LAB-5 22/10/24

Simulated Annealing Algorithm

Step-1 Initialisation

- Choose initial soln 'S'
- Set initial temp 'T'
- Set a stopping criteria (target achieved)

Step-2: Iteration

```
while( stopping criteria not met)
{
    - Generate neighbours: Create neighbouring solution "S'" from 'S'.
    - Calculate energy difference:
      Compute cost of E(S) and neighbouring E(S').
```

Step-3: Acceptance criteria:

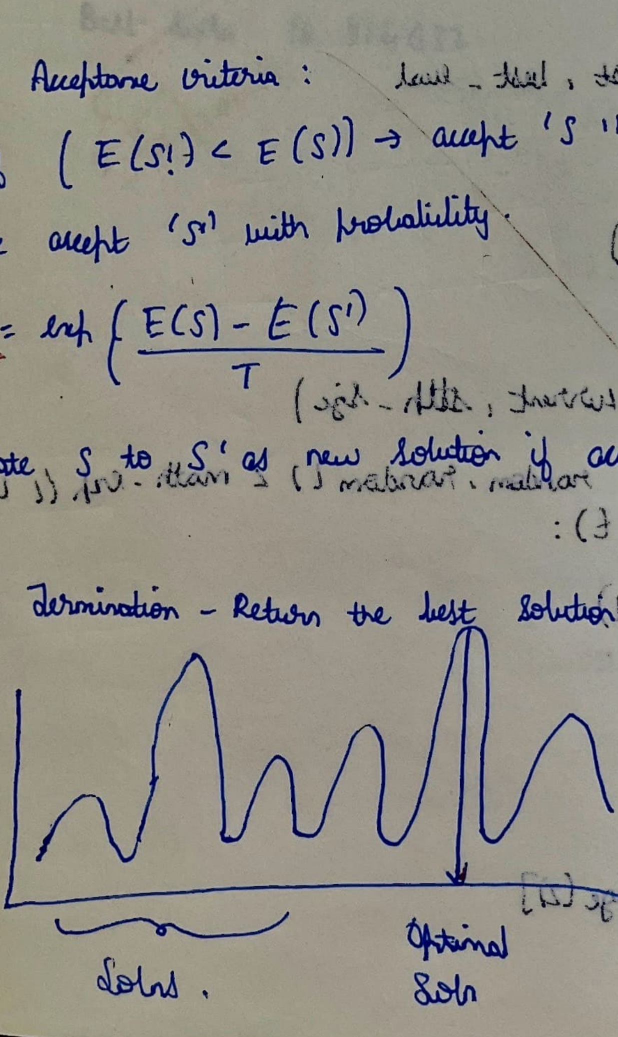
If $(E(S') < E(S))$ → accept 'S'' as new solution

else accept 'S'' with probability

$$P = \exp\left(\frac{E(S) - E(S')}{T}\right)$$

update S to S' as new solution if accepted

Step-4 Termination - Return the best solution found.



Proceed

Code -

```
import math
import random

def objective _function (x)
    return 10* len (x) + sum ([xi ** 2 - 10 * math. cos (2* math.pi * xi
        for xi in x])

def get_neighbour (X, step_size = 0.1):
    neighbour = x[:]
    index = random. randint (0, len(x) -1)
    neighbour [index] += random. uniform (- step_size, step_size)
    return neighbour

def simulated ( objective, bounds, n_iterations, temp) :
    best = [random. uniform ( bound[0], bound[1]] for bound in bounds]
    best _eval = objective (best)
     current, current _eval = best, best _eval
     scores = [best - eval]

  for i in range (n_ iterations )
     t = temp / float (i+1)
     candidate = get _neighbour (current, step _size)
     if eval < best_eval or random. random () < math. exp ((current
     eval - candidate_eval) / t):
     scores. append (best _eval)

  if i %100 == 0 :
     print ( )

bounds = [(-5.0, 5.0) for _ in range (2)]
n_ iterations = 1000
```

```
step-size = 0.1
temp = 10
print()
print()
```

OUTPUT -

```
Iteration 0    Temperature 10.0    Best evaluation 41.275
          100              0.099                   16.954
          200              0.050                   16.926
          300              0.033                   16.916
          400              0.025                   16.916
          500              0.020                   16.915
          600              0.017                   16.915
          700              0.014                   16.914
          800              0.012                   16.914
          900              0.011                   16.914

Best Solution: [3.98054, 0.9962213]
Best Score: 16.914632
```

22/10/24