Assignment 3 Report

Question 1

a

> 💡 [4 marks]For part (a) provide a summary of the representation for the solutions. Justify your choice.

This problem will be represented as a real-valued GAS.

A given phenotype can be represented as,

$$x =< x_1,\ x_2,\ x_3 >$$

where each gene can represents a given real-valued PID parameter;

$$x_1 = k_p,\ x_2 = T_I,\ x_3 = T_D.$$

Each gene has their own range of valid values,

$$2.00 < k_p < 18.00,$$

$$1.05 < T_I < 9.42,$$

$$0.26 < T_D < 2.37.$$

The real-valued GA representation was chosen over binary GAS as it is much simpler to work within python and provided faster performance.

b

💡 [5 Marks] For part (b) provide a summary of the fitness function that you used to evaluate a solution. Justify your choice.

The fitness function for the PID controller is a function of the ISE (integral squared error), the rise time t_r, the settling time t_s and the maximum overshoot percentage M_p. The fitness is the reciprocal of the sum of all these values. It was difficult to weight and normalize each output parameter accurately, therefore a simple sum was used instead.

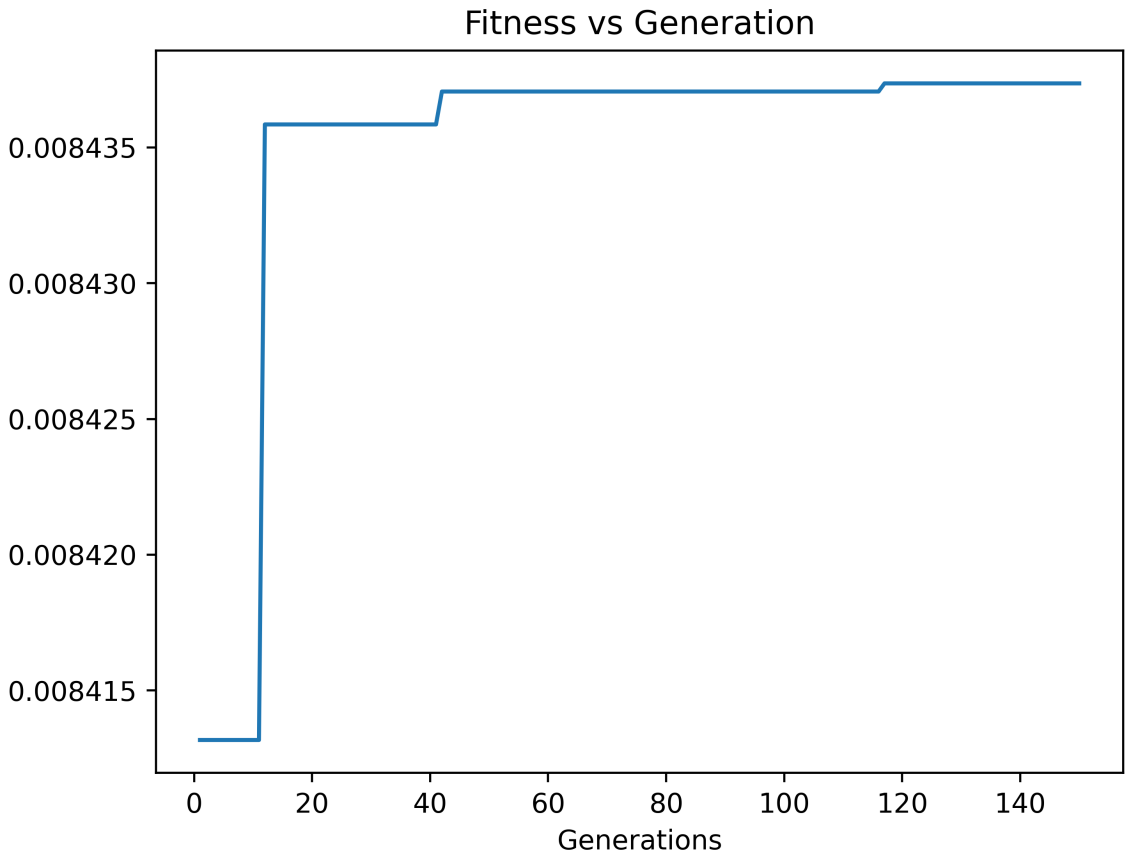
Each solution will have a fitness as follows,

$$fitness = \frac{1}{ISE + t_r + t_s + M_p}.$$

The reciprocal was taken as all the parameters should be minimized for a better performing controller, however, the genetic algorithm is defined to be maximizing. In the PID controller problem, the better solution is the one with the minimum of the output parameters.

c

💡 [20 marks] For part (c) provide a Plot(s) of the fitness of the best solution in each generation throughout the generations. That is the x-axis is generation no., the y-axis is the fitness value of the best solution in that generation.

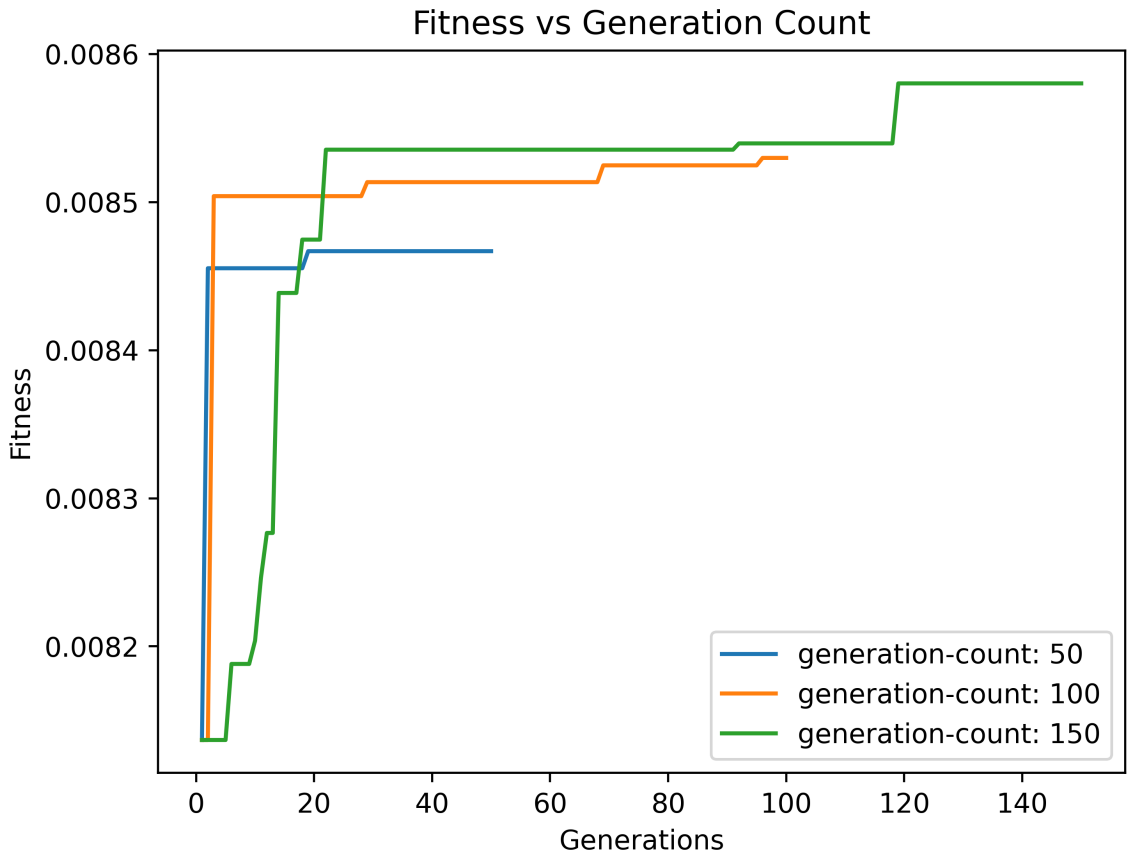


The graph shows the best fitness/generation for the default parameters presented in the question.

d

i

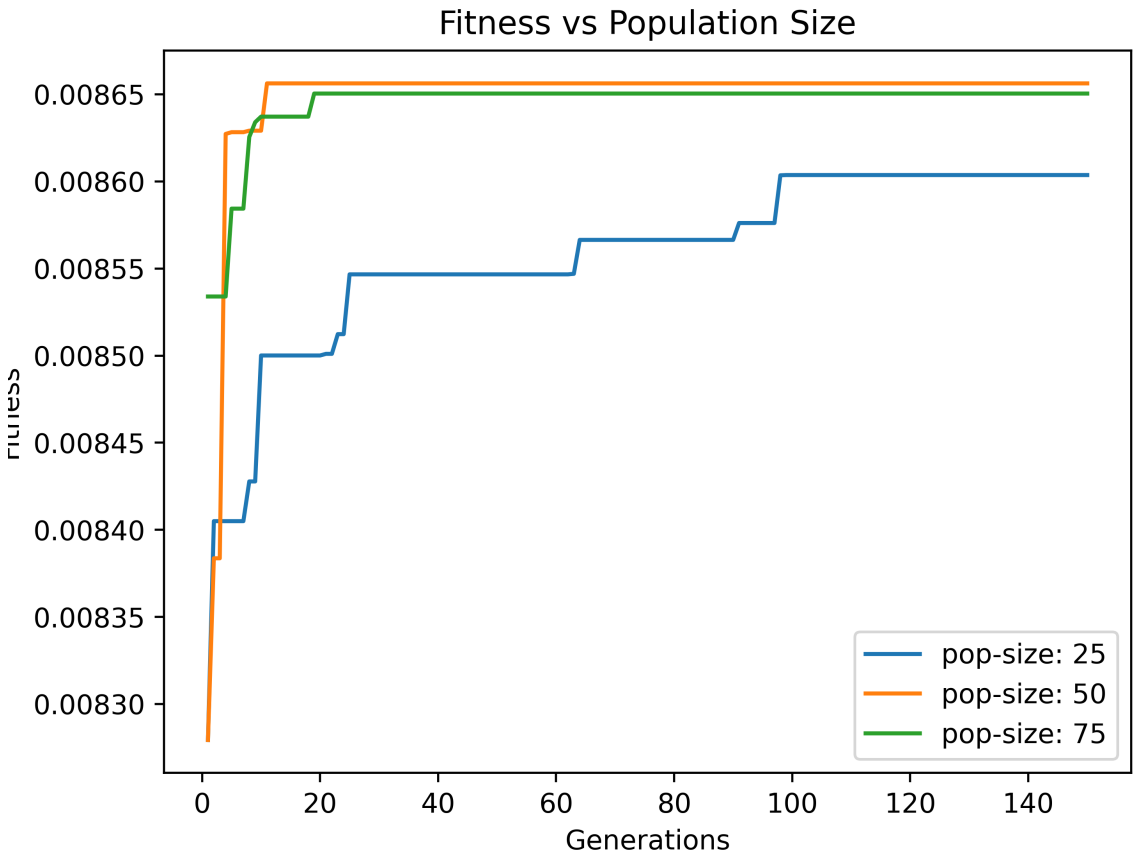
> 💡 [7 marks] Plot the progress of the solution fitness as a function of the 3 different values for the number of generations. Provide your insight into the behaviour of the search as the number of generations is increased.



As the generation count increases, the system stabilizes to better values. This can be seen when comparing a generation size of 150 to a generation size of 50. This is due to the fact that with more generations, there is more opportunity for individuals and the overall population to get better genes via crossover and mutation, which may lead to better fitness.

ii

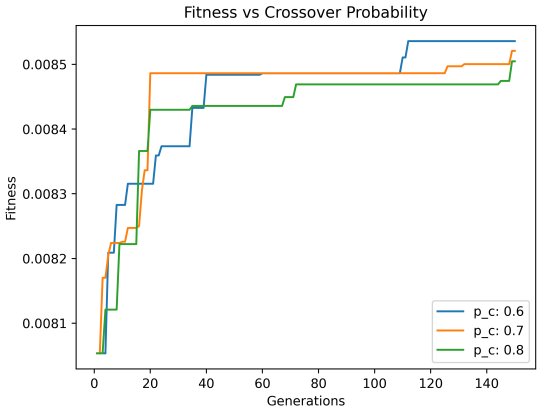
💡 [7 marks] Same as in (i)but for the population experiment.

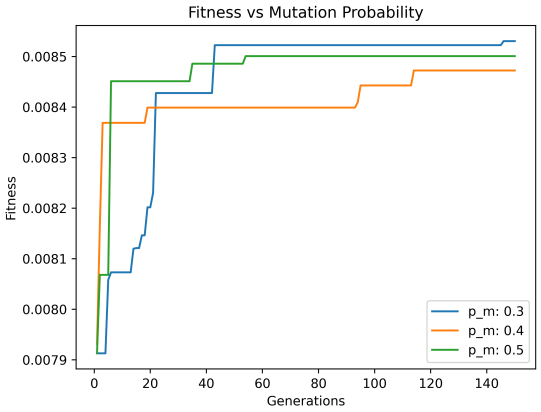


Populations with a size that is too small will pre-maturely converge to worse solutions. The reason for this is due to the fact is that there are not enough genes in the mating pool to provide encompass all the gene values. Additionally, since uniform crossover was used, the genes from crossover only swap known genes as opposed to arithmetic crossover which combines two genes to create a median gene. This allows worse solutions to converge (unless mutation helps create better solutions in the specified generations count).

iii

> 💡 [7 marks] Same as in (ii)but for the crossover/mutation experiment.





The changes in crossover probability seem minimal between the different values. The lower crossover probabilities converge faster, as can be seen when comparing $p_c = 0.6$ and $p_c = 0.8$. Having too large a crossover probability bias's the system to explore too much.

The crossover operator used is uniform crossover.

The change in mutation probability provides a minimal change between different values. The lower mutation probability converges faster, as can be seen when comparing $p_m = 0.3$ and $p_m = 0.5$. The mutation is analogous to exploitation, therefore a mutation rate that is too high or low will have the problem of too little too much exploitation.

The mutation technique used is a form that is analogous to bit-flipping, which will uniformly pick a new gene for each gene, each with probability p_m.

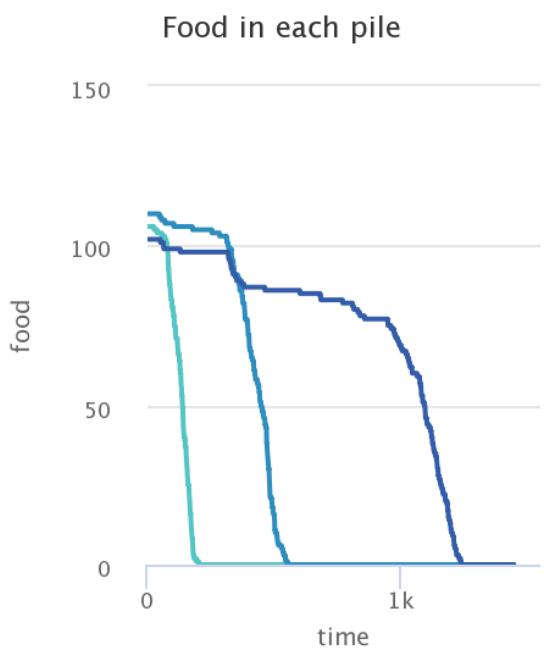
Question 2

a

> 💡 For part (a) report your observations on the behaviour of the colonies for the different parameters.

Control

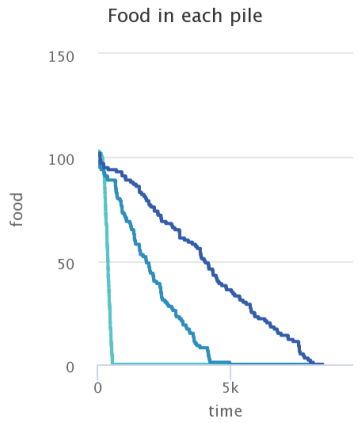
This graph shows the default



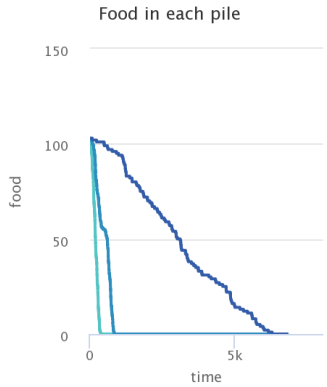
125-50-10

parameters of the NetLogo ant model. This is used as a comparison for the other experiments. The default parameters are a population of 125, diffusion of 50 and evaporation of 10.

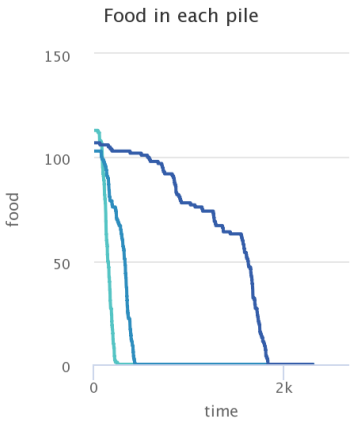
Food in Each Pile as a Function of Population



30-50-10



50-50-10



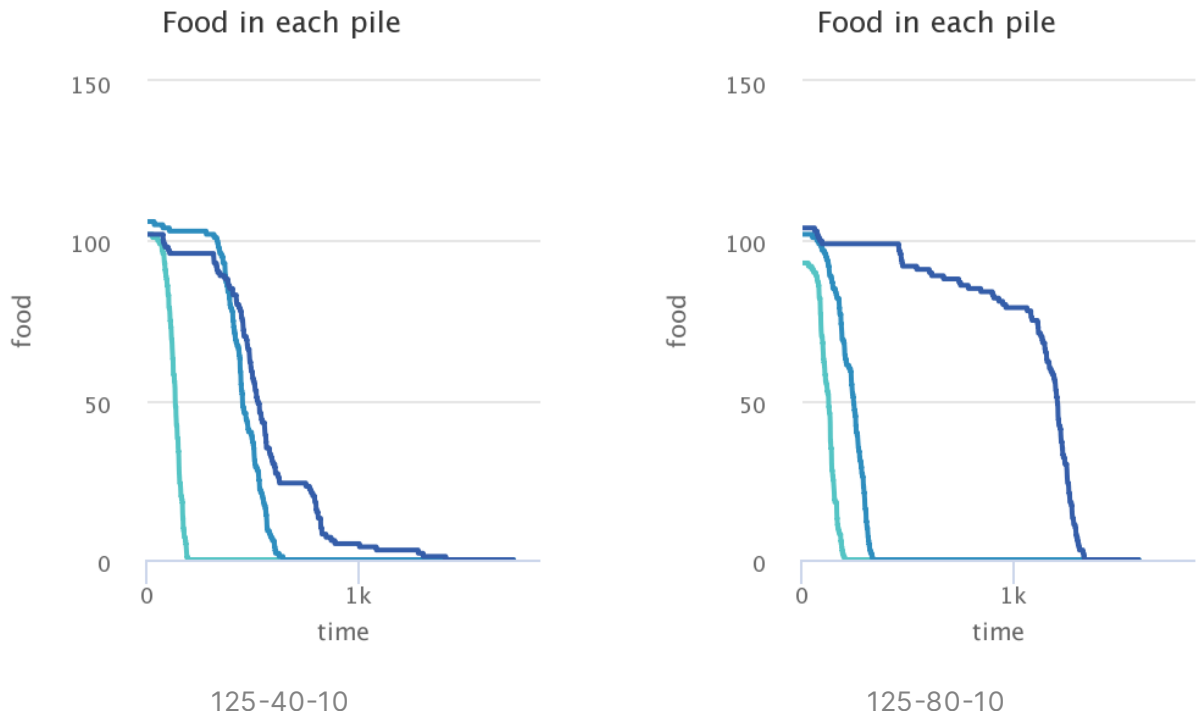
100-50-10

For this experiment, the ant population had a size of $[30, 50, 100]$. The rate of food source depletion is proportional to the number of ants in the simulation. With more ants, as can be seen when comparing the 100 population case to the

30 population case, the food sources deplete at a faster rate. This is due to the fact that with more ants, it is more likely that they will,

- a) find a pheromone trail and,
- b) contribute to a food source pheromone trail.

Food in Each Pile as a Function of Diffusion

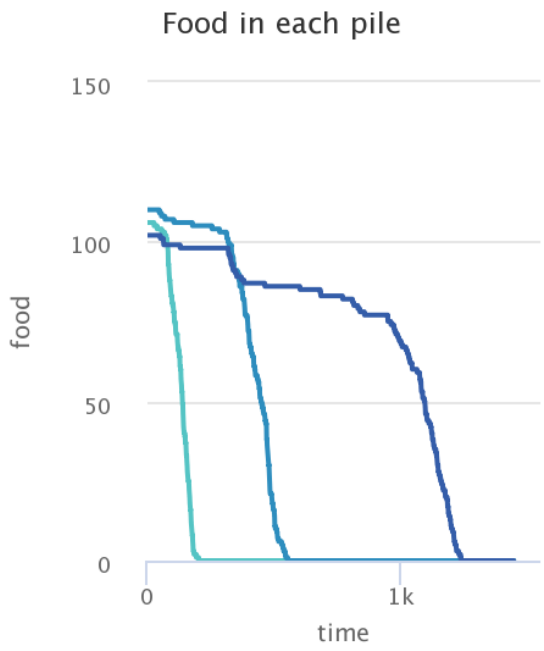


125-40-10

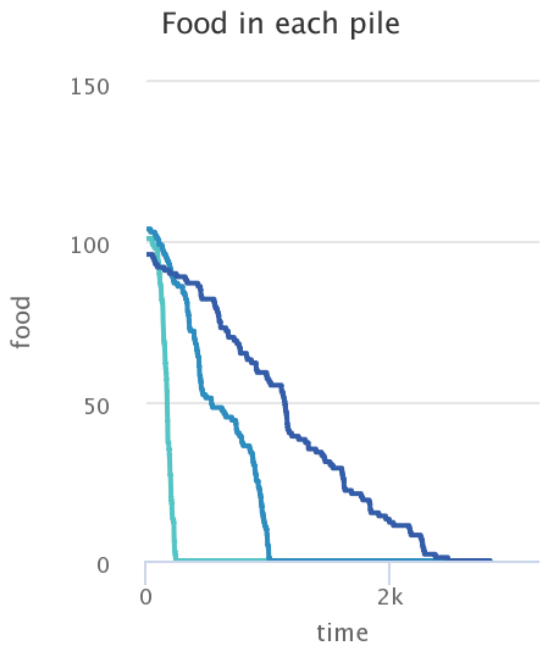
125-80-10

For this experiment, the diffusion rate had values of $[40, 80]$. With a larger diffusion rate, the ants took a longer time to find the last food source. The high diffusion rate allowed the ants to deposit too much unnecessary information into the environment. With more information (pheromones), it will be harder for each individual ant to discern and choose the best path to follow. This enabled the ants to stray/create new paths (further confusing subsequent ants). Therefore, the lower diffusion rate lead to a faster completion time.

Food in Each Pile as Function of Evaporation



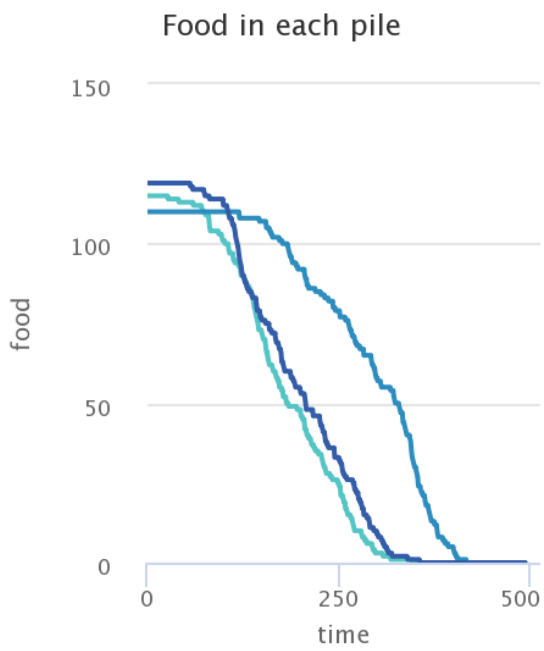
125-50-10

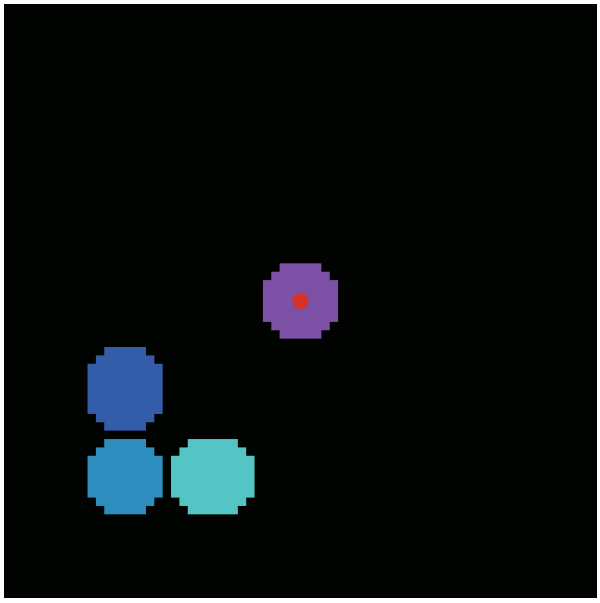


125-50-20

For this experiment, the evaporation rate had values of $[10, 20]$. With a larger evaporation rate, the ants take a longer time to deplete all the food sources. This is because the pheromones left by ants a lot faster, making it harder for other ants to find the trail to the food source. A higher evaporation rate enables the swarm intelligence of ants to 'forget' more quickly, and if it is too high, then newer ants may not be able to replenish the pheromone trail to good areas faster then it evaporates (therefore making really weak trails).

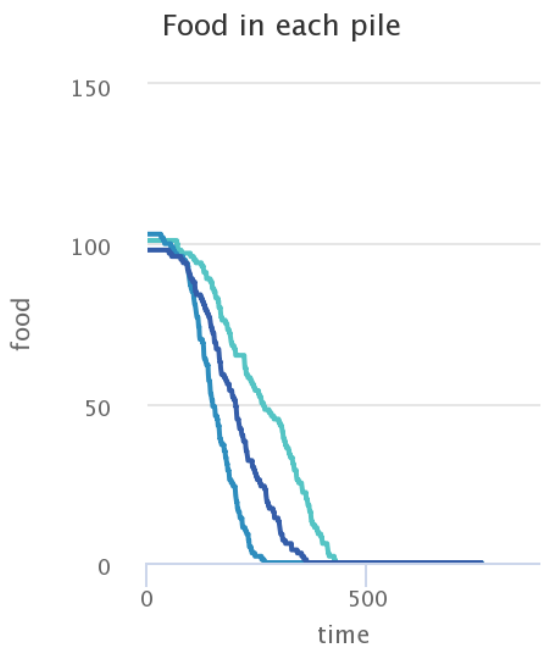
Food in Each Pile as a Function of Food Placement

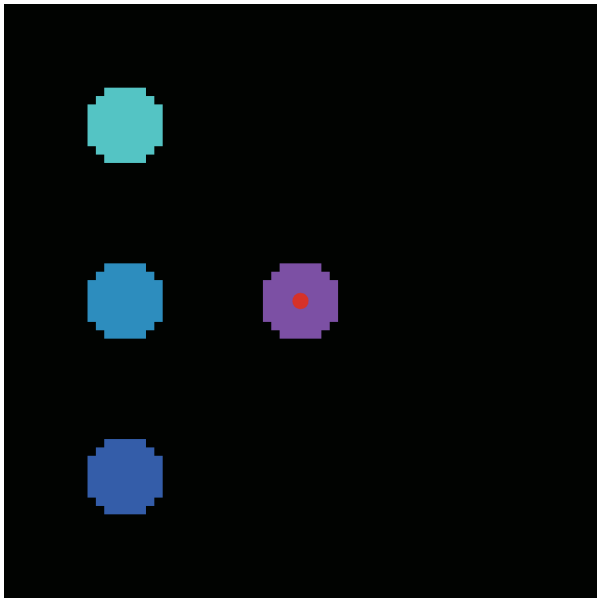




```
to setup-food  ;; patch procedure
  ;; setup food source one on the right
  if (distancexy (-0.3 * max-pxcor) (-0.6 * max-pxcor)) < 5
  [ set food-source-number 1 ]
  ;; setup food source two on the lower-left
  if (distancexy (-0.6 * max-pxcor) (-0.6 * max-pycor)) < 5
  [ set food-source-number 2 ]
  ;; setup food source three on the upper-left
  if (distancexy (-0.6 * max-pxcor) (-0.3 * max-pycor)) < 5
  [ set food-source-number 3 ]
  ;; set "food" at sources to either 1 or 2, randomly
  if food-source-number > 0
  [ set food one-of [1 2] ]
end
```

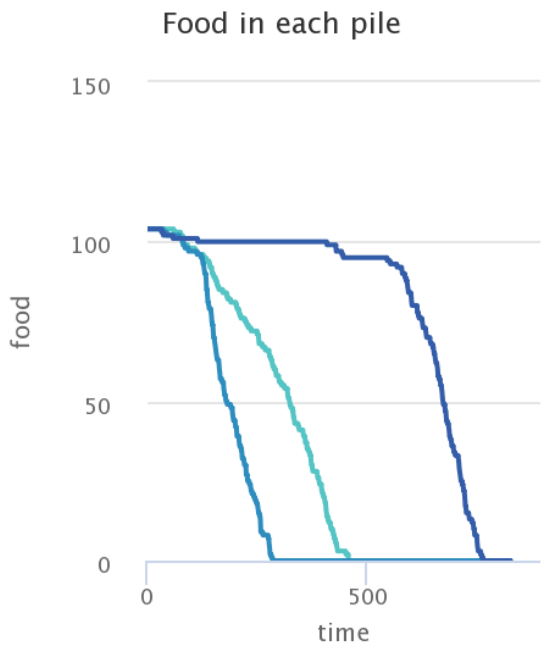
With food sources all near each other, the food sources depleted at the fastest rate. This is because the ants had to spend less time exploring for food.

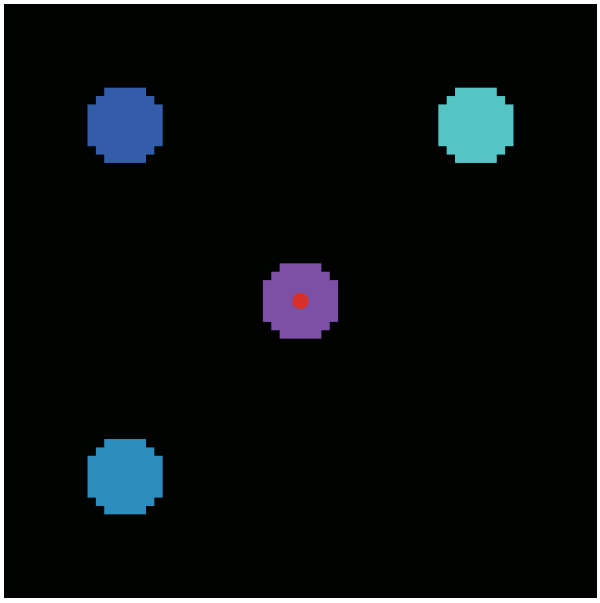




```
to setup-food  ;; patch procedure
  ;; setup food source one on the right
  if (distancexy (-0.6 * max-pxcor) (0.6 * max-pxcor)) < 5
  [ set food-source-number 1 ]
  ;; setup food source two on the lower-left
  if (distancexy (-0.6 * max-pxcor) (0 * max-pycor)) < 5
  [ set food-source-number 2 ]
  ;; setup food source three on the upper-left
  if (distancexy (-0.6 * max-pxcor) (-0.6 * max-pycor)) < 5
  [ set food-source-number 3 ]
  ;; set "food" at sources to either 1 or 2, randomly
  if food-source-number > 0
  [ set food one-of [1 2] ]
end
```

With equa-distant vertical spacing of food, the ants spent prioritised the food that was closest, opposed to the two farther out and equa-distant to centre food sources.





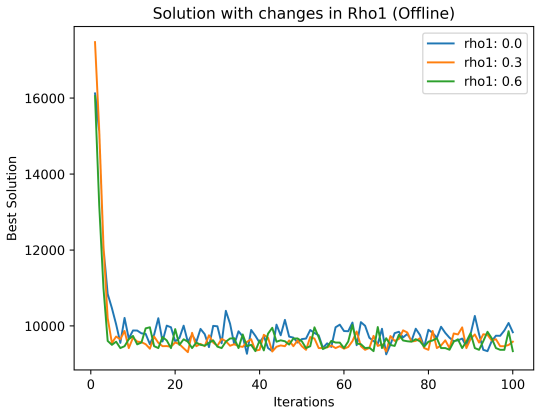
```
to setup-food  ;; patch procedure
  ;; setup food source one on the right
  if (distancexy (-0.6 * max-pxcor) (0.6 * max-pxcor)) < 5
  [ set food-source-number 1 ]
  ;; setup food source two on the lower-left
  if (distancexy (-0.6 * max-pxcor) (0 * max-pycor)) < 5
  [ set food-source-number 2 ]
  ;; setup food source three on the upper-left
  if (distancexy (-0.6 * max-pxcor) (-0.6 * max-pycor)) < 5
  [ set food-source-number 3 ]
  ;; set "food" at sources to either 1 or 2, randomly
  if food-source-number > 0
  [ set food one-of [1 2] ]
end
```

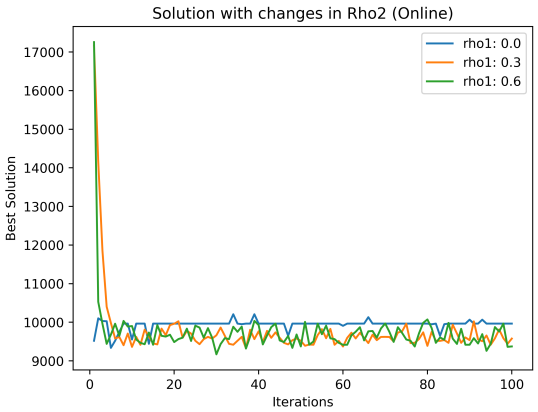
With all food sources equa-distant from the nest, the ants took the longest time. This is likely because the ants had trouble prioritising between the food sources until they picked one food source enough times to build a strong pheromone trail via the stochastic nature of the model

b

i

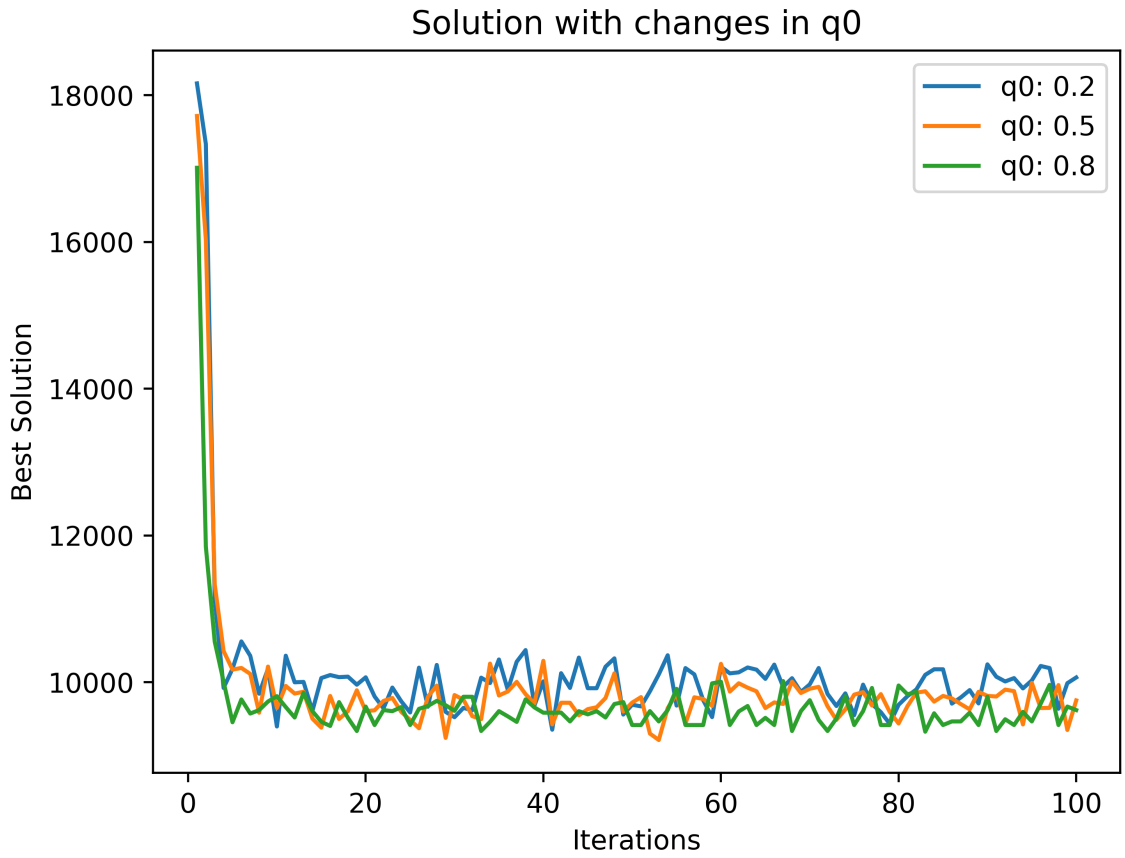
[5 marks] (a) Plot the solution progression for each pheromone persistence constant.





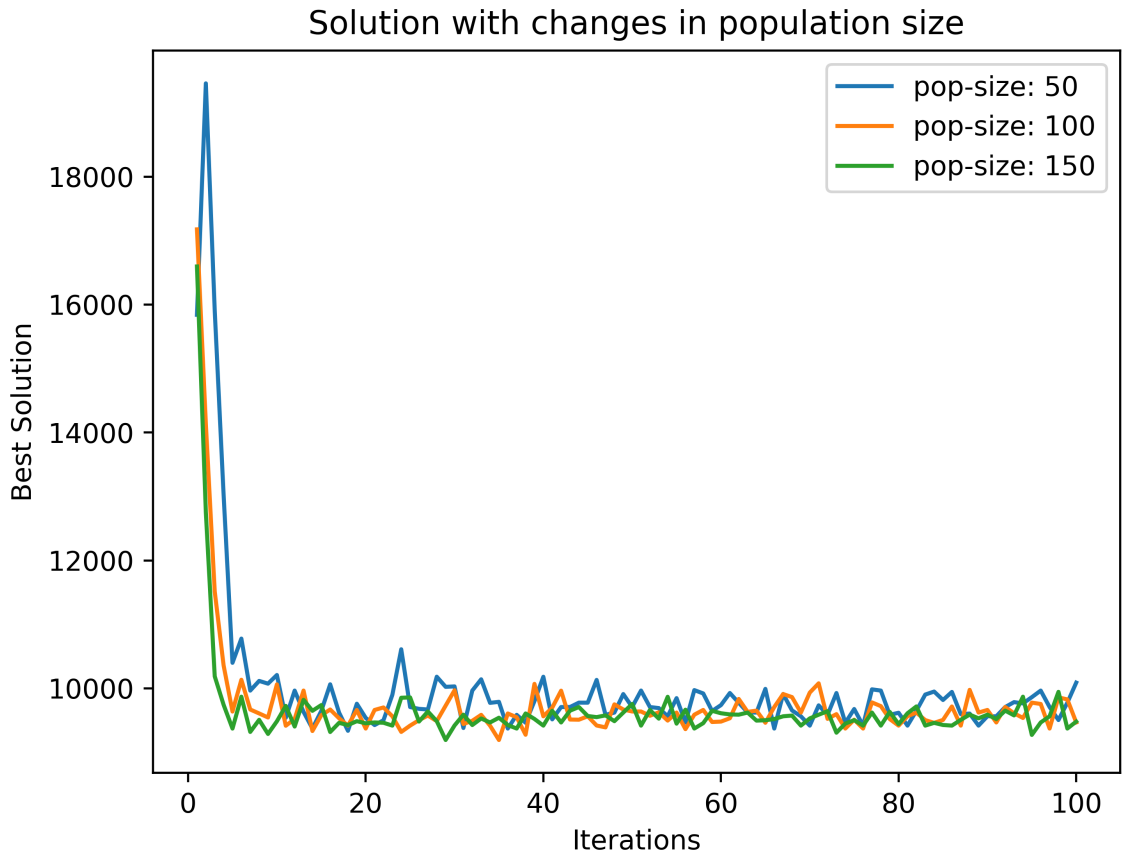
ii

[5 marks] (b) Same as in(3)(a) but for the state transition control parameter.



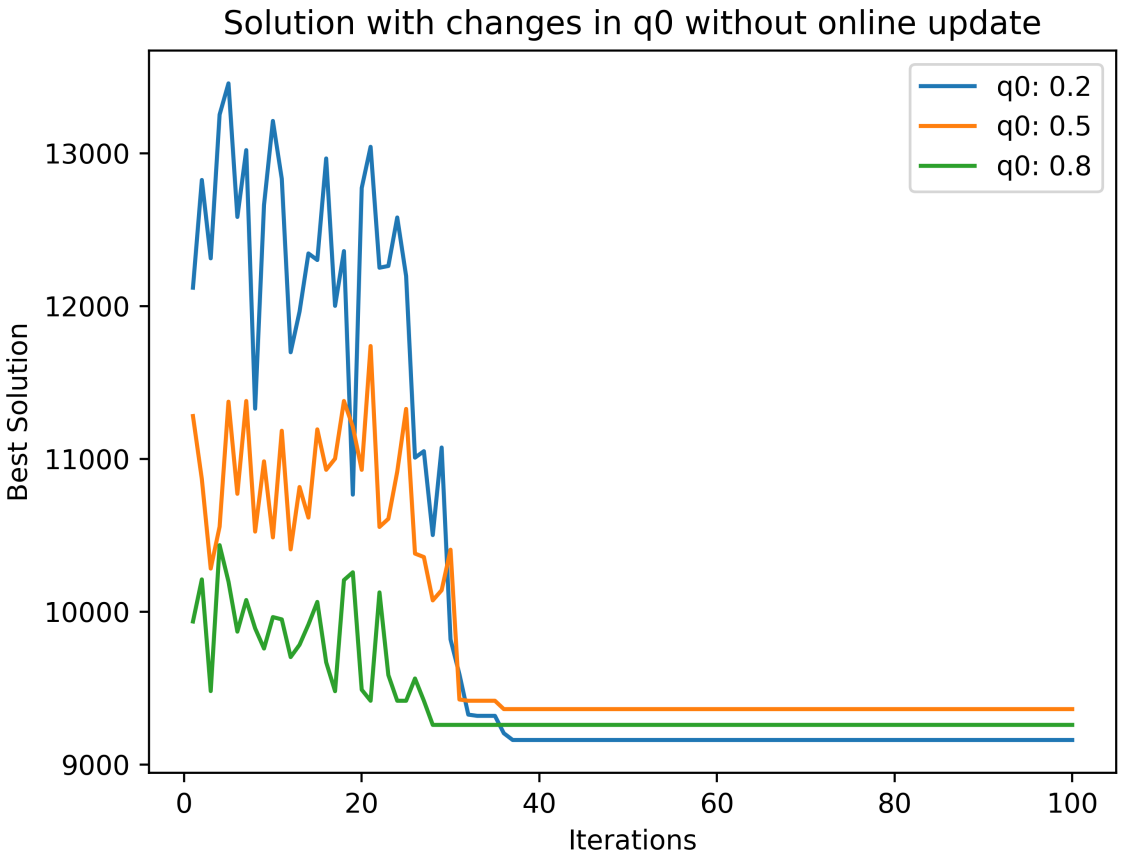
iii

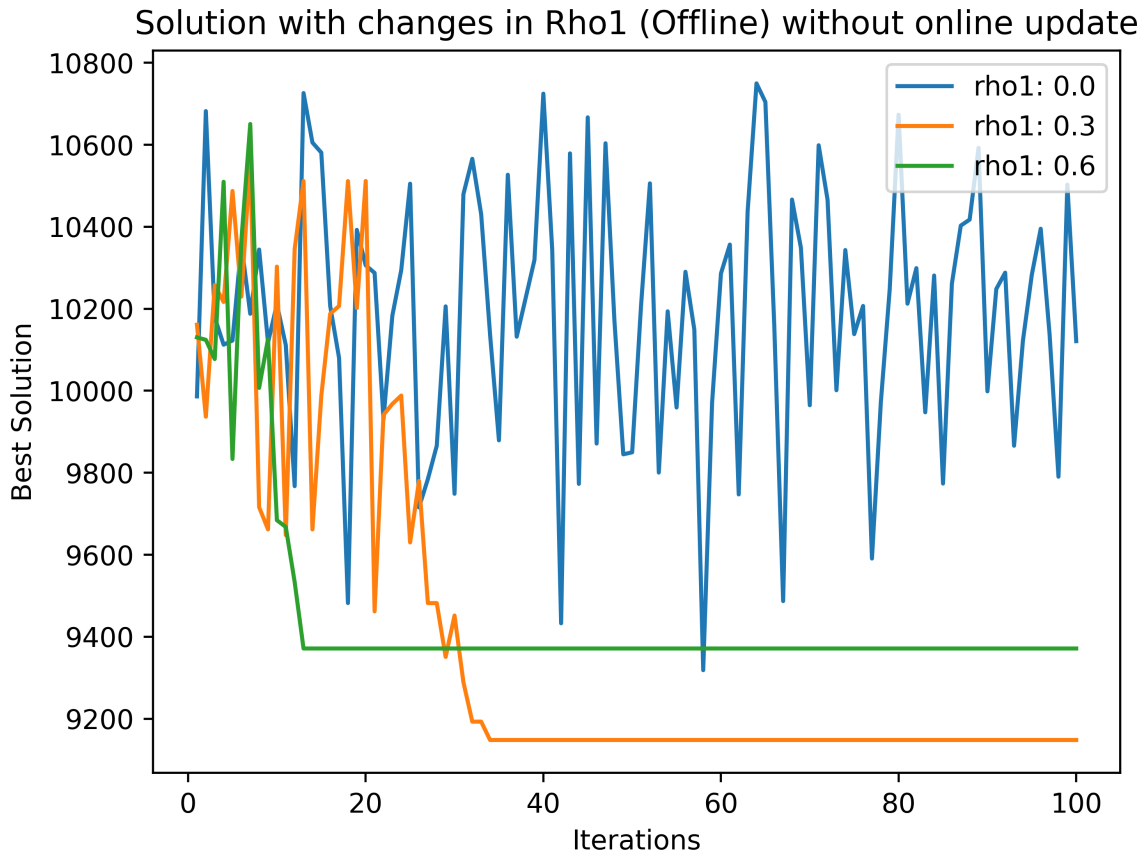
[5 marks] (c)Same as in (3)(a) but for the population size parameter.

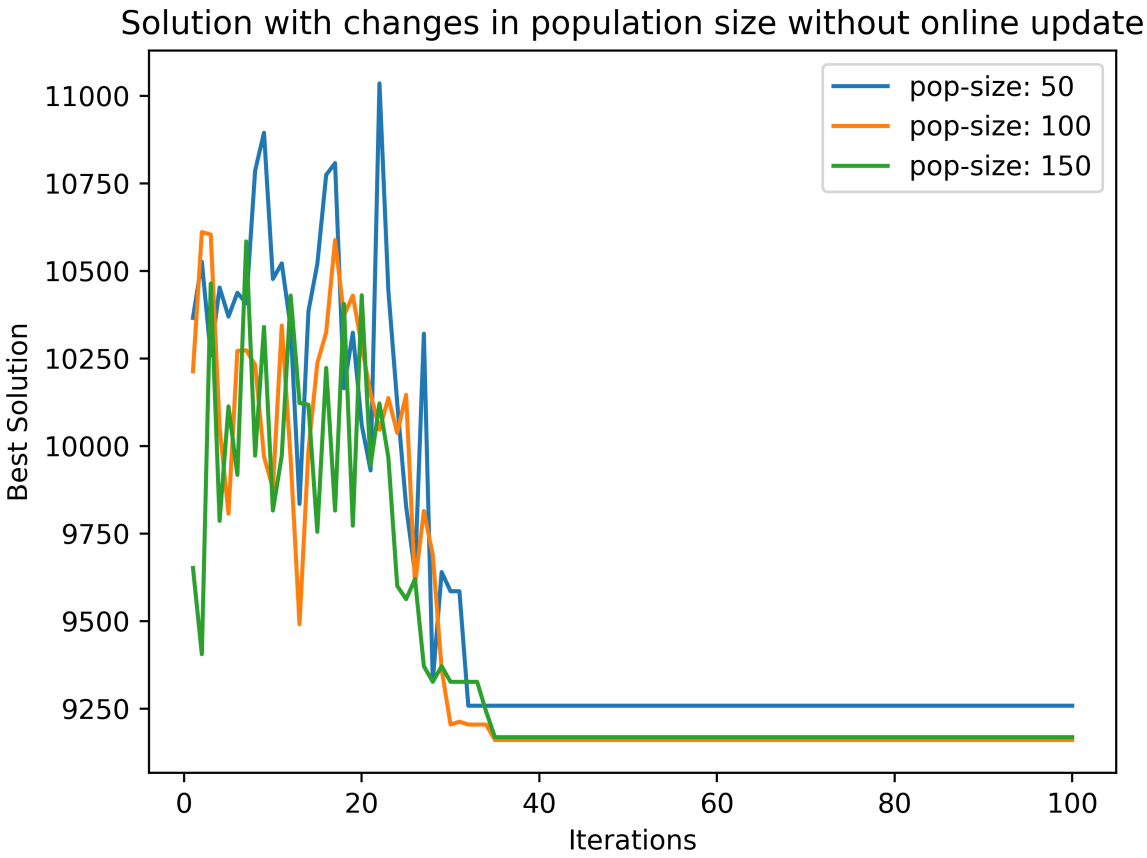


iv

> [5 marks] (d) Same as in (3)(a) but for the case where the pheromone update is turned-off for each case of the above three scenarios.


Solution with changes in q0 without online update
q0: 0.2
q0: 0.5
q0: 0.8
Best Solution
13000
12000
11000
10000
9000
0
20
40
60
80
100
Iterations


Solution with changes in Rho1 (Offline) without online update
rho1: 0.0
rho1: 0.3
rho1: 0.6
Best Solution
Iterations
10800
10600
10400
10200
10000
9800
9600
9400
9200
0
20
40
60
80
100



c

> 💡 [15 marks] Summarise your observations and conclusions on the behavior of your ACO algorithm as you change its control parameters.

Changing the pheromone persistence constant ρ_1 for offline persistence made marginal/un-noticeable difference in the progression of the best solution. Changing the pheromone persistence constant ρ_2 for online persistence made slightly more impact with smaller ρ_2 values causing less variation/flat areas on the graph. This is due to the fact that a small ρ_2 will cause the system to focus more on exploitation than on exploration.

When observing the effect of changing ρ_1 when online pheromone update is turned off, a larger difference is noticed. The system seems unstable in the beginning, however stabilises after some generations. This is because the ants do not have enough pheromone trail information in the first few iterations, and therefore will make less informed/more random path decisions. Having a larger ρ_1 value will give less importance to new pheromone deposits made by ants, therefore more ants are needed to strengthen the pheromones of a good trail (leading to higher confidence in taking said trail).

With online and offline updates, it can be seen that the state transition control parameter leads to better solutions the higher the value of q_0 is, albeit it is very marginal. This relationship is better viewed in the graph without online updates. In that graph, it is clear that larger values led to a better solution convergence. Increasing the value of q_0 biases the ants to exploit more. However, it seems that too much exploitation is not beneficial as the $q_o = 0.5$ case is very close in performance to the $q_0 = 0.8$ case.

With online and offline updates, it can be seen that the higher ant population sizes lead to better solutions, albeit it is very marginal. This relationship is better viewed in the graph without online updates. This is due to the fact that a larger population will have more ants participating in the modification of the environment, and thus working to get a better solution. With more ants it is more likely that ants will choose a good path, further strengthening the trail for subsequent ants in other iterations. However, it seems that the benefits of higher populations taper off after a threshold as the population size of $pop\_size = 100$ is very similar to the $pop\_size = 150$ case.