21/10/24

Bafna Gold
Date: Page:

Simulated Annealing:

Algorithm:

1. Initialize T = T_initial; x_best = x_current; and T_min with some minimum value near to global minimum.

2. while T > T_min:
   for i = 1 to max_iterations:
   // generate new_solution x_new by modifying x_current.

   ΔE = f(x_new) − f(x_current)
   if ΔE < 0:
   x_current = x_new
   else:
   // Probability
   P = exp(−ΔE/T)
   if accepted, set
   x_current = x_new

   if f(x_current) < f(x_best):
   x_best = x_current

   T = α * T
   ↳ cooling rate.

3. // Return x_best as the final solution.

Prod
22/10/24

CODE:

```
import random
import math

def objective_function(x):
    return x**2

def simulated_annealing(initial, initial_temp,
        cooling_rate, max_iterations):
    current_solution = initial
    current_temp = initial_temp

    best_solution = current_solution
    best_value = objective_function(best_solution)

    current_value = best_value

    for i in range(max_iterations):
        new_solution = current_solution +
            random.uniform(-1, 1)
        new_value = objective_function(new_solution)

        delta_value = new_value - current_value

        if delta_value < 0 or random.random()
            < math.exp(-delta_value / current_temp):
            current_solution = new_solution
            current_value = new_value

        if new_value < best_value:
            best_solution = new_solution
```

```
        best_value = new_value

    current_temp *= cooling_rate
return best_solution, best_value

best_solution, best_value = simulated_annealing():
print("The best x: {}, It's corresponding f(x):
      {best_value}, Temparature: {temp}")
```

OUTPUT:

The best x: 0.0024631 , It's correspondly f(x) = 8.780101300e-6 Temperature: 1.86375×10^{-6}.

20/10/24