Particle Swarm Optimization

Samuel Steinberg

CS420

April 16th, 2019

Introduction:

This project is the implementation of the Particle Swarm Optimization (PSO) algorithm. The algorithm is meant to optimize by having a population of particles iteratively trying to get an improved position and solution, taking error (quality) into account. Particles are bound by the search space they are moving in and is dependent on position and velocity (these represented as vectors in 2D space). They consist of a position, velocity, best position found, and a set of update rules. The PSO algorithm combines private and public knowledge (best solution each individual has found and best solution group has found, respectively) of particles and allows the sharing of abstract space. There were two positional update formulas used in this project, each producing unique results. In addition, different parameter combinations were utilized to investigate a cause-effect relationship and to see how well the swarm would perform. The PSO can be seen in nature with flocks, herds, schools of fish, and other phenomena where it is impossible to coordinate with such population numbers (sometimes in the millions) or to be led by a single individual.

Theory and Methods:

The project takes parameters representing the number of particles, inertia, cognition and social parameters, world width and world height, maximum velocity, an error threshold, a number of runs, and a mode. These all play a key role in the algorithm and how well it optimizes to reach a solution. After a positioning the particles randomly throughout the search space and initializing the velocities, the program will enter the update main update loop. This loop will only stop when a specified number of epochs have occurred, or if there has been sufficient convergence and the quality set by the error threshold has be deemed met.

```
error_x += (position_x[k] - global_best_position_x)^2
error_y += (position_y[k] - global_best_position_y)^2
```

Figure 1: Error Summation for each particle k

The algorithm will halt when a sufficient measure of quality has been reached, meaning the particles have converged on the global best position. For each particle *k*, the error in both the x and y direction are calculated by calculating the difference between the particle's position at that axial coordinate and the global best's position there. This will give a total measure of quality that is scaled in Figure 2 below:

```
error_x = sqrt((1/(2*num_particles))*error_x)
error_y = sqrt((1/(2*num_particles))*error_y)
```

Figure 2: Quality measure after summation

This is one of the stopping conditions for the master loop and is the final error measure before the next iteration. If the x and y error drop below the error threshold, the positional update ends.

The velocity of each particle is updated and scaled in the following Figure (3 and 4):

```
velocity' = inertia * velocity + c_1 * r_1 * (personal_best_position - position) + c_2 * r_2 * (global_best_position - position)
```

Figure 3: New particle velocity calculation; c_1 and c_2 represent the cognition and social parameters, while r_1 and r_2 represent random numbers [0, 1]

Velocity is updated in both the x and y directions (since it is a vector) and is the product of inertia and previous velocity summed with the product of cognition, a random number, and the difference between the personal best position and its current position. This is then summed with the product of the social parameter, a random number, and the difference between the global best position and current position. Each random number is within the range [0,1], inertia maintains direction, cognition represents bias towards a particles personal best, and social is the bias towards public best. The velocities in both directions are then scaled to the max velocity in Figure 4:

```
if velocity_x^2 + velocity_y^2 > maximum_velocity^2
  velocity = (maximum_velocity/sqrt(velocity_x^2 + velocity_y^2)) * velocity
```

Figure 4: Scaling of velocity in terms of the max velocity a particle is allowed

If the velocities are not scaled; positions will go beyond the scope of the search space and chaos will ensue. After obtaining the new velocity, it is possible to update the position of the particle. This is done by simply adding the directional velocity vector to the positional vector. The new positions are then compared to both the current global best position and the personal best position of the particle. To perform this, the following distance calculations are performed:

$$mdist = \sqrt{\max_x{}^2 + \max_y{}^2} / 2$$

$$pdist = \sqrt{(p_x - 20)^2 + (p_y - 7)^2}$$

$$ndist = \sqrt{(p_x + 20)^2 + (p_y + 7)^2}$$

Figure 5: mdist represents the global distance calculation, pdist and ndist represent personal best and local distances

Here, $\max_x$ and $\max_y$ represent global maximum positions, while p_x and p_y represent the current particles position. These distances are used to find $Q(p_x, p_y)$, which measures the quality of the position of a particle p. For the two quality problems in this project, Figure 6 represents a quality measure where there will on be one global maximum (Problem 1), and Figure 7 is a

quality measure producing two maxima; a global maximum and a local maximum. These Figures are located on the page below.

$$Q(p_x, p_y) = 100 \cdot \left(1 - \frac{pdist}{mdist}\right)$$

Figure 6: This will produce a single global maximum

$$Q(p_x, p_y) = 9 \cdot \max(0, 10 - pdist^2) + 10 \cdot \left(1 - \frac{pdist}{mdist}\right) + 70 \cdot \left(1 - \frac{ndist}{mdist}\right)$$

Figure 7: This will produce both a global maximum and a local maximum

The positional quality is used as a comparison tool to update the global best position, personal best position, and the amount of error in the algorithm (since this is indeed a quality measure). Figures 8 and 9 below represent the evaluation:

```
if(Q(position) > Q(global_best_position))
  global_best_position = position
```

Figure 8: Global best position updating

```
if(Q(position) > Q(personal_best_position))
  personal_best_position = position
```

Figure 9: Personal best position updating

The completion of the update loop will signify the end of the optimization algorithm. This will be due to a particle convergence on a maximum or by an epoch limit being reached without convergence. The project will then generate graphs to support the visualization of meaningful data measures.

Results and Discussion:

Experiments were conducted for quality equations shown in Figures 6 and 7. Figure 6 results in a single global maximum, while Figure 2 results in both a global maximum and a local maximum. Experiments were conducted with varying population sizes of particles, inertia values, social parameters, cognitive parameters, and levels of maximum velocity.

I. Problem 1

Problem 1 uses the positional quality calculation measured in Figure 6, which results in a single global maximum. In these experiments, the number of particles, maximum velocity, and inertia values played the most significant part and were most interesting in determining convergence. The social and cognitive parameters tended to behave as expected.

When the number of particles is very small (such as around 10), they will converge at a slower rate and it takes longer to find the global maximum. Usually they will not converge at all. When the number of particles is increased, however, convergence takes place much more rapidly, as shown in Figure 10 below:

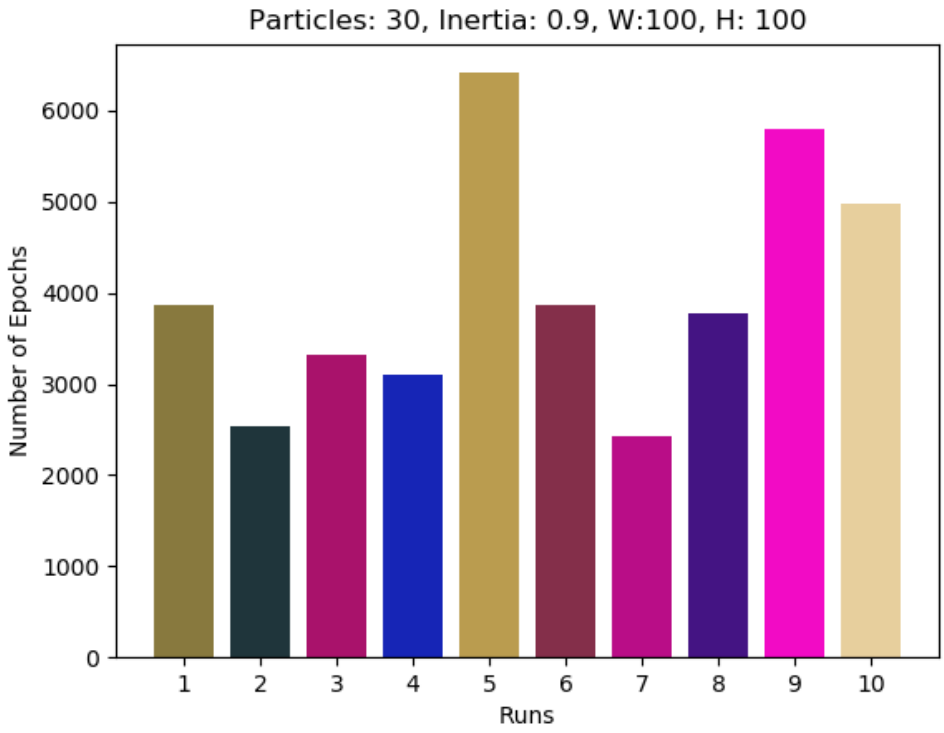


Figure 10: 30 Particles number of epochs to convergence.

In this experiment, 30 particles were used to serve as a medium sample size, by which to compare with others as a base case. In 10 experiments, it took an average of ~4000 epochs to converge, with a few runs taking 1000-2000 epochs higher. Though it took this many epochs, the particles found the global maximum quickly:

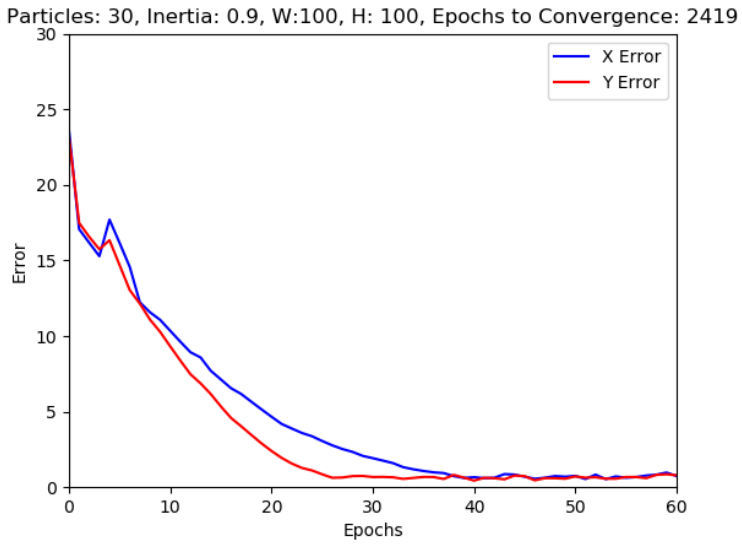


Figure 11: Error in X and Y direction for each epoch in Run 7

As seen in Figure 11 for Run 7, the error in the X and Y direction both reached extremely low levels by epoch 40 but do not meet the error threshold until epoch 2419. This could be explained by the inertia at .9, since this could cause particles to balk when near the maximum.

Increasing the particle count even further to 70 yields similar results at the same inertia value. Though the average epochs to convergence was around 500 more (~4500), this could be explained simply by it taking more time for 70 particles to converge as opposed to 30, especially since the stalling of particles will continue to occur around the global maximum for these as well:

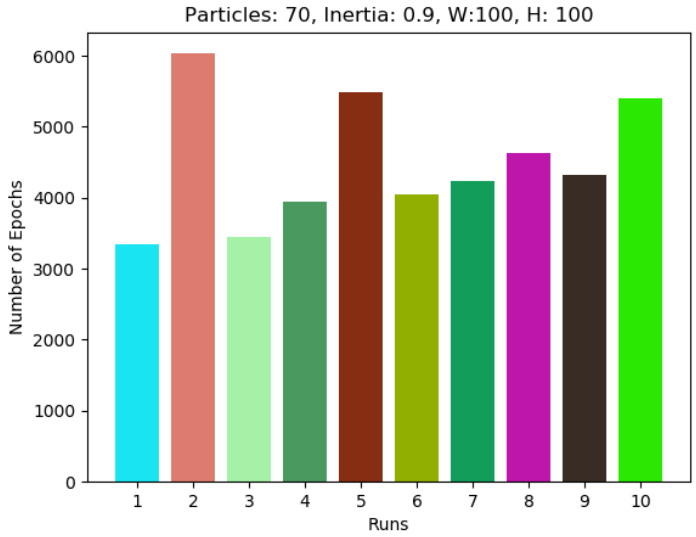


Figure 12: Epochs to Converge with 70 particles over 10 experiements

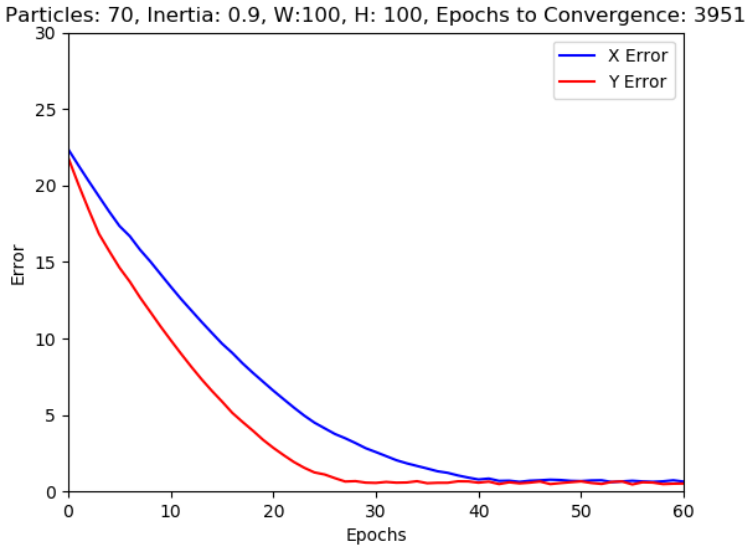


Figure 13: Error in X and Y direction for 70 particles

Inertia values also played a large role in Problem 1. Below is a representation of the base case particles (30) and their convergence when inertia is lowered to just 0.6:

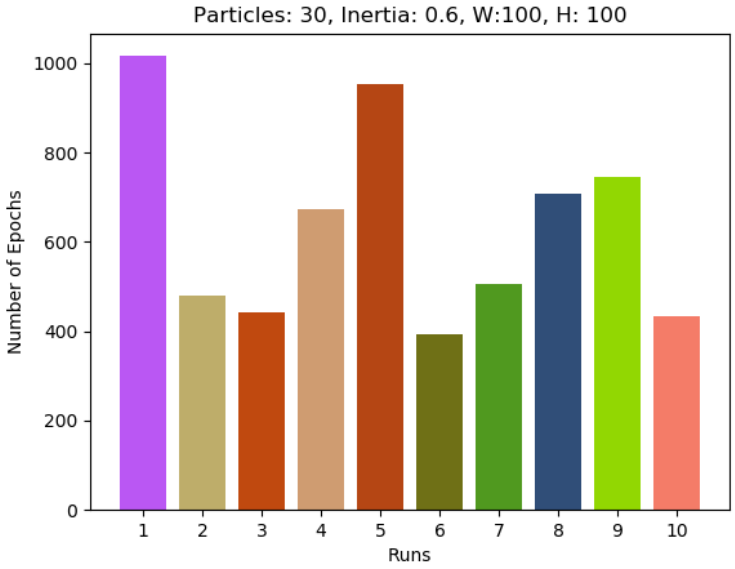


Figure 14: Epochs to Converge significantly lower

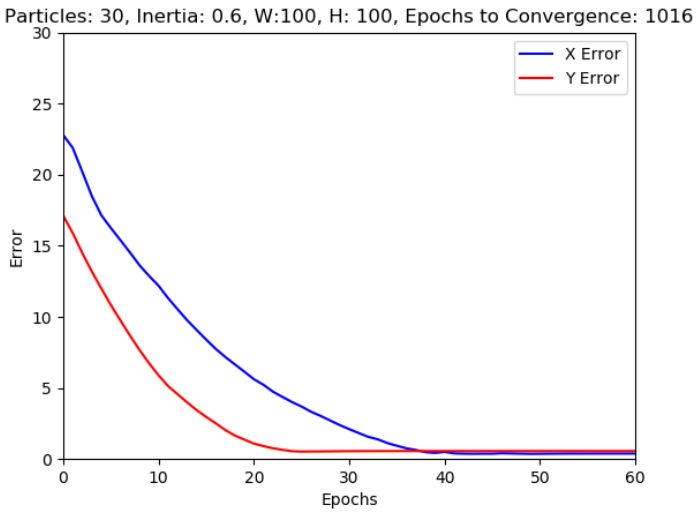


Figure 15: Note that the error is still stalling around 40 epochs, but will still converge quicker

The lower inertia values resulted in an average number of epochs around 600. This is a significant difference from the .9 inertia graphs, since the lower value causes less stalling even though as seen in Figure 15 there is still a significant number of epochs until it will converge. To amplify the effect that inertia has, experiments were also conducted with varying degrees of particles and velocities. As shown in Figure 16 below, a very low inertia plays a significant effect on convergence speed:

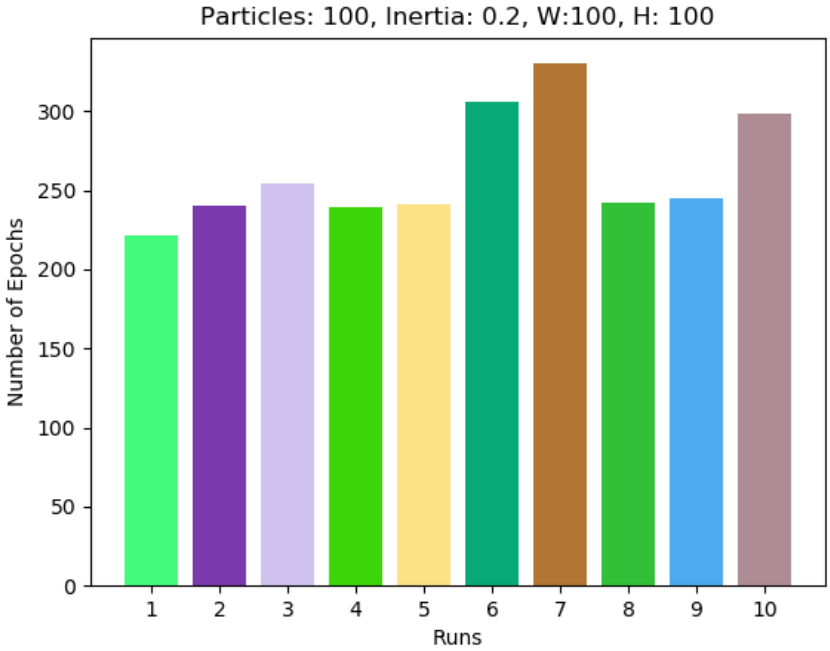


Figure 16: Inertia's influence felt on high number of particles

Even with a large population of 100 particles, a very low inertia effect allows particles to converge much faster and with less hang time around the error threshold. Comparing this to Figure 14 above; Figure 16 converges around ~350 epochs faster with three times the number of particles. This can be attributed to a lower inertia value and faster convergence when around the global maximum.

Another useful relationship to take into consideration for this project was maximum velocity and also its relationship with inertia. Maximum velocity is the limit on how much a particle can more for an iteration (epoch). In Figure 17 below, maximum velocity is increased to 8 in otherwise base case conditions:

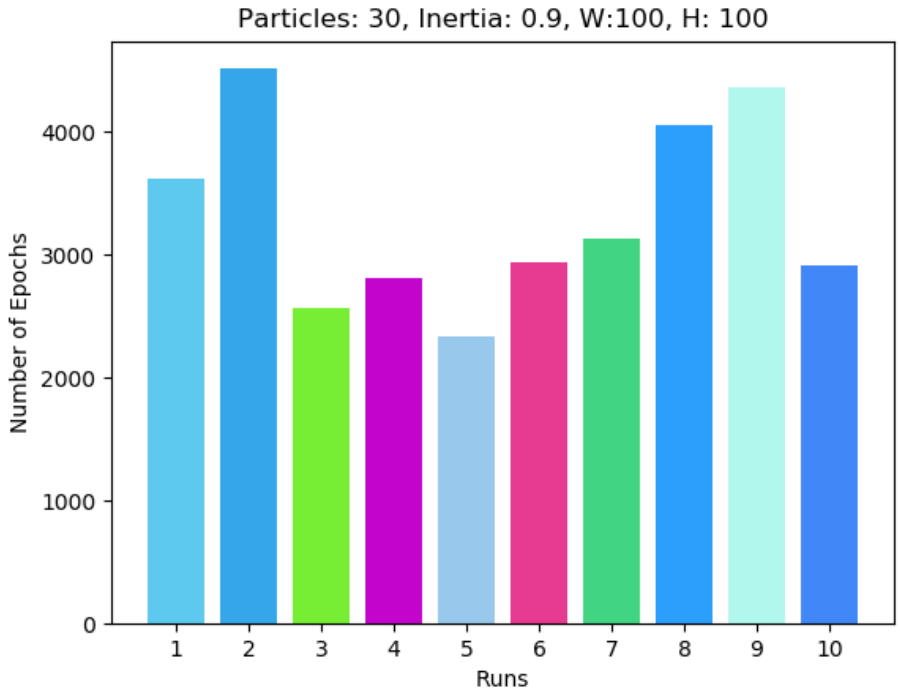


Figure 17: Maximum velocity of 8. Note the decrease in epochs to convergence.

With higher maximum velocities particles converge around 1000 epochs faster. This can be explained by maximum velocity allowing a particle to move more with a greater value, hence it can get to the area of the global maximum quicker.

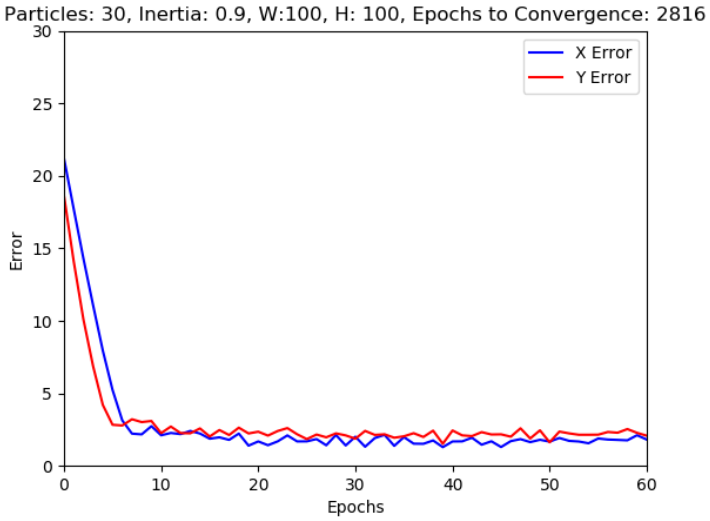


Figure 18: Note the faster decrease in error followed by more erratic changes

The movement speed can also be seen in Figure 18 above, where error rapidly decreases and also incurs much more erratic movements as it nears the error threshold. This is due to being allowed to move further and not having as constricted a search space. Without as many inertia restrictions, however, particles converge even faster with a higher maximum velocity, and less chaos is present in the error.

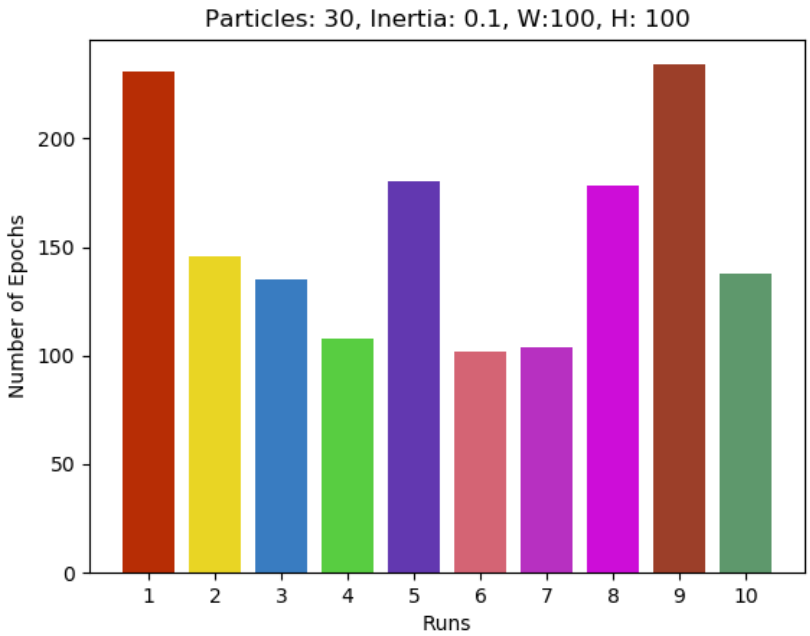


Figure 19: Significantly fewer epochs to convergence with less inertia.

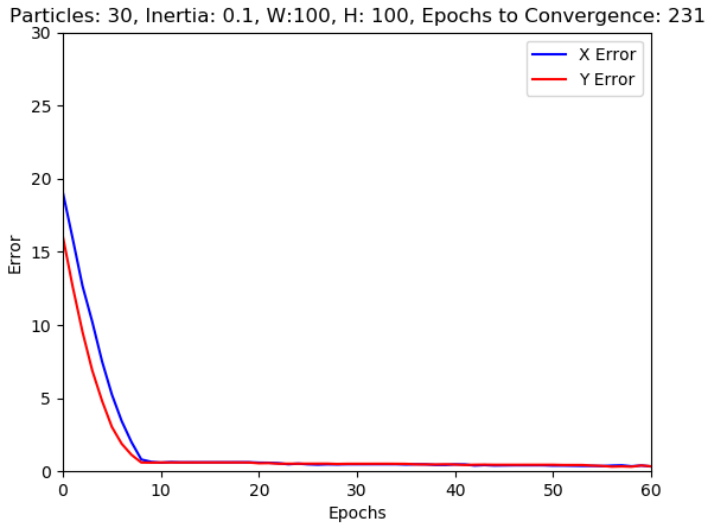


Figure 20: Run 1 shows a fast-initial convergence to the global maximum, and a less chaotic error in general with lower inertia.

As shown in Figures 19 and 20 convergence is significantly faster with less inertia, and error is also less erratic error after the initial rush to the maximum. With less resistance put up by the inertia, the particles can ascend on the maximum faster with some experiments taking less than 100 epochs. The average number of epochs was around ~150.

The cognitive and social parameters tended to behave as expected. The cognitive parameter turns towards the private best (a particles personal best), while the social turns towards the public best (global maximum). When cognition is high, particles will not flock to the global maximum as quickly as they normally would, and as such would take longer to converge. Figure 21 represents a very high cognition as 4 in otherwise normal circumstances:

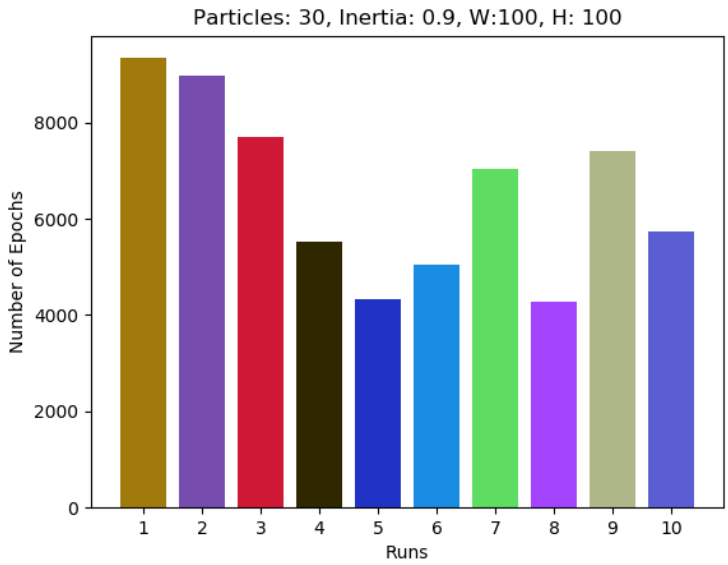


Figure 21: Note the greater number of epochs it takes to converge with a higher cognition

Due to the bias towards their own best, on average it took ~6,600 epochs to converge; over 2,000 more than they would have with a normal cognition. Notably when cognition is lowered is turned off the and the social parameter is dominant the number of epochs almost halves:

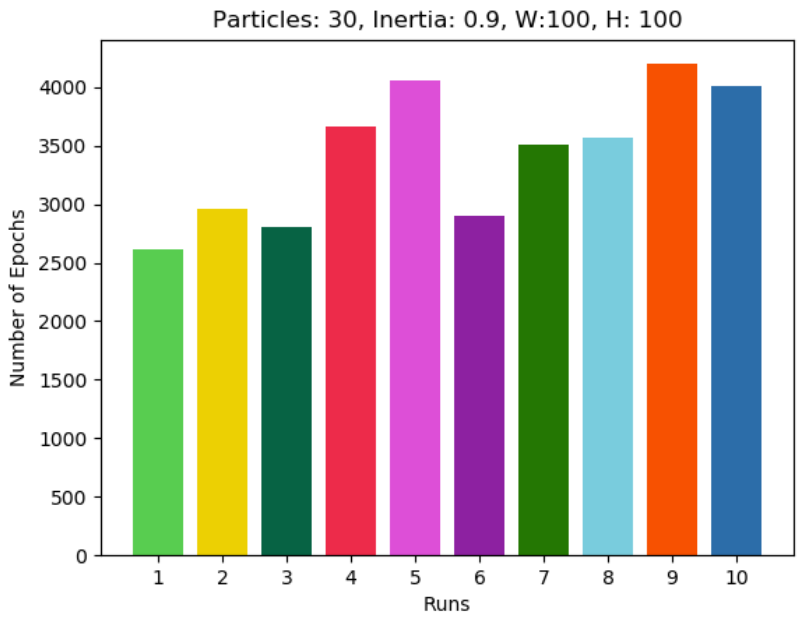


Figure 22: Since the social parameter is allowed to take control, it will converge sooner.

As cognition lowers and particles are more concerned with the global maximum and the best position for their population as a whole, the average epochs to converge drops to ~3,000. The social parameter seems to be the stronger of the two parameters in terms of converging to the global max. With this information it was not a surprise that when social was turned off and particles did not communicate with each other at all, there was no convergence at all.

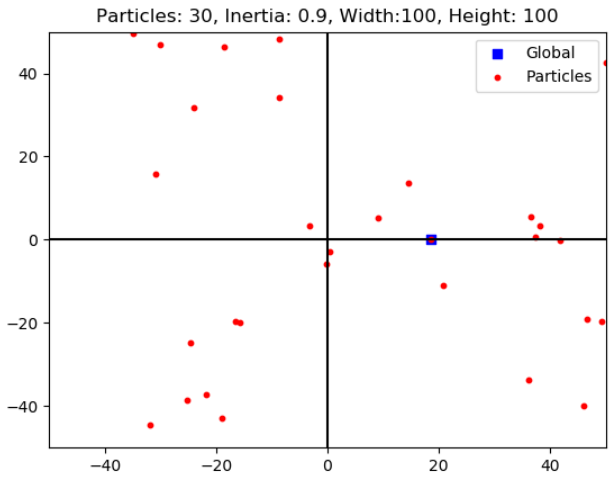


Figure 23: Initial positioning of population

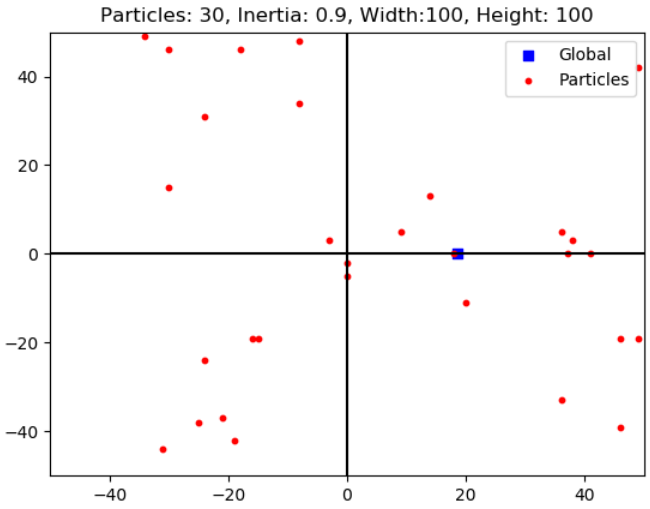


Figure 24: Epoch 10, note the small update in positioning

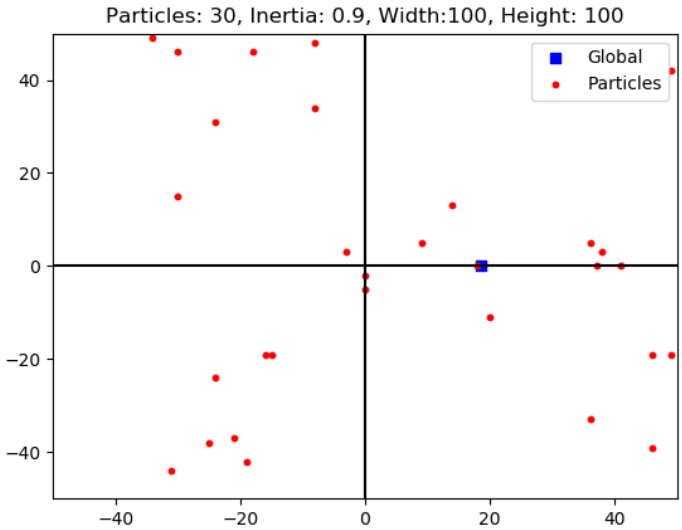


Figure 25: Epoch 100, population has completely stalled at this point.

The plots in Figures 23, 24, and 25 represent positioning for the initial grid, 10th update, and 100th update respectively. Each run number is denoted in its caption. With no communication between individuals, the swarm will stall, and convergence will never occur. Particles stall out when only looking out for their own personal best position and not the global maximum. As shown, there were only miniscule updates, if any, made by particles with no social parameter.

II. Problem 2

Problem 2 uses the positional quality obtained from Figure 7, where there will be two maxima; one local and one global. Base case tests and tests where parameters were isolated to measure performance were each tested extensively.

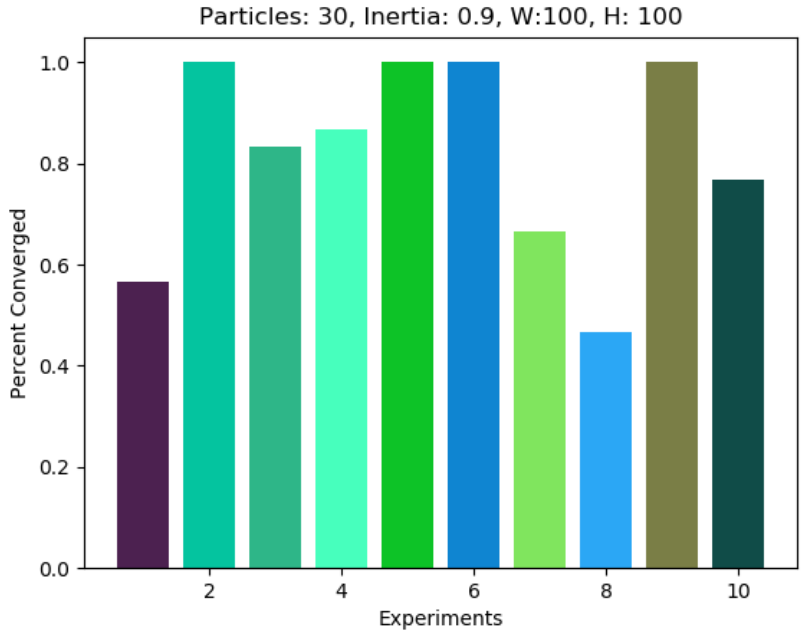


Figure 26: Percentage of particles converged over 10 experiments

For the base case, Figure 26 is a measurement of what percentage of the particles converged on a maximum. In this project a converged particle was a particle with .001 of the maxima.

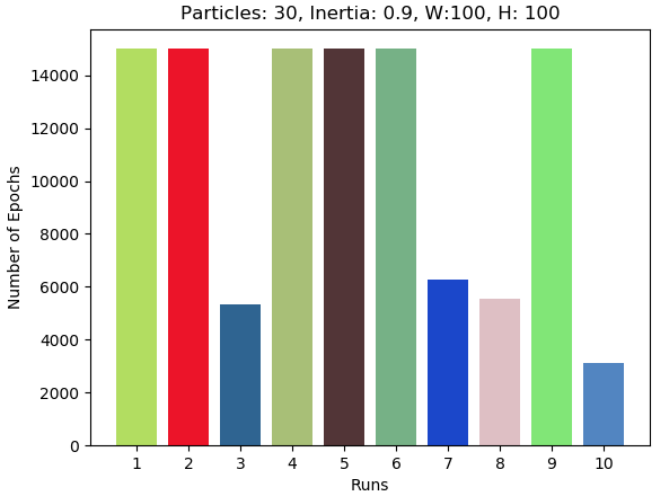


Figure 27: Convergence seemed to be very reliant on which maxima it converged on

Large numbers of particles had a much lighter effect on convergence than they did in Problem 1, aside from very small populations. Populations that fully converged tended to be on the local maxima:

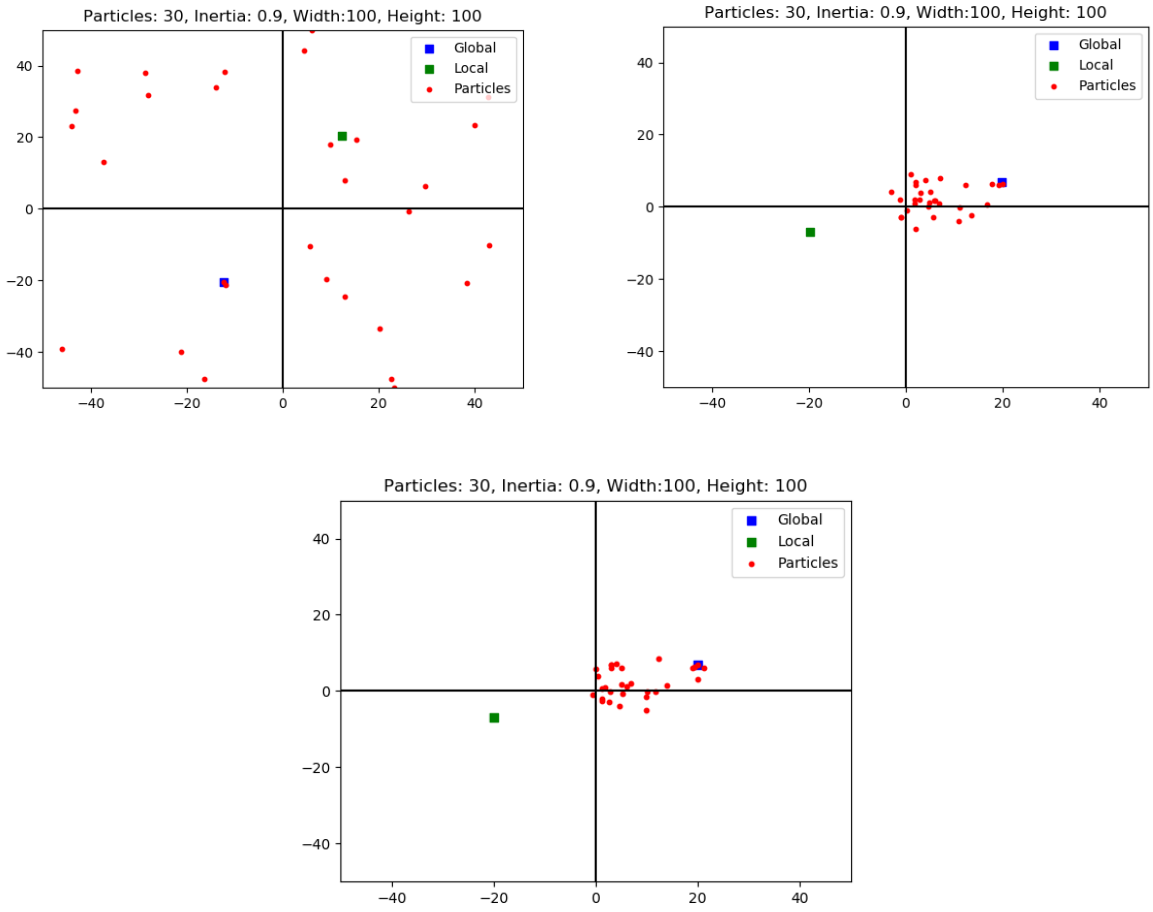


Figure 28: (Top left to right) Initial positioning, 10 epochs, 100 epochs. Notice the clustering here, mainly around origin but biased to global maxima

Here, not all particles converged, but were still biased towards the maximum. This is due to the presence of the local maxima, an influence which did not exist in Problem 1. Note that in Problem 1 population sizes played a large role in convergence and speed of convergence, and the lack of influence it has in Problem 2 could be explained by this. See the following page for Figures representing a sample size of 300, or 10 times the size of the above.

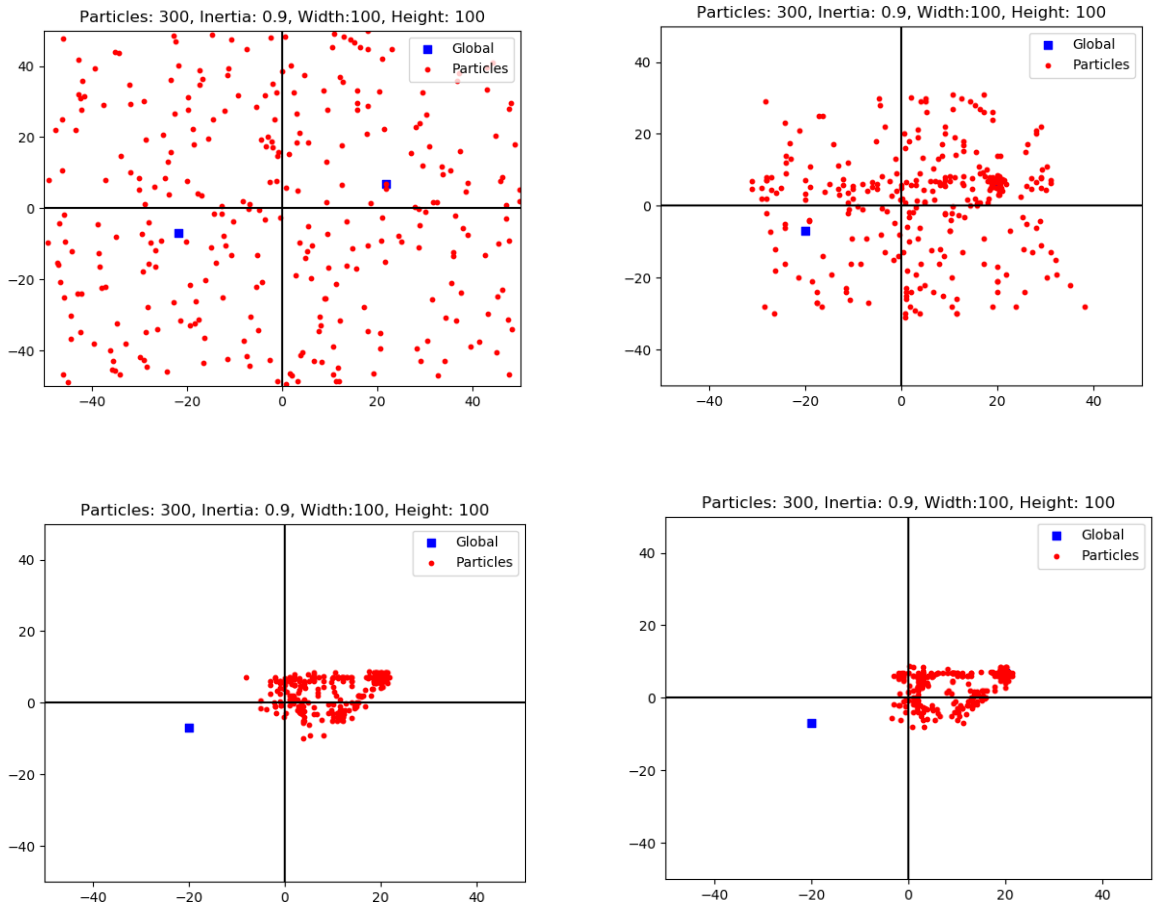


Figure 29: Initial swarm (top-left), 10 epochs (top-right), 40 epochs (bottom-left) and 100 epochs (bottom-right)

As can be seen when contrasting Figures 28 and 29; their behavior is very similar. They both first swarm towards the maxima's, and then mass around the global maximum. A larger sample size such as this is helpful in understanding the swarming of particles and can be easily equated to a swarm of bees massing around a nest in nature.

Inertia values had a similar effect in Problem 2 like they had in Problem 1, with a decreasing inertia seemingly leading to a faster convergence. Figure 30 on the following page represents an inertia at 0.6. With less of an inertia effect, particles found it easier to settle down around the global maximum and have to deal with less of an offsetting effect.

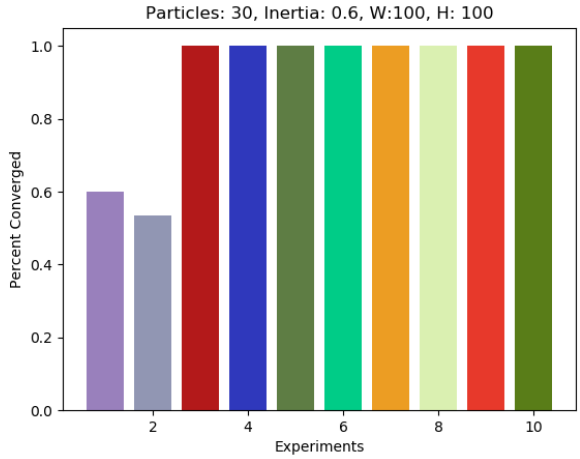
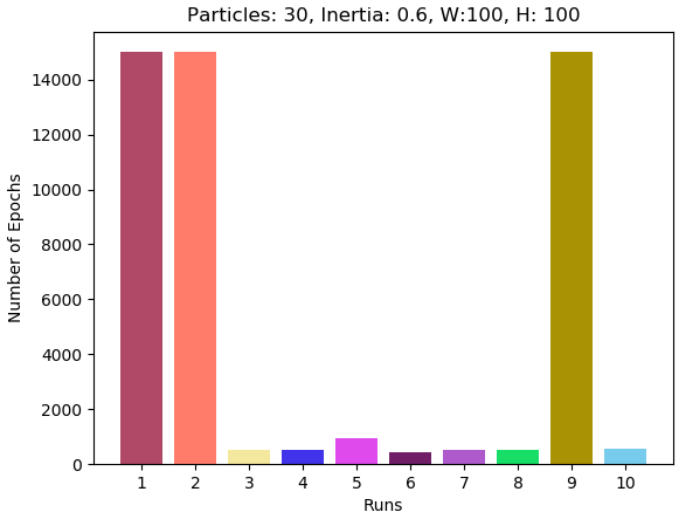


Figure 30: Note that all particles converged for 8 of the 10 experiments with a lower inertia value

Particles fully converged with a lower inertia for most of the experiments as seen in Figure 30. Additionally, those also converged much faster than in the base case experiments as seen in Figure 31 below.



Note Run 9 is an outlier that took around 14,700 epochs to fully converge, while the other converged runs all took under 1,000. To further display inertial effect, it was lowered to 0.4 where convergence took place even quicker, with this run taking 208 epochs:

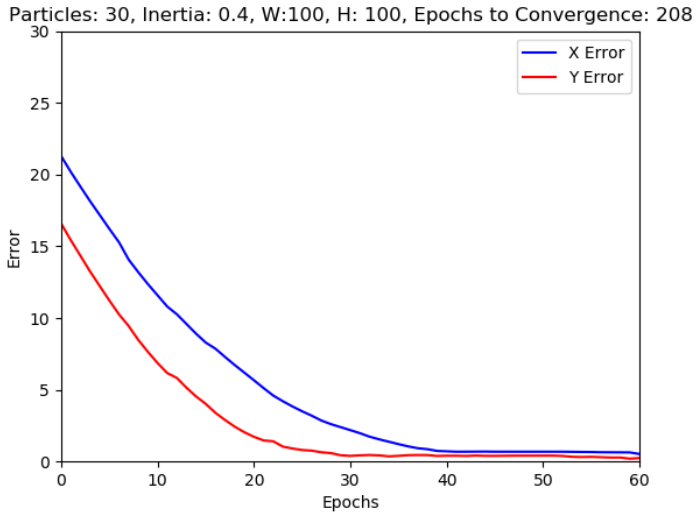


Figure 31: Note with a lowered inertia convergence will not stall out as long

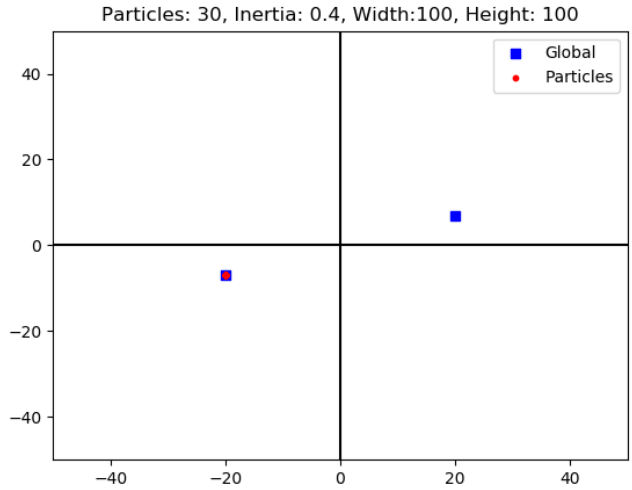


Figure 32: All particles converged on local minimum

Figure 31 shows a similar X/Y error curve to many previous runs but will experience less hang time around the error threshold due to the decreased inertia. This leads to a faster convergence, this time to the local minimum.

Maximum velocity was also a significant factor in determining performance, as an increased maximum led to significant more fully converged or nearly-fully converged populations. For example, when maximum velocity is bumped to 7 there are many more particles around the global maximum as compared to the base case:

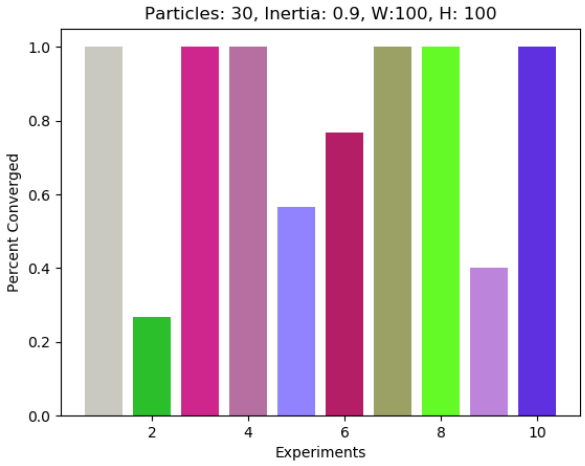


Figure 33: Note that many of the low convergence percentages were due to strictness in distance

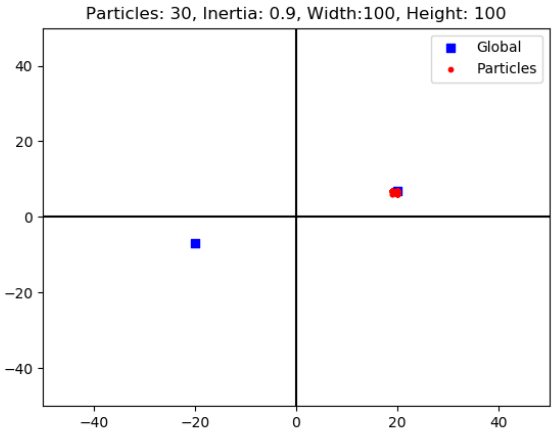


Figure 34: Convergence around global maximum with higher maximum velocities

With more freedom of movement, it is not surprising that even with the influence of a local minimum particles were able to move closer to and converge on the global maximum. Note that in Figure 33, many of the experiments were not considered converged due to strict distance standards. It should be obvious visually, however, of the effects a higher maximum velocity has due to Figure 34. The influence of the local minimum is virtually offset.

The cognitive and social parameters were also quite similar to Problem 1. Higher cognitive values lessen the impact that the social parameter has, thereby leading to particles looking out for their own personal best at the expense of communicating with each other to go to the global best and find the best candidate solution quicker. On the other hand, lower cognitive values favored those in the initial vicinity of the global maximum much more than those further away and led to a significant communication disconnect in which particles converged and which did not.

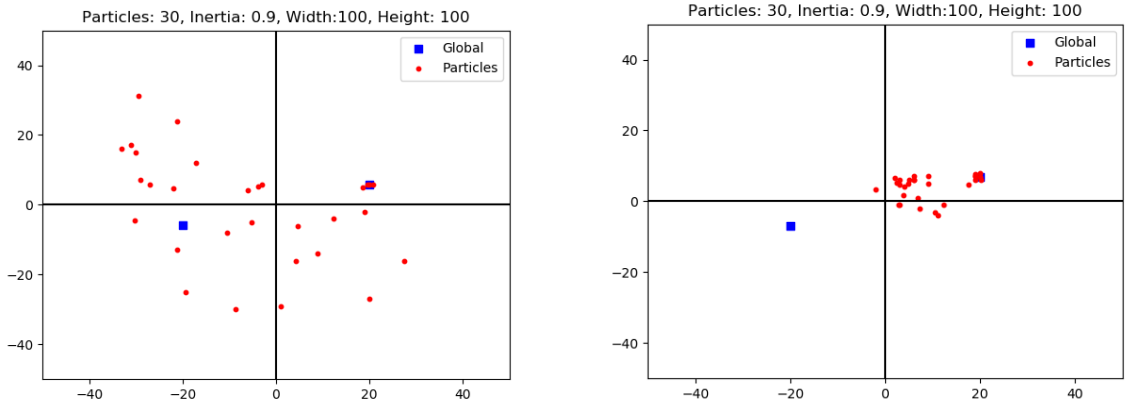


Figure 35: On the left is the search space after 10 epochs, on the right is after 40 epochs. This is with a cognitive parameter of 1.5

The low cognitive value heavily favors the particles in the vicinity of the global best, especially since it gives the social value more influence to bias the values to the position of the best particle. Particles beginning closer to the global maximum converged quite smoothly and efficiently. Contrast this with a higher cognitive value of 3.5 below:

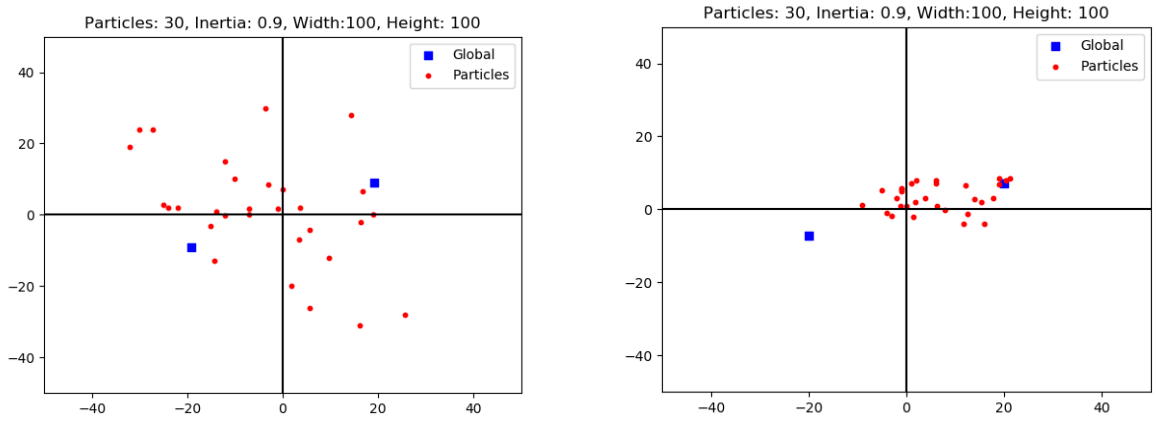


Figure 36: Notice how much greater distance is between the particles than in Figure 35

Since private knowledge is favored about public knowledge in this experiment, it will take longer for the population to converge, if it even converges in the first place. Note the greater distances between one another the particles have in Figure 36 compared to Figure 35, due to the public knowledge represented by the social parameter not valued as high.

Valuing the social parameter highly (public knowledge) tended to have a much greater effect on convergence and how quickly particles converged. Communication between particles to collaborate on descending on the best possible position was usually better than particles looking out for their own best positions.

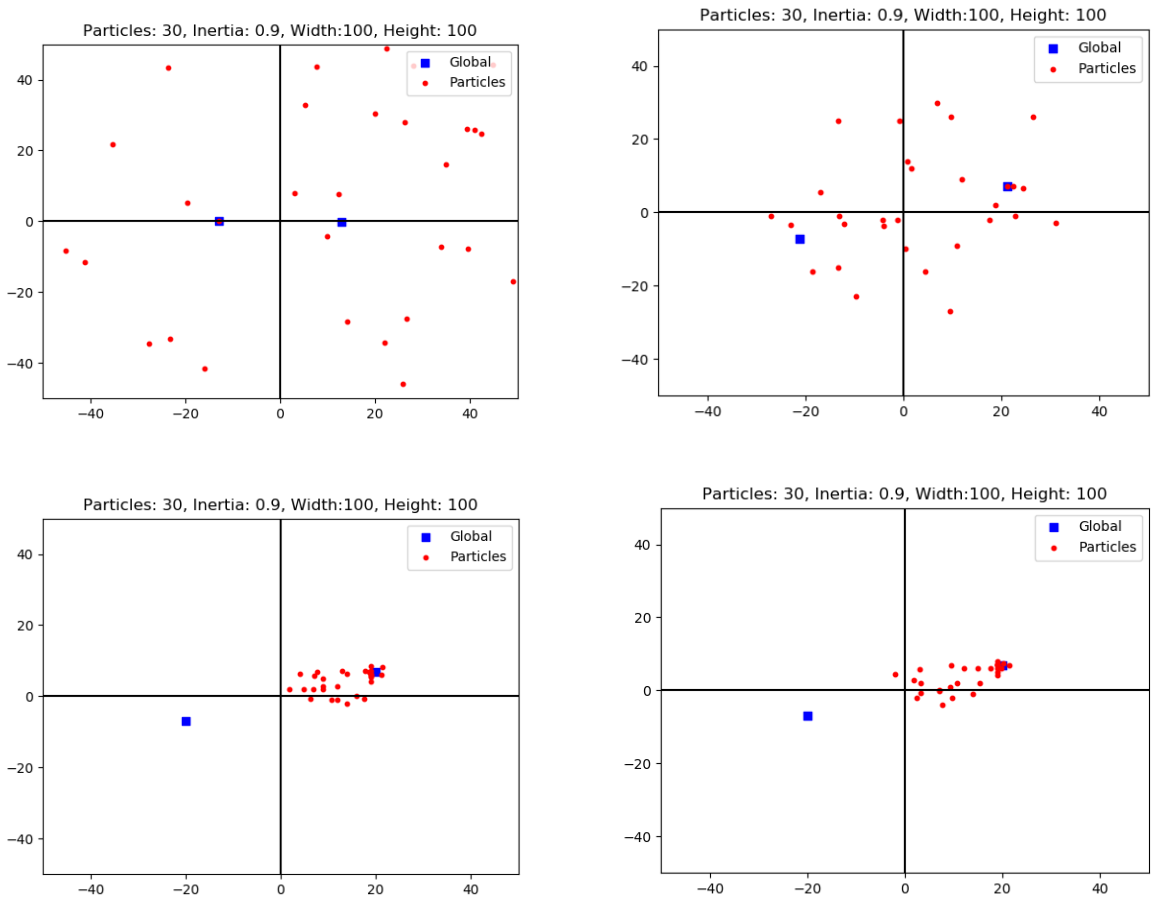


Figure 37: Initial swarm (top-left), 10 epochs (top-right), 40 epochs (bottom-left) and 100 epochs (bottom-right)

With a strong public knowledge and moderate cognitive knowledge (3.5 and 2 respectively) this swarm converged. They converged relatively efficiently, especially for Problem 2. Below shows its X/Y error to epoch chart:

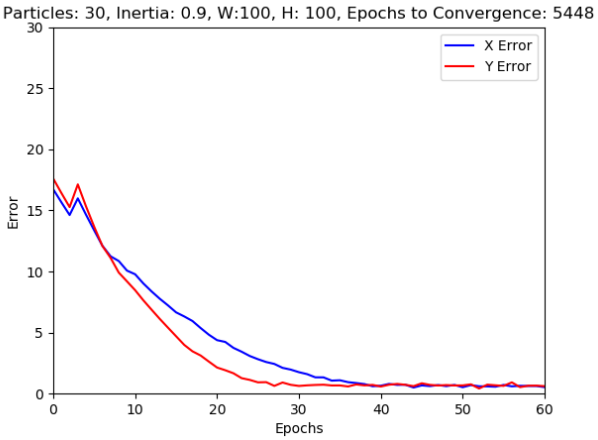


Figure 38: Note that it took 5448 epochs to converge. Convergence in general is rare for Problem 2 cases

The opposite could be said about smaller social values, and are best illustrated with the same particle positioning charts at epochs 1, 10, 40, and 100:

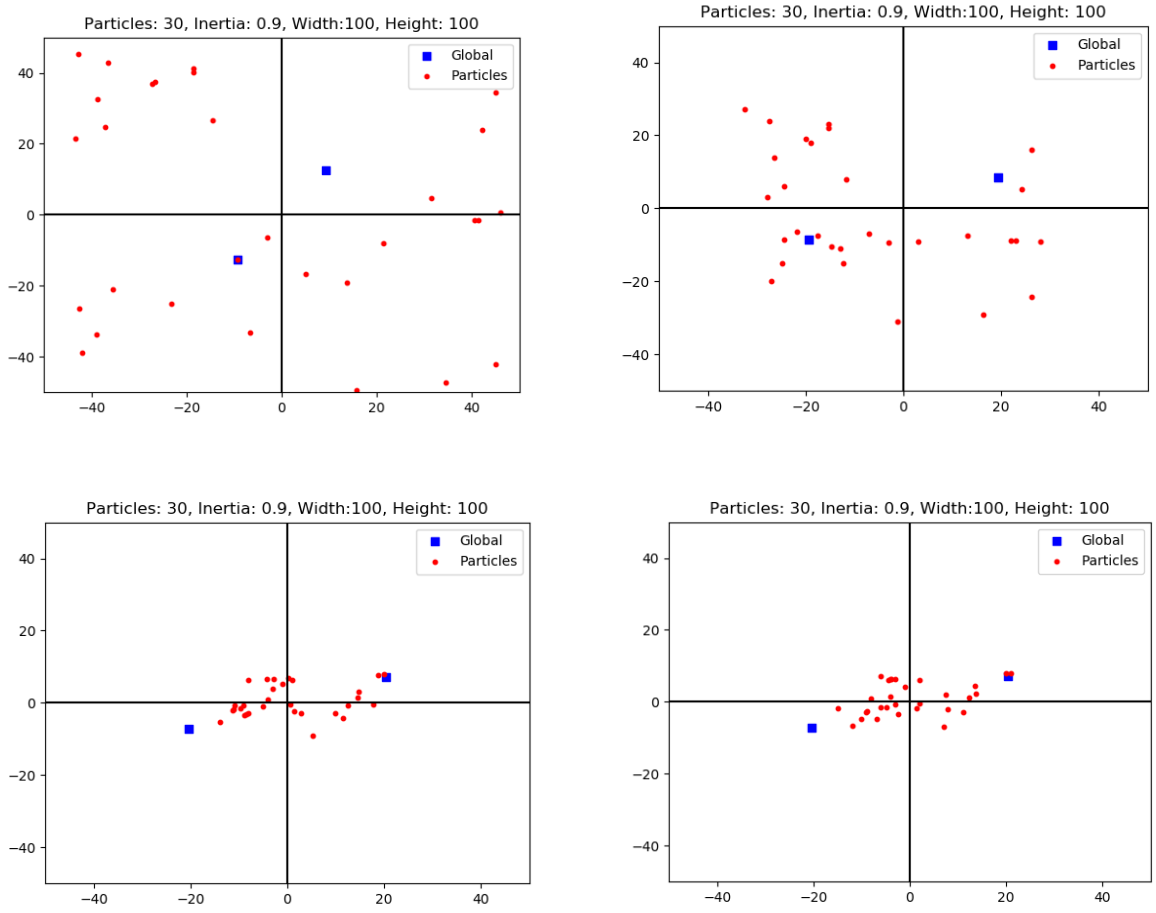


Figure 39: Initial swarm (top-left), 10 epochs (top-right), 40 epochs (bottom-left) and 100 epochs (bottom-right)

Note that the low communication between particles led to few particles in the vicinity of the global maximum. Additionally, there are many particles merely lingering in their positions between epochs 40 and 100. These factors led to high error and a lack of convergence.

III. Decreasing Inertia

The steady decreasing of inertia for each epoch had profound effects on the algorithm. This experiment set was conducted with a linear decrease of .001 and led to a significant decrease in the number of epochs until convergence, and was conducted for the conditions of both Problem 1 and Problem 2. All experiments for this set were completed using base case conditions aside from the number of particles in order to discover the true potency of the decrease in inertia. The algorithm will now allow particles to explore globally for better solutions and will be less likely to be trapped by the local maxima.

In Problem 1 for true base case conditions (with 30 particles), the algorithm converged significantly quicker:

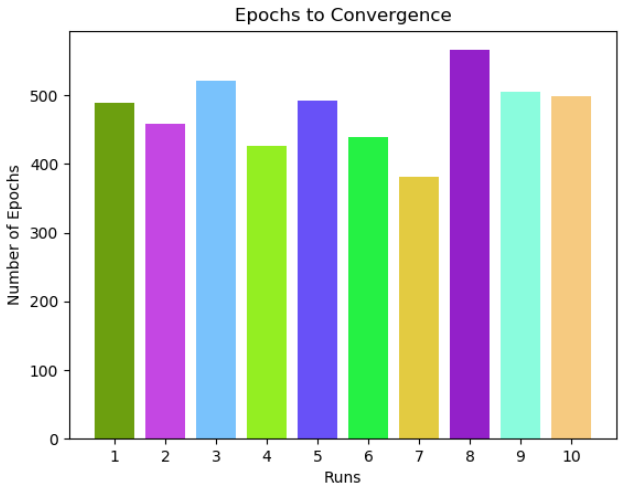


Figure 40: Averaged around 4000 less epochs than constant inertia

This is an incredible display of optimization, where the lowering of system energy allowing a more global search for particles and thus an easier time finding the global maximum. The optimization also relates to a problem that Problem 1 could not solve before: the convergence of very small populations. Due to this greater global search space, particles can communicate and search easier and more quickly, thus leading to convergence. See Figure 41 on the following page:

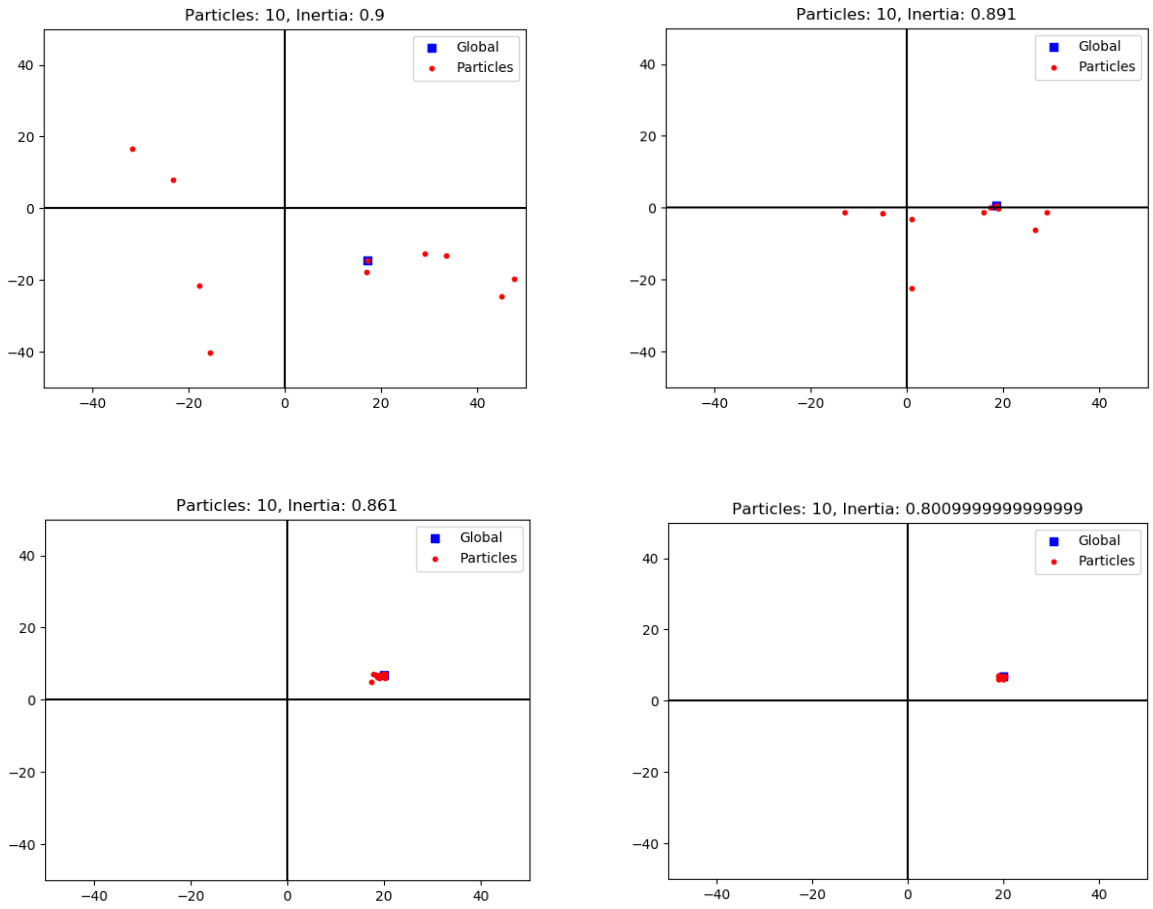


Figure 41: Initial swarm (top-left), 10 epochs (top-right), 40 epochs (bottom-left) and 100 epochs (bottom-right)

As seen in the movement of the particles, decreasing inertia linearly can be a great optimizer and fix many of the communication issues lingering with low numbers of particles. Additionally, there is not a great difference in the number of epochs until completion with large particle sets as opposed to normal or small sizes:

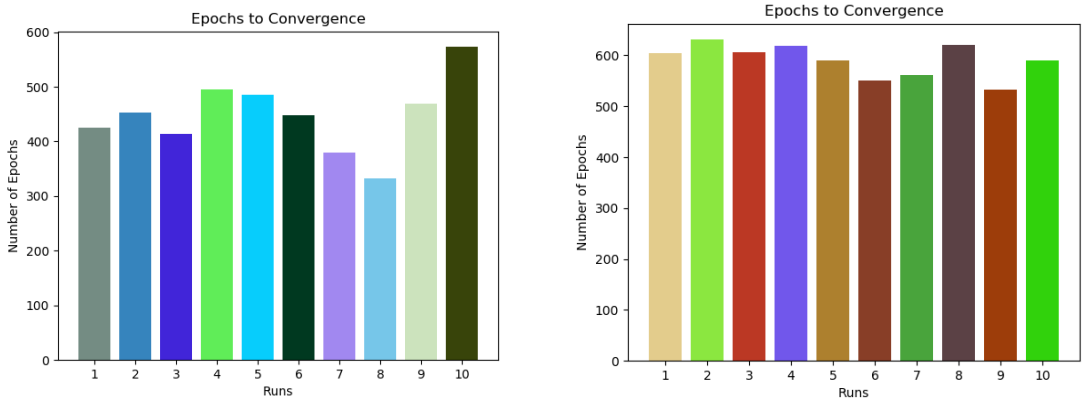


Figure 42: On the Left is the number of epochs for 10 particles, on the right is for 200

For Problem 2, the updated algorithm decreased the likelihood that particles would fall into the local maxima trap, though these did still occur. When they did and were converged, similar to Problem 1 they were optimized.

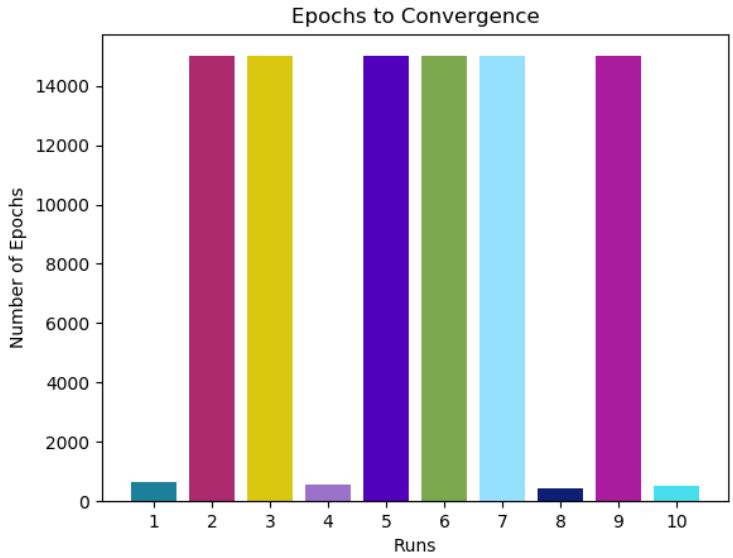


Figure 43: Note that runs were less likely to go to the local maxima, though when they did it was significantly quicker with a decreasing inertia

Multiple experiment sets were created for testing decreasing inertia, with the set below investigating the same base case parameters. Note that even those that do not fully converge in Figure 44 are still significantly greater in the percentage than in the previous sets.

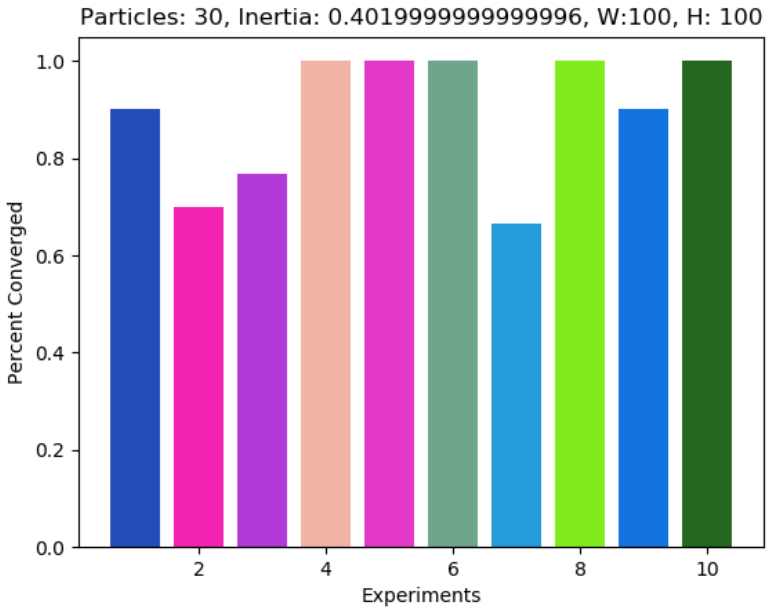


Figure 44: Note that the lowest convergence percentage is over 60%, which is still significant.

This is especially amplified when public knowledge and maximum velocity are increased, illustrated in Figure 45 below:

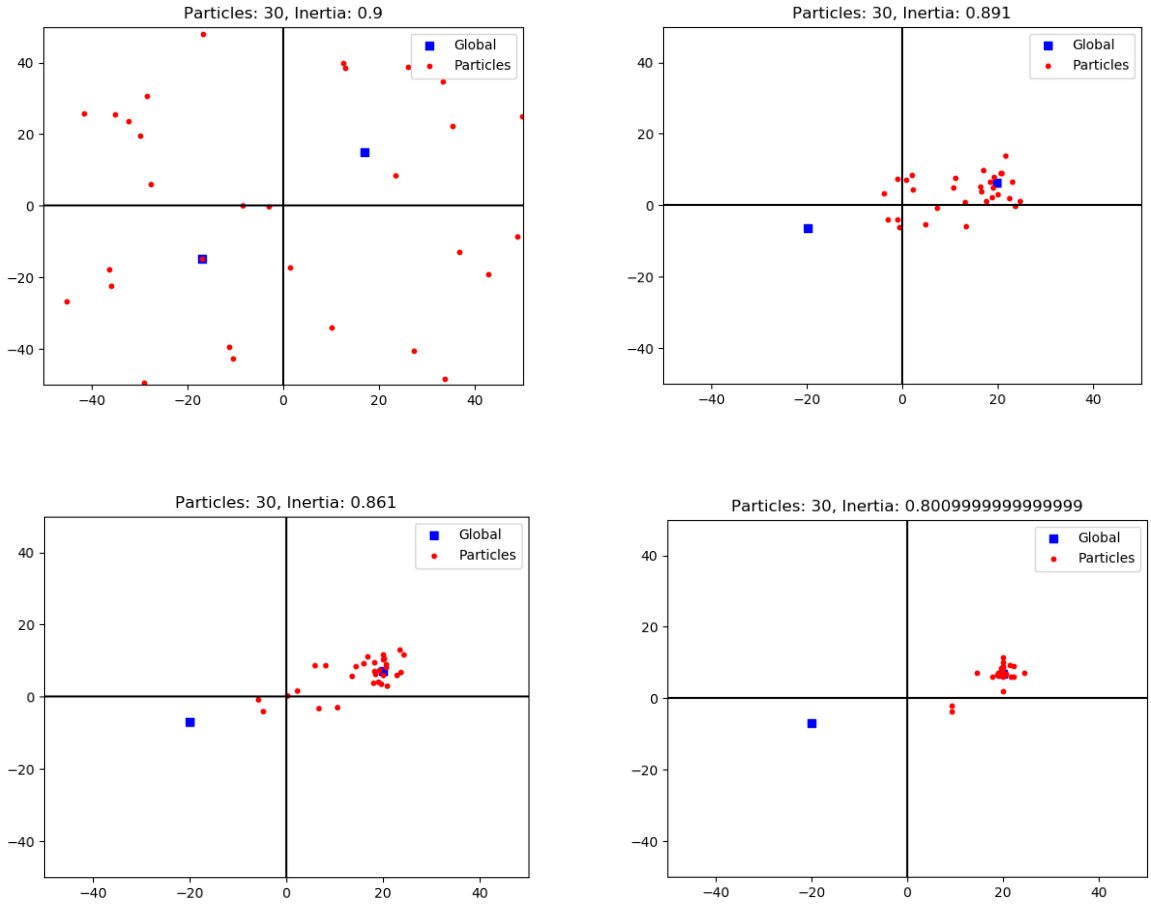


Figure 45: Initial swarm (top-left), 10 epochs (top-right), 40 epochs (bottom-left) and 100 epochs (bottom-right)

This was one of the rarer experiments where the Problem 2 positional quality formula produced a convergence on the global maximum. It seems that because the decrease of inertia linearly leads to an increase of search space; the maximization of public knowledge and more room for velocity can produce ideal results.

Conclusions:

Throughout each problem and experiment set, the biggest factor when it came to convergence in this project was inertia. The parameters for number of particles, maximum velocity, social (public knowledge), and cognitive (private knowledge) all made impacts as well, though none were felt quite like inertia. Problems were investigated for the effects of these parameters on convergence for two distinct problems; who had their own positional quality formulas. Problem 1 produced a single global maximum where convergence was very likely, while Problem 2 produced two maximums (a global and a local) where convergence usually occurred in the less ideal local maximum. In addition to these problems, another aspect was added; the linear decrease in inertia for each epoch, which usually led to a significantly optimized algorithm.