PSO and Multi-Funnel Landscapes: How cooperation might limit exploration

Andrew M. Sutton Darrell Whitley Monte Lunacek Adele Howe
Department of Computer Science
Colorado State University
Fort Collins, CO 80523
{sutton,whitley,lunacek,howe}@cs.colostate.edu

ABSTRACT

Particle Swarm Optimization (PSO) is a population-based optimization method in which search points employ a cooperative strategy to move toward one another. In this paper we show that PSO appears to work well on "single-funnel" optimization functions. On more complex optimization problems, PSO tends to converge too quickly and then fail to make further progress. We contend that most benchmarks for PSO have classically been demonstrated on single-funnel functions. However, in practice, optimization tasks are more complex and possess higher problem dimensionality. We present empirical results that support our conjecture that PSO performs well on single-funnel functions but tends to stagnate on more complicated landscapes.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search; G.1.6 [**Numerical Analysis**]: Optimization—*Global Optimization*

General Terms

Performance

Keywords

Optimization, Swarm Intelligence, Evolution Strategies

1. INTRODUCTION

Particle Swarm Optimization (PSO) is a population-based optimization method where search points employ a cooperative strategy to move toward one another. There have been few theoretical studies of PSO, and the theory that does exist suggests that PSO appears to be estimating a simple gradient based on local sampling.

PSO has primarily been tested on functions that we can describe as having a "single-funnel" topology. Intuitively, a "funnel" is a global trend on a landscape that consists of a basin of clustered local optima. A single-funnel function may be multimodal, but has a distinct overall bowl-like structure. A search method with a reasonable probability of moving between adjacent local optima should be able to locate the global optimum of a single-funnel function. On the other hand, a function that contains more than one funnel can pose a problem to strategies that rely on local information by deceptively leading the search into a suboptimal basin.

PSO has shown promising results on single-funnel benchmarks. In practice however, many real world optimization tasks are more complex. Real problems often possess higher dimensionality than is commonly used in benchmarks and usually little is known of the search space structure or the location of the global solution.

In this paper we present and test the following three hypotheses:

1. **On multi-funnel functions, PSO will tend to converge to the funnel that contains the majority of the swarm at initialization.** PSO's dependency on cooperative search suggests an inherent bias toward exploration within the span of the swarm.
2. **PSO may exhibit more of a tendency to stagnate on multi-funnel functions than an algorithm that uses more explorative informed mutation, or even uninformed mutation.** PSO moves through the search space using a position update rule that Shi and Eberhart have called "mutation with a conscience" [19, 20]. That is, mutation uses information about the locations of other population members to adapt direction and step size. However, this position update rule creates a tendency to move toward points that have already been discovered.
3. **PSO might struggle with real-world functions that possess landscapes of greater complexity than those of single-funnel benchmarks.** This is suggested by the previous two hypotheses.

We will test the first hypothesis by sectioning a known multi-funnel function into funnel regions and examining region convergence controlling for region initialization. We will test the second and third hypotheses by comparing PSO's performance with two algorithms. The first uses informed mutation and the second uses uninformed mutation but employs recombination. We test each algorithm on a suite consisting of single-funnel, multi-funnel, and real world problems.

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.

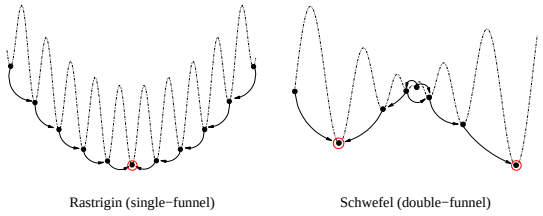

Figure 1: One dimensional function graphs.

The remainder of this paper is organized as follows. We present a more formal definition of funnels. We give a brief background of the PSO algorithm and discuss some issues with benchmarking and behavior. We introduce an algorithm that, like PSO, updates its motion through the search space intelligently. We then discuss a suite of test functions and real world problems on which we test our hypotheses. Finally we present empirical results.

2. FUNNELS

Often, real-world problems possess idiosyncratic features that can present challenges to traditional optimization strategies. For instance, Wales [21] suggests that many optimization problems in computational biology are difficult because there are multiple local optima that form distinct, spatially separate clusters in the search space. Assuming a search algorithm has the ability to move between local optima, the number of local optima may not be important [17]. Instead, the more difficult problems will be those with spatially distinct clusters of local optima. If the best local optima are all clustered together, a function is said to have one funnel. If there are spatially distinct clusters of local optima, the function is said to have multiple funnels.

There is no single precise definition of a funnel. Doye *et al.* suggest that a funnel "consists of a collection of local minima such that all monotonically descending sequences of successively adjacent local minima entering the funnel terminate at the funnel bottom" [4]. In a later work, Doye discusses funnels in the context of the atomic cluster with the smallest potential energy for a given set of molecules; a funnel is characterized by "a set of downhill pathways that converge on a single low-energy structure or set of closely related low-energy structures" [5].

Locatelli *et al.* offer a slightly different view of funnels [14]. We construct a graph G where each vertex represents a local optimum in the search space. Let V represent the set of local optima and $v_i \in V$ denote a particular local optimum. Let the function $d(v_i, v_j)$ denote the distance between points v_i and v_j in the space and the function $f(v_i)$ denote the fitness of point v_i. There is a directed edge $e(v_i, v_j)$ in G for all $v_i, v_j \in V$ such that $d(v_i, v_j) < r$ and $f(v_i) > f(v_j)$, where r is a radius of influence.

A funnel bottom point is a node with no outgoing directed edge. An example of this graph for the one dimensional Rastrigin and Schwefel functions is given in Figure 1. Locatelli's method requires locating all the relevant local optima in the search space and is dependent on choice of r.

3. PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization was developed by Kennedy and Eberhart [13] and was inspired by the social behavior of artificial life programs. In PSO, a population of search points called *particles* "fly" across the surface of the fitness function. Information about promising regions of the function is shared over social channels, and particles update their velocities in such a way to direct their motion toward other particles in fitter regions and toward previously discovered points.

On an n dimensional fitness function, each particle is defined by two vectors in $\mathbb{R}^n$: its position $\vec{x}$ and velocity $\vec{v}$. During each iteration of the algorithm, the following state transition rule is applied:

$$\vec{v}_t = \chi\left(\vec{v}_{t-1} + r_1\phi_1\left(P_{bst} - \vec{x}_{t-1}\right) + r_2\phi_2\left(N_{bst} - \vec{x}_{t-1}\right)\right)$$
$$\vec{x}_t = \vec{x}_{t-1} + \vec{v}_t$$

Here, χ is an inertial coefficient, and ϕ_1 and ϕ_2 are acceleration constants. The variables r_1 and r_2 are drawn from a random uniform distribution on $[0, 1]$.

This state transition causes particles to update their velocities to effect movement toward their "remembered best position" (P_{bst}) and toward the fittest particle in their *sociometric* neighborhood (N_{bst}). The acceleration constants ϕ_1 and ϕ_2 are employed to control relative motion toward other points.

In 1998 Angeline noted that generally PSO has fast early convergence, but is slow to fine tune a solution [1]. The PSO population tends to quickly cluster together on multimodal functions causing early stagnation. Shi and Eberhart [20] attempt to correct this by using a self adaptive strategy to adjust the inertia weights used by PSO. Attempting to correct the same problem, Riget and Vesterstrøm [18] apply an attraction and repulsion mode that controls how the particles interact depending on the diversity of the swarm. Should the diversity fall below a predefined threshold, the velocity transition rule is inverted and the particles begin to repel each other. Later, if the diversity increases beyond another threshold, the particles return to the original attractive velocity rule.

Benchmarks and PSO

Historically, the performance of PSO has primarily been demonstrated on a subset of fairly standard benchmark functions. Most PSO literature [1, 20, 2, 8, 23, 12, 15] presents empirical results on the sphere, Rastrigin, Rosenbrock, Schaffer, and Griewank functions. Riget and Vesterstrøm [18] also add Ackley's function to the mix, but Ackley's function differs little from the other test functions. One can show that the sphere, Rastrigin, Schaffer, Griewank, and Ackley's functions are all single-funnel functions with relatively small distances between local optima.

In all of the single-funnel landscapes on which we have seen PSO tested the global optimum is known and centered at or near the origin. The cooperative behavior in PSO produces a tendency to fly toward other population members. This is a result of the velocity update rule being a linear combination of other positions in the cluster (including best-seen points). Methods that use this kind of averaging to incorporate information from population members also exhibit a tendency to perform best when the optimum lies near the center of the initialization region, which is often the origin [16]. If PSO is already roughly centered at initialization it only needs to *swarm in* to find the global optimum. However, if the location of the optimum is *not* known (as is the case in real-world applications), there is no guarantee that

the initialization region will contain the optimum [1]. Monson and Seppi [16] have demonstrated that several variants of PSO perform poorly when the optimum is moved away from the initialization center.

We base our first hypothesis on these results. On multi-funnel functions, this vulnerability would also be exposed if the initialization region did not contain or only partially contained the funnel with the optimal solution.

4. COVARIANCE MATRIX ADAPTATION

We are interested in comparing PSO with an algorithm that uses informed mutation based on local structural information, but does not directly bias its search motion toward other individuals of the population. *Covariance Matrix Adaptation Evolution Strategy*, or CMA-ES, is an evolution strategy that adjusts mutation direction and step size based on how the population is moving through the search space.

Typically *evolution strategies* maintain a population of μ parents which produce λ offspring based on random mutation distributions that center around the parents. In a (μ, λ) strategy, each generation is chosen by selecting the μ best of only the λ offspring. This differs from a $(\mu + \lambda)$ strategy in which each generation is chosen from both parents and offspring. In both cases, the best μ members of the population are selected.

In traditional evolution strategies, mutation distributions are determined using a set of *strategy parameters* that are encoded along with the *object parameters* on the chromosome that is being evolved.

CMA-ES calculates both step size as well as a rotation of the search space which are used to guide the search. Step size is determined in part heuristically. Information is collected about the path (i.e., the sequence of sampled points) that the search is generating. If the path is longer than expected (e.g., much longer than the distance that has been traversed), the search steps are probably parallel and the mutation strength should increase. If the path is shorter than expected, the steps are likely anti-parallel and mutation strength should decrease. The expected length under *random selection* is simply the expected length of a random normal vector ($E||N(0, I)||$). Hansen *et al.* [9] use the following approximation.

$$E||N(0,I)|| = \chi_n = \sqrt{n}\left(1 - \frac{1}{4n} + 1 - \frac{1}{21n^2}\right)$$

Given this estimation, the strategy parameter defining the global step size is updated as follows:

$$\sigma^{g+1} = \sigma^g \cdot \exp\frac{c}{d}\left(\frac{||\vec{p}^{g+1}|| - 1}{\chi_n}\right)$$

Here, d is the damping factor, whose default is 1.

CMA-ES also uses a covariance matrix to explicitly rotate and scale the mutation distribution [10]. Hansen and Ostermeier define the reproduction phase from generation g to generation $g + 1$ as:

$$x_k^{(g+1)} = \langle x \rangle_\mu^{(g)} + \sigma^{(g)}\mathbf{B}^{(g)}\mathbf{D}^{(g)}z_k^{(g+1)}$$

where $z_k^{(g+1)}$ are randomly generated from an $N(0, I)$ distribution. This creates a set of base points that are rotated and scaled by the eigenvectors ($\mathbf{B}^{(g)}$) and the square root of the eigenvalues ($\mathbf{D}^{(g)}$) of the covariance matrix C. The single global step size $\sigma^{(g)}$ is calculated and is used to scale the distribution. Finally, the points are translated to center around $\langle x \rangle_\mu^{(g)}$, the mean of the μ best parents of the population.

To compute covariance, CMA-ES uses a time dependent portion of the path. The path updates after each generation using a weighted sum of the current path, $p_c^{(g)}$, and a vector that points from the mean of the μ best points in generation g to the mean of the μ best points in generation $g + 1$. A principle components analysis on the evolution path is used to update the covariance matrix.

For larger populations, CMA-ES uses a rank-μ-update that calculates the covariance of the μ best individuals

$$\mathbf{Z}^{(g+1)} = \frac{1}{\mu}\sum \mathbf{B}^{(g)}\mathbf{D}^{(g)}z_i^{(g+1)}(\mathbf{B}^{(g)}\mathbf{D}^{(g)}z_i^{(g+1)})^T$$

Assuming $\mathbf{Z}^{(g+1)}$ is the covariance of the μ individuals, and $\mathbf{P}^{(g+1)}$ is the covariance of the path, the new covariance matrix is

$$\mathbf{C}^{(g+1)} = (1 - c_{cv})\mathbf{C}^{(g)} + c_{cv}\left(\alpha_{cv}\mathbf{P}^{(g+1)} + (1 - \alpha_{cv})\mathbf{Z}^{(g+1)}\right)$$

where c_{cv} and α_{cv} are constants that weight the importance of each input. This covariance matrix estimation allows CMA-ES to evolve elliptical mutation distributions which makes CMA-ES very efficient on poorly scaled unimodal surfaces.

5. TEST FUNCTIONS

Hooker [11] has argued that the best forms of testing are hypothesis driven. We have selected test problems that address our hypotheses. The first set of functions are synthetic with two single-funnel and two multi-funnel topologies. The second set are two real-world problems with complicated structures. The first real-world problem is a known multi-funnel function and the second is a high dimensional "black-box" optimization problem.

Synthetics

The two single-funnel functions are Griewank and Rastrigin. Both are popular benchmarks in the PSO literature. Griewank is a bowl-shaped quadratic with cosine perturbations:

$$f_1(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

Similarly, Rastrigin is a cosine-modulated sphere function:

$$f_2(\vec{x}) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10)$$

The two multi-funnel functions are Schwefel and Rana. Rana is defined as follows:

$$f_3(\vec{x}) = \sum_{i=1}^{n-1} f_{rana}(x_i, x_{i+1})$$

where $f_{rana}(x, y) = x\sin\alpha\cos\beta + (y+1)\cos\alpha\sin\beta$ with $\alpha = \sqrt{|-x+y+1|}$ and $\beta = \sqrt{|x+y+1|}$. Schwefel is defined as:

$$f_4(\vec{x}) = \sum_{i=1}^{n}\left(-x_i\sin\left(\sqrt{|x_i|}\right)\right)$$

In order to eliminate axis-parallel symmetry, we rotated all test functions 20 degrees.

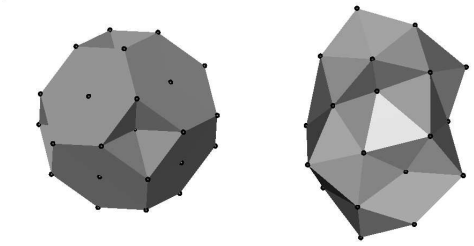

Figure 2: The two lowest energy atomic Lennard-Jones clusters with 38 atoms.

Lennard-Jones clusters

We selected Lennard-Jones cluster optimization to test performance on a real-world problem with a multi-funnel topology. The optimization of Lennard-Jones clusters involves finding the atomic cluster with the smallest potential energy, based on the distance between all the molecules of the cluster. The Lennard-Jones potential is

$$E = \sum_{i<j}^{N} \left(\left(\frac{1}{r_{ij}} \right)^{12} - \left(\frac{1}{r_{ij}} \right)^{6} \right)$$

where r_{ij} is the Euclidean distance between the centers of atoms i and j, and N is the number of atoms in the cluster. This equation defines the simple interactions between the atoms which results in molecular clusters with compact geometries [5].

The energy surface of the Lennard-Jones potential is highly multimodal and the number of local optima grows with problem size. The 38 atoms test problem is particularly interesting because it has been shown to be more difficult than larger clusters of size $N = 60$ [5]. The 38 atom test problem has two very competitive solutions that have a distinctly different atomic structure [6] resulting in two distinct clusters of local optima. Figure 2 shows the two most effective clusters of size $N = 38$.

Spacetrack scheduling

We selected a "black-box" parameter optimization variant of the spacetrack scheduling problem to examine performance on a second real application. The spacetrack scheduling problem concerns the allocation of phased-array radar power to tracking tasks in time. Radar operators are given predictive data based on past observations that include positional information (direction in which to steer the radar beam), object range, and a set of time intervals during which the target object is in transit through the visibility cone of the radar.

Phased-array radars are able to steer their beam electronically, and may track several objects simultaneously by interleaving pulses within a given period of time. However, selection of tracking times must be performed judiciously such that system duty cycle constraints are not violated. Pulse energy is correlated with pulse width in time, so objects that are more distant may require longer pulses. This means that the range of objects being simultaneously tracked affects the number of interleaved tracking operations that can feasibly occur.

The likelihood of a tracking task failure corresponds directly to its signal to noise ratio (SNR) which is a function of its target object's position with respect to the radar array and changes as the target object moves through space. Therefore, choosing tracking times to occur during intervals of high SNR is desired to maximize the probability of tracking success. However, duty cycle constraints make it impossible to schedule all tasks during their peak SNR time and track times must also be selected to produce feasible resource usage.

Since the problem is oversubscribed, we apply a constraint relaxation by allowing resource infeasibility. We pose the problem as an instance of parameter optimization in which we must find a set of tracking times that minimizes the number of resource violations while maximizing total signal to noise ratio.

For an n-task scheduling problem, a set of tracking times is encoded as a vector $\vec{v}$ in $\mathbb{R}^n$. To prevent time-infeasible schedules, we define a mapping F that maps vectors in $\mathbb{R}^n$ to a set $\vec{s}$ of feasible tracking times in the schedule.

That is, $\vec{s}_i = F(\vec{v}_i)$ for $i = 1 \ldots n$. Where F is a feasibility transformation defined as follows. Let

$$W_i = \{(a_{i,1}, b_{i,1}), (a_{i,2}, b_{i,2}), \ldots, (a_{i,|W_i|}, b_{i,|W_i|})\}$$

be the ordered set of visibility windows for the object corresponding to track i. That is, an ordered pair $(a_{i,j}, b_{i,j}) \in W_i$ denotes the earliest tracking opportunity and the last tracking opportunity in window j for track i. Let $V(i,j)$ be the total number of possible start times (with a resolution of one second) for track i before or during the j^{th} visibility window. In particular, $V(i,j) = \sum_{k=1}^{j} ((b_{i,k} - d_i) - a_{i,k}) + 1$. The i^{th} component of the real vector $\vec{v}$ is translated into a feasible start time of the i^{th} track in the following way. Let m_i be the minimum number such that $V(i, m_i) \geq \vec{v}_i \mod V(i, |W_i|)$. Then $\vec{s}_i = F(\vec{v}_i) = a_{m_i} + (\vec{v}_i \mod V(i, |W_i|)) - V(i, m_i - 1)$. Note that we define $V(i, 0) = 0$.

Let SNR_{total} denote the sum of the SNR quantities associated with each track's object at the assigned time in $\vec{s}$. Let PEN_{thres} represent the number of times the SNR drops below threshold for each task. Then the fitness f of a solution is defined to be a quotient of penalty by total SNR.

$$f(\vec{s}) = \frac{PEN(\vec{s})}{SNR_{total}(\vec{s})}$$

where :

$$PEN(\vec{s}) = w_{feas} \cdot PEN_{feas}(\vec{s}) + w_{thres} \cdot PEN_{thres}(\vec{s})$$

Here w_{feas} and w_{thres} terms are weighting coefficients.

The idea is that better (lower) fitness solutions correspond to more promising sets of start times: those with lower violations but higher likelihood of tracking success. The optimization problem is to find a vector $\vec{v} \in \mathbb{R}^n$ that minimizes $f(F(\vec{v}))$.

6. EMPIRICAL RESULTS

Does PSO tend to converge to the funnel that contains the majority of the swarm at initialization?

We ran PSO on a two-dimensional non-rotated instance of Schwefel: one of the multi-funnel problems. We designated two regions for initialization. Region 1 consisted of a disk of radius 400 around the funnel bottom at coordinates (421,-303). Region 2 lies within a disk of radius 400 around another funnel bottom at coordinates (421,421), which is the

global optimum. We ran 1000 trials of the classical PSO algorithm. In 500 of the trials we initialized 80% of the swarm in region 1 and 20% in region 2. In the remaining 500 trials we initialized 80% of the swarm in region 2 and 20% in region 1. We performed this test three times controlling for swarm size 10, 20, and 50. To measure the statistical significance of the results, we performed a chi-square test on the trial counts.

The results (and p-values) of the trials are in Table 1. The table shows statistically significant evidence that the region that contains the majority of the swarm at initialization influences into which region the particles converge.

A plot of the particle trajectories over a contour of the function surface from one trial is illustrated in Figure 3. In this figure, note that a few particles move very near to the global optimum but do not explore further due to the attractive force of the majority of the swarm in the suboptimal funnel.

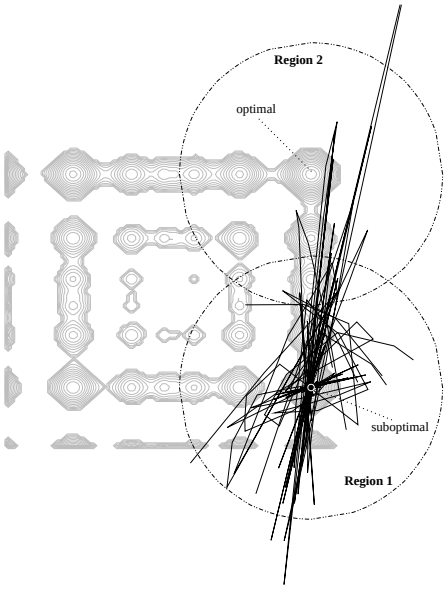

Figure 3: Particle trajectories and regions on 2 dimensional Schwefel function.

How does PSO compare to an algorithm that uses more explorative informed mutation? How does PSO compare to an algorithm that uses uninformed mutation?

We compared PSO's performance to CMA-ES and GENITOR, a steady state genetic algorithm that uses rank-based selection [22]. Our use of CMA-ES is motivated by the fact that it uses an "informed" mutation strategy to move through the space by addressing how the population as a whole is moving. We chose GENITOR to examine an algorithm that uses uninformed mutation, but employs a recombination strategy.

We ran each algorithm for 200,000 evaluations and 30 trials. We employed three variants of PSO: classical PSO, Riget and Vesterstrøm's attractive-repulsive PSO (ARPSO), and a PSO that used random restarts to diversify after detecting stagnation (PSOR).

We selected a population size of 100 for PSO, ARPSO, PSOR, and GENITOR. For CMA-ES, we set $\mu = 125$ and $\lambda = 250$. For PSO, we selected acceleration constants of $\phi_1 = 2.8$, $\phi_2 = 1.3$ as recommended for an "off the shelf" PSO in [2]. We used Clerc's constriction factor [3] to set the inertial coefficient:

$$\chi = \frac{2}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|} = 0.7298$$

where $\phi = \phi_1 + \phi_2$. For GENITOR we used a real-value chromosome representation with HUX crossover, Gaussian mutation with a rate of 0.5, and a linear selection bias of 2.

We tested the algorithms on 30 dimensional Griewank, Rastrigin, Rana, and Schwefel functions. To explore behavior in higher dimensions we tested the algorithms on 500 dimensional Rana and Rastrigin functions.

We expected PSOR to perform better than ARPSO because it re-diversifies more quickly. ARPSO will potentially waste several evaluations in the repulsion phase trying to attain diversity. PSOR is able to gain large amounts of diversity instantaneously by performing a random restart. Our results tend to reflect this. One argument for ARPSO is that actual progress is made during the repulsive phase. In this case, we would expect ARPSO to perform well when exploration is relatively local. However, when more global exploration is needed, PSOR may be able to attain it more quickly.

The convergence plots for the synthetic test functions are shown in Figures 4 and 5. To test the significance of the difference between the top performing PSO method and both GENITOR and CMA-ES, we perform one-sided t-tests and report the p-values in the convergence plots. PSO consistently converges quickly and performs fairly well on the single-funnel functions. On the multi-funnel landscapes, PSO clearly converges into sub-optimal basins. The performance of ARPSO is comparable while PSOR appears to be making slow progress. CMA-ES, on the other hand, makes dramatic progress on the multi-funnel benchmarks. GENITOR performs fairly well on the multi-funnel functions, but seems to struggle on the single-funnel set.

On the more complicated landscapes, PSO may also be having trouble with the scale of the functions due to our use of a static inertial coefficient. On functions where scale changes dramatically, the update rule may be too inelastic to compensate for changes in local features. Angeline [1] points out that the inflexibility of this parameter may serve to impede optimization beyond a particular granularity. This could be mitigated by applying a temperature schedule to the coefficient to change its intensity during the course of the search (see [7] and [20]).

How does PSO compare with other algorithms on real-world problems?

We tested on two real world problems: a 31 dimensional Lennard-Jones cluster and a 1000 task variant of the spacetrack problem. We used the same population parameters mentioned above. On the Lennard-Jones problem, we ran each algorithm for 200,000 evaluations and 30 trials. On the spacetrack problem we ran each algorithm for 200,000 evaluations and 10 trials and initialized the population in a random uniform distribution of radius 10000 about the origin.

To test whether origin-centric initialization affects PSO's performance on the spacetrack problem we also tried initial-

Population size 10

		80% initialized in	
		reg. 1	*reg. 2*
Converged to	*reg. 1*	**376**	43
	reg. 2	36	**388**
	other	88	69
χ^2 test p-value < 0.001			

Population size 20

		80% initialized in	
		reg. 1	*reg. 2*
Converged to	*reg. 1*	**352**	56
	reg. 2	74	**395**
	other	74	49
χ^2 test p-value < 0.001			

Population size 50

		80% initialized in	
		reg. 1	*reg. 2*
Converged to	*reg. 1*	**329**	59
	reg. 2	135	**415**
	other	36	26
χ^2 test p-value < 0.001			

Table 1: Region initialization trials on 2 dimensional Schwefel test function.

izing populations in promising regions of the problem space. We used a greedy activity selection method to create a vector of known good start times for a spacetrack schedule and initialized the population in a uniform random distribution centered on this vector. Similarly, we extracted the fittest point of 1,000,000 random samples and initialized the population in a uniform random distribution around the point.

Since the spacetrack problem possesses a highly irregular structure, we tried different inertial coefficients for PSO and discovered that it performed significantly better with a *low* inertial coefficient when initialized near a seed. This is not surprising since the magnitude of the inertial coefficient affects the granularity of the search [19]. We found that an inertial coefficient of $\chi = 0.8$ on the randomly initialized trials and $\chi = 0.1$ on the near-seed trials performed best.

The plots for the Lennard-Jones problem and the spacetrack problem are shown in Figure 6. The statistical significance of the difference between the top performing PSO method and both GENITOR and CMA-ES are reported as one-sided t-test p-values in the convergence plots. CMA-ES significantly outperforms the rest of the algorithms on the Lennard-Jones problem. GENITOR seems to do very well on the scheduling problem. This may be because GENITOR's recombination strategy is able to exploit information about good partial schedules.

We also have evidence that initialization region on the spacetrack problem affects the performance of PSO significantly. Using a vector of promising start times as the center of the initialization region caused the particle methods to perform better. This is likely due to the fact that the majority of the swarm lies in a better part of the space.

7. CONCLUSIONS

In this paper we have presented evidence that PSO tends to stagnate early into suboptimal solutions on multi-funnel landscapes. Our first hypothesis was that the percentage of initial swarm in a funnel region will influence to which region the swarm eventually converges. We demonstrated, with basic PSO, that the choice of funnel region in which the majority of the swarm is initialized does have an impact on which region the swarm converges. This may be a result of the algorithm's "cooperative" behavior. A small number of population members in a good funnel may be virtually ignored by a swarm whose majority is cooperating in another (potentially suboptimal) funnel.

Our second hypothesis was that the position update rule in PSO may inhibit exploration on multi-funnel functions. We conjectured that algorithms that use more explorative informed mutation, or uninformed mutation may perform better on multi-funnel instances. We showed that PSO performs exceptionally well on single-funnel functions, but struggles in comparison to CMA-ES in lower dimensional multi-funnel cases. CMA-ES also performs well on higher dimensional multi-funnel problems, exhibiting marked improvement after the PSO methods appear to stagnate.

Our third hypothesis was that PSO might have trouble with more complicated real world problems. We demonstrated that CMA-ES does tend to outperform the PSO methods on two real world problems. GENITOR performs well on the scheduling problem, and may be using recombination to exploit partial solutions.

Our results indicate that multi-funnel landscapes may pose a problem to the traditional PSO strategy. This work raises two important research questions, 1) Is PSO better suited to single-funnel optimization problems? 2) If not, what strategies can be employed that mitigate the apparent bias toward exploration within the span of the swarm?

8. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0117209 and by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-03-1-0233. We wish to thank the GECCO reviewers for their insightful comments and suggestions.

9. REFERENCES

[1] P. Angeline. Using selection to improve particle swarm optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 1998.

[2] A. Carlisle and G. Dozier. An off-the-shelf PSO. In *Proc. Workshop on Particle Swarm Optimization*, Indianapolis, IN, 2001.

[3] M. Clerc and J. Kennedy. The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evolutionary Computation*, 6(1):58–73, 2002.

[4] J. Doye, R. Leary, M. Locatelli, and F. Schoen. Global optimization of Morse clusters by potential energy transforms. *INFORMS Journal on Computing*, 16(4):371–379, 2004.

[5] J. P. Doye. Physical perspectives on the global optimization of atomic clusters. In *Global optimization – select case studies*, Kluwer, 2006.

[6] J. P. Doye, M. A. Miller, and D. J. Wales. The double-funnel energy landscape of the 38-atom Lennard-Jones cluster. *Journal of Chemical Physics*, 110(14), April 1999.

[7] R. C. Eberhart and Y. Shi. Comparison between genetic algorithms and particle swarm optimization. In *Evolutionary Programming VII*, pages 611–616.

[8] R. C. Eberhart and Y. Shi. Comparing inertia weights and constriction factors in particle swarm optimization. In *Proc. of the 2000 Congress on Evolutionary Computation*, pages 84–88, Piscataway, NJ, 2000.

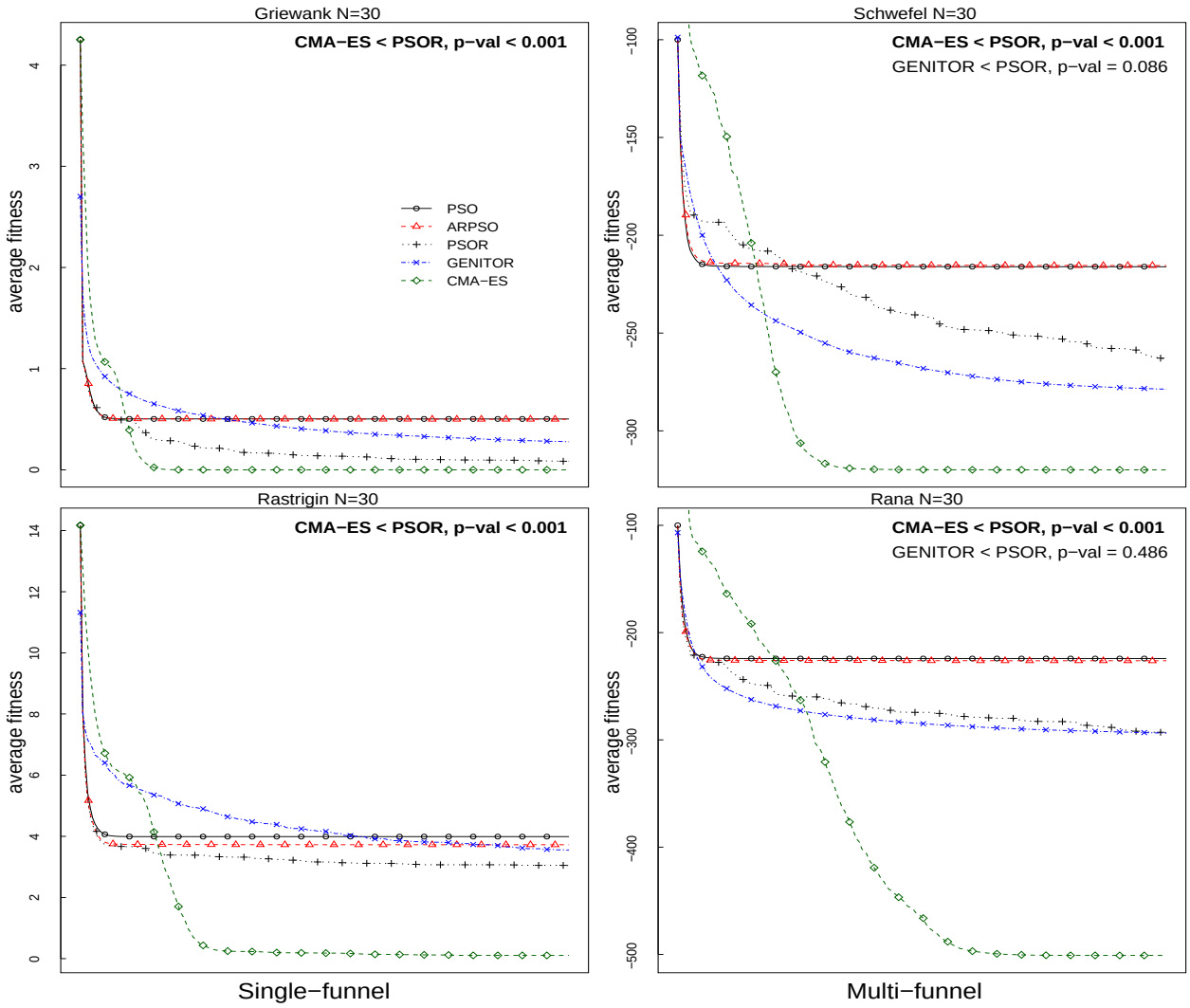

Figure 4: Convergence plots for 30 dimensional synthetic problems.

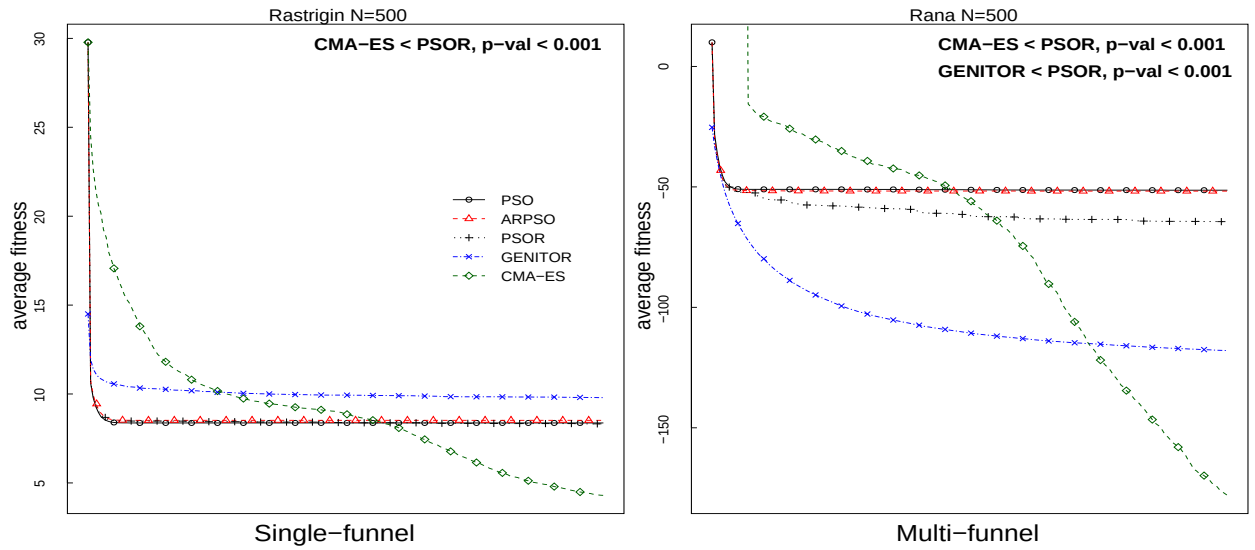

Figure 5: Convergence plots for 500 dimensional synthetic problems.

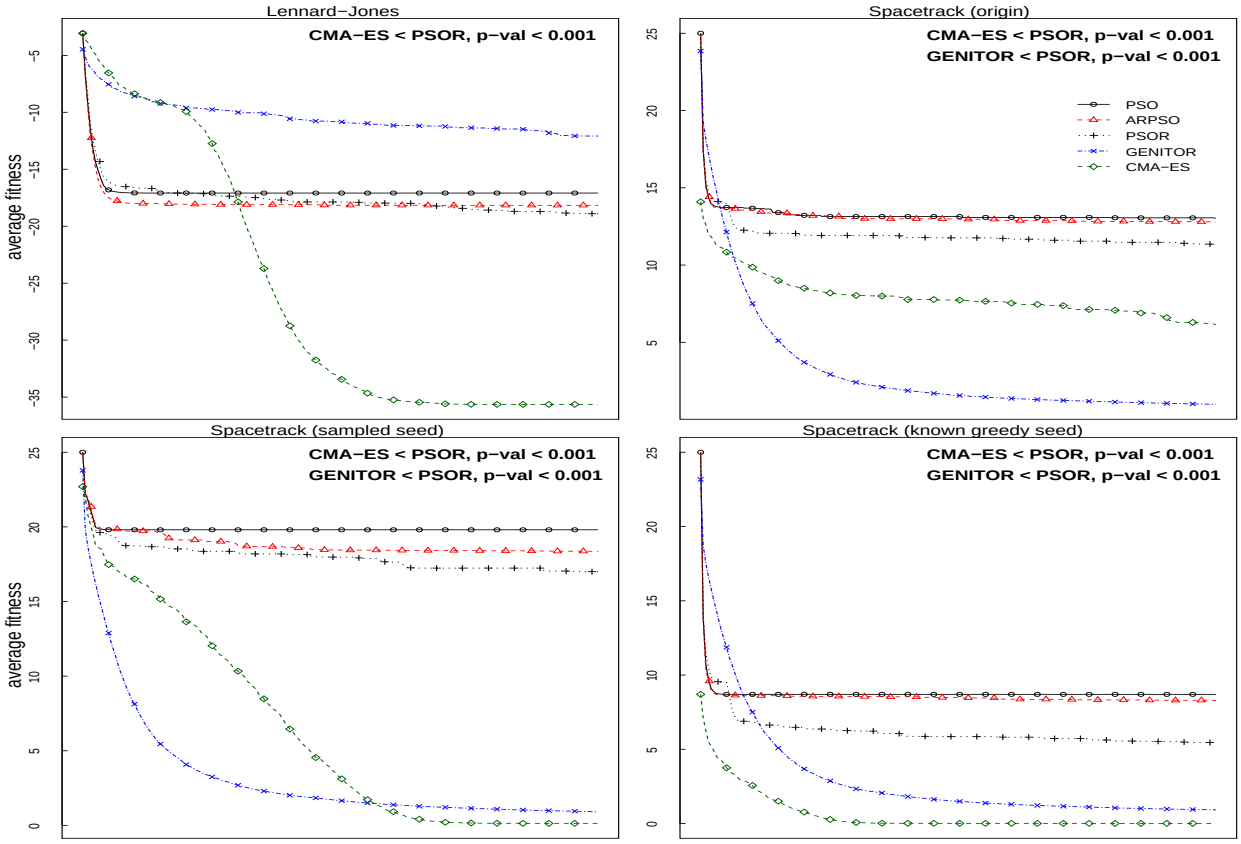

Figure 6: Convergence plots for real world problems.

[9] N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In *PPSN*. Springer, 2004.

[10] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[11] J. Hooker. Testing heuristics: We have it all wrong, 1996.

[12] J. Kennedy. Bare bones particle swarms. In *Proc. IEEE Swarm Intelligence Symposium*, 2003.

[13] J. Kennedy and R. Eberhart. Particle swarm optimization. In *International Conference on Neural Networks*, pages 1942–1948, Perth, Australia, 1995.

[14] M. Locatelli. On the multilevel structure of global optimization problems. *Computational Optimization and Applications*, 30, 2005.

[15] R. Mendes, J. Kennedy, and J. Neves. The fully informed particle swarm: Simpler, maybe better. *IEEE Trans. Evolutionary Computation*, 8(3):204–210, 2004.

[16] C. K. Monson and K. D. Seppi. Exposing origin-seeking bias in pso. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2005*, 2005.

[17] P. M. Pardalos and F. Schoen. Recent advances and trends in global optimization: Deterministic and stochastic methods. In *Proceedings of the Sixth International Conference on Foundations of Computer-Aided Process Design*, 2004.

[18] J. Riget and J. Vesterstroem. A diversity-guided particle swarm optimizer - the ARPSO, 2002.

[19] Y. Shi and R. C. Eberhart. Parameter selection in particle swarm optimization. In *Evolutionary Programming VII*, pages 591–600.

[20] Y. Shi and R. C. Eberhart. Empirical study of particle swarm optimization. In *Proceedings of the Congress of Evolutionary Computation*, volume 3, pages 1945–1950, 1999.

[21] D. J. Wales. Energy landscapes and properties of biomolecules. *Physical Biology*, 2:S86–S93, 2005.

[22] D. Whitley. The GENITOR algorithm and selection pressure. In *Third International Conference on Genetic Algorithms*, 1989. Morgan Kaufman.

[23] Z.-L. Y. Xiao-Feng Xie, Wen-Jun Zhang. Adaptive particle swarm optimization on individual level. In *International Conference on Signal Processing*, 2002.