CS420 Project 6:
Particle Swarm Optimization

Robby Ferguson

May 6, 2016

Introduction

For Project 6, I was tasked with utilizing the technique of particle swarm optimization (PSO) to optimize two different problems. Inspired by swarm behavior observed in nature, PSO involves solving a problem by allowing a simulated population of potential solutions (each of which is represented by a particle within the population) to move throughout the search-space according to basic position and velocity functions. At any given point in time, each particle's current position in the search space is rated according to some sort of quality or fitness function. Higher fitness values correspond to positions in the space that further optimize the particular problem defined by the quality function.

Rather than move randomly about the space, each particle's movement is guided by two primary factors, dubbed the "social" and the "cognition" components. The cognition component influences a particle's movement toward the current best position in the search-space according to its personal search history, and the social component influences a particle's movement based on the current best position that any particle in the population has located within the search-space. In this way, each particle has a reference to the best location that it has seen thus far in addition to the best location that has been found by any particle.

The two quality functions that this simulation was designed to optimize are defined in the following way:

Problem 1:

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

Problem 2:

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

where *pdist, ndist,* and *mdist* are given by

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

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

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

Although they represent fairly simplistic functions, both Problem 1 and Problem 2 can be used to test the general characteristics of the PSO algorithm implemented. Problem 1 has a single maximum at (20, 7), and Problem 2 has two maxima: one at (20, 7), which is a global maximum, and one at (-20, -7), which is a local maximum.

By its nature, there are many variable parameters that are used to create a given PSO system, and the primary objective of Project 6 was to understand how each of these parameters generally influences the overall performance of the system. According to the project description, the main parameters to consider were the number of particles in the system, the inertia of the particles (which is used to calculate particle velocities), the cognitive component, and the social component. After experimenting with parameter combinations and the resulting data, I determined that another parameter, the maximum velocity, also seemed to play an important role in the behavior of the systems studied, so I chose to treat it as an additional system parameter and also explored its influence on the system performance in the same manner as the previously mentioned parameters. The maximum velocity acted as a threshold at which particle velocities were manually scaled down proportionally to the maximum velocity whenever they became too large. In order to draw relevant conclusions about how each of these parameters contribute to a given PSO system's performance, many variations for these parameter combinations were explored.

Methodology

My simulation program incorporates three different options for runtime behavior. If the correct number of arguments is provided on the command line, the program runs a single simulation with those parameters and outputs the resulting data to the given data file. Particles are initially placed randomly in the search-space. As the simulation runs, particles update their velocity and movement values, and individual fitness values are recalculated using the current problem's quality function. Each particle keeps track of its own best location thus far, and a pointer is continuously updated to point to the current best particle in the population, which is the one that has received the highest fitness value overall up to this point. Additionally, the overall error for the x and y coordinates of each particle with

respect to the current best particle in the population is recalculated every update iteration. This updating process continues for a maximum of 1000 epochs. However, if the error values for both the x and y particle coordinates drop below a predefined threshold of 0.001, the system halts because the majority of particles have converged to approximately the same location as the current best particle. The particular iteration at which the simulation halts the updating process is referred to as the convergence epoch.

Although the program can run a single simulation at a time if the proper arguments are provided, it can also run several together. If no arguments are provided, the program waits for input parameters on standard input. For each parameter combination given on standard input, the program runs a simulation with those arguments. For testing, I created a file that contained many different sets of simulation parameters and could redirect that file into my program as standard input, allowing for multiple simulations to be run together. The program also looks for particular flags as it parses the input lines that serve various purposes. Most importantly, one flag acted as a delimiter between different sets of simulations. Each set could be categorized as a certain experiment (e.g., the effects of varying the particle count), and the data for each simulation in a single experimental set were all written into a single data file for easier analysis of different experiments.

My program also offered a third option for runtime behavior. In order to determine a good base set of parameter values for Problem 1 and for Problem 2, I added the functionality to run a batch set of simulations based on the enumeration of parameter combinations. I created another input file that simply contained sets of values where each set contained some possible values for each of the variable simulation parameters used to build a given PSO system. The following parameter values were used:

$$Particle\ Count = \{10, 20, 30, 40, 50, 100\}$$
$$Inertia = \{0.1, 0.25, 0.5, 0.75, 0.9\}$$
$$Cognition = \{0.5, 1.0, 1.75, 2.0, 3.0, 4.0\}$$
$$Social = \{0.5, 1.0, 1.75, 2.0, 3.0, 4.0\}$$
$$Maximum\ Velocity = \{1, 2, 3, 5, 7, 10\}$$

The program simply enumerated each of the parameter combinations that could be produced from these value sets and ran 4 different simulations with each combination (to try to account for outlier behavior) before averaging some resulting data across those 4 runs. This enumeration resulted in 6840 different parameter combinations. Each of these simulations was ranked according to its average convergence epoch (across the 4 different runs that used the same parameter values). For Problem 2, an additional ranking criterion was added such that no more than 1 of the 4 simulation runs could converge to the local maximum position of (-20, -7) in order for that parameter combination to be considered for ranking. This criterion was introduced in an attempt to eliminate parameter combinations that tended to converge to the local maximum, rather than the global maximum.

The remaining valid parameter combinations were placed in a list and sorted by convergence epoch. The list contained only the current 10 best parameter sets (those that resulted in the lowest average convergence epoch). This ranking process was done for both Problem 1 and Problem 2, and the resulting parameter combinations are shown in the tables below.

Problem 1 Best Simulations (when averaged over 4 runs):						
Rank (Sim #)	Avg Convergence Epoch	Particles	Inertia	Cognition	Social	Maximum Velocity
1	14	50	0.1	0.5	1	7
2	15	20	0.1	0.5	1	10
3	15	30	0.1	0.5	1	10
4	16	20	0.1	0.5	1	7
5	16	20	0.1	1.75	1	7
6	16	20	0.1	1.75	1	10
7	16	30	0.1	1	1	7
8	16	30	0.1	1	1	10
9	16	40	0.1	0.5	1	10
10	16	40	0.1	1	1	7

Problem 2 Best Simulations (when averaged over 4 runs):						
Rank (Sim #)	Avg Convergence Epoch	Particles	Inertia	Cognition	Social	Maximum Velocity
1	30	100	0.1	0.5	1.75	7
2	35	100	0.25	0.5	1.75	5
3	36	40	0.25	0.5	1.75	5
4	36	100	0.1	0.5	2	5
5	37	30	0.1	1	2	5
6	40	100	0.25	0.5	2	5
7	41	50	0.25	1	2	5
8	41	100	0.1	0.5	1.75	5
9	44	40	0.25	1	1.75	5
10	44	100	0.25	0.5	1.75	7

According to these results, the best simulation parameters (of those examined in the batch runs) for Problem 1 and Problem 2 are given as follows:

Problem 1 Base Parameters

Particles: 50
Inertia: 0.1
Cognition: 0.5
Social: 1.0
Maximum Velocity: 7

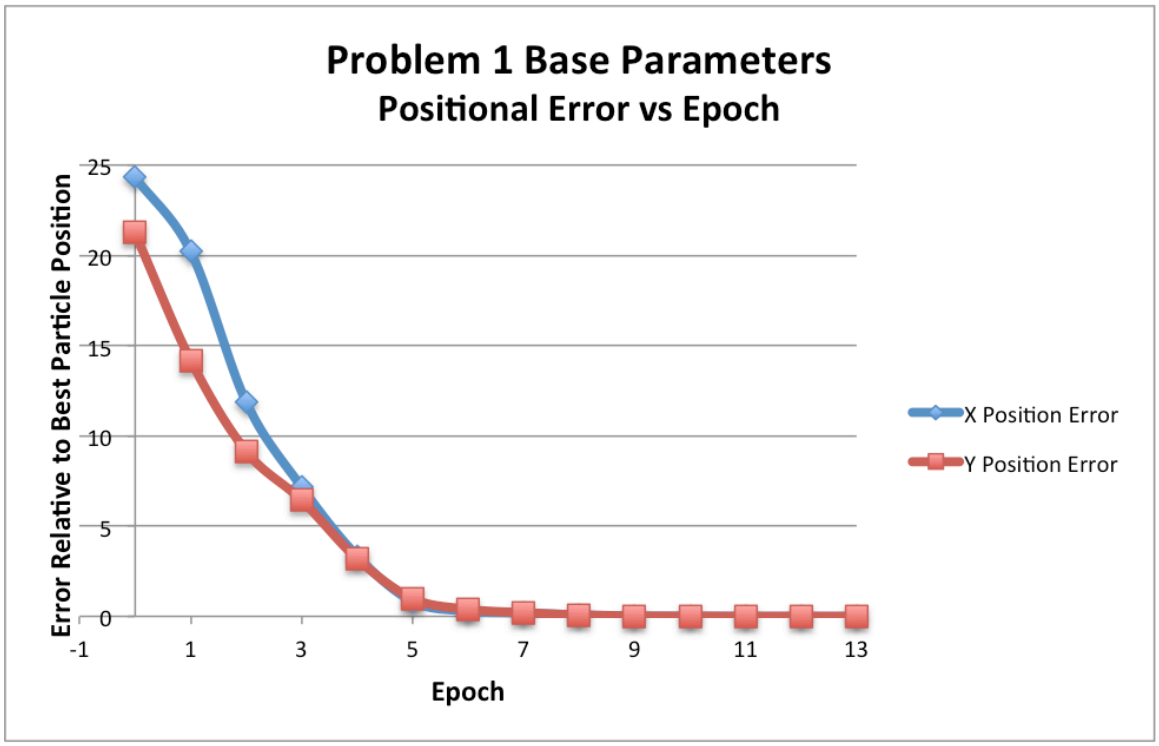
Problem 2 Base Parameters
Particles: 100
Inertia: 0.1
Cognition: 0.5
Social: 1.75
Maximum Velocity: 7

These are the parameter sets that were used as the "base" values for each problem because, on average, they converged in the shortest amount of time and converged to the global maximum more often than the local maximum (for the Problem 2 parameters). When examining the effects of a given parameter value, I used these base values and altered a single parameter to several different values, while keeping the other values constant.

Results/Conclusions

Problem 1

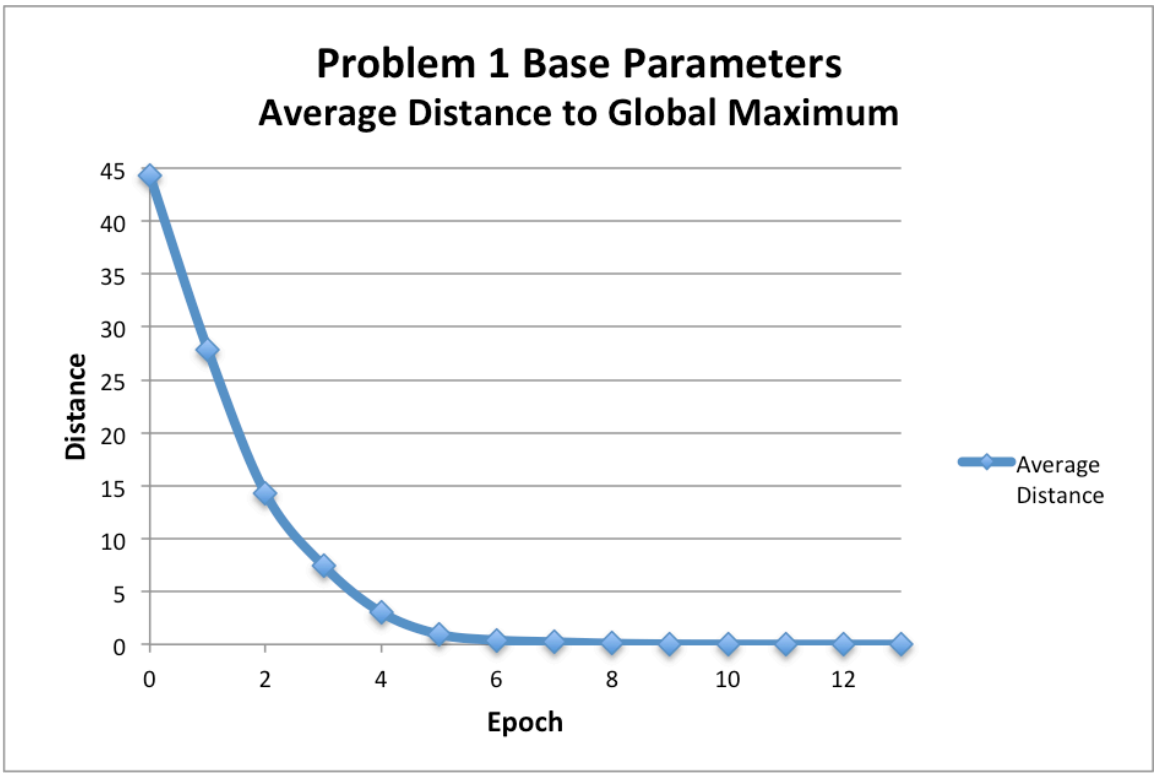
To start, I examined some of the characteristics of a typical run of the base parameter set for Problem 1 (described above). No parameters were altered at this point. Below is the error in the x and y coordinates of the population of particles at each epoch up to the simulation's convergence epoch.



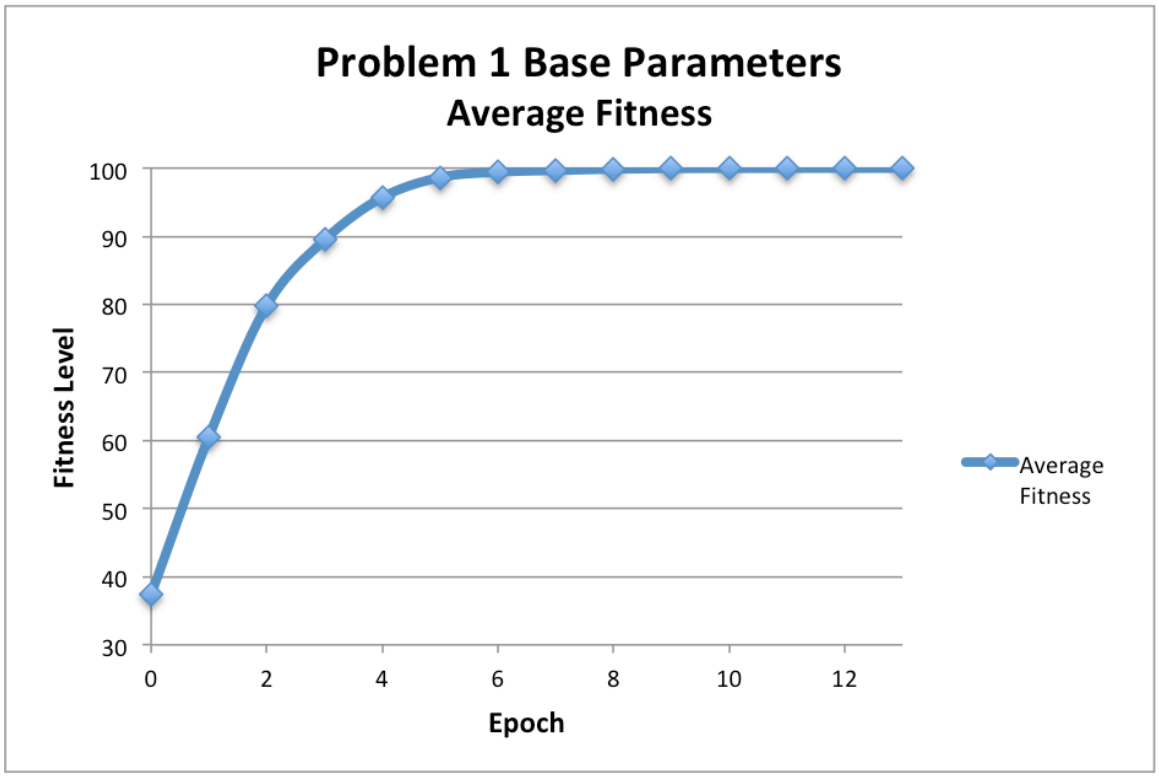
By epoch 14, the positional error values in this particular simulation dropped below the error threshold, and the simulation halted. From the above figure, the error values descended rather quickly, approaching miniscule error values for both

coordinate components around epoch 6. This behavior reinforces the fact that this parameter combination seems to produce efficient systems that rapidly converge to the maximum for Problem 1.

In accordance with this observation, the following chart shows the average distance of each particle in the population to the global maximum as time progresses.



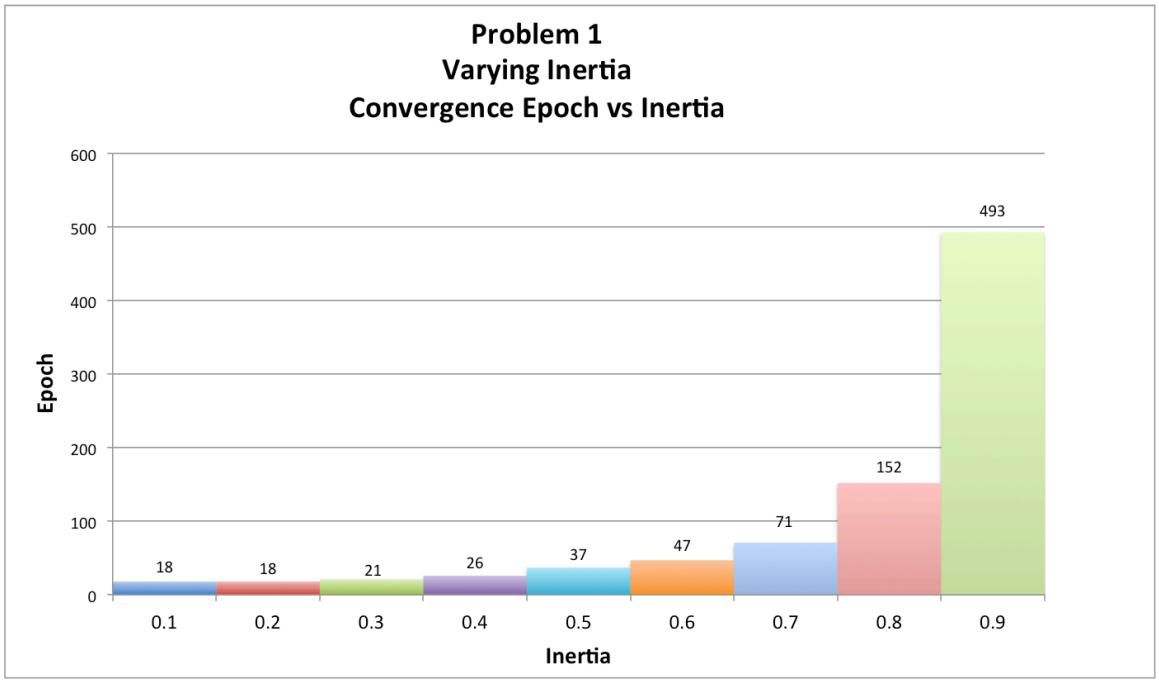
Collectively, the particles in this population seem to move almost directly to the maximum value with little interference. As an additional metric of system's performance, the average fitness of the population according to the quality function defined for Problem 1 is shown below.



These results suggest that the base parameters chosen for Problem 1 already seem to perform quite well. However, the performance could still be improved. To determine the contribution of each parameter to the system's effectiveness, each was individually examined.

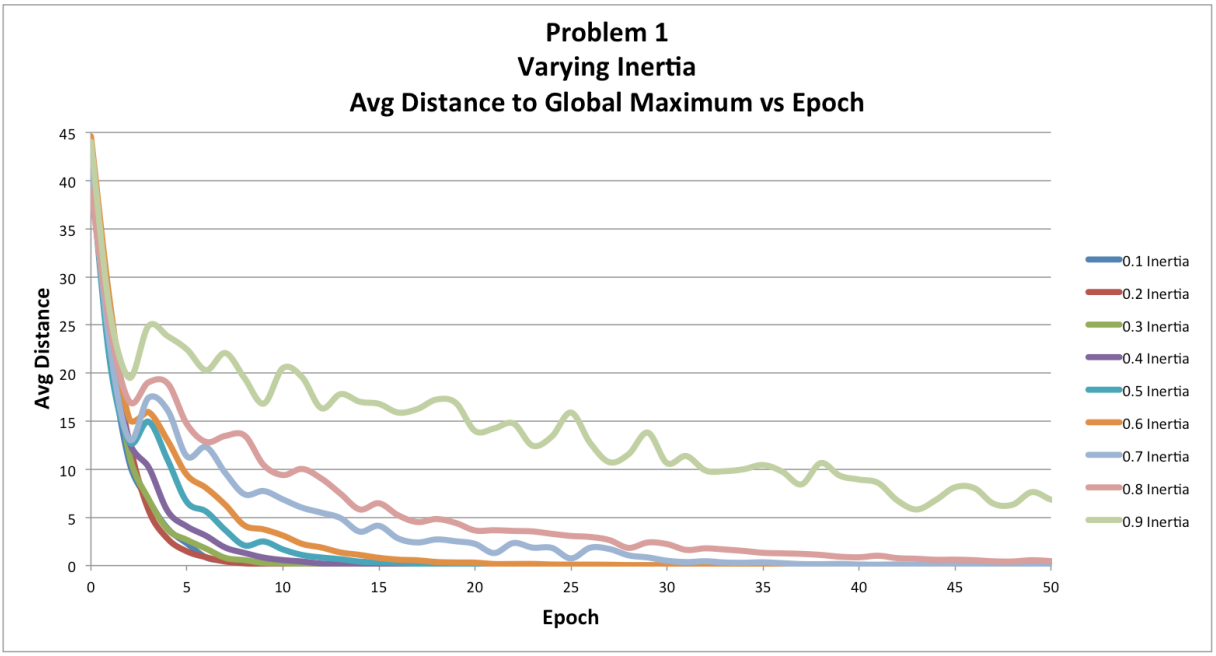
First, the inertia parameter was varied using the following values: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9. During my initial experimentation and testing, I noticed that the inertia seemed to play a significant role in the overall behavior of the system. As such, I chose a fairly large number of different variations to test.

A single simulation run was done for each of these 9 variants to the inertia value. All of the other system parameters remained constant according to the base parameter set for Problem 1. Below is a figure showing the particular epoch at which each of these systems converged.



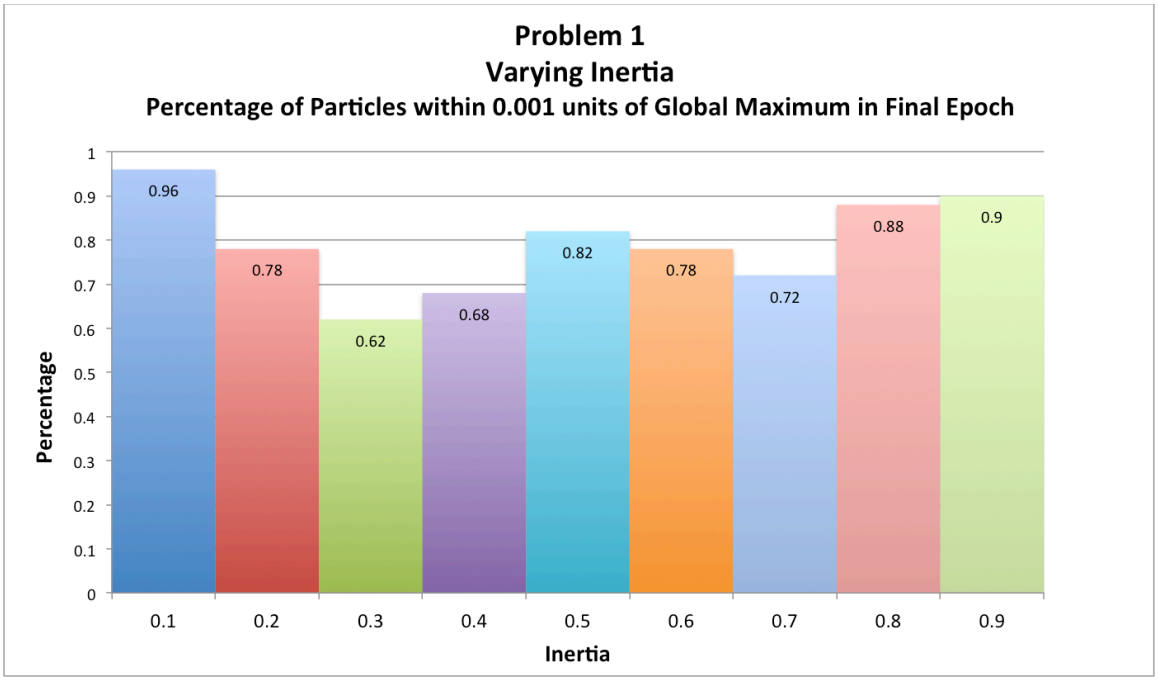
The simulations with the 0.1 and 0.2 inertia values converged most quickly at 18 epochs. As the inertia value of each simulation increased, so did the corresponding convergence epoch. Inertia values from 0.6 to 0.9 seemed to produce a much more drastic increase in the time to convergence for the respective simulations. More specifically, higher inertia values in the upper half of the range examined increase the time it takes to converge at a nearly exponential rate. The system using an inertia value of 0.9 did not converge for 491 epochs, which is starkly different from the 18 epochs it took the 0.1-inertia simulation to converge.

In general, the primary goal of the PSO algorithm is to optimize the given quality function. In this case, that would correspond to particles converging to the global maximum point in the search-space. In the following figure, the average distance of each particle in the population to the global maximum is shown with respect to each epoch.



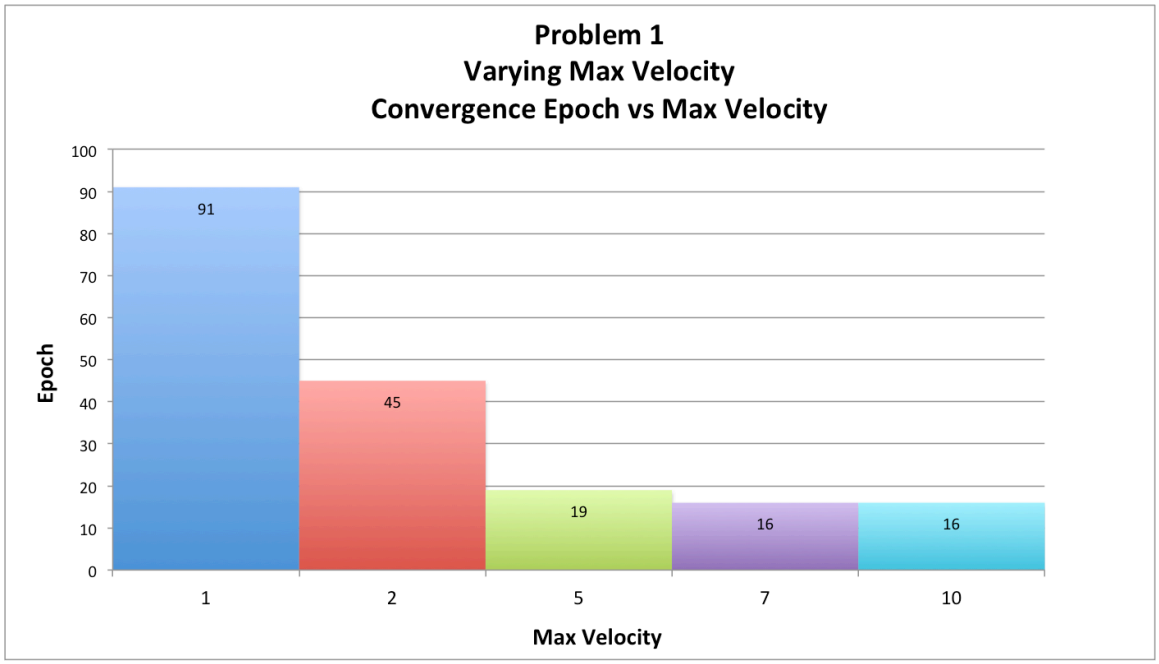
The above chart was limited to show only the first 50 epochs, even though the 0.9-inertia system took much longer to converge. The 0.2-inertia system seemed to decline the fastest, but inertia values of 0.1, 0.2 and 0.3 all seem to result in similar behavior. The particles in these simulations converged very quickly and nearly directly to the global maximum. As the inertia value increased, the average distance became much less steady. Because the inertia was directly related to how the velocity of each particle was updated with each new epoch, higher inertia values seemed to cause particle velocities to fluctuate more sporadically at each time step. As a result, it likely made it more challenging for particles to consistently move in a given direction.

Although the convergence epoch is a fairly good measure of the efficiency of a particular system to reach the global maximum, I also recorded the fraction of particles that were within 0.001 units of the global maximum whenever the system had halted due to convergence. This measure would provide a better metric for the accuracy of the system, rather than just its efficiency. These results are shown in the following figure:



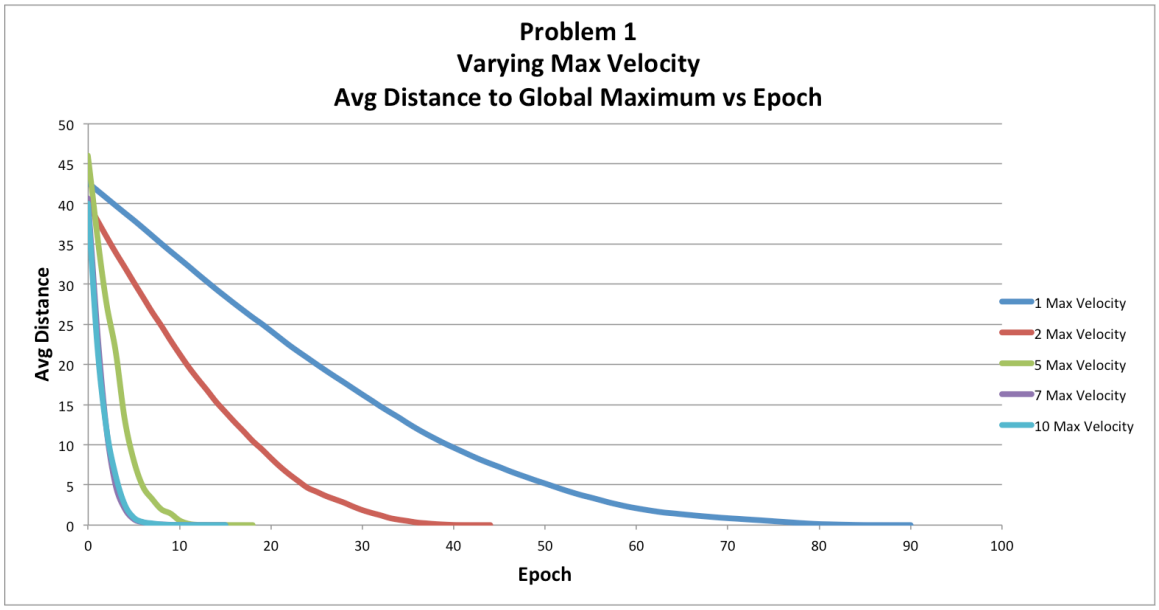
Although the converge epochs for the 0.1-inertia and the 0.2-inertia systems were the same, the percentage of particles in each population that were "close" to the global maximum (within the 0.001-unit threshold) upon convergence is fairly different. 96% of the particles in the 0.01-inertia system were close to the global maximum, but only 76% of the particles in the 0.02-inertia system were considered close. From the above graphs, the mid-range values for inertia seemed to be the least accurate in the end, but inertia values of 0.1 and 0.9 (with 0.8 following closely behind) produced the most accurate systems where nearly all of the particles in the system were within 0.001 units of the global maximum. Although the inertia value of 0.9 took much longer to converge than the other systems, it made up for the lack of efficiency by being more accurate overall. However, the 0.01 inertia value still seemed to perform the best.

Ultimately, the effects of varying the remaining parameters were examined in the same way as the inertia parameter above. The second parameter that I analyzed was the maximum velocity. Although the project description did not explicitly state to study this parameter, I noticed that the maximum velocity had a very noticeable effect on how the systems behaved. The maximum velocity was varied with the following values: 1, 2, 5, 7, and 10. Below is a figure showing the resulting convergence epochs for these simulations.

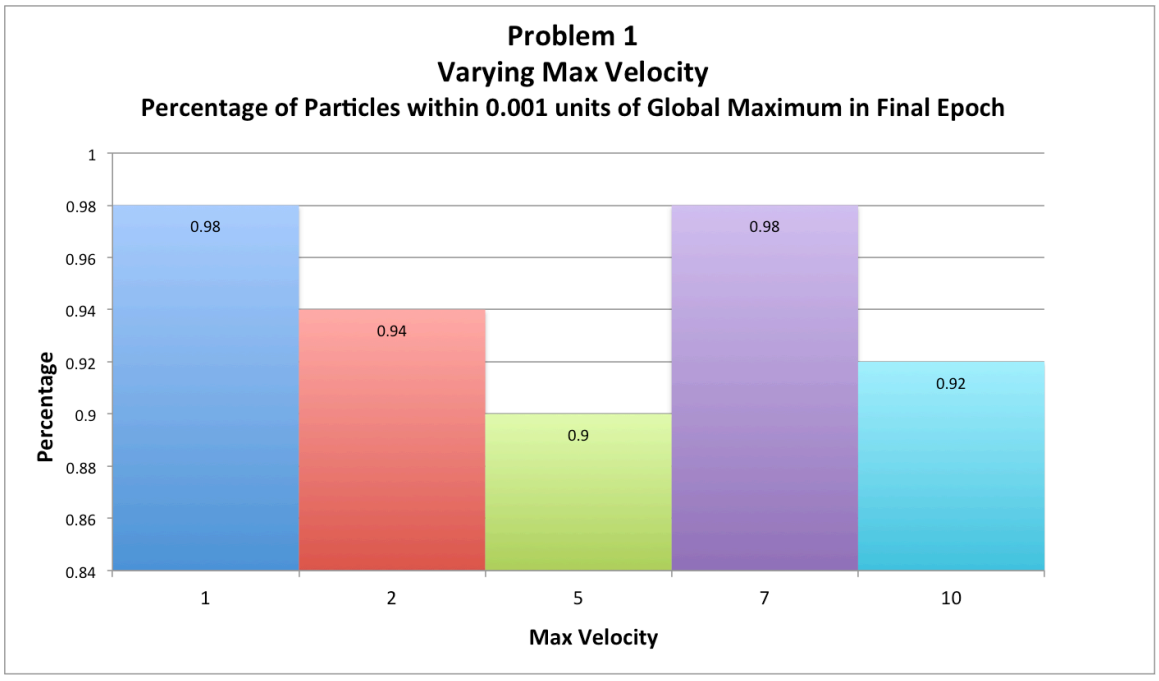


From the figure, as the maximum velocity increases, the time it takes for the system to converge decreases. Because the maximum velocity is used as a threshold for a particle's velocity, lower maximum velocity values will mean that particles cannot move as much in a single time step. The natural conclusion to draw from this fact is that moving less each time step means that the particles will need more time steps to reach their destination—the global maximum. Maximum velocity values of 7 and 10 both seem to work well with respect to the convergence speed of a system.

This same fact can be seen from a different perspective in the chart below that shows the average distance to the global maximum over time. The distances decrease steadily at each time step for each maximum velocity value, but the higher maximum velocity parameters take much longer to reach low distances than lower maximum velocity parameters do.



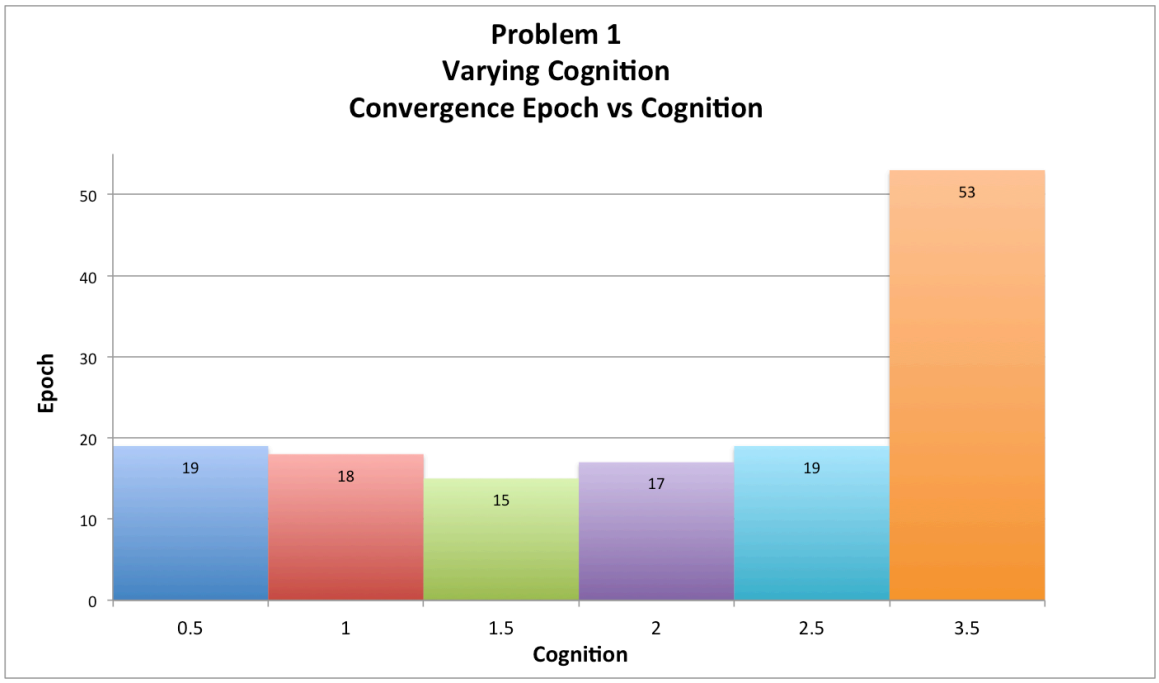
The previously described data suggest that the maximum velocity values of 7 and 10 are most efficient, but the general accuracy of each value can be seen in the following figure:



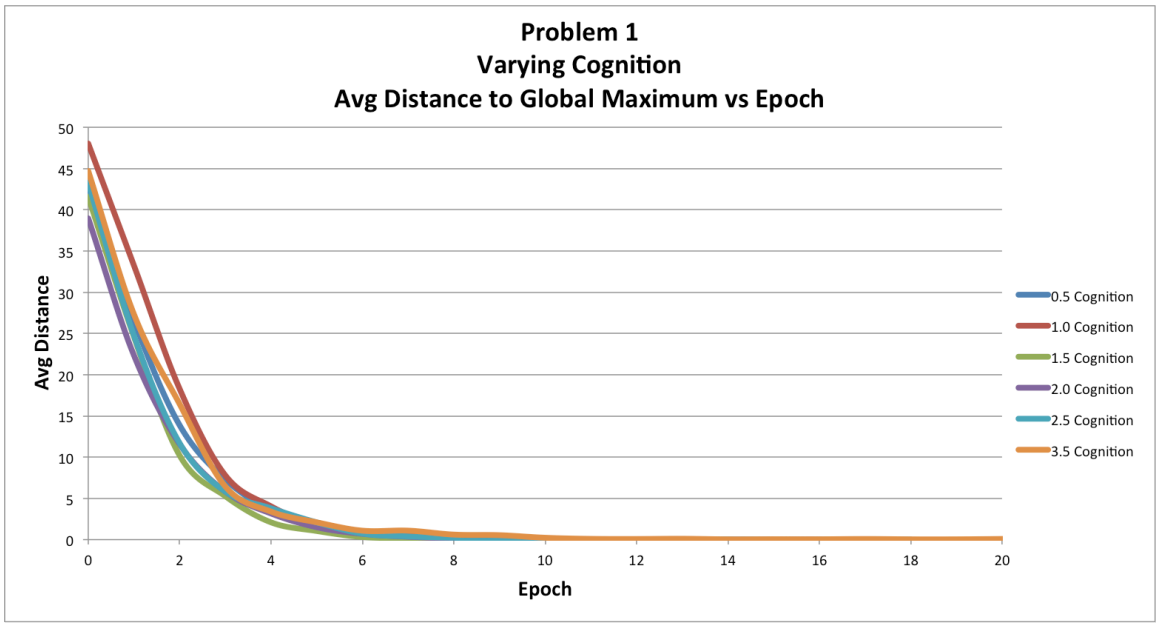
The two maximum velocity values that seem to produce the most accurate systems are, in fact, 1 and 7. Although a maximum velocity of 1 was much slower to converge, it allowed for particles to slow down more easily and stay within the 0.001 distance threshold when they actually reached the global maximum. Interestingly enough, a maximum velocity of 7 was just as accurate as 1, but increasing the value to 10 made the system less accurate. However, the range of

values across all of the parameter values tested is 0.9-0.98, so, in reality, the difference between these maximum velocity values is not particularly notable and could simply be due to more or less favorable initial conditions, rather than the velocity values themselves. It seems that the maximum velocity may not have much influence on the accuracy of the system, but has a fairly large influence on the convergence speed.

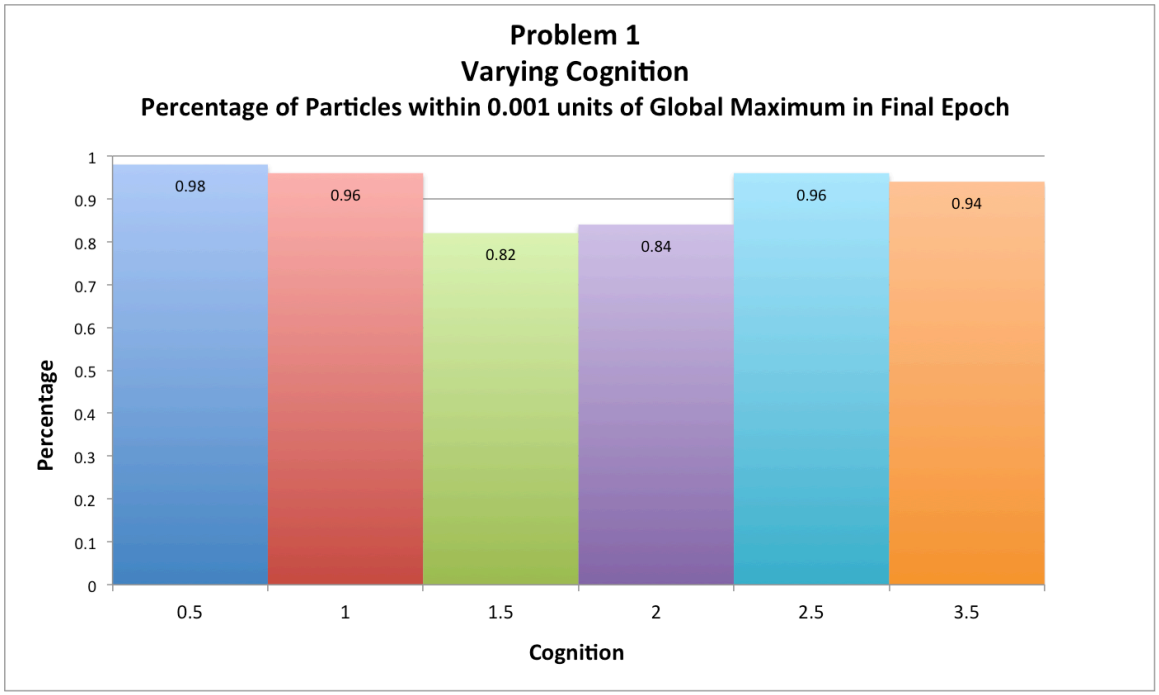
Next, the cognition parameter was varied. The values used were 0.5, 1.0, 1.5, 2.0, 2.5, and 3.5. The resulting convergence epochs for each of these simulations can be seen below.



Oddly enough, most of the cognition values did not have much of an influence on when each system converged, with the exception of a cognition value of 3.5, which obviously took quite a bit longer for the simulation to halt. The average distance to the global maximum, which is depicted below, supports this observation.

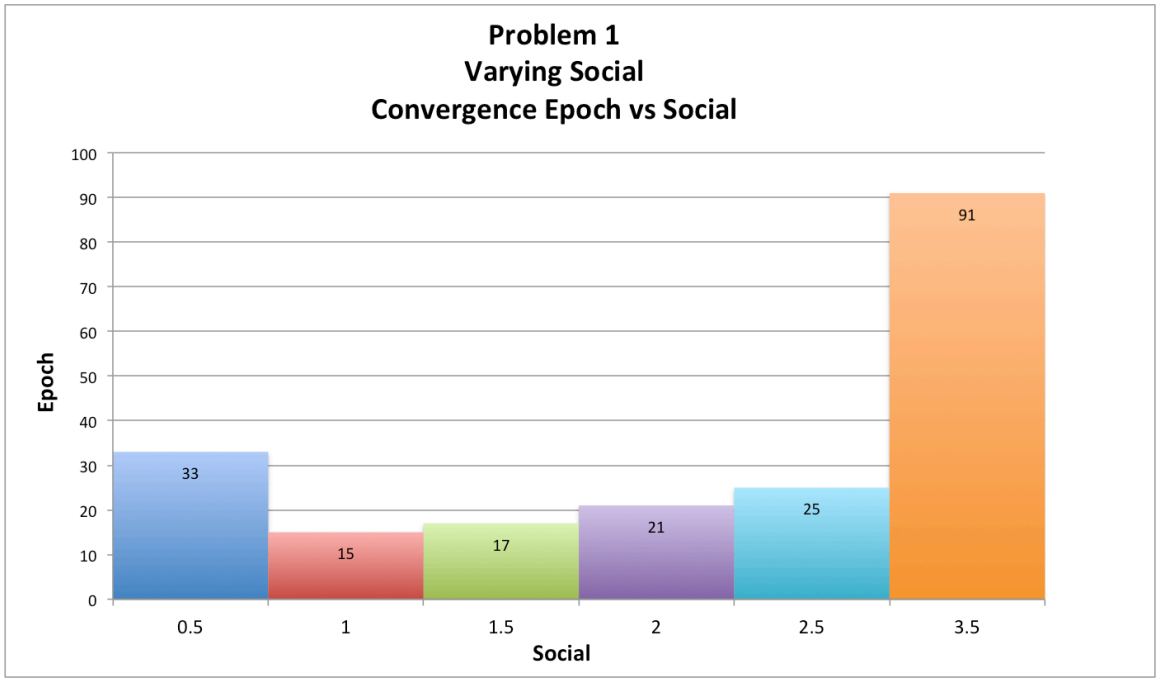


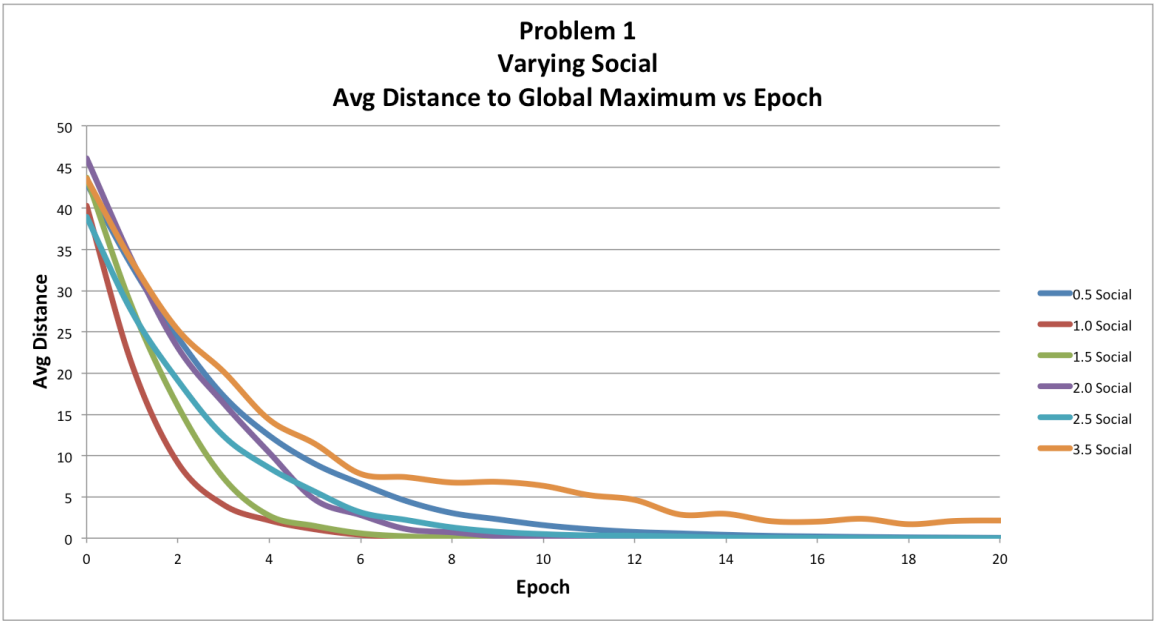
It appears that all of the particles in each simulation approached the global maximum at very similar rates. Although the cognition value of 3.5 took much longer to actually converge, it still displayed similar behavior to the other simulations with respect to the average distance from the figure. Perhaps the 3.5-cognition simulation fluctuated very closely to the error threshold required for the simulation to halt without actually going below that threshold.



The above figure shows how those cognition variables influenced the accuracy of the system upon convergence, but these results are also uninteresting. The cognition value of 0.5 was slightly more accurate than the other values, but the other simulations still performed relatively well. Cognition values of 1.5 and 2 were the least accurate. Overall, the cognition parameter had less of an influence on the system as I expected. Perhaps this is due to the other parameters that were used.

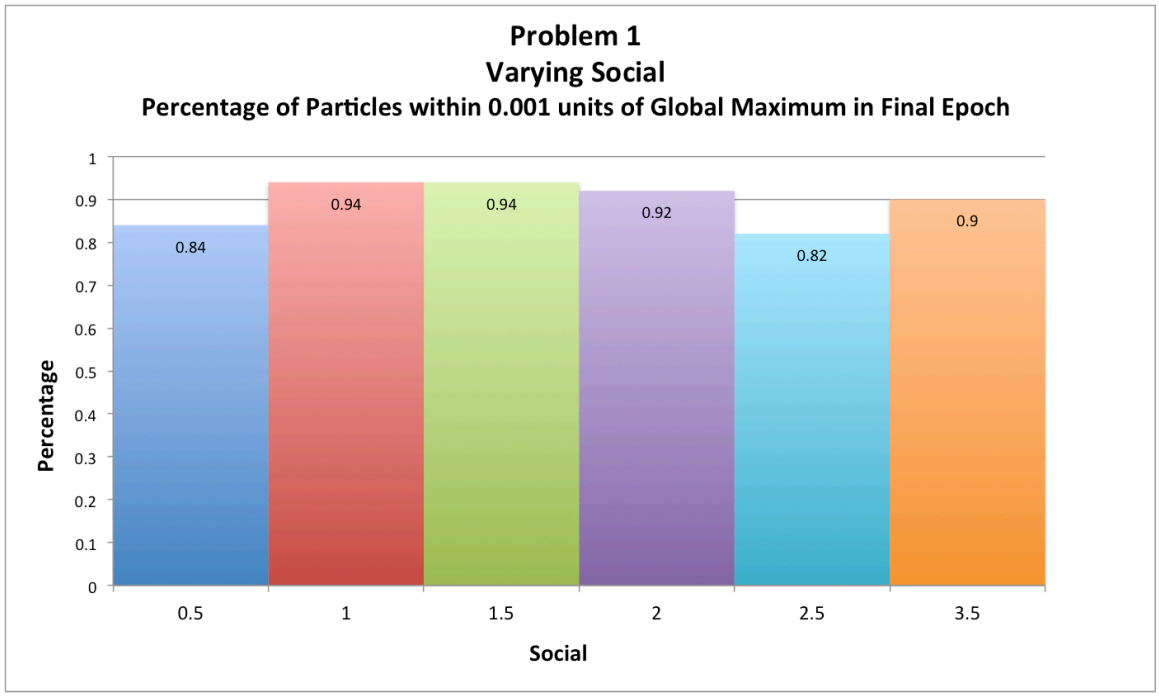
After the cognition parameter, I examined the social parameter, which was varied using the same values as the cognition parameter: 0.5, 1.0, 1.5, 2.0, 2.5, and 3.5. Interestingly enough, the social parameter variations performed similarly to the cognition parameter variations. The final convergence epochs and the average distance from the global maximum at each epoch are shown in the following figures:



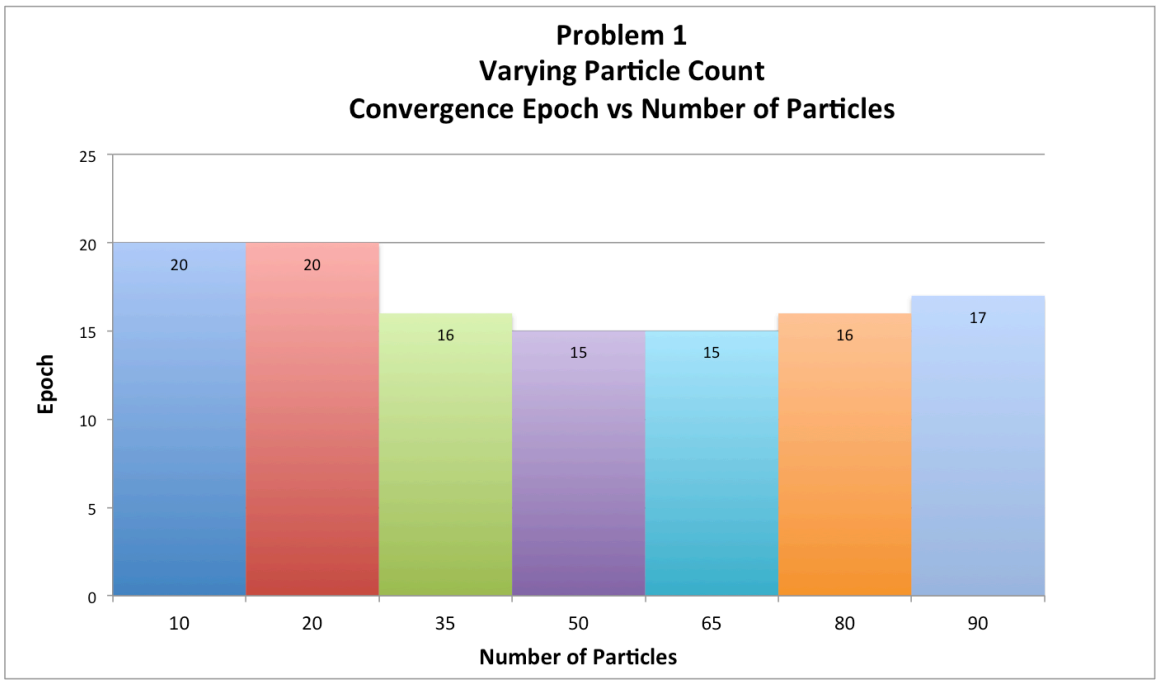


The social values on each end of the range that was used performed the worst with respect to the actual epoch at which the simulation halted, but the 3.5-social simulation took much longer to converge than the 0.5-social simulation. A social parameter value of 1 seemed to help the simulation converge most quickly. The graph showing the average distance to the global maximum reflects similar behavior. The 1.0-social simulation approached the global maximum notably faster than any of the other social variants. The simulation with the social value of 3.5 seemed to approach the global maximum much less steadily than the other simulations, which suggests that a social value that high negatively impacted the particle population's ability to converge to the correct point, possibly because there was too much interference from the current "best" particle at each iteration.

With respect to the percentage of particles that were close enough to the global maximum upon convergence (shown in the figure below), social values of 1, 1.5, and 2.0 were fairly close to each other for the highest percentage. These three values are likely the best parameter combinations to produce a more accurate PSO system.

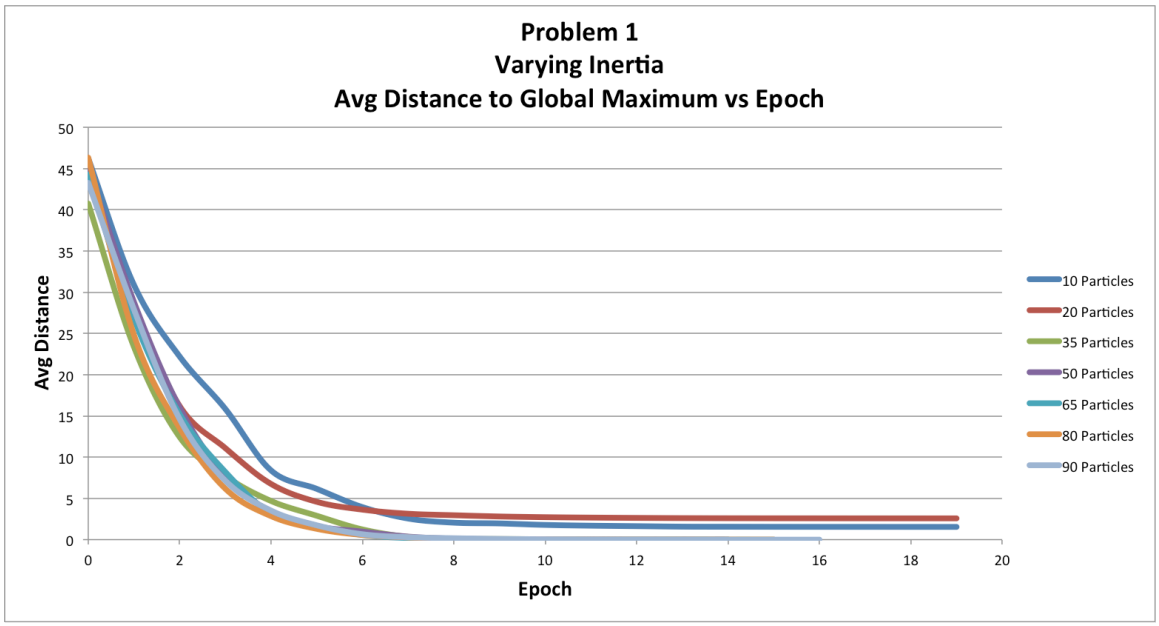


The final parameter that was studied was the number of particles in a given population, which was varied with values of 10, 20, 35, 50, 65, 80, and 90. The following figure shows the convergence epochs for each of these simulations.

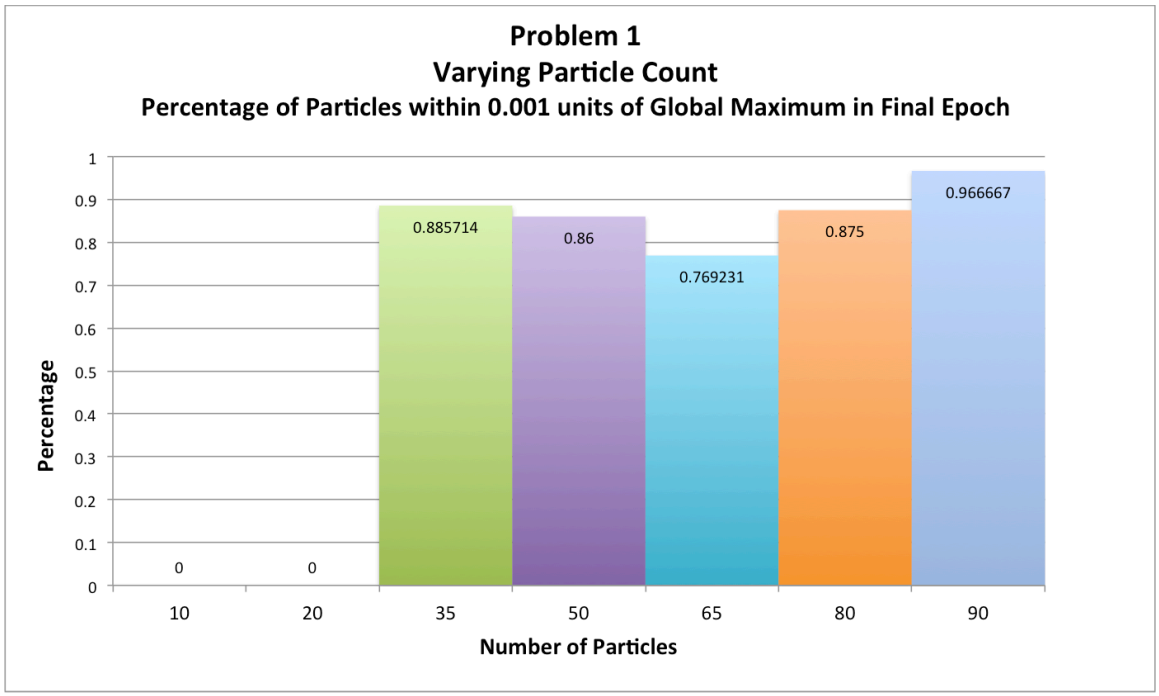


From the data, it appears that the number of particles did not have a very large influence on the convergence rate of the system. The figure below shows the average distance to the global maximum, and it also shows that most of the

variations to the particle count displayed similar behavior to one another with respect to convergence speed. Lower population sizes performed the worst.



Although the convergence rate was not heavily affected by the particle count, the figure below shows that the accuracy of these systems was influenced much more strongly.

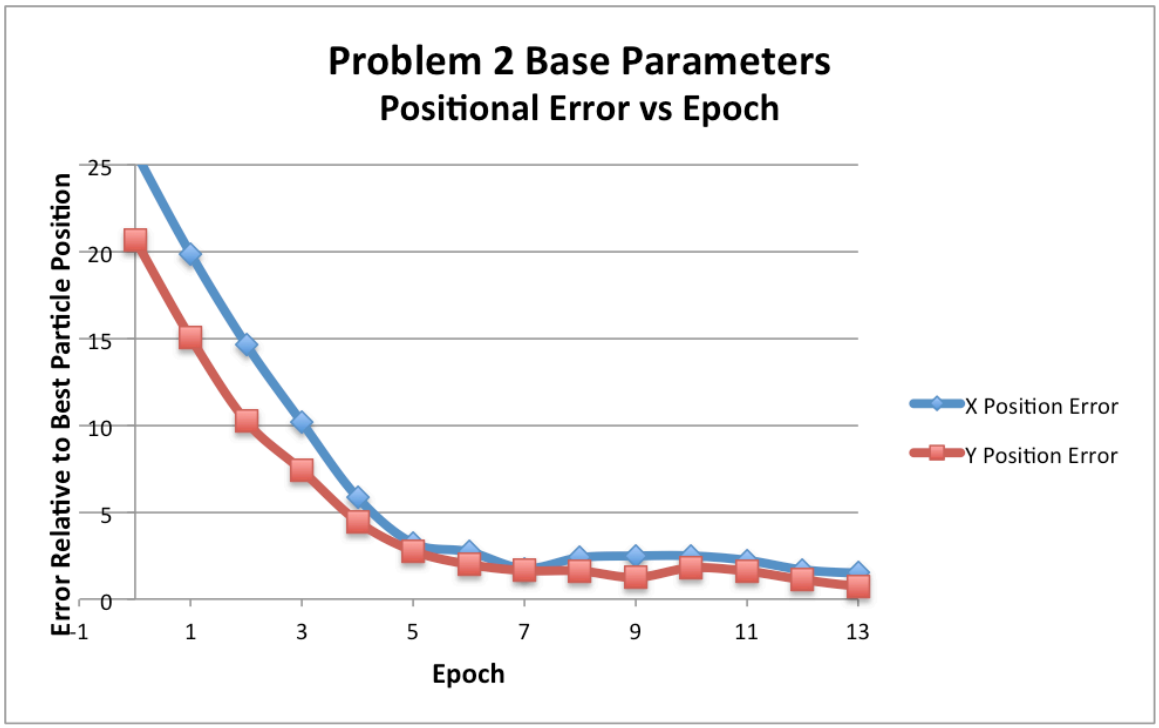


When comparing the percentage of particles that were within 0.001 units of the global maximum, population sizes of 10 and 20 actually had no particles that fell

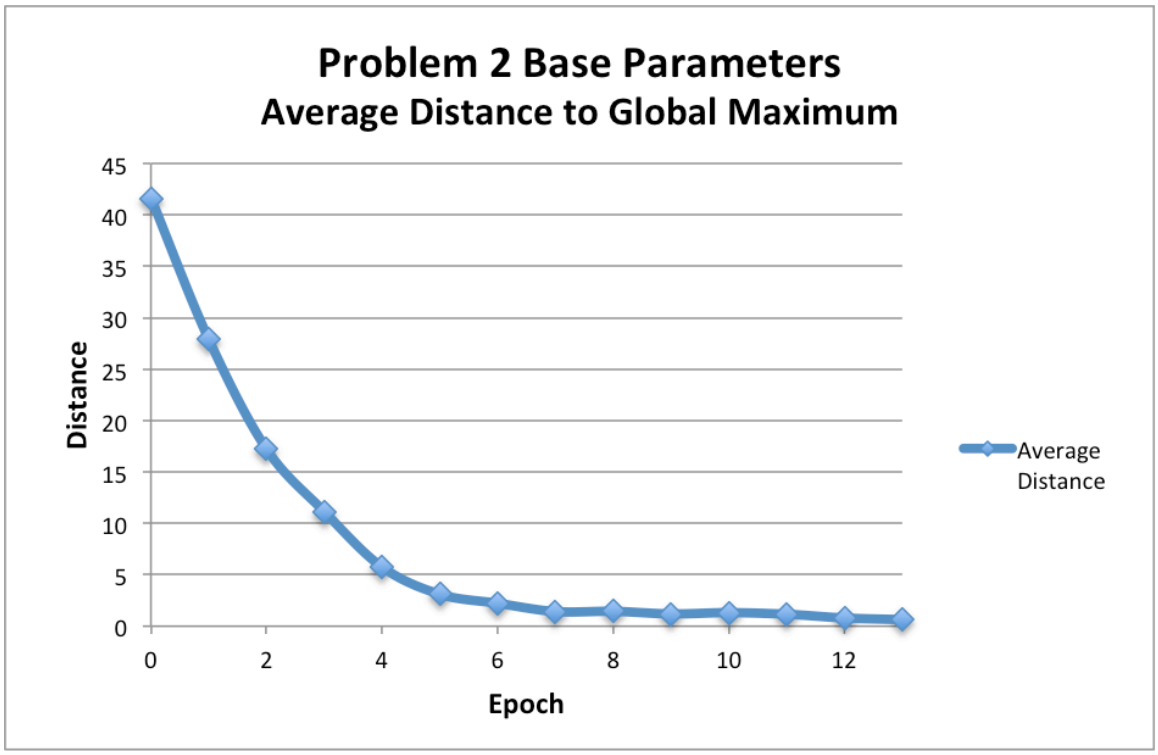
into that range. Although their error values dropped low enough for the system to halt, the particles were still relatively far away from the actual maximum point in the search-space. This accuracy issue was resolved for all of the other values for population size. It appears that a larger population size helped produce more accurate simulations.

Problem 2

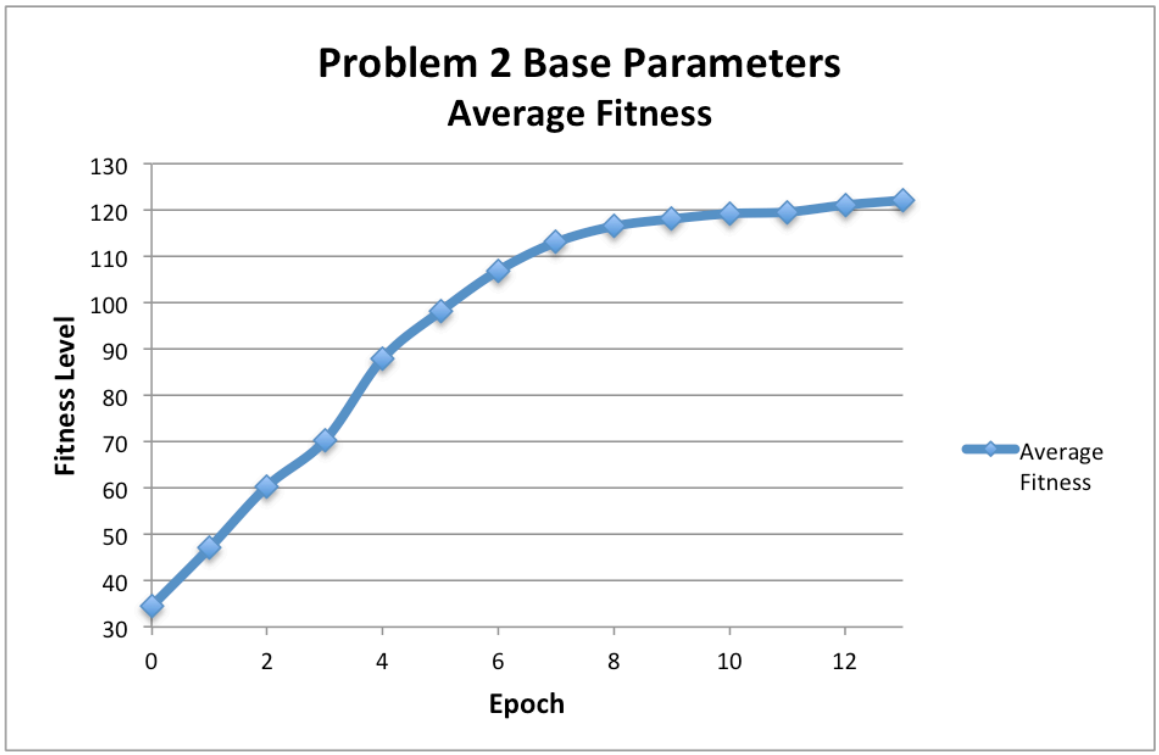
Problem 2 was investigated in exactly the same manner as Problem 1. Using the base parameter combinations described above, each system parameter was independently varied one at a time, while the other parameters were kept constant. Below is a chart that depicts the positional error of the x and y coordinates of the population of particles with respect to the global best particle at any given point in time.



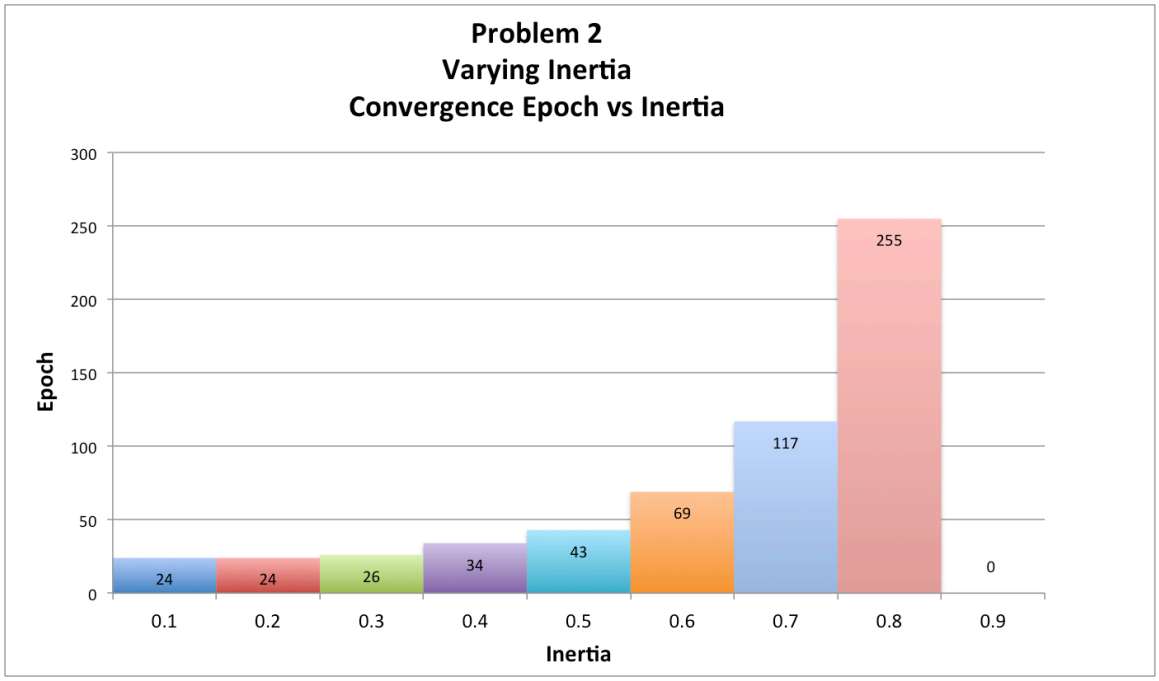
In the base test, the error drops relatively quickly, but not as quickly as the base parameters for Problem 1. However, rather than drop consistently, the error fluctuates back up for a short period of time. Although the error does not consistently decrease, the average particle distance to the global maximum does steadily decline until the system halts, as shown in the figure below.



Similarly, the average fitness (next figure) shows that the particles in this simulation continuously become more fit over time.

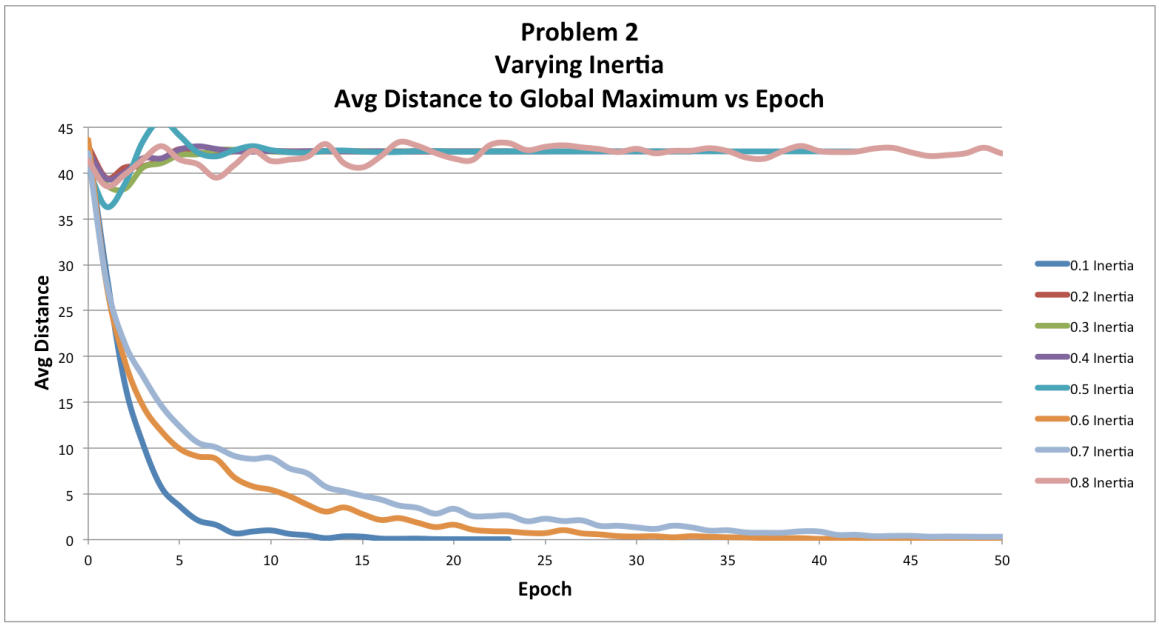


Like in Problem 1, the first parameter to be examined was the inertia, which was varied using values 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9. The following figure depicts the epoch at which these systems converged.



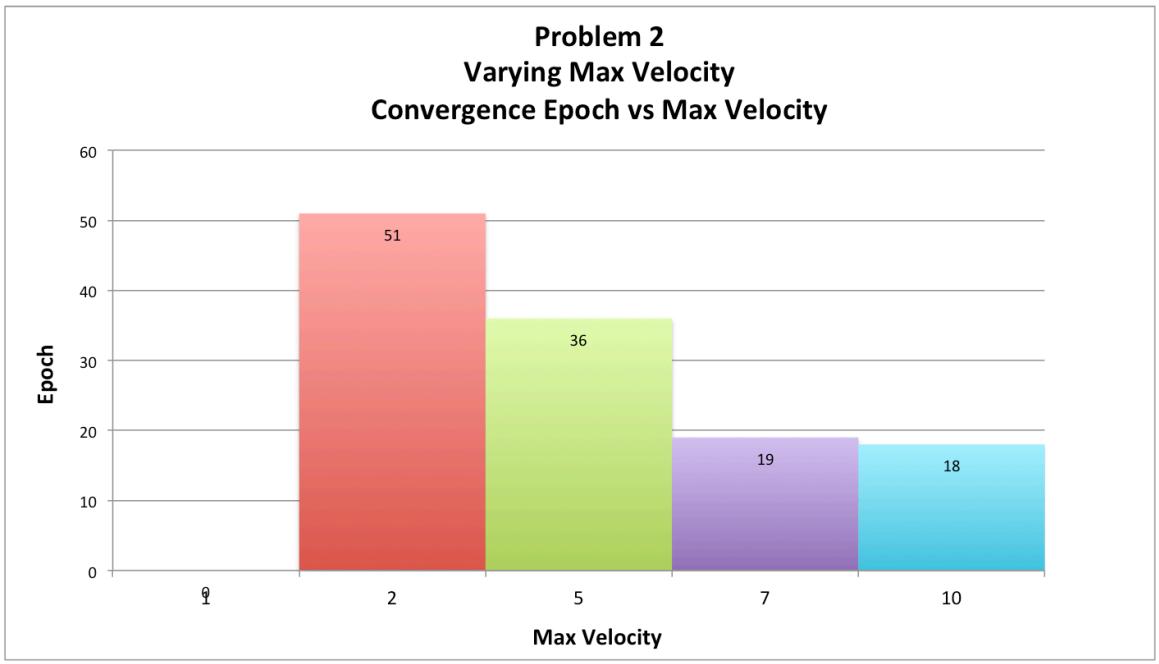
It would appear that low inertia values fare much better than higher values with regard to the convergence speed of a particular system. As the inertia increases, the particles take much longer to settle down to a particular location in the search-space. The simulation with an inertia value of 0.9 actually did not converge within the 1000-epoch limit.

Because Problem 2 incorporates a local maximum in addition to the global maximum position, convergence rates are not necessarily reflective of how quickly the particles converge near the global maximum. The halting condition within the simulation is based on the total error in the x and y coordinates of the particles in the population. However, this error is calculated with respect to each particle's distance from the current best particle's position. The particles may converge to the same location as one another, causing the simulation to halt, but this location may now be near the local maximum at (-20, -7), rather than at the global maximum of (20, 7). This phenomenon is shown in the following chart.

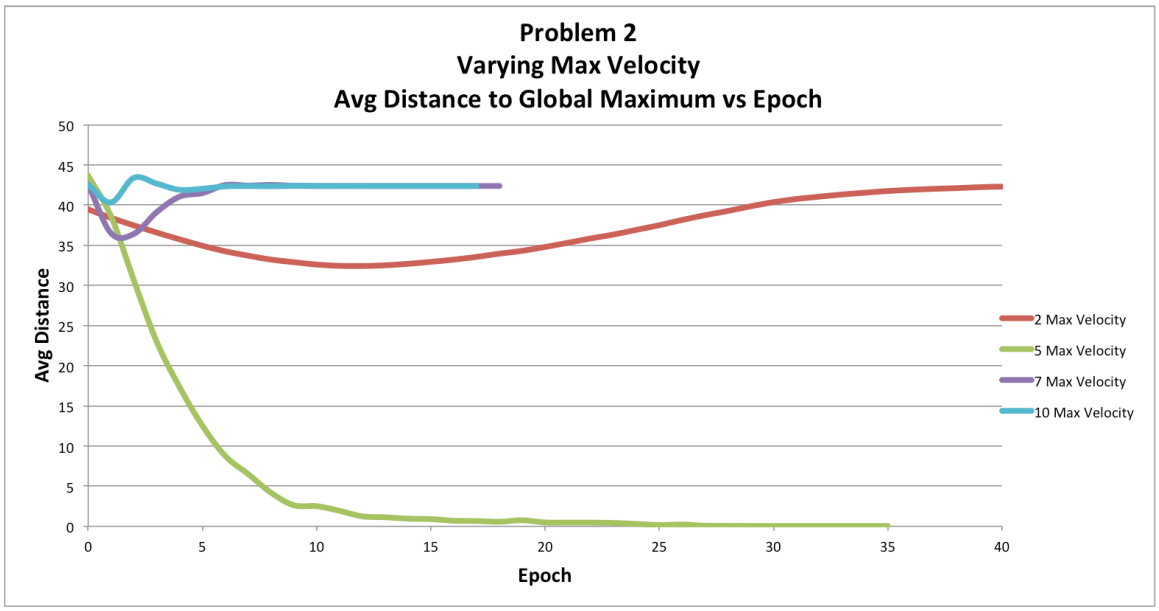


From the above graph, the simulations with inertia values of 0.7, 0.6, 0.1 converged toward the global maximum, while the other simulations converged toward the local maximum, which is why the average distance for those simulations remains very large, even though the simulation halted.

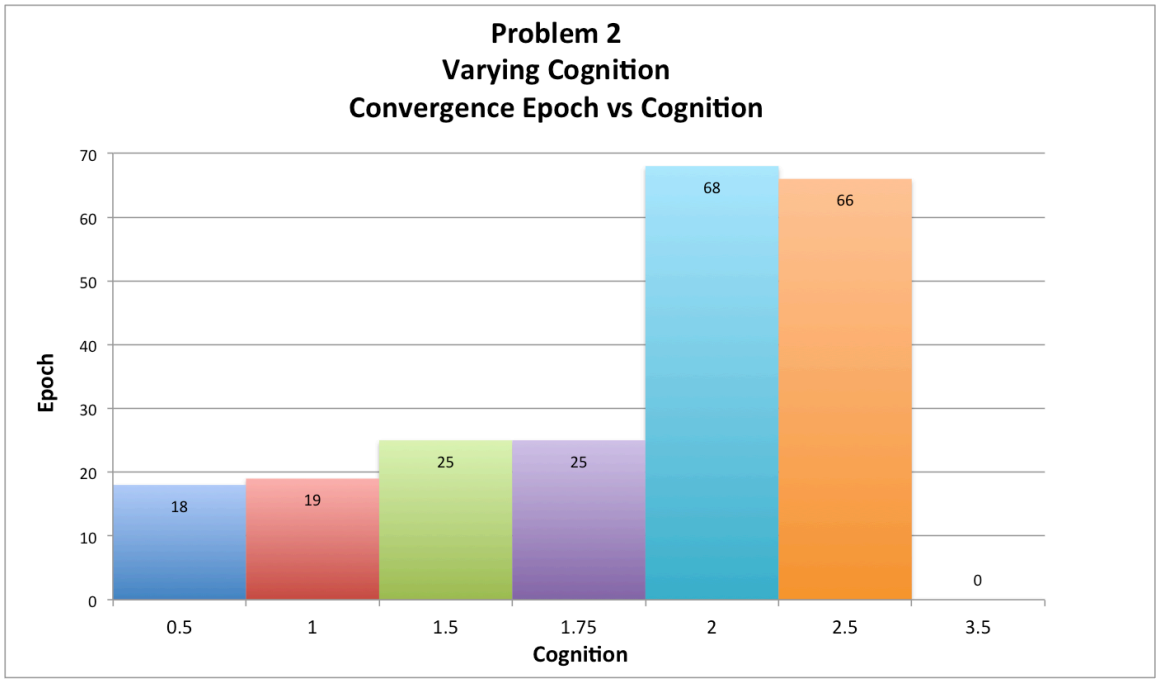
After studying the inertia value, I examined the maximum velocity using values of 1, 2, 5, 7, and 10. The convergence epochs are shown below.



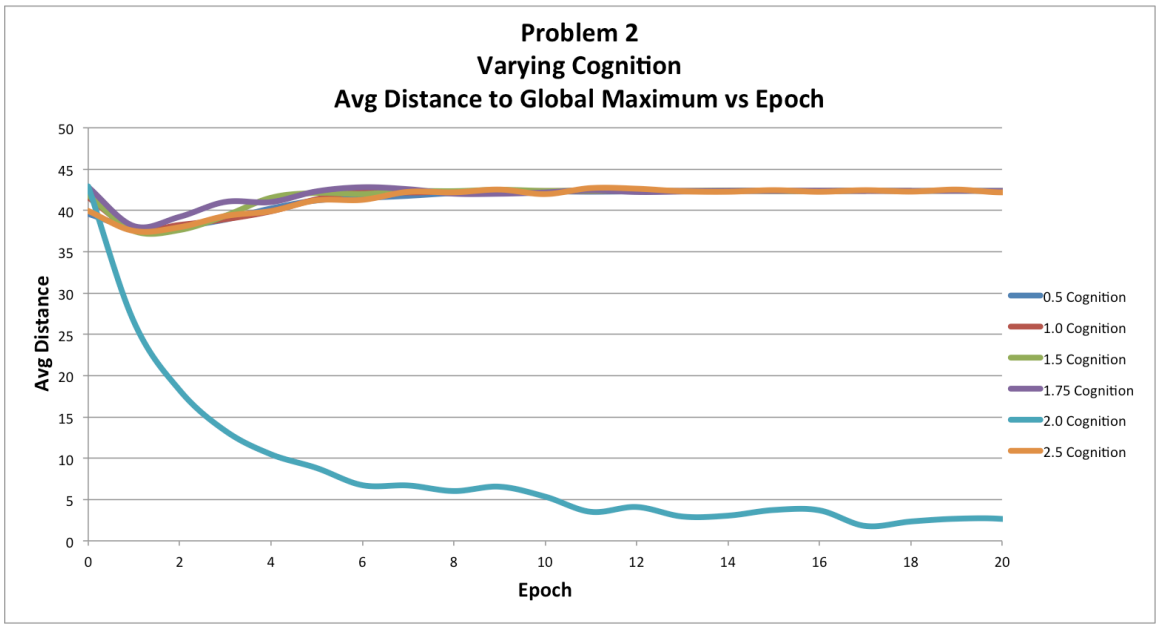
The previous figure shows that a lower maximum velocity caused the simulations to take much longer to converge. In fact, a maximum velocity of 1 did not converge within 1000 epochs. The average distance of each particle from the global maximum (below) shows that the simulation with a maximum velocity of 5 was the only one in this set to converge to the correct location.



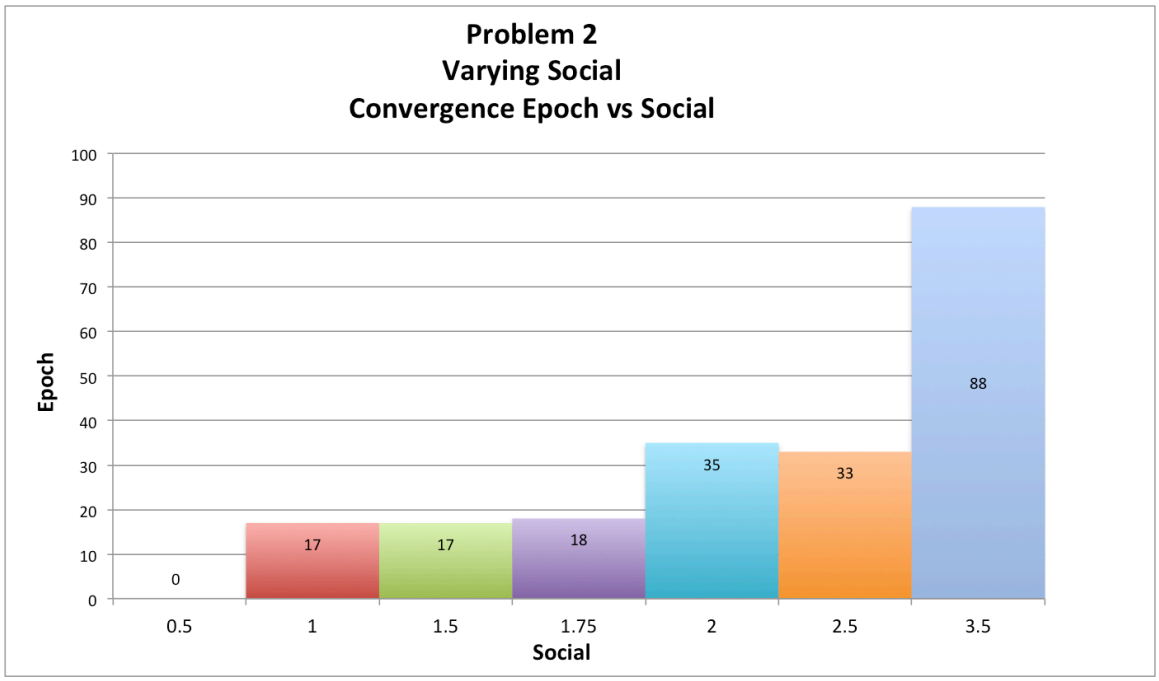
When examining the cognition parameter, values of 0.5, 1, 1.5, 1.75, 2, 2.5, and 3.5 were used. The convergence epochs for these simulations are depicted below.



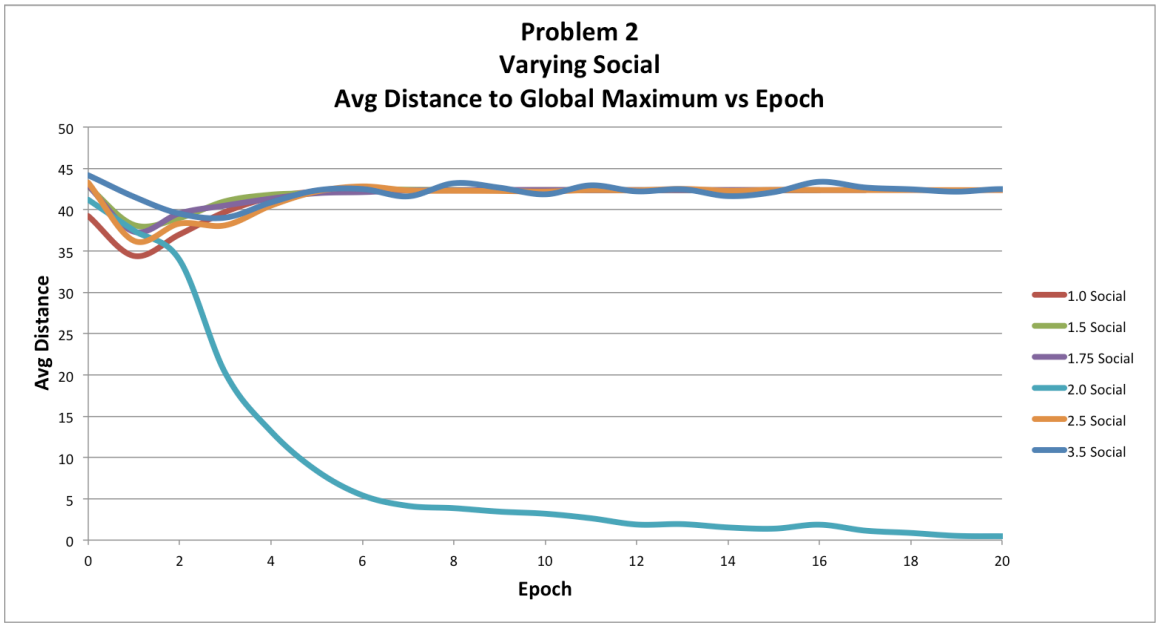
Lower cognition values converged most quickly, but values in the range of 0.5-1.75 seem to produce similar results for convergence speed. Cognition values of 2 and larger took drastically longer to converge. The 3.5-cognition simulation did not converge within 1000 epochs. Again, the average distance from the global maximum is graphed below. It would appear that the only simulation to converge to the correct location was the one with a cognition value of 2.0.



Like the cognition parameter, the social parameter was varied across the values 0.5, 1.0, 1.5, 1.75, 2.0, 2.5, and 3.5. The convergence rates for these parameter values followed a similar trend to those of the cognition values. Lower social values tended to converge more quickly, with the exception of the simulation using a social value of 0.5, which did not converge in 1000 epochs. Social values of 1, 1.5, and 1.75 performed almost equally well from an efficiency standpoint.



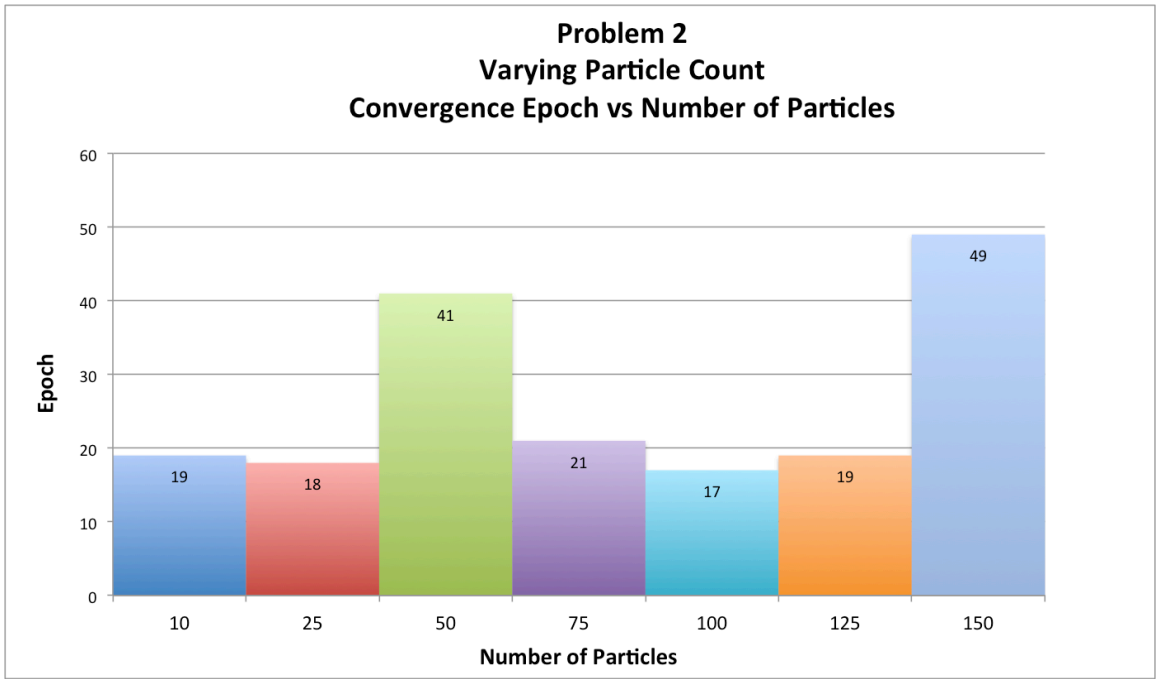
Below is a chart depicting the average distance of each particle to the global maximum. Interestingly, the only simulation to converge to the global maximum, rather than the local maximum, was the simulation using a social parameter value of 2.0, which was the exact same as the cognition set in which the 2.0-cognition simulation was the only one to converge to the global maximum.

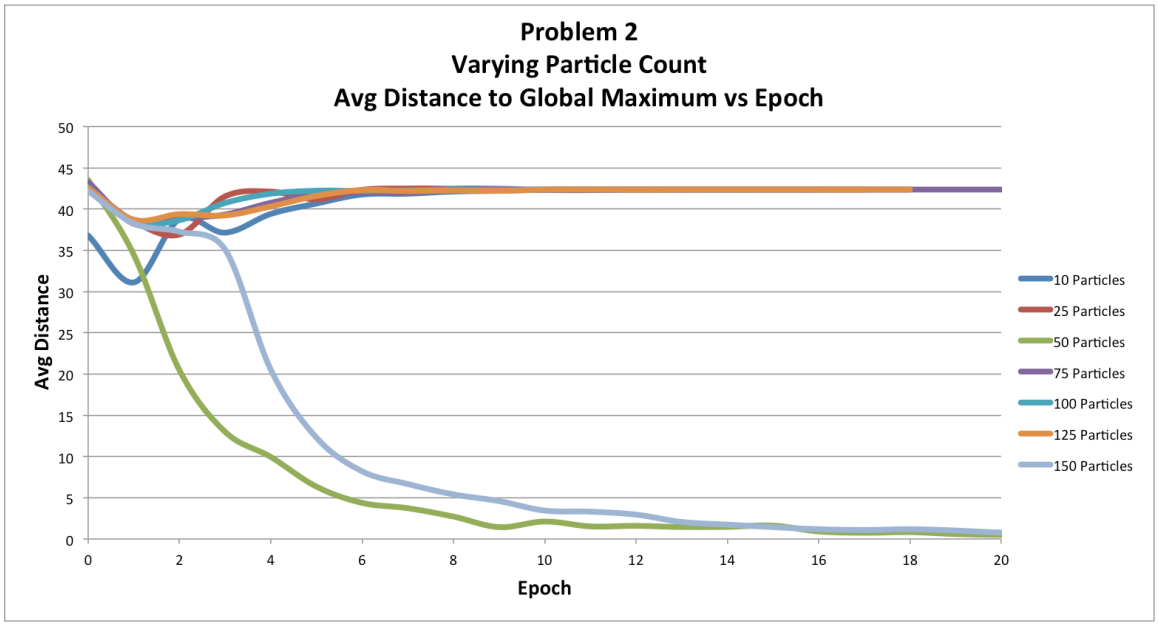


Perhaps social and cognition parameter values near 2.0 allow for the system to more easily avoid incorrect convergence to the local maximum, instead of the global

maximum. It may help balance the particles' personal and social influences when calculating velocities at each time step.

Following the social parameter, I varied the particle count with the values 10, 25, 50, 75, 100, 125, and 150. From these parameter combinations, most of the simulations converged in a comparable number of epochs (according to the figure below), with the exception of population sizes of 50 and 150, both of which took nearly twice as long to converge.





From this set of simulations, the average-distance chart above shows that the simulations using population sizes of 50 and 150 were the only two simulations to converge to the correct location. Although the other simulations converged more quickly, they actually went to the wrong location in the search-space. The 50-particle and 150-particle simulations likely took longer to converge because the particles had to overcome being trapped by the local maximum, rather than the global maximum.

Overall, it is more difficult to draw meaningful conclusions from the resulting data of the Problem 2 experiments. Most of the simulations seemed to converge to the local maximum, rather than the global maximum. Whether or not these systems converged to this incorrect location likely depended heavily on the initial conditions of the system. As a result, the effects of the actual parameters that were being varied was challenging to discern from the behavior caused by this local-maximum convergence.