The Dispersion Metric and the CMA Evolution Strategy

Monte Lunacek
Department of Computer Science
Colorado State University
Fort Collins, CO 80523
lunacek@cs.colostate.edu

Darrell Whitley
Department of Computer Science
Colorado State University
Fort Collins, CO 80523
whitley@cs.colostate.edu

ABSTRACT

An algorithm independent metric is introduced that measures the dispersion of a uniform random sample drawn from the top ranked percentiles of the search space. A *low dispersion* function is one where the dispersion decreases as the sample is restricted to better regions of the search space. A *high dispersion* function is one where dispersion stay constant or increases as the sample is restricted to better regions of the search space. This distinction can be used to explain why the CMA Evolution Strategy is more efficient on some multimodal problems than on others.

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

Measurement

Keywords

Dispersion, CMA-ES

1. INTRODUCTION

It is well known that all search algorithms have some particular bias: since there is no general purpose search method that works well on all problems, all search methods must be biased to solve particular types of problems. But rarely do the developers of new heuristic search methods document on which types of problems their algorithm is likely to yield good performance. Of course there are exceptions. For example, Hansen and Kern [10] noticed that the performance of an *Evolution Strategy with Covariance Matrix Adaptation*, or CMA-ES, could fail if the underlying structure of a multimodal test problem was unpredictable. Since we do not have good general metrics that allow us to classify optimization problems into different subclasses, it is difficult to explain why some problems are apparently more difficult than others.

One common observation is that some test functions, such as Rastrigin's, are considered difficult because they are highly multimodal, but they display an underlying unimodal structure that in fact makes the test function relatively easy. Addis *et al.* points out that certain multimodal test functions can be seen as perturbations of a more simple underlying structure that has a low number of local optima [1]. Furthermore, it is the underlying problem structure that makes these problem easier or more difficult, not the actual number of local optima [15].

Covariance Matrix Adaptation uses principle components analysis to determine the variance associates with different samples of points from the search space, including the best points in the current offspring population as well as points along a defined "path" or trajectory as the search progresses. This makes CMA-ES extremely good at detecting and exploiting local structure. It is also insensitive to rotations of the search space that can cause serious degradation in performance for many other local search methods and evolutionary algorithms. CMA-ES has proven to be very effective on many well known test functions, and on several real world applications.

Despite the emphasis on exploiting local information in CMA-ES, Hansen and Kern have empirically shown that CMA-ES also performs well on multi-modal functions where there exists a "global topology." CMA-ES is more prone to fail when the global structure is less predictable [10]. Recently, several different research communities have started to pay attention to how the features of the "global topology" in multimodal optimization problems can impact the performance of search algorithms. However, concepts aimed at defining "global topology" are often vague and are also often intractable in the general case because they require knowledge that would seem to require enumeration of much of the search space.

In this paper we introduce the *dispersion* metric, then use this to classify functions as high or low dispersion functions. We then use this classification to predict the efficiency and effectiveness of CMA-ES. We show that CMA-ES works very well on low dispersion functions, but has difficulties with high dispersion functions.

The *dispersion* of a function measures how close together a sample of points are, where the sample of points is constructed so as to represent an aspiration threshold expressed

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

as a target percentile of the search space. For example, we can measure the dispersion over points estimated to be in the top 5% of the search space and compare this to points estimated to be in the top 0.5% of the search space. Dispersion can be computed using a relatively modest number of samples from the objective function.

We also explore how and why function *dispersion* affects the performance of CMA-ES. Our results indicate that the adaptive step-size heuristic, called *cumulation*, does not function as intended when the best regions of the search space are too spread out (i.e., too disperse).

2. DEFINING FUNCTION DISPERSION

This paper introduces an algorithm independent metric that measures the *dispersion* of a function. Dispersion is measured based on a sample drawn from the best points in the search space. The sample of points is constructed such that they represent an aspiration threshold expressed as a target percentage of the search space. This allows us to use the dispersion metric in two ways: First, we can compare the dispersion of different functions at a particular threshold; second, we can calculate the dispersion of a single function at different thresholds. In the second case, we might measure the dispersion over points estimated to be in the top 5% of the search space and compare this to points estimated to be in the top 0.5% of the search space. A low dispersion function is defined to be one where the dispersion measure decreases as the sample is restricted to better regions of the search space. A high dispersion function is one where dispersion stays relatively constant or even increases as the sample is restricted to better regions of the search space.

The dispersion metric gives us a means to determine a significant feature of the global topology of a function. Are the best points in the search space, including the best local optima for multimodal functions, localized or disperse in the search space? Some highly multimodal functions have dispersion measures that are approximately the same as a simple unimodal sphere function: when this is the case, empirically experiments confirm that low dispersion functions are easy to optimize.

To make the dispersion metric useful, we define it to be a function based on a sample of the search space. There exists other recently introduced metrics that attempt to measure similar trends in the global topology, but these measures have the unrealistic requirement that the search space be enumerated, or at least that the best local optima are located and enumerated. These metrics are briefly reviewed in the next section.

We calculate dispersion by drawing a uniform random sample of points from the search space and calculate approximate thresholds that breaks the sample into subsets representing the best X% of the seach space. For example, if we sample 100,000 random points, we can define a 10% threshold to be the maximum fitness of the best 10,000 points. A 1% threshold is the maximum fitenss of the best 1,000 points. As the percentage decreases, the threshold also decreases. Figure 1 shows an example of this technique for one-dimensional versions of the Rastrigin function and the so-called Schwefel function. Notice that the *threshold* leaves only the best local optima.

In this way, a threshold is determined by a sample size, s_v, and a percentage p. At this threshold, *dispersion* refers to the average pair-wise distance between the best $p \times s_v$ points in the sample. In practice, computing the average pair-wise distance of $p \times s_v$ points can be computationally costly. For example, if $s_v = 100,000$ and $p = 0.01$, then $s_v \times p = 1000$. This implies computing the distance between $1,000^2/2 - 1,000 = 499,000$ points.

To make the dispersion calculation more efficient, we define $s_b = s_v \times p$ to be a fixed sample size. Our dispersion metric is: given an objective function, $f(\vec{x})$, a fixed sample size, s_b, and a variable sample size, s_v, *dispersion*(s_b,s_v,$f(\vec{x})$) calculates the average pair-wise distance between the best s_b points of the random sample, s_v. The variable sample size, s_v can be expressed as a function of the fix sample size s_b and the desired percentage, p: $s_v = s_b/p$. For example, given a fixed sample size of $s_b = 100$, a variable sample of size $s_v = 100/0.01 = 10,000$ is needed to create a $p = 0.01$ threshold. Below is the dispersion pseudo code which emphasizes clarity instead of efficiency.

Dispersion(s_b,s_v,$f(\vec{x})$)

---

input
Integer s_b – the fixed sample size
Integer s_v – the variable sample size
$f(\vec{x})$ – the objective fitness function

variables
allPoints – a vector of random $\{\vec{x}, f(\vec{x})\}$ pairs.
bestPoints – a vector of the best $\{\vec{x}, f(\vec{x})\}$ pairs.

for $i = 1$ to s_v **do**
  Create a random point, $\vec{x}$.
  Evaluate its *fitness*, $f(\vec{x})$.
  Add $\{\vec{x}, f(\vec{x})\}$ to allPoints.
end for

bestPoints $\leftarrow$ best s_b points of allPoints
return *average pairwise distance*(bestPoints)

Decreasing the threshold can change the dispersion metric of a function at a particular threshold. To understand why this happens, consider the one dimensional Rastrigin and Schwefel problems shown in Figure 1 . The horizontal line represents the threshold and the gray regions specify the basins of attraction that are below threshold. Notice that on the Rastrigin function, lowering the threshold implies that the basins of attraction that we are measuring will get closer together. This means that the expected pairwise distance between the best points of our sample will decrease. On the Schwefel function, however, as the threshold decreases, the distance between basins of attraction actually increases. This implies that lowering the threshold will cause the dispersion to increase.

The point estimate of dispersion given in dispersion pseudo code is not invariant with respect to scale. That is, it is possible for two algorithms with completely different underlying structures to have similar *dispersion* measures for a given sample size. However, decreasing the aspiration threshold will change this point estimate, and this change is a more accurate measure of the underlying topology. We calculated the dispersion of several classic benchmark test functions: Sphere, Rastrigin, Schwefel, Rana, F101, Ackley, Bohachevsky, Griewank, and Shaffer [18][10]. We computed

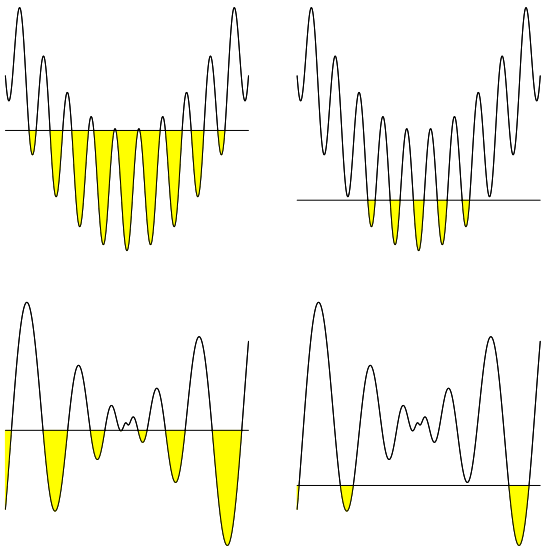

Figure 1: A decrease in fitness threshold will tend to decrease the dispersion on the Rastrigin function (the top figures) and increase the dispersion of the Schwefel function (the bottom figures).

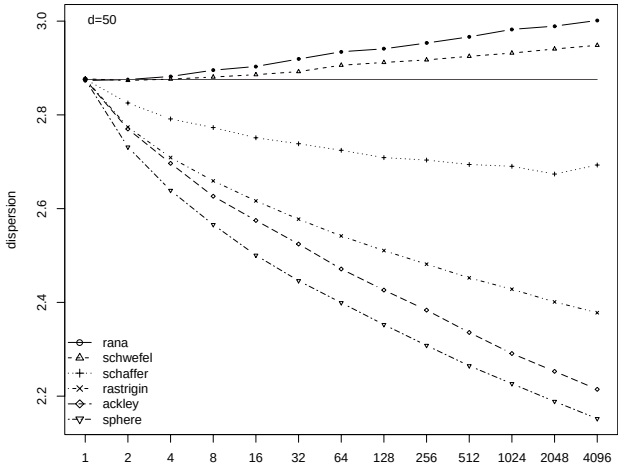

Figure 2: The dispersion of each test function computed for different sample sizes ($\times 100$). The horizontal indicates the dispersion of each sample before selection is applied: *dispersion*$(100, 100)$. Because we rescaled the boundaries of each function such that the range was $(0, 1)$, the initial dispersion using the first 100 points is the same for all functions.

the dispersion of the best $s_b = 100$ points for a set of increasing sample sizes; starting with $s_v = 100 \cdot 2^0$ we doubled the sample size each iteration until we reached $s_v = 100 \cdot 2^{12}$ samples (e.g. 100, 200, 400, 800, ..., 204800, 409600). This yields both a point estimate of dispersion at a particular threshold as well as information about how the dispersion changes as the threshold decreases. In order to make cross-function comparisons more meaningful, we rescaled the best 100 points based on the boundaries of the function such that the new minimum and maximum parameter values are 0 and 1 respectively. Then, we computed the distance for all $(100^2/2 - 100) = 4,900$ pairs of points and averaged this number. We repeated this 30 times for each function in 20, 50, and 100 dimensions. Figure 2 shows the average dispersion as a function of sample size for several 50 dimensional functions. The solid horizontal line indicates the average dispersion of the first 100 random points, denoted *dispersion*$(100, 100)$. We found that as we increased the sample size, the dispersion increased on the Schwefel, Rana and F101 functions. The opposite was true for all the remaining functions. Increasing the sample size actually decreased the dispersion of these functions.

We computed the change in dispersion for each benchmark function by subtracting the lowest threshold dispersion value in our set from the average dispersion value when no selection is applied. Specifically, we computed the value: *dispersion*$(100, 409600)$ - *dispersion*$(100, 100)$. On functions where the dispersion increases as we decrease the threshold (by increasing the sample size s_v), we should get a positive number. When the dispersion decreases as we decrease the threshold, we will get a negative number. Figure 3 shows the change in dispersion for several benchmark test problems. This allows us to clearly distinguish between *low* and *high* dispersion functions. For the remainder of the paper, low dispersion is associated with functions whose change in dispersion is less than zero. Low dispersion functions include the Schaffer, Rastrigin, Bohachevsky, Ackley, and Griewank functions. The Schwefel, Rana, and F101 are referred to as high dispersion functions.

2.1 Related Work

Lennard-Jones clusters are a popular benchmark test function for configuration optimization problems [4]. The goal is to find the atomic cluster with the smallest potential energy. The potential energy of the system is modeled 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. The search space contains a large number of equivalent solutions because there are $N!$ ways to order the N atoms in any local minima [8]. The 38 atoms test problem is particularly interesting because it has been shown to be more difficult than larger clusters because it has two very competitive solutions that have a distinctly different atomic structure [6] resulting in two distinct clusters of local optima.

A *funnel* is a topographical feature that is used to explain why some Lennard-Jones cluster instances are more difficult than others [6]. The exact definition of a funnel is vague. Doye describes a funnel as "a set of downhill pathways that converge on a single low-energy structure or set of closely related low-energy structures" [5]. An alternative definition suggests 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".[3]

It has been suggested that many applications in computational biology are more difficult because the energy function

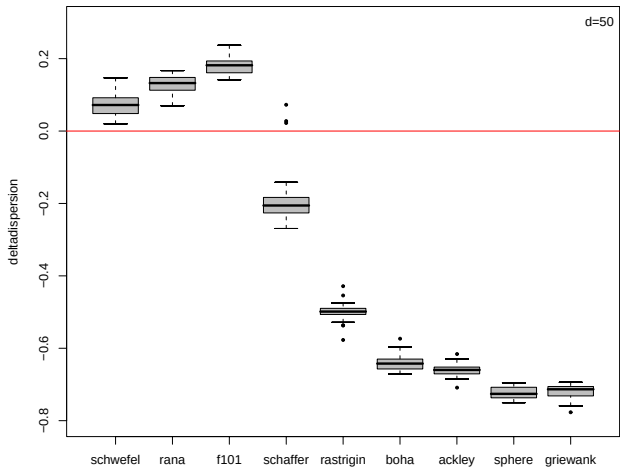

Figure 3: The change in dispersion of each test function.

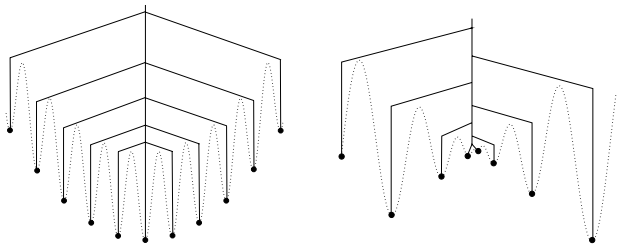

Figure 4: The disconnectivity graph for the Rastrigin (left) and Schwefel (right) functions.

appears to have multiple local optima that form distinct, spatially separate clusters in the search space [17] and that this has more impact on problem difficulty than the number of local optima [15].

Doye and Wales use disconnectivity graphs to visualize the funnel structure of various energy landscapes [6]. The disconnectivity graph is actually a tree whose leaves are the local optima. Two or more leaf nodes are connected by a node if they exist in the same basin of attraction. Doye uses different energy levels to define basins of attraction. A split in the tree implies that every local optima below this point is reachable by a path whose fitness does not exceed the given energy threshold. In other words, the minimum *saddle point* of the path connecting two local optima is lower than the energy threshold. Figure 4 shows the disconnectivity graphs for the one dimensional Rastrigin and Schwefel functions. Notice that the Rastrigin function has a single dominate stem. This is indicative of a single funnel topology. On the other hand, the Schwefel function has several stems that split early at high energy levels and has at least two funnels.

Disconnectivity graphs have been applied to discrete optimization problems where landscape concepts, such as local optima, basins of attraction, and saddle points are clearly defined [7]. Flamm *et al.* extend these critical definitions so that disconnectivity graphs, which they call *barrier trees*, can be generated for highly *degenerate* discrete problems [7].

Locatelli et al. offers a different view of *funnel* landscapes [13]. A graph is defined where the nodes of the graph are the local optima of the function and an directed edge is

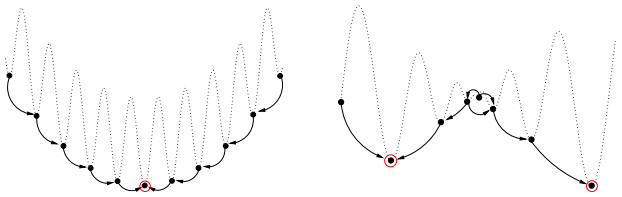

Figure 5: Locatelli's view of a funnel landscape.

extended from node X to node Y if $f(X) > f(Y)$ and X and Y are adjacent local optima. Local optima that are within distance r of each other are said to be adjacent. A funnel bottom is a node with no outgoing directed edges. An example of this graph for the one dimension Rastrigin and Schwefel functions is given in figure 5. One drawback to this method is that Locatelli's graphs will change dramatically depending on what distance measure r is used.

One problem with *disconnectivity graphs*, *barrier trees*, and Locatelli's method is that they all requires locating the relevant local minima in the search space. This cannot be efficiently done for "black box" optimization problems where the underlying mathematical function is unknown or does not exist. Even when derivative information exists and can be used to catalog all the *local optima* and *saddle points*, these methods are limited to relatively small problems. Doye *et al.* admit that finding all the minima for large clusters is impossible: "We therefore stopped searching once we were confident that we had obtained an accurate representation of the low-energy regions of the [Potential Energy Surface] " [4]. Hallam and Prügel-Bennett point out that exhaustive enumeration of the search space raises concerns regarding the usefulness of *barrier trees* [9]. To mitigate this problem, they use a branch and bound technique to find only the best local optima. But this technique is still limited to relatively small problems. Hallam and Prügel-Bennett construct barrier trees for MAX-SAT problems with 40 variables.

Our *dispersion* metric is appealing because it requires a relatively modest number of samples to observe a noticeable change in dispersion, even on problems with 500 parameters. Furthermore, *disconnectivity graphs* and *barrier trees* do not directly address the *distance* between local optima but rather the *barrier height* between local optima.

Several test function characteristics have been identified that help explain why some problems are more challenging than others. Dispersion is another way of distinguishing multimodal problems that compliments other characteristics. For example, Salomon [16] and Whitley [18] both noticed that many of the benchmark problems are *separable*. Salomon [16] rotated some of the common separable benchmark problems to create similar landscapes with a high degree of parameter interaction. He found the breeder genetic algorithm (BGA) was significantly less effective when the coordinate system is rotated in $n-$dimensional space. Salomon used the degree to which a problem is separable to explain under what conditions the BGA was likely to perform well.

In the same way, we use the dispersion metric to identify the types of multimodal problems for which CMA-ES is most likely to be successful.

3. THE CMA EVOLUTION STRATEGY

The canonical *evolution strategy* is an iterative process where a population of μ distinct parents produce λ offspring based on mutation distributions that are centered around the parents. Only the best μ offspring are chosen to be the parents of the next generation. This is known as a (μ, λ) selection strategy.

One common belief is that population-based methods are better at optimizing multimodal functions because they tend to explore more of the fitness landscape before their population converges to a more compact and globally competitive region search space. This approach is appealing because, by comparing the values of each candidate solution, population-based algorithms have a better perspective of the entire search space than purely *local search* methods.

Hansen *et al.* interpret any Evolution Strategy that uses *intermediate recombination* as a local search strategy [12]. Intermediate recombination creates a single parent based on the average position of the current population. Sometimes a weighted average is used to determine the position of the best parent. The next generation of offspring are based on a mutation distribution that surrounds this single parent. Once the initial mutation distribution has decreased, this variation of the traditional evolution strategy will behave much like a local search algorithm. In this paper, we use the default log-weighted intermediate recombination for CMA-ES, denoted (μ_W, λ)-CMA-ES.

Ostermeier *et al.* introduce *cumulative step-size adaptation*, or CSA, as a means of adapting a global step-size [14]. CSA uses information from previous generations as a reliable way of evolving step sizes. The sequence of consecutive steps is called *cumulation*.

The *evolution path* is a vector that points in the *recent* overall direction search has taken. Assuming that $\langle \vec{x}^g \rangle$ is the mean of generation g, then the evolution path is calculated as

$$\vec{p}^{g+1} = (1-c) \cdot \vec{p}^g + \sqrt{c(2-c)} \cdot \frac{\sqrt{\mu}}{\sigma^g} \left(\langle \vec{x}^{g+1} \rangle - \langle \vec{x}^g \rangle \right)$$

where $\langle \vec{x}^g + 1 \rangle - \langle \vec{x}^g \rangle$ is the vector representing the current step. The relative importance of previous steps is weighted by c, the cumulative time parameter. When $c = 1$, no history is considered and only the most recent step contributes to the current cumulation. The value for c is usually on the order of $1/\sqrt{n}$ to $1/n$. The value $\sqrt{c(2-c)} \cdot \frac{\sqrt{\mu}}{\sigma^g}$ is a normalization constant [12].

Hansen *et al.* reasons that if the evolution path is longer than expected, the search steps are probably parallel and the mutation strength should increase. Otherwise, the evolution path is shorter than expected, meaning the steps are likely anti-parallel. The 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.* uses 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.

Covariance Matrix Adaptation, or CMA, uses a covariance matrix to explicitly rotate and scale the mutation distribution [12]. 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 as described in the CSA algorithm 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 evolution path. The evolution 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.

When a larger population (λ) is used, the best μ individuals may help describe the topology around the mean of the current generation [11]. This may increase the accuracy of the covariance matrix estimation. In order to exploit this information, CMA-ES uses a rank-μ-update, that calculates the covariance of the steps that lead to the best μ individuals:

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

This information, along with the evolution path, $p_c^{(g)}$, is used to update the covariance matrix. Assuming $\mathbf{Z}^{(g+1)}$ is the covariance of the steps leading to the best μ individuals, and $\mathbf{P}^{(g+1)}$ is the covariance of the evolution 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.

4. GLOBAL SEARCH WITH CMA-ES

Hansen and Kern found that a large population size and rank-μ updates improved the performance of CMA-ES on multimodal functions. They state that, "If the local optima can be interpreted as a perturbations of an underlying unimodal function, then larger populations can detect this global topology" [10]. Furthermore, they state, "A strong asymmetry of the underlying function jeopardizes a successful detection and can lead to failure".

We proposal that dispersion is one way to quantify and identify functions with underlying unimodal surfaces. We also predict that asymmetric functions will tend to have higher dispersion.

The distribution used by CMA-ES is initially isotropic. As a result, the initial value of σ is critical for exploration. Hansen and Ostermeier suggest that the quality of solutions found by CMA-ES using a small initial step-size are often determined by the location of the starting point [12]. Hansen and Kern further emphasize that small initial step-sizes can have a considerable impact on the performance of CMA-ES[10]; they set the σ_0 between 20% and 40% of the length

of the constrained region of the search space. Auger and Hansen also test CMA-ES on multimodal functions and set $\sigma_0 = 50\%$ of the constrained region [2]. Hansen *et al.* summarizes that "A larger step-size improves the global search performance of a local search procedure"[12]. In this paper, we use an initial step-size suggested by Auger and Hansen which is $\sigma_0 = (U - L)/2$, where L and U are the lower and upper bounds of the constrained search space.

Each test function is usually bound constrained. As implemented here, the test functions have a boundary penalty to insure that a solution is in the feasible region. This method, described by Hansen *et al.* [10], is the standard way of handling boundary conditions when testing CMA-ES on multimodal functions [10, 2]. The penalty is proportional to the number of parameters that fall outside the boundary. Therefore, the penalty for a point where one parameter is outside the feasible region is less than a point where all the parameters are outside the feasible region.

Hansen and Kern [10] summarize that a larger population size considerable improves the performance of CMA-ES on all problems with one notable exception, the Griewank function. Here, they found that a larger population performed better in lower dimensions, but a smaller population was more effective in higher dimensions. One explanation for this is that the Griewank function actually gets easier as dimensionality increases [18]. We can also use dispersion to explain this apparent anomaly. In low dimensions (e.g. two), the dispersion of the Griewank function is much higher than the sphere function. However, referring back to Figure 3, we can see that the dispersion of Griewank and the sphere are nearly indistinguishable. This is not a shortcoming of our metric, but rather an indication of how smooth the high dimensional Griewank actually is.

In order to understand the relationship between dispersion and the performance of CMA-ES, we ran CMA-ES with several different population sizes on all of the benchmark test functions described earlier. Our hypothesis is that CMA-ES will perform relatively well on the test functions that we categorized as *low dispersion* and will be less effective on the *high dispersion* functions.

Figure 6 shows the convergence of CMA-ES on six of the test functions. In every case, we used restarts to ensure that each instance of CMA-ES ran for exactly $200,000$ evaluations. The population size was varied using $\lambda = 50, 100, 250, 500$. On all the *high dispersion* functions, the larger values of λ produced the most effective solutions. However, none of these represent solutions that are close to the optimal. This is consistent with the observations made by Hansen *et al.*[10]; when the underlying problem structure is less predicable, the performance of CMA-ES suffers. CMA-ES was much more successful on the *low dispersion* functions. The Schaffer and Bohachevsky were easily solved for all values of λ. On Rastrigin's function, CMA-ES was more successful with larger values of λ, but was able to get relatively close to the optimal solution even for small population sizes. Referring back at Figure 3, we can see that dispersion gives a reasonable prediction of how well CMA-ES will perform on multimodal surfaces.

In the conclusion of their paper, Hansen and Kern suggest starting with a small population and increasing its value with each restart of CMA-ES [10]. Auger and Hansen have implemented this strategy, which they refer to as IPOPCMA, where the population doubles each time a restart occurs [2]. Starting with the default population size of $\lambda = 4+3\cdot log(n)$, each restart was initialized with a population size twice that of the previous instance. For example, in 20 dimensions, the population sequence would be $12, 24, 48, 96, 193, 348, ...$ The authors assert that this essentially makes CMA-ES parameter-free because the population size does not need to be tuned for each problem [2].

Looking at the convergence graphs for the Shaffer and Bohachevsky functions in Figure 6, it is not too surprising that this type of strategy would perform well on these functions where smaller population sizes are equally effective, and therefore, preferred because they have a faster rate of convergence. But when the test suite contains mostly *unimodal* and *low dispersion* multimodal functions, this average behavior can be misleading. On *high dispersion* functions, incrementally increasing the population size incurs dramatic increase in convergence time. For example, Hansen *et al.* found that CMA-ES used over 250,000 evaluations in order to solve the 10 dimensional Schwefel function. Similarly, Auger and Hansen found that when they increased the maximum number of evaluations from $300,000$ to $900,000$ the performance of the 30 dimensional Rastrigin function greatly increased.

One conclusion here is that on *high dispersion* functions, where larger values of λ are likely to be the most effective, IPOPCMA will require an unrealistic number of evaluations just to start using a more effective value for λ. While this efficiency may be satisfactory on some problems, it infeasible for others.

5. UNDERSTANDING CONVERGENCE

On high dispersion functions, the performance of CMA can sometimes be quite poor; the solutions are less effective when compared to other functions and there is a noticeable increase in the overall efficiency. Intuitively, it makes sense that finding the global optimum of a high dimension, high dispersion function is difficult. There are simply too may "funnels" that can trap even the most effective search algorithm. On low dispersion functions, like Rastrigin's, every starting position is in the only (and optimal) *funnel*.

Although we can reason why CMA-ES is less effective, there does not appear to be an easy explanation as to why it is also less efficient in terms of convergence. Figure 7 shows the normalized mean convergence of CMA-ES with a population size of $\lambda = 500$ on four 20 dimensional benchmark test functions: Ackley, Rastrigin, Schwefel, and the Rana function. Notice that for the high dispersion Rana and Schwefel functions, it takes well over 100,000 evaluations to converge. Although $\lambda = 500$ seems too large for a 20 dimension problem, we have already seen that the best solutions on high dispersion functions are likely to be discovered with larger populations.

One reason the convergence is slow is that CMA-ES tends to "waste" evaluations on points that are infeasible, even on low dispersion functions like Rastrigin and Ackley. This is because the initial σ value will often create points that are outside the boundary of the problem. Of course, this evaluations are not wasted in the sense that they are not useful, but they cannot offer any improvement in fitness and can therefore can be seen as inefficient. On average, over 10% of the $200,000$ evaluations used on the 20D Rana and Schwefel functions were infeasible. This is almost three times higher than that of the low dispersion functions we tested. This

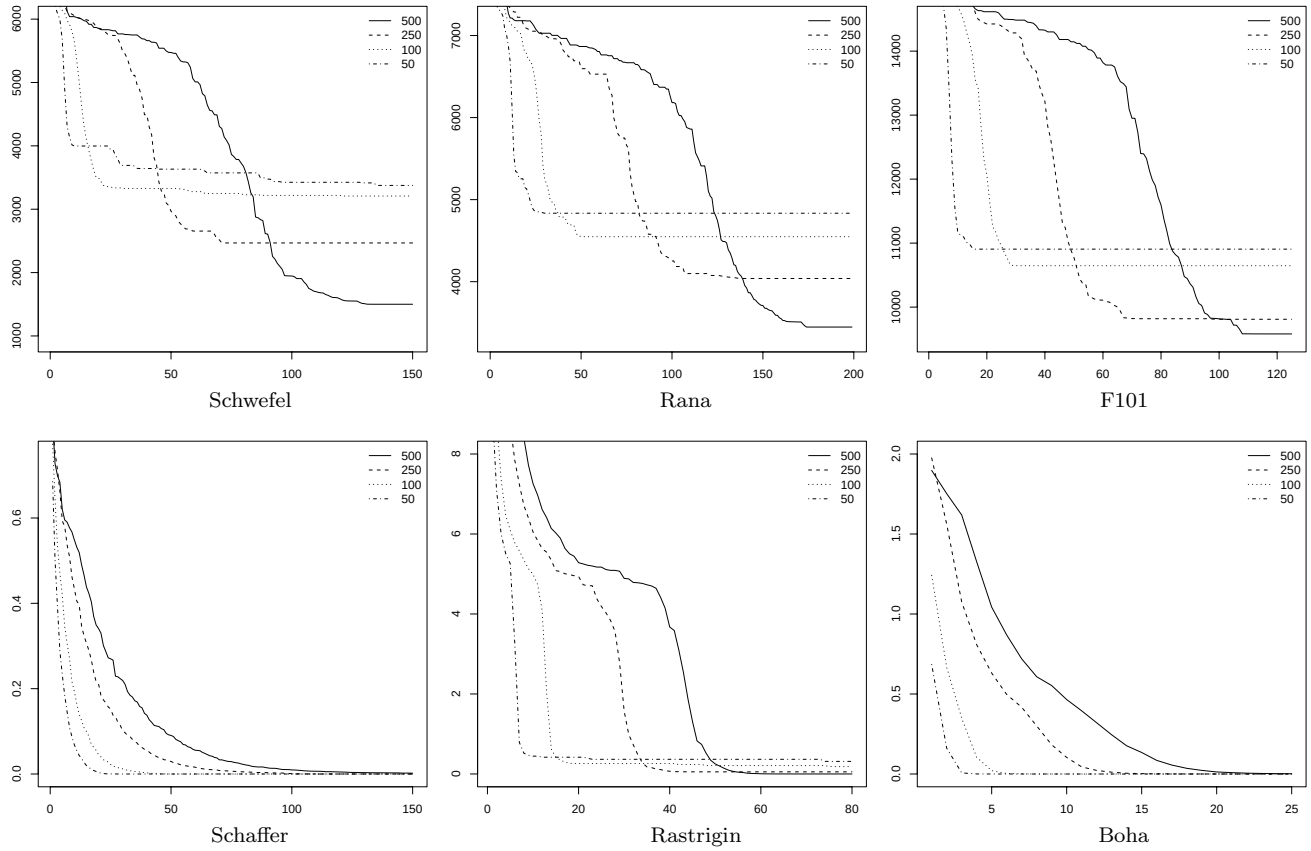

Figure 6: The median fitness vs. number of evaluations ($\times 1000$) for CMA-ES on several 20D benchmark test functions. The larger population size is the most effective on the *high dispersion* functions, which includes Schwefel, F101, and Rana. This is also true for the *low dispersion* and highly multimodal Rastrigin function. The *low dispersion* Shaffer and Bohachevsky functions are easily solved for all values of λ.

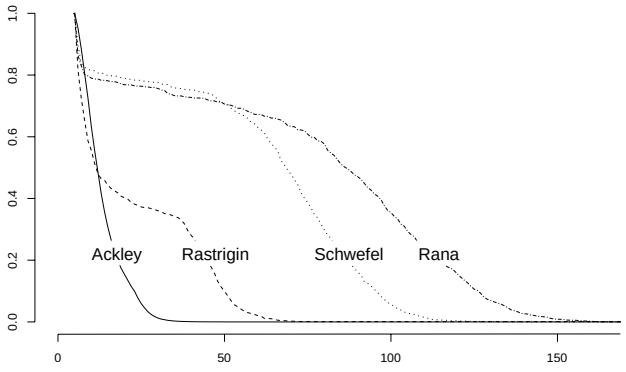

Figure 7: Scaled fitness vs. number of evaluations ($\times 1000$) for select functions.

means that CMA-ES requires almost $20,000$ evaluations just to find the boundaries of the problem.

One enticing potential remedy to this problem is to simply re-sample each infeasible point until it becomes feasible. But in high dimensions, sampling a point in the feasible space using a large σ is too improbable to be a practical solution. Unfortunately, decreasing σ will also decrease the effectiveness of CMA-ES.

A high percentage of infeasible offspring does not sufficiently account for the inefficient behavior of CMA-ES on high dispersion functions because this also occurs on *low* dispersion functions. There must be other factors that decrease efficiency. Recall that CMA-ES uses a distinct global step size, σ, to appropriately scale its covariance matrix distribution. Hansen and Ostermier state that this is necessary for two reasons [12]. First, the learning rate on the covariance matrix is too slow. Second, the optimal step length cannot be approximated well by the the covariance matrix alone. This is why the actual sample population created by the distribution defined by the covariance matrix is rescaled based on the current step size, σ. When search is making progress, the evolution path is longer than expected, and σ grows. On the sphere function, for example, σ will increase as CMA-ES approaches the minima during exploration. Once the algorithm begins to exploit a local optima, the evolution path is likely to be smaller, and σ will shrink. This creates a region of higher density during exploitation.

We looked at the values of σ as a function of time on the standard test functions described earlier. The step-size adaptation on the low dispersion functions behaved much like that of the sphere. This is not too surprising because we know that the underlying topology is unimodal. However, we found that on the higher dispersion functions, the step-

size grew rather large, on average between two and three times its initial value. At the same time, the actual distribution of offspring was decreasing. This has two important implications. First, the distribution of the covariance matrix is responsible for the decrease in actual sample distribution. This is a concern because Hansen *et al.* asserts that the distribution defined by the covariance matrix is sub-optimal in value and often too slow [12]. Second, if σ is steadily increasing, the cumulative path length is remaining above its expected length. This implies that the adaptive *cumulation* heuristic cannot decide when to stop exploring.

6. DISCUSSION AND CONCLUSION

We have defined a new metric, *dispersion* that quantifies the proximity of the best regions in the search space. On low dispersion functions, the best regions of the search space become more localized as the sample size increases (threshold decreases). The opposite is true for high dispersions functions: as the sample size increases, the best regions of the search space tend to become more disperse.

Several researchers have noticed that CMA-ES is inefficient and ineffective on multimodal functions where the underlying structure of the problem is unpredictable. Dispersion quantifies "underlying" topology. This allows us to separate functions into *high* and *low* categories and predict how efficient and effective CMA-ES will be. Future work will extend this by looking at how other evolutionary algorithms perform on high dispersion problems.

We have identified two reasons why CMA-ES is less efficient on *high dispersion* functions. First, many of the initial offspring are not in the feasible region. CMA-ES can waste up to 20% of its evaluations simply finding the boundaries of the problem. This is really a constraint handling problem, yet it is not clear how to address this issue in order to improve the efficiency of CMA-ES. Second, and probably more serious, we found that the step size adaptation mechanism does not work as expected on high dispersion functions. Instead, the distribution defined by the covariance matrix is primarily responsible for the decrease in the actual distributions that creates the offspring.

7. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0117209 and Sandia National Labs. We thank Nikolaus Hansen for providing us with his C++ implementation of CMA-ES.

8. REFERENCES

[1] B. Addis, M. Locatelli, and F. Schoen. Local optima smoothing for global optimization. *Optimization Methods and Software*, 20(4-5), 2005.

[2] A. Auger and N. Hansen. A restart cma evolution strategy with increasing population size. In *Proceedings of IEEE Congress of Evolutionary Computation*, 2005.

[3] 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.

[4] J. Doye, M. Miller, and D. Wales. Evolution of the potential energy surface with size for lennard-jones clusters. *Journal of Chemical Physics*, 111:8417, 1999.

[5] J. P. Doye. Physical perspectives on the global optimization of atomic clusters. In *Chapter in forthcoming Kluwer book Global optimization – select case studies*, 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] C. Flamm, I. L. Hofacker, P. F. Stadler, and M. T. Wolfinger. Barrier trees of degenerate landscapes. *Z. Phys. Chem.*, 216:155–173, 2002.

[8] K. Foreman, A. Phillips, J. Rosen, and K. Dill. Comparing search strategies for finding global optima on energy landscapes. *Journal of Computational Chemistry*, 20(14), 1999.

[9] J. Hallam and A. Prügel-Bennett. Large barrier trees for studying search. *IEEE Trans. Evolutionary Computation*, 9(4):385–397, 2005.

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

[11] N. Hansen, S. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 2003.

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

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

[14] A. Ostermeier, A. Gawelczyk, and N. Hansen. Step-size adaptation based on non-local use of selection information. In *PPSN*, pages 189–198. Springer, 1994.

[15] 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.

[16] R. Salomon. Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. *BioSystems*, 39:263–278, 1996.

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

[18] D. Whitley, S. B. Rana, J. Dzubera, and K. E. Mathias. Evaluating evolutionary algorithms. *Artificial Intelligence*, 85(1-2):245–276, 1996.