ for each unvisited city j calculate probability of moving from i to j
*) Use the probability to select next city
*) Add selected city to tour & mark as visited.

4) Calculate total length of each tour & keep track of best tour found so far.

5) Update pheromones:
pheromones * = (1-ρ)
pheromone increase = Q / tour_length

Pseudocode

1) Initialize parameters:
   - No. of ants
   - No of iteration
   - Pheromone matrix with small initial values
   - α for pheromone influence
   - β for the heuristic influence
   - evaporation rate (ρ)

2) Initilize pheromon matrix & distant matrix
   - set initial & compute dist.

3) Repeat for each iteration
   i) place the ants at random city
   ii) construct a complete tour
   - calculate probabilities
   - calculate the length of tour

5) update pheromones -
   - evaporate pheromones
   - reinforce pheromones

4) Return the best solution

Op - Best route - {3, 1, 0, 2, 4, 3}
Best dist - 22. so ...

21/11/24

LAB-4

Cuckoo search

Pseudocode

```
initialize_parameter() {
    num_nests = no. of nests
    max_iterations = Max no of iteration
    pa = Discovery rate of alien eggs
    low_bound, upper_bound = search space bounds
}.
# Generate initial popu
nests - Randomly generate num nests
solutions within bounds.

# Evaluate fitness of each solution
fitness = Evaluate fitness for each
    solution in nests

# Iterate to improve solution
    for iteration in
    range (max_iterations):
        new soln = Generate a new soln by levy flight
            from a random nest.
        Evaluate
# Replace the old solution if new is better
        new_fitness = Evaluate fitness of new_solution.

# Replace the old solution if new is better
    if new_fitness is better than fitness of chosennest:
        Replace the chosen nest with new_solution
```

Abandon some nests with probability pa & replace them
for each nest:
if Random probability < pa:
replace nest with a new random soln.

Update the best soln found
update best_solution if new best fitness is found

Return the best soln & its fitness
return best_solution,
best_fitness.

Op - Best soln - [-0.6366, 0.657]
Best fitness - 0.83822..

28/11/24

LAB-5

Grey Wolf Optimizer

Pseudocode

1) Initialize parameters:
- population size
- no. of iterations
- lows & upper bounds
- Objective function to minize

2) Randomly initialize the positions of grey wolves within the search space.

3) Evaluate the fitness of each wolf & identify:
- Alpha (best soln)
- Beta (second best soln)
- Delta (third best soln)

4) Update the position of each wolf using the hunting mechanism:
- compute the coefficient A & C:
A = 2a * random() - a
C = 2 * random()
- calculate the position updates - α, β, Delta
D_alpha = |C1 * Alpha_position
- current_position|
X1 - Alpha_position - A1 * D_alpha
New_pos = (X1 + X2 + X3)/3

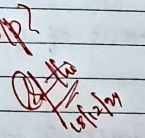
5) Clip the position to remain within bound.

6) Evaluate the fitness of updated pos & update Alpha, Beta & Delta accordingly.

7) Reduce the parameter 'a' linearly from 2 to 0 over iterations.

8) Repeat steps 4-7 until the stopping condition is met

9) Return Alpha (best solution) & its fitness



op - optimal fitness = 2.5816773
optimal soln = [-1.08909 -15, -1.181 -15]

18/12/24

LAB-6

Parallel Cellular Algo & Program

Pseudocode

1) Initialize parameters:
   - population size
   - Max iterations
   - Mutation rate
   - crossover rate
   - neighbourhood structure (2D grid)
   - Bound of the search space

2) Randomly initialize the population within the search space

3) Evaluate the fitness of each individual in the population

4) While stopping cond not met:
   a) for each individual in the popu
      i) select parent from the individual's neighbourhood using a selection strategy
      ii) Perform crossover b/w the selected parent
      iii) Mutate the offspring
      iv) Replace the individual

   b) sync update across the population

5) Return the best soln & its fitness

Objective function - $f(x) = \sum x_i^2$

op - Best soln - [-0.5305, 0.452]
Best fitness 0.2002455.

18/12/24

LAB - 7

Optimization via Gene Expression Algo

Objective fuction $f(x) = \Sigma x_i^2$

1) Initialize parameter:
   - population size
   - no. of generations
   - Mutation rate
   - Crossover rate
   - Gene length & no of genes
   - objective funct to min or max.

2) Randomly generate the initial popn of chromosons:
   - each chromosone contains a seq of genes
   - Decode each chromosom to produce an expression tree or candidate soln.

3) Evaluate the fitness of each chrosone using the objective function

4) Repeat for each generation:
   a) select parent based on fitness
   b) Apply genetic operations:
      i) Mutation
      ii) crossover
      iii) Reproduction
   c) Decode offspring into new soln & evaluate their fitness.

5) Return the best soln & its fitness after max. generation.

18/12/24

Op - Best soln = -4.872922
fitness = 23.7453