Not Mr. Roger's Neighborhood: Bench-marking PSO Performance with Different Neighborhood Topologies

Dustin Hines, Duncan Gans, David Anderson

5 October 2018

Particle swarm optimization (PSO) is a valuable tool for approximating the solution to complex, multidimensional problems. PSO techniques are inspired by the movement of bird flocks as they seek the highest concentration of insects. This algorithm involves many different particles "flying" through a solution space along trajectories influenced both by the best solution they have found and the best solutions found by other particles in the swarm, their "neighbors". In this paper, we investigate the effect of four different neighborhood topologies on PSO performance by testing a standard PSO algorithm on three different functions. Specifically, we tested how effectively global, ring, von Neumann, and random topologies minimized Rosenbrock, Ackley, and Rastrigin benchmark functions. We found that different topologies were better suited to solving different problems. Specifically, random topologies performed best when optimizing Ackley, random topologies were best for Rosenbrock, and global topologies were best for Rastrigin.

1 Introduction

Particle swarm optimization (PSO) is a nature-inspired approach that can be effective at approximating optima of complex multi-dimensional functions. PSO was initially inspired by the flocking behavior of birds, though as with all nature inspired techniques there is not an exact analogy between the behavior observed in nature and the implementation of the algorithm. Generally speaking, a swarm of particles is created that "flies" through solution space searching for optimal solutions. Like birds communicating the locations of optimal foraging spots, the particles in the PSO algorithm remember the best solutions found both by themselves (the particle's personal best) and by neighboring particles in the swarm (the neighborhood best) to inform future exploration. One question that immediately arises from such a setup is how to determine neighborhoods; that is, how to decide which particles influence a given particle's position updates. The most obvious approach to neighborhood setup is to make every particle influence every other particle, otherwise known as a "global" topology. This approach has the advantage of being somewhat analogous to how birds may communicate information about optimal foraging locations. However, it is not clear if constructing neighborhoods globally is the most efficient neighborhood setup for solving analytical problems, especially on functions with many local optima. We considered three additional neighborhood topologies (ring, von Neumann, and random) and investigated their performance on three standard PSO benchmark functions to reveal how neighborhood topology impacts the performance of PSO.

In particular, we hypothesized that varying neighborhood topologies might help with the perennial problem of early convergence in multi-modal problems. A major weakness of PSO algorithms is the tendency to converge to sub-optimal solutions when there are many local optima. To evaluate the effectiveness of each topology we tested PSO algorithms implementing each of our test topologies against three standard benchmark functions in 30 dimensions: Rosenbrock, Ackley, and Rastrigin. For the global, ring, von Neumann, and random topologies, we used 16, 30, and 49 particles for 20 runs on each of the benchmark functions for 10,000 iterations of the algorithm. This is a total of 4x3x20x3, or 720 runs of the PSO algorithm throughout our testing. To ensure that we can further investigate the difference in performance of neighborhoods that converge to similar solutions after the 10,000 iterations, we measure deviation from the optimum solution every 1000 iterations to compare how quickly PSO approximates the optimum solution using each neighborhood.

We found that the effect of topology on PSO performance varied by the benchmark problem. For topologies that favored heavy exploration, random neighborhoods tended to perform the best (Ackley, Rastrigin) and global neighborhoods performed the best on Rosenbrock where quicker convergence was valuable.

First, in section 2, we will detail how PSO algorithms work. Then, in section 3, we will discuss the three benchmark functions that we use. Next in section 4, we will discuss our experimental methodology. In section 5, we will discuss our results. We discuss the opportunity for further work in section 6. Finally, in section 7, we conclude by briefly summarizing our results.

2 Particle Swarm Optimization (PSO)

2.1 Inspiration

Particle Swarm Optimization is an optimization and search tool tenuously modeled from flocks of birds searching for high concentrations of insects to eat. Although individual birds do not now know the locations with the highest concentrations of insects, they do know where the best location they have found on their own, as well as where other birds are finding higher qualities of insects. This leads to birds being biased towards both their personal experience and the experiences of the birds around them. With particle swarm optimization, the birds are replaced by particles in a solution space of variables, and the higher concentrations of insects are replaced by solutions with higher fitness. The particles explore the solution space, gradually moving towards the best solution they have found, as well as the best solution their neighborhood has found. As these particles fly through the solution space, they begin by largely exploring and then shifting towards exploitation.

2.2 Particle Initialization

The first step of PSO is the initialization of the particles. The first option within initialization is choosing a larger population size. Choosing a larger population will increase the odds that one of the randomly generated particles will be generated close to the optimum, and with more particles more of the solution space will be explored. The main drawback will be run time, with more particles meaning more function calculations, and velocity changes. After choosing the population size, the particles locations and velocities are then randomly generated in the bounds of the solution space. However, for our experiments, the bounds will not include the actual solution. This ensures that particles will have to explore to find the optimum, rather than just being randomly generated near it. However, when using a PSO algorithm, one would not intentionally generate the individuals away from where they think the optimal solution might be. After the particles are initialized, you iteratively go through the following steps, keeping track of the global best as you try to continually improve it.

2.3 Updating Velocity with Biases

The first step in the iterative process is updating velocities. This is where the bias towards the personal best and the neighborhood best comes into play. Each individual particle keeps track of its own personal best (*pbest*) and the best value found by particles in its neighborhood (*nbest*). The new velocity will be calculated by adding the current velocity vector to weighted velocity vectors pointing towards *pbest* and *nbest* respectively. The weight of these is determined by ϕ_1 and ϕ_2, where ϕ_1 denotes the bias towards the personal best and ϕ_2 denotes the bias towards the neighborhood best. By adding the current velocity, the personal best velocity, and the neighborhood best velocity, one gets a new velocity vector biased by the neighborhood and personal bests. In our implementation, we use the following "constriction" function to update particle velocities: $\vec{v}_{i+1} = \chi[\vec{v}_i + \vec{u}_1 \otimes (\vec{p}_i - \vec{x}_i) + \vec{u}_2 \otimes (\vec{n}_i - \vec{x}_i)]$ where $\vec{v}_i$ is the current velocity, $\vec{p}_i$ is the vector to the current personal best, $\vec{n}_i$ is the vector to the current neighborhood best, $\vec{u}_1$ is a vector with random values in the range [0, ϕ_1), and $\vec{u}_2$ is a vector with random values in the range [0, ϕ_2). χ is a constant known as the "constriction factor", which serves to balance local and global search. For our purposes, we set $\chi = 0.7298$, a common constant for constriction based PSO.

2.4 Updating and Evaluating the Function at the New Position, and Updating Bests

Once the velocity for each particle has been updated, new velocity vectors are added to each particles current location to calculate the particles' new locations. This amounts to the vector addition: $\vec{x}_{i+1} = \vec{x}_i + \vec{v}_{i+1}$ where $\vec{x}_i$ is location before updating, $\vec{x}_{i+1}$ is the new location, and $\vec{v}_{i+1}$ is the updated velocity at $\vec{x}_i$. The benchmark function is then evaluated at the new location to determine the optimality of each particle's new location. If the function evaluation deems the current location more fit (i.e. closer to an optima) than either the personal best or the neighborhood best, those bests are then updated to correspond with the particles current location.

2.5 Neighborhood Methods

Lots of PSO research concerns itself with what neighborhoods to use. Changing the neighborhood will have an impact on the extent of exploration vs exploitation, and the ability of the algorithm to avoid and escape local optimum.

2.5.1 Global

The first of these is the global neighborhood. In this case, each particle is biased towards its own personal best, and the global best. This means that every particles neighborhood best is biased towards the same location. This works well when the number of local optimum are minimal, and it is more important to converge quickly. Because there is only one best, the flock of particles will move as a single swarm, rather than a set of swarms.

2.5.2 Ring

The next neighborhood is the ring neighborhood. In this one, the particles are in a list, and its neighborhood are the $\mathrm{N_k}/2$ (Typically $\mathrm{N_k}$ is 2) particles on either side of the initial particle. This means that information about the global best is disseminated slowly, giving the particles and neighborhoods plenty of time to explore before being alerted to what the global best is. Higher $\mathrm{N_k}$ will lead to a quicker dissemination of the global best, and therefore converge quicker, whereas an $\mathrm{N_k}$ of two will be slow to converge.

2.5.3 Von Neumann

The von Neumann is like the ring method, but instead of the particles being arranged in a list, they are put into a grid. In this case, the number of neighbors is $\mathrm{N_k}$, with there being $\mathrm{N_k}/4$ neighbors above, below, to the right of, and to the left of the home particle. Von Neumann will allow exploration and for the global to be dispersed relatively slowly but will be dispersed quicker than the ring Method. Furthermore, because the neighborhood involves the particles above and below a given particle, it can jump in multiple directions, essentially letting the global best jump to very different locations in the solution space.

2.5.4 Random

Although there are several versions of random neighborhoods, we will focus on one. In this version, neighborhoods are of size k, and each particle will randomly choose the k-1 particles that are in its neighborhood. This neighborhood is regenerated with some probability (we will use .2) so that the particle has a new neighborhood of random particles. Since each particle generates their own neighborhood randomly at random times, it means that even though particle B is in the neighborhood of particle A, particle A might not be in the neighborhood of particle B. This will also do well at balancing convergence because the global best will only spread to a new neighborhood at the start after every five iterations (assuming a probability of .2), with the rate of spread increasing exponentially as the global best is passed on to more neighborhoods, which can then pass it on to even more neighborhoods. Furthermore, by decreasing the probability and the size of K, you can make the exploration last even longer before exploitation begins. This method does a great job of simulating annealing by starting out with lots of exploration, and moving towards more exploitation

3 Benchmark Problems

To assess the efficacy of differing PSO neighborhood topologies we tested how effectively implementations of each topology found global mimima for the Rosenbrock, Ackley, and Rastrigin benchmark functions. These functions are all non-convex n-dimensional functions commonly used as tests for optimization algorithms (Bratton and Kennedy 2007). Qualitative descriptions of each function are provided below.

3.1 Rosenbrock

The Generalized Rosenbrock function, also known as Rosenbrocks Valley, is a uni-modal function defined by the function $f = \sum_{i=1}^{D} 100(x_i + 1 - x_i^2)^2 + (x_i - 1)^2$ where D is the number of dimensions. When rendered in two dimensions, it resembles a valley with a hill at one end, resulting in the formation of a parabolic trough that runs across the valley floor (Fig. 1). This trough is quite easy to find, but has an extremely slight slope to it which makes convergence to the local minimum difficult. For PSO, Rosenbrock provides a good test of an implementations ability to find optima in situations where differences between values are very slight, making it difficult to determine the direction in which a swarm should move.

3.2 Ackley

The Ackley function is an n-dimensional function defined by the equation $f = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i) + 20 + e$. When rendered in two dimensions, the Ackley function roughly resembles egg-carton functions with a single global minimum in the center (Fig. 2). The problem is difficult to optimize for two reasons: first, its outer region is nearly flat in net elevation change, with large changes in elevation only

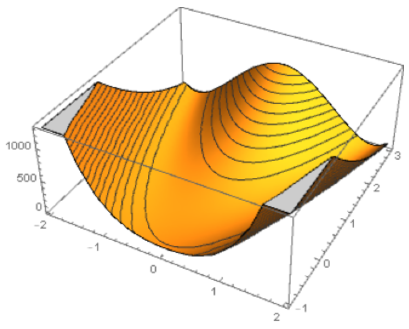

Figure 1: Three dimensional plot of a Rosenbrock function where D=2. Note the steep hill at the end of the valley and the parabola shaped trough at the base.

occurring in the immediate vicinity of the optima. This feature makes it difficult for PSO to determine which direction to move initially or when the swarm is in a diffuse state. Moreover, the outer landscape is covered in small hills and valleys, providing particles ample opportunities to get stuck on local optima. The combination of these two factors make premature convergence quite easy, though if a swarm is able to find its way into the vicinity of the global minimum convergence to the minimum can be quite rapid.

3.3 Rastrigin

The third test-function we used was a generalized Rastrigin function, defined in n dimensions as $f = \sum_{i=1}^{D}[x_i^2 - 10\cos(2\pi x_i) + 10]$ The Rastrigin macrostructure is slightly spheroid, with a global minima located at the origin. At a finer scale, Rastrigin is revealed to be highly multimodal, with local minima dispersed evenly across the space (Fig. 3). For a particle to reach the global minimum, it is likely that it will have to cross several of these local minima. This poses a serious challenge for PSO due to the high likelihood of particles getting stuck on one of these minima, leading to premature convergence.

3.4 Summary of Benchmarks

The three problems we used for testing all are standard tests of optimization algorithms that can be tested in n-dimensions. Rosenbrock tests the ability of a swarm to find optima when differences in optima are small, while Ackley and Rastrigin test a swarms ability to move across a landscape where there are many local optima and thus many opportunities for premature convergence. It is worth noting that while these are good tests of the ability of a particle swarm to find optima in strange multi-dimensional landscapes, their relevance to the swarms performance when optimizing applied problems may be minimal. For example, in a real world situation the minute differences between the valley floor and the global minimum in Rosenbrock is may be of little practical significance. Nevertheless, for the purposes of testing the efficacy of various swarm topologies these functions serve as a good metric of PSOs effectiveness when optimizing difficult problems.

4 Experimental Methodology

For our experiments we focused on the effects of each topology on performance when optimizing the benchmark functions described above. We ran tests for each function in 30 dimensions. Preliminary testing revealed that this number of dimensions was good for testing as it proved computationally challenging enough to reveal subtle differences in performance between topologies. To test for optimality, we performed runs for each topology on each problem and calculated the median and mean solution found each 1000 iterations. We then compared how close these values were to the true optimum in each problem (0.0) to evaluate how accurate the solutions produced by each topology were. By plotting solutions found by each topology as a time-series, we also compared the speed at which each topology converged to an optimum.

Of course, varying other algorithm parameters can also effect performance, and these effects can be synergistic. For instance, given one swarm size, a certain topology might perform better than another whereas with a larger swarm size the two topologies might perform more equivocally. To address potential problems involving swarm size we decided to run tests with swarm sizes of 16, 30 and 49 particles. Given these swarm sizes, our von Neumann array dimensions were rounded to 4x4, 5x6, and 7x7 respectively.

When running preliminary tests, we also quickly determined that we had to specify different ranges for *vmax* based on the problem we were optimizing. To

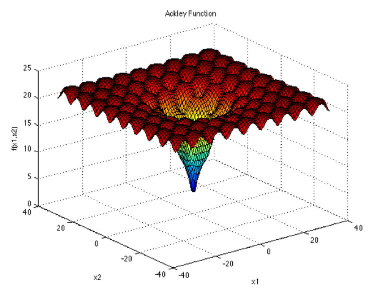

Figure 2: The Ackley function in two dimensions, courtesy of S. Surjanovic and D. Bingham, Simon Fraser University. Note the releatively flat, hilly exterior and the centralized minimum.

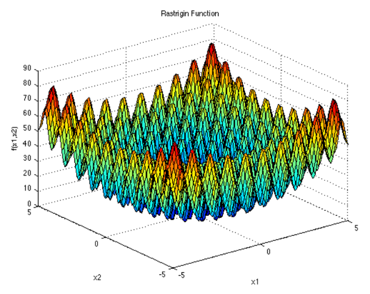

Figure 3: Rastrigin function with dimension 2. Note overall bowl shape punctuated by regular local minima that a particle must traverse to find the global minima. Courtesy of S. Surjanovic and D. Bingham, Simon Fraser University.

address this second problem, we assigned *vmax* to be 5.12 when optimizing Rastrigin, 32.768 when optimizing Ackley, and 2.048 when optimizing Rosenbrock based on qualitative observation of swarm performance and input from experts in the field.

For each combination of parameters (topology x swarm size x benchmark evaluated) we performed 20 runs of 10,000 iterations each, keeping track of the best solution found every 1000 runs. For all random topology tests, neighborhood size was set to 5.

5 Results

5.1 Ackley

For the Ackley benchmark, the random topology was able to reach the true minima, while the ring, von Neumann, and global toplogies all got stuck in local minimas around 20. Between von Neumann, ring, and global, global was able to achieve lower values, and sink deeper into the local minimum, while von Neumann got the next lowest, followed by the ring toplogy. As far as sizes, for the global, von Neumann, and ring topologies, a size of 30 performed the best, followed by 49, and then 16 performing the worst. With the random topology, larger sizes lead to slower convergence but slightly lower minimas, while smaller sizes lead to quicker convergence, but didn't converge quite as low. The speed of convergence was comparable between the topologies but to different minimas. With random, it quickly reached below zero, but continued optimizing beyond that point, whereas with global, ring, and Von Neuman, it quickly approached about 20, and then further optimized to the true local minima.

5.2 Rastrigin

The 30 dimensional Rastrigin benchmark was difficult for the PSO algorithm to solve regardless of the neighborhood topology. As shown in figure 6, von Neumann performed the worst overall, with slightly better optima after the 10,000 iterations as the swarm size increased. The global topology did not achieve more optimal values with more PSO iterations, which suggests that the global topology converged upon a solution within the first 1000 iterations for each population size. This is consistent with the observed behavior of the global topology. The ring topology achieved solutions comparable to the global topology, but continued to achieve better results though additional increments of 1000 iterations. The random

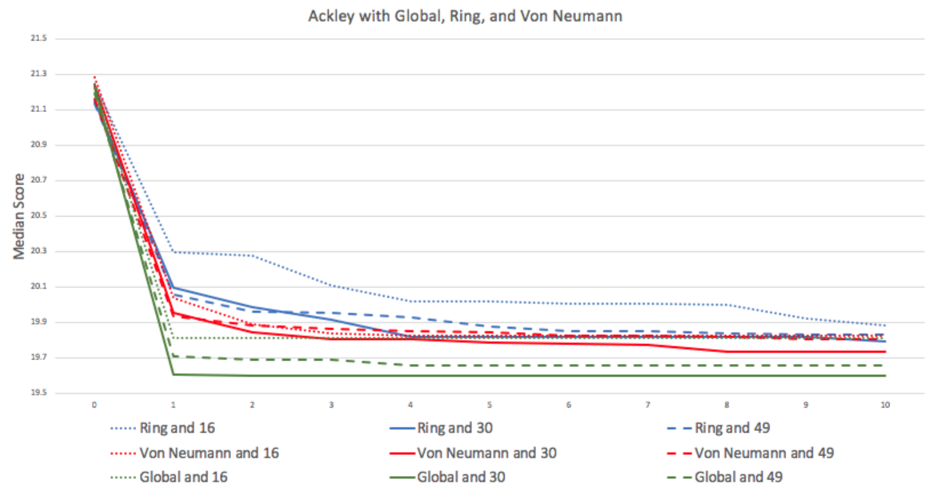

Figure 4: Ackley function performance with 30 dimensions, 1000 iterations, and various topologies and swarm sizes. As can be seen, ring, von Neumann, and global all converged at local minima for the median run.

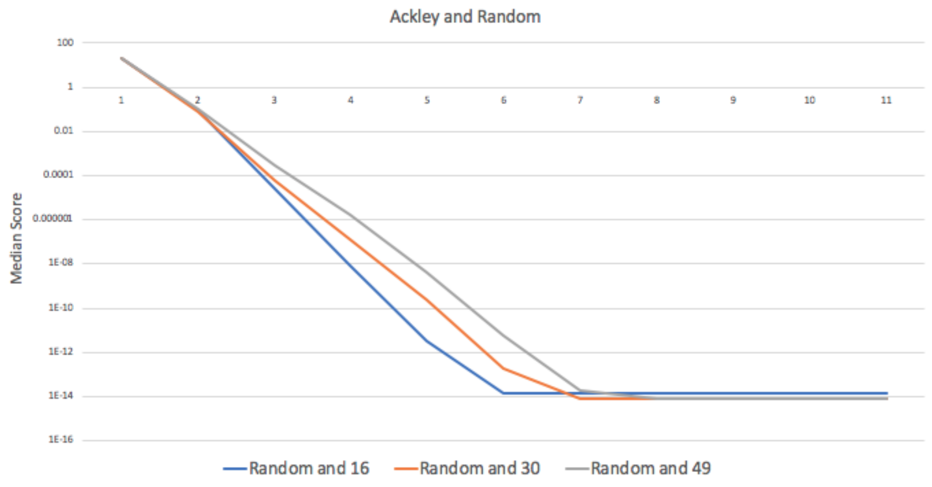

Figure 5: Ackley function performance with 30 dimensions, 1000 iterations, random topology and three swarm sizes. The random topology was able to continue approaching 0 until E-14.

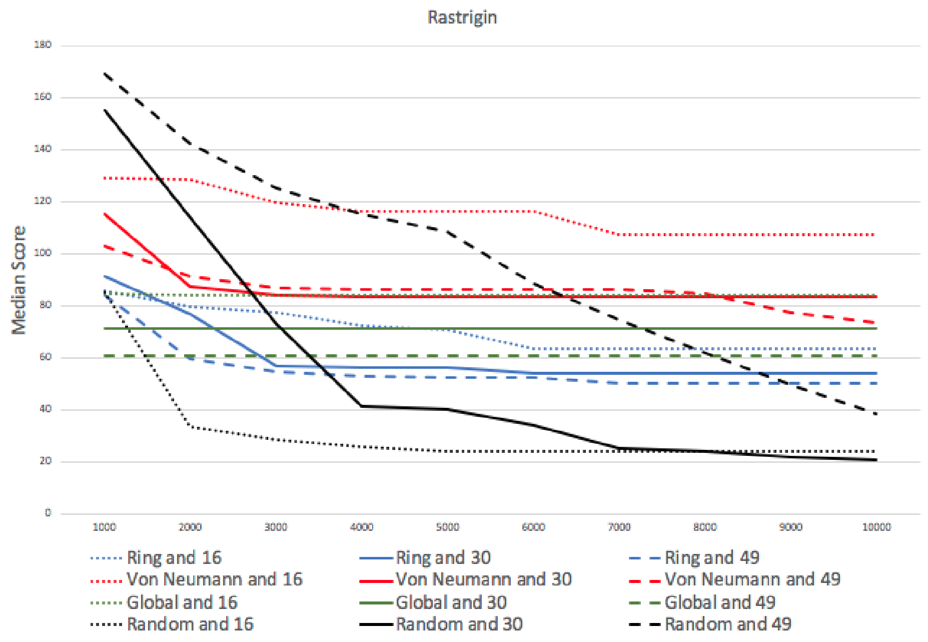

Figure 6: PSO performance on the Rastrigin benchmark using the global, ring, von Neumann, and random topologies measured by the median value at each interval of 1000 iterations across the 20 tests for each parameter combination.

topology had the most interesting behavior. The random topology achieved the most extreme increase in solution quality with a swarm size of 16. The swarm size of 49 with the random topology converged more quickly than with a swarm size of 30, but the swarm size of 30 resulted in a better median function value after 10,000 iterations. Overall, the global topology performed the best after 1000 iterations with the random topology performing the best after the full 10,000. Of all the combinations of swarm size and topology, a swarm size of 49 with the random topology found comparatively "good" solutions with the fewest iterations of the algorithm.

5.3 Rosenbrock

For the Rosenbrock benchmark, the global topology was able to converge faster and for a longer duration towards the minimum, while the speed of convergence of the other three topologies was slow and plateaued. As shown in figure 7, between von Neumann, ring, and random, von Neumann had the largest variance in performance based on size, but overall, ring performed better than von Neumann, which performed better than random. For the global, von Neumann, and ring topologies, swarm sizes of 16 performed worse, but there wasn't notable variation between 30 and 49. With the random topology, the size of 16 performed best with the sizes of 30 and 49 performing similarly. In terms of convergence rates, global continued converging and reaching lower values, while the other three converged quickly at the start, and then their convergence began to plateau.

6 Discussion

6.1 Ackley

Random topologies proved to be far and away the best for optimization of the Ackley function. This makes sense, given that information transfer in random topologies is relatively slow due to constant mixing. This allows the function to explore more and avoid early convergence. Given that the main difficulty of Ackley is converging prematurely in a local minima, the slow convergence and constant mixing of random topologies allowed the to sidestep the problem of early convergence and thus find solutions near the true minimum.

Size did appear to have an impact on random topologies when optimizing Ackley. When size is smaller, the topology is more similar to global, because information is spread between the particles at a faster rate than with more particles, especially when the neighborhood size is fixed. This means that lower sizes will converge more quickly, but potentially not find the best solutions. With the non-random topologies, global generally performed best, followed by von Neumann, and then ring. This behavior is to be expected: in fixed topologies like these, convergence to local minima in a multimodal problem is almost inevitable. However, with a relatively large neighborhood size there are more particles to pull trapped particles out of local minima, thus slowing rates of premature convergence. As neighborhood size shrinks (i.e. as we transition from global towards ring-sized neighborhoods) fewer and fewer neighbors are outside of a given local optima, leading to entire neighborhoods being trapped at a local optimum and ultimately leading to early convergence.

It is worth noting that the typical results presented here do obscure some of the variation found in our data. For instance, the von Neumann and ring topologies both got to the true minimum once, but this is hidden in the median values. These data should not be misconstrued as suggesting that non-random topologies *can't* find true minima, but rather that in multimodal problems they tend to converge to local optima.

The effects of swarm size on performance were more complex. A swarm size of 30 performed best, followed by 49, and then 16. It is unclear why moderate swarm sizes performed best, but it may be an artifact of sample size.

Overall, we found that Ackley favored algorithms that were able to explore more rather than converge to exploit minima found early on. Of the topologies tested, the random topology was the most exploratory and indeed produced the best and most efficient results.

6.2 Rastrigin

Rastrigin proved to be a difficult benchmark in 30 dimensions. As shown in figure 6, none of the median values approached the true minima (0.0) for any swarm size/neighborhood combinations. The random topology produced the best results after 10,000 iterations, which is consistent with the idea that the random topology is intended to heavily explore the solution space. What's more, with a swarm size of 49, the random topology was able to quickly achieve relatively better results than the other topologies. Thus, from our results, it appears that the random topology with a swarm size of 49 is the most competitive swarm size and topology combination to optimize the Rastrigin benchmark.

It is worth noting that random topology with a swarm size of 16 continued to achieve better values throughout the 10,000 iterations and could potentially achieve more optimal results if given more time. Further testing is necessary to investigate this. For the other topologies, von Neumann and ring did not have very notable performance characteristics. Ring consistently performed better than von Neumann, but both performed worse than random at 10,000 iterations and worse than global at 1000 iterations. For time sensitive problems where solutions need to be calculated in fewer than 1000 iterations, a swarm size of 49 with the global topology would achieve the best results with these constrictions. Overall, the Rastrigin function is a challenging multimodal benchmark problem that demonstrates the ability of the random topology to explore a difficult solution space with many local minima.

6.3 Rosenbrock

Relative to Ackley and Rastrigin, Rosenbrock proved to be a relatively easy function to optimize for all neighborhood topologies. The global topology performed best, with median solutions producing minima smaller than one for all tested swarm sizes after the full 10,000 iterations. The global topology also decreased quickly: with a swarm size of 16, global consistently beat out each other topology after just 7000 generations, while with a swarm size of 30 global consistently beat out other topologies after just 4000 generations. With a large population size, global was more optimal than other topologies at every run time.

With a larger swarm size, von Neumann topologies also found relatively optimal solutions reasonably quickly. Notably, von Neumann topologies appeared to be extremely sensitive to swarm size: with a small swarm size, von Neumann consistently performed the worst of all topologies.

In contrast, ring topologies were the least sensitive to changes in swarm size. Ring topologies were consistently able to find minima in the 10s after 10,000 iterations, though they decreased more slowly than global topologies and only found more optimal solutions when both swarm size and iterations were severely limited.

Random topologies appeared to be the worst for optimizing Rosenbrock. Though they did decrease over long run times, particularly with larger swarm sizes, the rate of decrease was observed to be small. More importantly, optima found by random topologies were almost never as good as those found by other topologies, such as global or ring.

Overall, the Rosenbrock function serves as a test of how well different topologies optimize a relatively simple, unimodal solution space where differences between solution fitness is slight. In such a topology, more information is always better; almost all particles will immediately converge to the valley and then face the challenge of moving down the very shallow parabola at the base to the minimum. With more particles exploring this straightforward valley, information about "which way is down" and which direction to go will be transmitted more readily with large amounts of communication and cooperation between particles. This idea corresponds with our results, where global topologies converged to the most

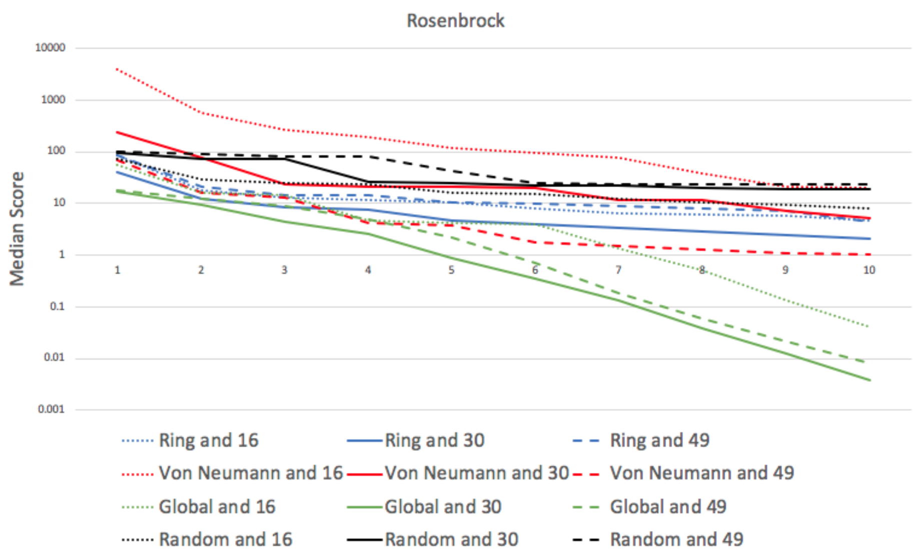

Figure 7: PSO performance on the Rosenbrock benchmark using the four specified topologies. Performance is measured as the median value found every 1000 iterations for twenty 10,000 iteration tests.

optimal solutions at the fastest rate, whereas random topologies performed relatively poorly. Thus, for subtle unimodal problems, simple global topologies appear to work quite well.

7 Further Work

This project presents many interesting opportunities for future work. First, random neighborhood size and swarm size could be allowed to co-vary. Since random neighborhood size is not currently increased with population size, it is possible that increasing population size has little effect on randomized topologies since small random neighborhoods are not able to take advantage of the relatively greater amount of information available from a larger swarm. It is possible that there is an ideal ratio of random neighborhood size to swarm size that should be maintained as swarm size increases rather than maintaining a constant neighborhood size. On a related note, it could likely be effective to dynamically change neighborhood sizes throughout the course of the algorithm. This would help to balance exploration and exploitation throughout the iterations.

In keeping with the theme of dynamism, it would be interesting to explore the effect of “shaking” up the particles after they have converged. One simple way this could be achieved by setting particle velocity values to a much higher value if they fall below a certain threshold, essentially implementing a problem-specific *vmin*. This would cause the particles to fly away from positions they have converged to. The neighborhood best and personal best values would be preserved, so as the particle made it’s way back to these locations in the solution space, it might find better solutions.

Efficacy of different topologies may vary in alternative PSO approaches. A commonly used variant of PSO is ”Inertia PSO”, where the current velocity of a particle is given it’s own unique coefficient in velocity updates rather than multiplying the entire update vector by a constriction factor. It is likely that topologies would perform similarly in inertia PSO as in the variant implemented here because mathematically constriction PSO is really a special case of inertia PSO. However, this is not necessarily the case, and particularly in more far-flung PSO variants (e.g. quantum PSO) certain topologies may perform better.

8 Conclusion

Particle swarm optimization proved to be a reasonably effective algorithm for optimizing many-dimensional complex functions. It’s versatility and intuitive implementation make it a nature-inspired algorithm of choice for many problems. However, the effects of varying neighborhood topologies remains unclear.

Here, we tested the efficacy of four different topologies when optimizing three standard PSO benchmark problems. Overall, we found that different topologies perform better on different types of problems. In particular, topologies that encourage exploration over exploitation, such as randomized topologies, perform well on multimodal problems where premature convergence is a serious issue. In contrast, topologies favoring exploitation and communication, such as globally linked topologies, perform best on unimodal problems where convergence is often slow (e.g. due to tiny differences in fitness between solutions). Additionally, changes to topology or problem type respond differently to increasing or decreasing population size, though these differences can be more nuanced and complex.

These differences *matter*. In many cases, selection of the right topology for the problem was the difference between finding a true minimum and finding a solution that was several times worse. Unfortunately, there is no mechanical way to select the best topology for a given problem. While we can come up with various rules to *suggest* what might be a good topology (i.e. favor exploration for multimodal problems), it is likely that every problem has a unique best topology that can only be found via testing. Thus, for the time being it is our task as computer scientists to continue to test various approaches to novel problems and document which ones work, which don't, and why. Here, we provide a step in that direction. There are many more ahead.

Swarm Size	Topol-ogy	Bench-mark	1000	2000	3000	4000	5000	6000	7000	8000	9000	10k
16	global	ack	19.81	19.81	19.81	19.81	19.81	19.81	19.81	19.81	19.81	19.8
16	global	ras	84.57	84.57	84.57	84.57	84.57	84.57	84.57	84.57	84.57	84.57
16	global	rok	6xE8	54.66	16.12	14.24	4.609	4.177	4.023	1.378	0.511	0.041
16	ring	ack	20.30	20.28	20.11	20.02	20.02	20.01	20.00	19.99	19.92	19.88
16	ring	ras	85.93	79.70	77.61	72.55	70.91	63.68	63.68	63.68	63.68	63.68
16	ring	rok	86.63	17.66	12.79	11.71	10.37	8.175	6.621	6.280	5.671	4.577
16	VN	ack	20.04	19.89	19.84	19.82	19.82	19.82	19.82	19.82	19.82	19.82
16	VN	ras	128.9	128.8	119.9	116.4	116.4	116.4	107.5	107.5	107.5	107.5
16	VN	rok	3985	579.6	259.4	187.9	121.7	93.98	75.95	38.82	21.22	20.46
16	rand	ack	0.091	2e-5	9e-9	e-12	e-14	e-14	e-14	e-14	e-14	e-14
16	rand	ras	85.18	33.83	28.85	25.87	24.37	23.88	23.88	23.88	23.88	23.88
16	rand	rok	71.27	28.97	24.60	23.47	16.42	15.03	12.30	10.67	9.327	8.049
30	global	ack	19.61	19.60	19.60	19.60	19.60	19.60	19.60	19.60	19.60	19.60
30	global	ras	71.64	71.64	71.64	71.64	71.64	71.64	71.64	71.64	71.64	71.64
30	global	rok	16.85	9.230	4.377	2.544	0.864	0.349	0.134	0.040	0.012	0.004
30	ring	ack	20.10	19.99	19.92	19.82	19.82	19.82	19.82	19.82	19.82	19.79
30	ring	ras	91.69	76.85	57.09	56.63	56.63	53.96	53.96	53.96	53.96	53.96
30	ring	rok	39.74	12.25	8.426	7.764	4.601	3.972	3.340	2.816	2.424	2.086
30	VN	ack	19.96	19.85	19.81	19.81	19.79	19.78	19.77	19.73	19.73	19.73
30	VN	ras	115.3	87.56	84.05	84.58	83.58	83.48	83.48	84.48	83.48	83.48
30	VN	rok	235.5	76.47	22.99	21.71	21.35	20.33	11.92	11.68	7.206	5.167
30	rand	ack	0.077	e-5	e-7	e-10	e-13	e-15	e-15	e-15	e-15	e-15
30	rand	ras	155.15	114.1	72.89	41.61	40.31	34.03	24.99	23.88	21.88	20.89
30	rand	rok	95.23	72.55	71.25	25.80	24.57	22.55	22.11	20.08	19.38	18.54
49	global	ack	19.71	19.69	19.69	19.66	19.66	19.66	19.66	19.66	19.66	19.66
49	global	ras	60.69	60.69	60.69	60.69	60.69	60.69	60.69	60.69	60.69	60.69
49	global	rok	17.89	12.42	9.110	4.945	2.253	0.702	0.181	0.058	0.021	0.008
49	ring	ack	20.06	19.96	19.96	19.93	19.88	19.85	19.85	19.84	19.83	19.83
49	ring	ras	84.28	59.80	54.72	52.78	52.27	52.27	50.51	50.51	50.51	50.44
49	ring	rok	85.63	20.97	14.42	14.20	10.44	9.793	8.967	8.059	7.067	4.433
49	VN	ack	19.93	19.88	19.86	19.85	19.84	19.83	19.82	19.82	19.81	19.81
49	VN	ras	103.1	91.54	87.16	86.56	86.56	86.56	86.56	84.60	77.61	73.63
49	VN	rok	68.77	16.28	12.92	4.125	3.854	1.757	1.540	1.295	1.084	1.008
49	rand	ack	0.101	e-4	e-6	e-9	e-12	e-14	e-15	e-15	e-15	e-15
49	rand	ras	168.9	142.6	125.1	155.5	108.6	88.56	74.88	61.71	49.50	38.49
49	rand	rok	99.13	88.31	81.11	79.54	41.65	24.20	23.95	23.59	23.45	23.24

Table 1: PSO Performance with the 3 Topologies on the 3 Benchmark functions at intervals of 1000 iterations up to 10,000 iterations measured by median function values over 20 tests rounded to 4 digits.