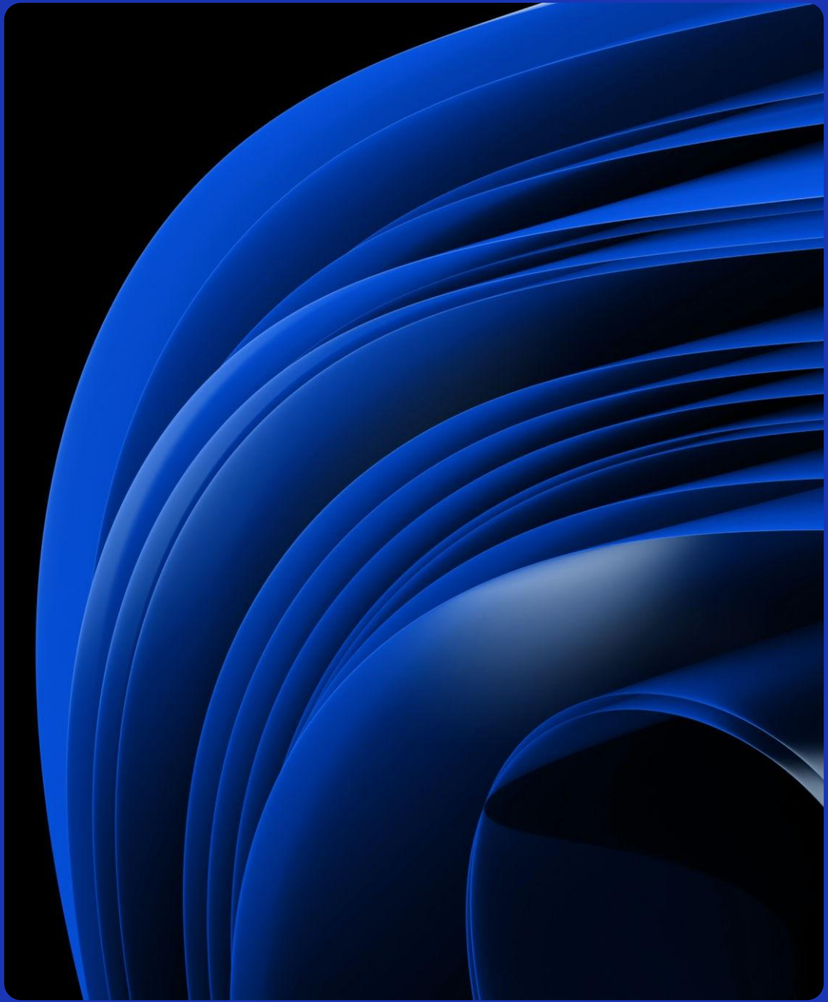
Artificial Intelligence - Group 1

Travelling Salesman Problem

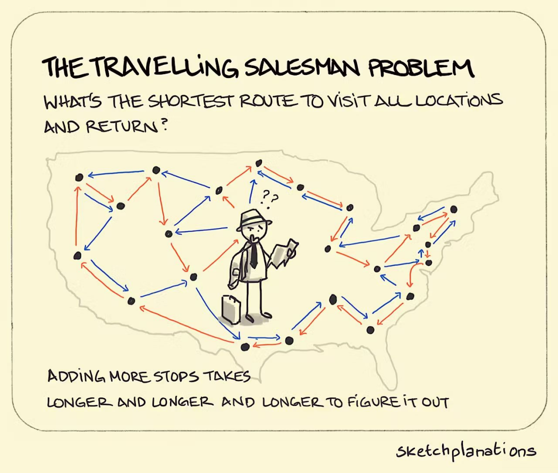
25/03/2025



Background

You are a salesperson, and you have a list of cities you want to visit. You wish to travel to each city exactly once then return home. What order should you visit the cities to minimize the total distance you travel?

The Traveling Salesman Problem (TSP) is a well-known optimization problem in computer science and operations research. It involves finding the shortest possible route for a salesman who must visit a set of cities exactly once and return to the starting point. TSP is widely studied due to its real-world applications in logistics, supply chain management, vehicle routing, and circuit board manufacturing.



The Ant Colony System (ACS) is an optimization algorithm inspired by the foraging behavior of real ants. It is a type of Ant Colony Optimization (ACO), a nature-inspired metaheuristic used to solve combinatorial optimization problems like the Traveling Salesman Problem (TSP)

ACS is based on how ants deposit and follow pheromones to find the shortest path between food sources and their nest. In an artificial system, artificial ants move through a solution space, updating a pheromone trail to guide future ants toward promising solutions.

Ants select the next city j from the current city i using the following probability formula:

$$P_{ij} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{k \in \mathcal{N}_i} [\tau_{ik}]^{\alpha}[\eta_{ik}]^{\beta}}, & \text{if } j \in \mathcal{N}_i \\ 0, & \text{otherwise} \end{cases}$$

where:

- P_{ij} = Probability of moving from city **i** to city **j**.
- τ_{ij} = Pheromone level on edge **(i, j)**.
- η_{ij} = Heuristic information, typically **1/d_{ij}** (where d_{ij} is the distance between cities **i** and **j**).
- α = Controls the influence of pheromone.
- β = Controls the influence of the heuristic distance.
- $\mathcal{N}_i$ = Set of unvisited cities.

Pheromone Update Rules

Local Pheromone Update (During Ant Movement)

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_0$$

- ρ = Evaporation rate $(0 < \rho < 1)$
- τ_0 = Initial pheromone level.

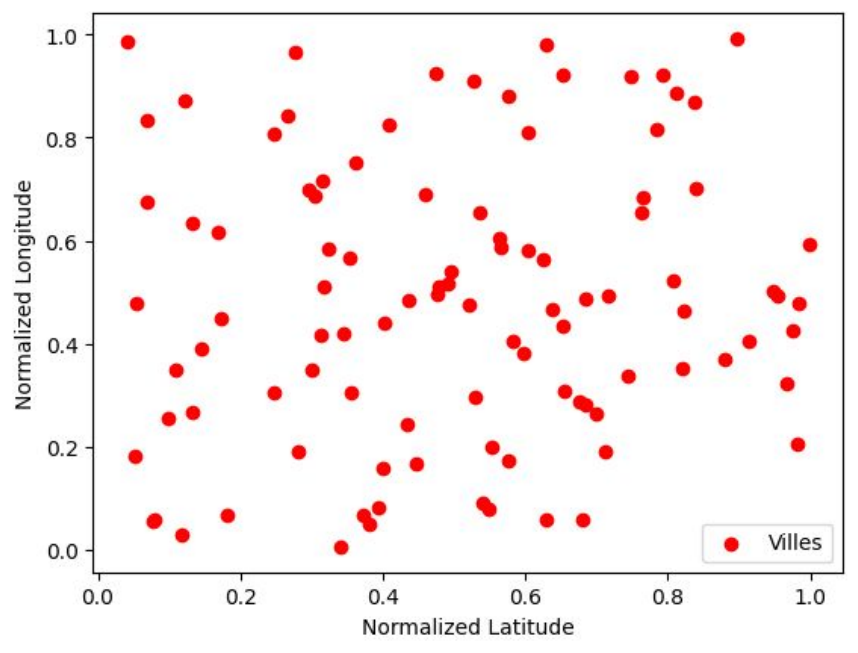
Global Pheromone Update (After All Ants Complete the Tour)

$$\tau_{ij} \leftarrow (1-\alpha)\tau_{ij} + \alpha\Delta\tau_{ij}$$

- α = Learning rate $(0 < \alpha < 1)$.
- $\Delta\tau_{ij} = \frac{1}{L^*}$, where L^* is the length of the best tour found so far.

The salesperson lives in a boring, flat, 2D Cartesian plane, and that each city can be described simply as existing at a single (x, y) location. Each datafile describes some number of cities, one city per line in the file, where each city has a index (first city is index-0) and a location

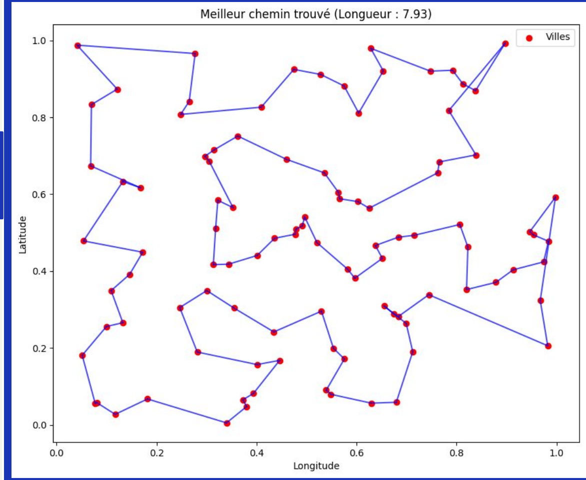
This dataset contains 100 cities.



Ant Colony System

Data

Discussion



5 Key Parameter Effects

Number of Ants (N_ANTS):
30 ants: 8.67 path length → 60 ants: 7.93 (8.5% better)
More ants = better solution space exploration

Pheromone Influence (ALPHA = 1):
Controls how strongly ants follow existing trails
α = 1 balances exploration vs. exploitation

Distance Influence (BETA = 2):
β > α prioritizes shorter connections over pheromones
Prevents random wandering, guides toward promising paths

Evaporation Rate (0.5):
50% pheromone decay prevents getting stuck
Enabled late improvements (8.54 → 7.93 after iteration 40)

Convergence Pattern:
Rapid initial improvement: 18.07 → 9.21 (10 iterations)
Middle refinement: 9.21 → 8.54 (by iteration 40)
Late breakthrough: 7.93 at iteration 60

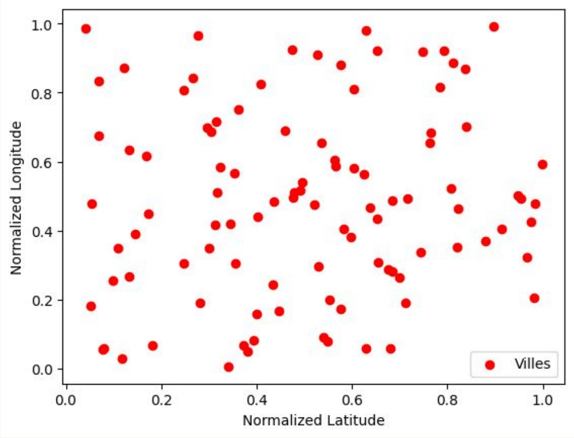
Conclusions

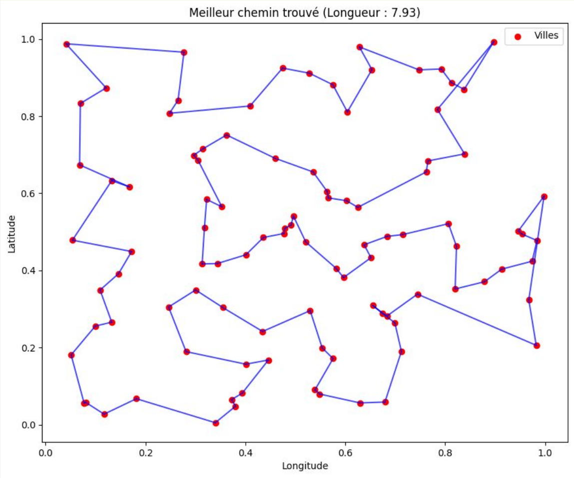
Successfully implemented Ant Colony Optimization for the TSP with 100 cities

Achieved significant improvement: 18.07 → 7.93 (56% reduction in path length)

Key findings:

- Doubling ants (30→60) yielded better solutions through increased exploration
- Parameter balance (α=1, β=2) critical for algorithm effectiveness
- Evaporation rate of 0.5 enabled continuous improvement beyond iteration 40





Conclusions

Successfully implemented Ant Colony Optimization for the TSP with 100 cities

Achieved significant improvement: 18.07 → 7.93 (56% reduction in path length)

Key findings:

- Doubling ants (30→60) yielded better solutions through increased exploration
- Parameter balance (α=1, β=2) critical for algorithm effectiveness
- Evaporation rate of 0.5 enabled continuous improvement beyond iteration 40

```
Aperçu des données :
          0         1
0  0.417992  0.344112
1  0.058938  0.679530
2  0.478459  0.053815
3  0.521159  0.806604
4  0.580632  0.602950
Iteration 0: Best Path Length = 18.07
Iteration 10: Best Path Length = 9.21
Iteration 20: Best Path Length = 8.68
Iteration 30: Best Path Length = 8.54
Iteration 40: Best Path Length = 8.54
Iteration 50: Best Path Length = 8.40
Iteration 60: Best Path Length = 7.93
Iteration 70: Best Path Length = 7.93
Iteration 80: Best Path Length = 7.93
Iteration 90: Best Path Length = 7.93
Iteration 100: Best Path Length = 7.93
Iteration 110: Best Path Length = 7.93
Iteration 120: Best Path Length = 7.93
Iteration 130: Best Path Length = 7.93
Iteration 140: Best Path Length = 7.93
Iteration 150: Best Path Length = 7.93
Iteration 160: Best Path Length = 7.93
Iteration 170: Best Path Length = 7.93
Iteration 180: Best Path Length = 7.93
Iteration 190: Best Path Length = 7.93
```