Biases in Particle Swarm Optimization

William M. Spears
Derek Green
Diana F. Spears
Computer Science Department
University of Wyoming
wspears@cs.uwyo.edu

April 11, 2010

Abstract

It is known that the most common versions of particle swarm optimization (PSO) algorithms are rotationally variant. It has also been pointed out that PSO algorithms can concentrate particles along paths parallel to the coordinate axes. In this paper we explicitly connect these two observations, by showing that the rotational variance is related to the concentration along lines parallel to the coordinate axes. We then clarify the nature of this connection. Based on this explicit connection we create fitness functions that are easy or hard for PSO to solve, depending on the rotation of the function.

1 Introduction

The popularity and variety of Particle Swarm Optimization algorithms has continued to grow at a rapid rate since the initial PSO algorithm was introduced in 1995 [3, 15, 16]. Recently, great strides have been made in understanding the theoretical underpinnings of the basic PSO algorithm (e.g., [12, 17]). However, there are still some behaviors exhibited by PSO that require further examination. For example, although it has also been pointed out that when running the traditional PSO algorithm, "most movement steps occurred parallel to one of the coordinate axes" [4], this behavior is not well explained theoretically. In this paper we examine this behavior and provide a theoretical explanation for why it occurs. Based on this explanation, we also show fitness landscapes in which the performance of PSO depends heavily on the rotation of the fitness function. Through these observations we hope to help users of PSO-based algorithms to better understand the effects of the biases inherent in PSO on their own particular problems.

1.1 The PSO Algorithm

The basic PSO algorithm [7] is usually described as follows. A swarm consists of N particles. Each particle i has a position at time t denoted by $\vec{X}_i(t) = (X_{i,1}(t), \ldots, X_{i,D}(t))$, where $\vec{X}_i(t)$ is a D-dimensional vector. Each particle i has a velocity $\vec{V}_i(t) = (V_{i,1}(t), \ldots, V_{i,D}(t))$, which is also a D-dimensional vector. The equations of motion are generally given as:

$$\vec{X}_i(t+1) = \vec{X}_i(t) + \vec{V}_i(t+1) \tag{1}$$

$$\vec{V}_i(t+1) = \omega\vec{V}_i(t) + c_1 r_1(\vec{P}_i - \vec{X}_i(t)) + c_2 r_2(\vec{G} - \vec{X}_i(t)) \tag{2}$$

$\vec{P}_i$ is the "personal best" position, or the position of best fitness ever encountered by particle i. $\vec{G}$ is the "global best" position ever found by all of the particles, or alternatively the best position ever seen within a neighborhood of particles. In this paper we will assume that all particles are neighbors. The best positions are updated when particles find positions with better fitness. The ω term, an "inertial coefficient" from 0 to 1, was introduced in [14]. The "learning rates" c_1 and c_2 are non-negative constants. Very often these are both set to 2.0. Finally, r_1 and r_2 are random numbers generated in the range of [0,1].

Looking again at equation (2), we point out an ambiguity which unfortunately continues to propagate throughout the literature. The ambiguity arises in the interpretation of the random numbers. In many papers it is not made clear when the random numbers are calculated. The random variables r_1 and r_2 in equation (2) may be interpreted as scalars or vectors. Figure 1 shows the two most common implementations seen in the literature. In both versions U(0,1) is a uniform random generator in the range of [0,1].

Version 1 is rotationally invariant, while Version 2 is not. James Kennedy, one of the creators of PSO, indicates that the second of the two versions is preferred, because it is considered to be more explorative [6]. Hence, the notation of Poli [12] is preferred:

$$\vec{V}_i(t+1) = \omega\vec{V}_i(t) + c_1\vec{r_1} \odot (\vec{P}_i - \vec{X}_i(t)) + c_2\vec{r_2} \odot (\vec{G} - \vec{X}_i(t)) \tag{3}$$

where $\odot$ represents component–wise multiplication.

In this paper we will show that updating the random numbers in the preferred way is the cause of the biased behavior. In other words, the rotational variance and coordinate axes bias of Version 2 are related. It is not our intention to suggest that PSO should not be used due to the bias, but rather that users of PSO should be aware of the bias, its cause, and how it might affect their particular needs.

1.2 Previous Analyses

An early formal analysis of PSO is given by [11]. In this analysis, the system is simplified by setting $\vec{P}_i = \vec{G}$, using a one-dimensional search space,

```
void pso1_one_step ()
{
  for (i = 1; i <= N; i++) {
    r1 = U(0,1);
    r2 = U(0,1);
    for (j = 1; j <= D; j++) {
      V[i][j] = w * V[i][j] + c1 * r1 * (P[i][j] - X[i][j]) + c2 * r2 * (G[j] - X[i][j]);
} } }

void pso2_one_step ()
{
  for (i = 1; i <= N; i++) {
    for (j = 1; j <= D; j++) {
      r1 = U(0,1);
      r2 = U(0,1);
      V[i][j] = w * V[i][j] + c1 * r1 * (P[i][j] - X[i][j]) + c2 * r2 * (G[j] - X[i][j]);
} } }
```

Figure 1: Two Versions of PSO

holding the random variables constant and examining a single particle. This one-dimensional, one particle analysis is examined further in [10]. In [1], Clerc notes the fact that PSO in its basic form, and hence many of its derivatives, is not rotationally invariant. The fact that PSO is rotationally variant is again noted in [18] and it is also mentioned that both versions of PSO are translationally invariant. Finally, in [4], it is suggested that the rotational variance of PSO causes the "bias to the coordinate axes".

1.3 Current Analysis

In this paper we expand on the work by Janson and Middendorf [4] and show that, for the standard PSO, the particular manner in which the random variables are applied causes both the rotational variance and the bias towards the coordinate axes. Finally, we create fitness landscapes that demonstrate some of the features that can cause difficulties for PSO, depending on the rotational angle of the system.

Throughout the paper our discussion will focus on the bearing of the velocity of the particles in the PSO as well as the change in velocity $\Delta\vec{V}_i(t) = c_1\vec{r_1} \odot (\vec{P}_i - \vec{X}_i(t)) + c_2\vec{r_2} \odot (\vec{G} - \vec{X}_i(t))$ The motivation for the latter focus stems from $\vec{F} = m\vec{a}$, where the acceleration of the particles is synonymous with calculating the force on the particles. Specifically, we will examine the expected bearing of $\Delta\vec{V}_i(t)$ to see if there is some angular bias that would explain the preference for the bearings associated with the coordinate axes.

2 Bias Parallel to the Coordinate Axes

In this section we illustrate the behavior of PSO on a simple radially symmetric fitness function. The fitness is the distance of a particle from the origin (the PSO is minimizing). Figure 2 shows snapshots for both the two-dimensional and three-dimensional functions. The particles are initially positioned using a Gaussian $\mathcal{N}(0, \sigma^2)$ with $\sigma = 1000.0$, which ensures a radially symmetric distribution. There are 100,000 particles and we used parameter settings $\omega = 1.0$, $c_1 = 2.0$, and $c_2 = 2.0$. PSO algorithms generally also include a maximum velocity V_{max}, which we set to 10.0.

What is interesting about the behavior on both problems is the concentration of particles along the axes, despite the radially symmetric fitness function and radially symmetric initial distribution of particles. There is no obvious reason for this concentration, and the PSO is behaving as if there are some additional structures in the functions that are aligned with the coordinate axes.

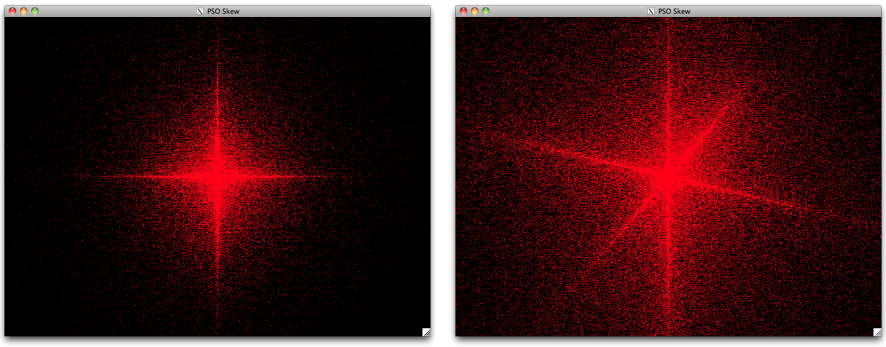

Figure 2: PSO on the distance function in 2D (left), and in 3D (right). Note the concentration of particles on the coordinate axes.

Naturally, a population size of 100,000 is unusually high for most PSO applications. In the next section we quantify the behavior more precisely, and show that it is not simply a "large population" effect. We will also show that is not a "small dimension" effect. Finally, we demonstrate that the settings of the PSO parameters have little effect on this axial concentration.

3 Analysis of Coordinate Axes Bias in PSO

In this section we examine the coordinate axis bias of PSO, by monitoring the bearing of the velocity of each particle for a number of time steps. For the first experiment we re-ran the PSO on the three-dimensional distance function, using the same Gaussian initialization of particles. However, the population size was lowered to a more standard size of 20. We allowed each PSO parameter to have one of two settings: $\omega \in (0.729, 1.0)$, $c_1 = c_2 \in (1.49, 2.0)$ and $V_{max} \in (2.0, 10.0)$.

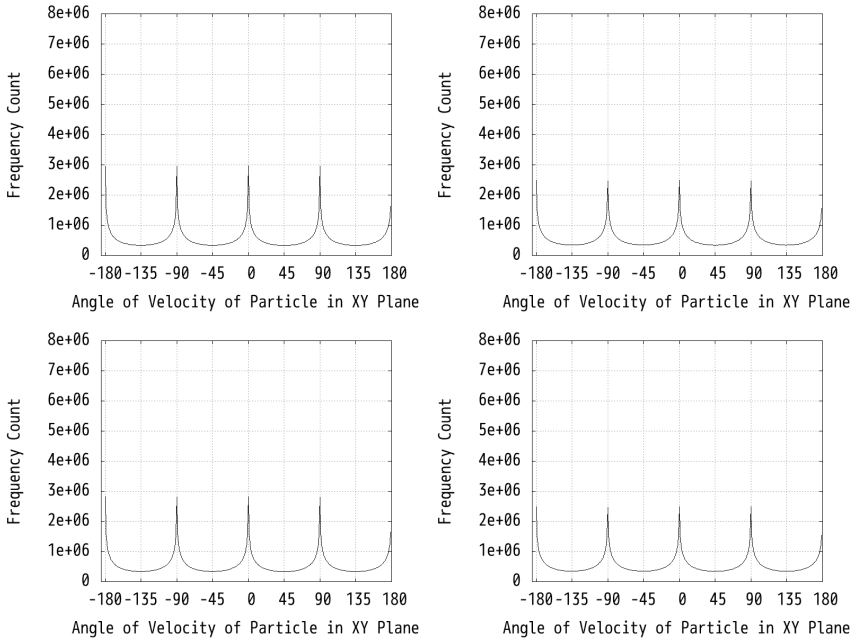

Figure 3: Angular bias with the 3D distance function in the XY plane. The x-axis represents the angle bins. The y-axis represents the frequency counts for the bins. On the upper left $c_1 = c_2 = 1.49$, and $\omega = 0.729$. On the upper right $c_1 = c_2 = 1.49$, and $\omega = 1.0$. On the lower left $c_1 = c_2 = 2.0$, and $\omega = 0.729$. On the lower right $c_1 = c_2 = 2.0$, and $\omega = 1.0$.

This yielded a total of eight different sets of parameters and experiments. The values of $\omega = 0.729$ and $c_1 = c_2 = 1.49$ are noted in [2].

The algorithm was run for 1000 steps. Each time step we computed the bearing of the velocity for each particle in the XY and YZ planes. We maintained 360 bins for each plane, ranging from $-180°$ to $180°$ degrees. Each bin maintained a count of how many times that bearing was observed (rounding real values appropriately). All eight experiments yielded very similar results. Each experiment was run over 10,000 independent trials, to provide good sampling for the bins.

Figure 3 presents the four graphs where $V_{max} = 2.0$, in the XY plane. The horizontal axis is the bearing in degrees, while the vertical axis is the number of times that bearing occurred. We see a strong bias parallel to the coordinate axes. The values of the standard PSO parameters (c_1, c_2, and ω) have little effect on this bias, although some effect is noticeable. Through this paper the results are similar when $V_{max} = 10.0$, so we omit those graphs.

Figure 4 shows the results for the YZ plane. Not surprisingly, Figures 3 and 4 are identical, since there is no preference for any particular axis in PSO.

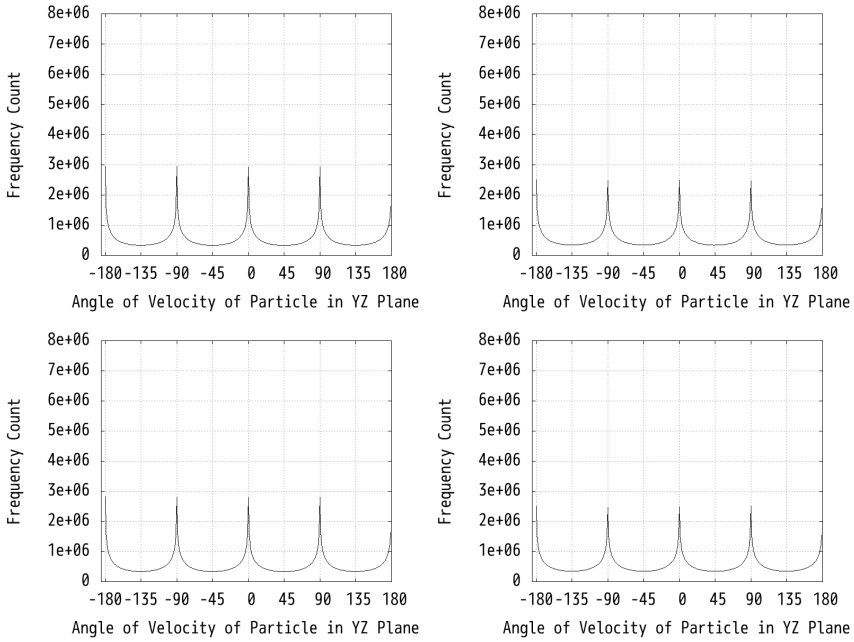

Figure 4: Angular bias with the 3D distance function in the YZ plane. The x-axis represents the angle bins. The y-axis represents the frequency counts for the bins. On the upper left $c_1 = c_2 = 1.49$, and $\omega = 0.729$. On the upper right $c_1 = c_2 = 1.49$, and $\omega = 1.0$. On the lower left $c_1 = c_2 = 2.0$, and $\omega = 0.729$. On the lower right $c_1 = c_2 = 2.0$, and $\omega = 1.0$.

This is in contrast to the results shown in [4], which shows a mild preference for the x-axis (we suspect there was an error in their code).

Finally, we tried the same experiment with the distance function in 20 dimensions. Each time we analyze the bearing of the velocity of a particle, we uniformly randomly choose two different axes (i.e., a random plane). Figure 5 shows the results. Again, PSO exhibits a strong preference for bearings aligned with the global coordinate axes despite the radial symmetry of the function and the initial conditions. With 20 dimensions the different parameter settings have almost no effect on the bias.

In summary, we have shown that, in the presence of radial symmetry in both the fitness function and the initial conditions, the particles in PSO are drawn strongly towards bearings parallel to the coordinate axes. This result is consistent with that of [4]. The behavior is similar regardless of the number of particles, the values of the standard PSO parameters, and the number of dimensions. Hence we can consider it to be a "core" behavior.

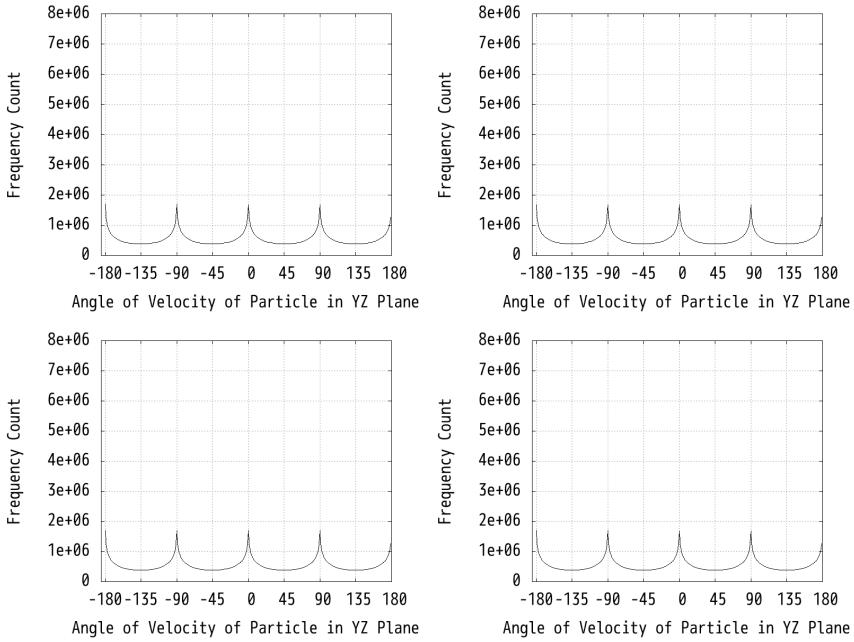

Figure 5: Angular bias with the 20D distance function. The x-axis represents the angle bins. The y-axis represents the frequency counts for the bins. On the upper left $c_1 = c_2 = 1.49$, and $\omega = 0.729$. On the upper right $c_1 = c_2 = 1.49$, and $\omega = 1.0$. On the lower left $c_1 = c_2 = 2.0$, and $\omega = 0.729$. On the lower right $c_1 = c_2 = 2.0$, and $\omega = 1.0$.

4 Analysis of Particle Trajectories

The previous section indicated that particles often travel along bearings parallel to the coordinate axes. But do they simply stay along one bearing, or switch bearings? In this section we examine the trajectories of individual particles for the two-dimensional distance function. One trial is run, for 300 steps of the PSO algorithm. $V_{max} = 2.0$ and, as before, we examine the four combinations of values for $c_1 = c_2$ and ω. The random seed for each of the four experiments is the same, so the initial location of the particles is the same for each experiment. Rather than follow the trajectory of only one particle (for all four experiments), we follow the trajectory of a different particle for each experiment, to demonstrate generality of the behavior.

Figure 6 shows the results. All four graphs show similar behavior. The velocity of a particle tends to remain parallel with a coordinate axis, although there is considerable noise. When there are changes, the velocity often switches to a direction approximately parallel to the other axis. In the upper right graph the particle switches (through a sequence of rather large changes) from

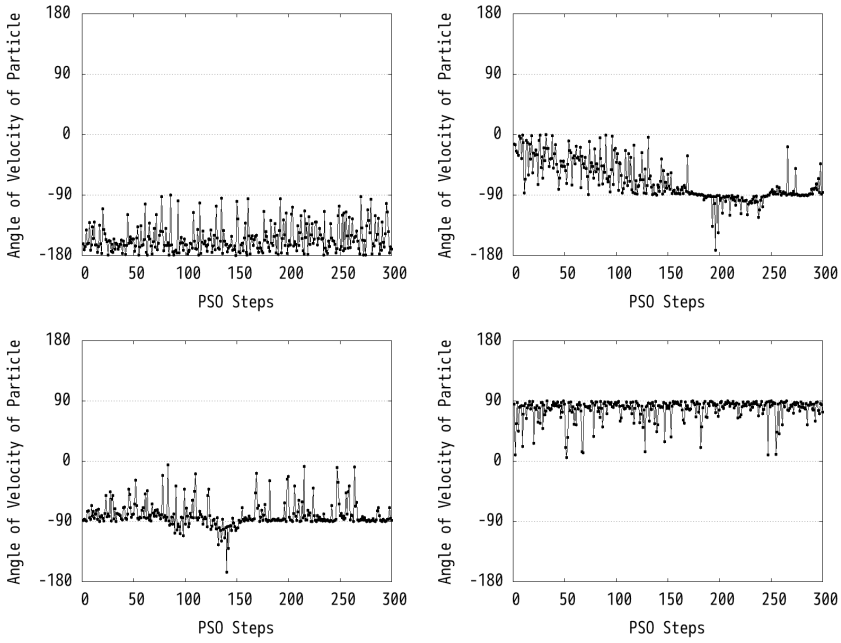

Figure 6: The bearing of individual particles on the distance function, over the first 300 steps of the algorithm. The x-axis represents the step number. The y-axis represents the angular bearing of a particle. On the upper left $c_1 = c_2 = 1.49$, and $\omega = 0.729$. On the upper right $c_1 = c_2 = 1.49$, and $\omega = 1.0$. On the lower left $c_1 = c_2 = 2.0$, and $\omega = 0.729$. On the lower right $c_1 = c_2 = 2.0$, and $\omega = 1.0$.

an alignment parallel to the x-axis to an alignment parallel to the y-axis.

Again, there is no obvious reason for there to be *any* preferred alignments when using the distance function for fitness. To see if this behavior holds on other fitness functions, we chose the Ackley and Griewank functions. Neither of these functions have any obvious biases parallel to the coordinate axes. To avoid any potential "origin-seeking bias" [9], both functions have their optima at (500,500).

Figures 7 and 8 show the trajectories on the Ackley and Griewank functions. The graphs are qualitatively similar to those shown for the distance function. Again, the trajectories of particles tend to remain parallel with coordinate axes, although there is quite a bit of noise.

Given these results, and the fact that they appear even when the fitness function and initialization are both radially symmetric, there must be a core PSO bias that favors trajectories that are parallel to coordinate axes. We provide a theoretical explanation in the next section.

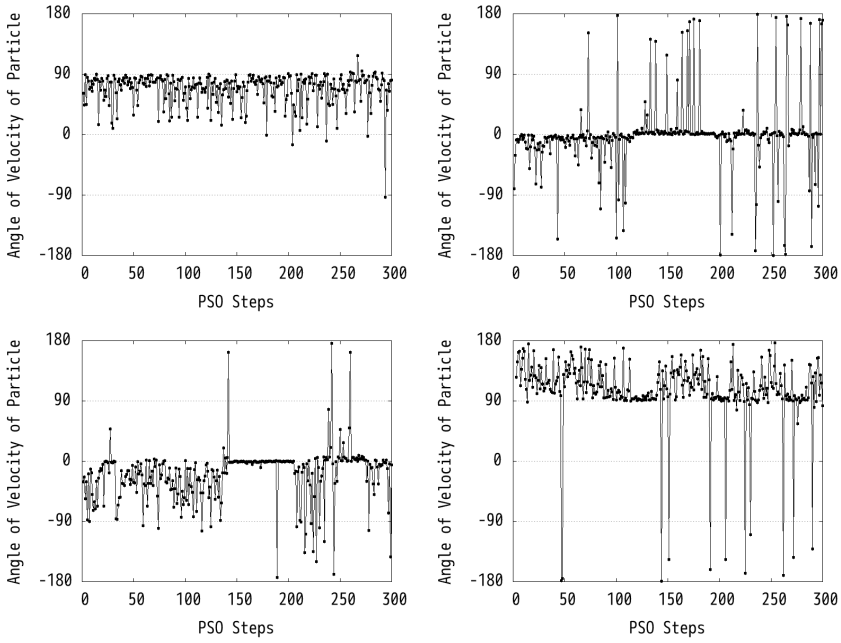

Figure 7: The bearing of individual particles on Ackley's function, over the first 300 steps of the algorithm. The x-axis represents the step number. The y-axis represents the angular bearing of a particle. On the upper left $c_1 = c_2 = 1.49$, and $\omega = 0.729$. On the upper right $c_1 = c_2 = 1.49$, and $\omega = 1.0$. On the lower left $c_1 = c_2 = 2.0$, and $\omega = 0.729$. On the lower right $c_1 = c_2 = 2.0$, and $\omega = 1.0$.

5 Theoretical Explanation

It is clear that the PSO algorithm itself must provide a bias towards bearings parallel to the coordinate axes. This has to occur where the velocity is updated. We ignore the first term, $\omega\vec{V}_i(t)$, because this does not change the bearing of the velocity vector. However, $\Delta\vec{V}_i(t) = c_1\vec{r_1} \odot (\vec{P}_i - \vec{X}_i(t)) + c_2\vec{r_2} \odot (\vec{G} - \vec{X}_i(t))$ can change the bearing. This can be thought of as a Newtonian force $\vec{F} = m\vec{a}$, where the acceleration of the particles is synonymous with calculating the force on the particles. Specifically, we will examine the expected bearing of $\Delta\vec{V}_i(t)$ to see if there is some angular bias that would explain the preference for the bearings associated with the coordinate axes.

Since we have shown that the bias holds regardless of the number of dimensions, we will focus on an analysis in two dimensions. Consider again the bearing of the change in velocity of each particle (we omit the time variable to simplify notation):

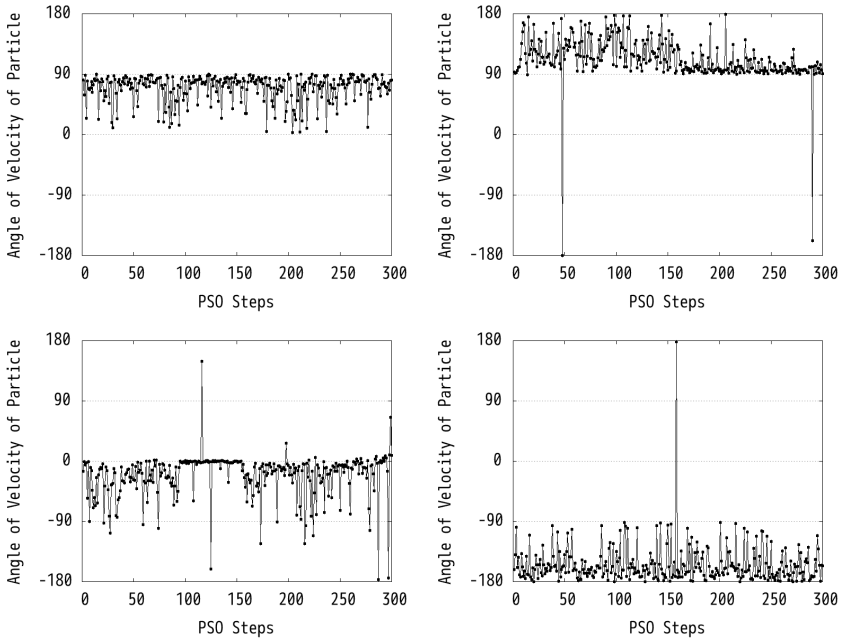

Figure 8: The bearing of individual particles on the Griewank function, over the first 300 steps of the algorithm. The x-axis represents the step number. The y-axis represents the angular bearing of a particle. On the upper left $c_1 = c_2 = 1.49$, and $\omega = 0.729$. On the upper right $c_1 = c_2 = 1.49$, and $\omega = 1.0$. On the lower left $c_1 = c_2 = 2.0$, and $\omega = 0.729$. On the lower right $c_1 = c_2 = 2.0$, and $\omega = 1.0$.

$$\Phi = \arctan\left(\frac{c_1 r_{1,2}(P_{i,2} - X_{i,2}) + c_2 r_{2,2}(G_2 - X_{i,2})}{c_1 r_{1,1}(P_{i,1} - X_{i,1}) + c_2 r_{2,1}(G_1 - X_{i,1})}\right) \tag{4}$$

In the above equation $r_{i,j}$ represents the jth component of $\vec{r_i}$. Hence, each is implemented via a separate function call to a U(0,1) random number generator.

Also, in most implementations of PSO, $c_1 = c_2$, so:

$$\Phi = \arctan\left(\frac{r_{1,2}(P_{i,2} - X_{i,2}) + r_{2,2}(G_2 - X_{i,2})}{r_{1,1}(P_{i,1} - X_{i,1}) + r_{2,1}(G_1 - X_{i,1})}\right) \tag{5}$$

A baseline bearing can be obtained by assuming that there are no random numbers generated:

$$\Theta = \arctan\left(\frac{(P_{i,2} - X_{i,2}) + (G_2 - X_{i,2})}{(P_{i,1} - X_{i,1}) + (G_1 - X_{i,1})}\right) \tag{6}$$

This bearing baseline corresponds to simple vector addition, with the change in velocity of a particle determined by the position of the personal and global

best positions. We will examine the expectation $E[\Phi - \Theta] = E[\Phi] - \Theta$ to see if it is equal to zero, and if it varies with Θ. We refer to $E[\Phi] - \Theta$ as the "angular bias" of PSO. If the expectation is not zero, angular biases are present. We consider various cases:

5.1 Case 1

In the first case $\vec{X}_i = \vec{G} \neq \vec{P}_i$. This situation is not possible, since if a particle is at the global best it must also be at it's personal best.

5.2 Case 2

For the second case $\vec{X}_i = \vec{P}_i = \vec{G}$. This can only occur with the global best particle. In this situation $\Delta\vec{V}_i = \vec{0}$ and there is no change in bearing.

5.3 Case 3

For the third case $\vec{X}_i = \vec{P}_i \neq \vec{G}$. This is often the situation for one of the non-global best particles. In this situation:

$$\Phi = \arctan\left(\frac{r_{1,2}(P_{i,2} - X_{i,2}) + r_{2,2}(G_2 - X_{i,2})}{r_{1,1}(P_{i,1} - X_{i,1}) + r_{2,1}(G_1 - X_{i,1})}\right) \tag{7}$$

Since $\vec{X}_i = \vec{P}_i$ we can simplify:

$$\Phi = \arctan\left(\frac{r_{2,2}(G_2 - X_{i,2})}{r_{2,1}(G_1 - X_{i,1})}\right) \tag{8}$$

The baseline bearing is:

$$\Theta = \arctan\left(\frac{(G_2 - X_{i,2})}{(G_1 - X_{i,1})}\right) \tag{9}$$

We computed the expectation $E[\Phi] - \Theta$ and its standard deviation via simulation. Two points (X and G) were distributed via our $\mathcal{N}(0, 1000^2)$ distribution, and Θ was computed. Then Φ was computed after two calls to U(0,1). This was repeated four billion times. Figure 9 shows the results. For both graphs, the horizontal axis is the baseline bearing Θ in degrees. The left graph gives the measured difference $E[\Phi] - \Theta$ in degrees. The right graph gives the standard deviation of that difference in degrees.

For the purposes of verification of our simulation, we were able to calculate the closed form of $E[\Phi] - \Theta$. Note that:

$$\Phi = \arctan\left(\tan(\Theta)\left[\frac{r_{2,2}}{r_{2,1}}\right]\right) \tag{10}$$

Let u denote $r_{2,2}$ and v denote $r_{2,1}$. Then, due to the independence of the two random variables $E[\Phi]$ is expressed as the integral:

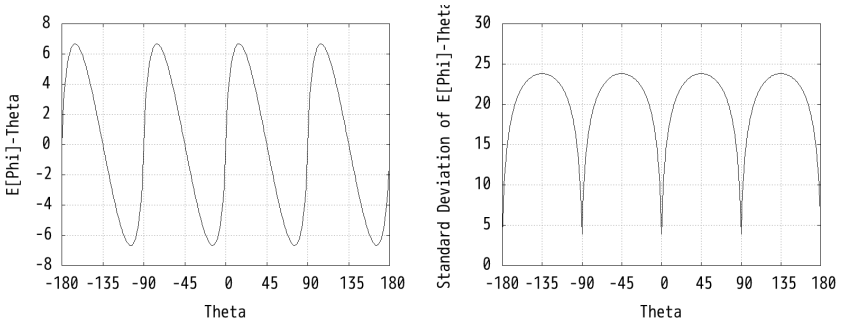

Figure 9: The expected angular bias (left) and the standard deviation (right) of PSO on Case 3, determined via simulation.

$$E[\Phi] = \int_0^1 \int_0^1 \arctan\left(\tan(\Theta)\frac{u}{v}\right) du\ dv \tag{11}$$

With some manipulation it can be shown that $E[\Phi] - \Theta = 0$ when $\Theta = k\pi$, where $k \in \mathbb{Z}$. Otherwise:

$$E[\Phi] - \Theta = \frac{\log((1+\tan^2\theta)(\tan\theta - \cot\theta)) - 2\tan\theta\log(\tan\theta)}{4} \tag{12}$$

The empirical simulation results have an average absolute error of only 0.006 degrees from theory – hence this confirms the correctness of the simulation.

Let us examine Figure 9 more carefully. Again, the left graph shows that the difference $E[\Phi]-\Theta$. This is zero when $\Theta = k\pi/4$, where $k \in \mathbb{Z}$. The difference is positive between 0° and 45°, and negative between 45° and 90°. This indicates a bias that tends to push particles *away* from directions parallel to the coordinate axes and *towards* directions parallel with the diagonals, which is opposite that which we expected! However, the right graph indicates that the variation in this difference is very low along directions parallel with the coordinate axes and highest along directions parallel with the diagonals.

Hence, an interesting picture emerges of two contrasting biases, *skew* and *spread*. Skew is the difference $E[\Phi]-\Theta$. Spread is the standard deviation of that difference. If a particle is moving approximately parallel to the coordinate axes, it moves away towards the diagonals (skew). However, if a particle is moving approximately parallel to the diagonals, the high variance (spread) is likely to change that direction quickly. Hence, directions parallel to the coordinate axes act as unstable equilibria (or weak "basins of attraction") for the particles. These basins are weak because the particles tend to change direction towards bearings parallel with the diagonals. But the bearings parallel with the diagonals are even more unstable, due to the very high variance. The combination of both biases creates a situation where bearing parallel to the coordinate axes are preferred, but with a lot of noise in the bearings. This is precisely what we saw in the experiments earlier in the paper.

Note that this discussion is independent of the fitness function. However, from the perspective of function optimization, certain functions will match the biases well, while others will not. We will test our understanding by creating fitness functions that are hard for PSO. However, first we must confirm that these biases are the same for the remaining two cases.

5.4 Case 4

For the fourth case $\vec{X}_i \neq \vec{P}_i = \vec{G}$. This is the behavior of the global best particle when it has moved past the best position. In this situation:

$$\Phi = \arctan\left(\frac{r_{1,2}(P_{i,2} - X_{i,2}) + r_{2,2}(G_2 - X_{i,2})}{r_{1,1}(P_{i,1} - X_{i,1}) + r_{2,1}(G_1 - X_{i,1})}\right) \tag{13}$$

Since $\vec{P}_i = \vec{G}$ we can simplify:

$$\Phi = \arctan\left(\frac{(r_{1,2} + r_{2,2})(P_{i,2} - X_{i,2})}{(r_{1,1} + r_{2,1})(P_{i,1} - X_{i,1})}\right) \tag{14}$$

The baseline bearing is:

$$\Theta = \arctan\left(\frac{(P_{i,2} - X_{i,2})}{(P_{i,1} - X_{i,1})}\right) \tag{15}$$

Again, we computed the expectation $E[\Phi] - \Theta$ and its standard deviation via simulation. Two points (X and P) were distributed via our $\mathcal{N}(0, 1000^2)$ distribution, and Θ was computed. Then Φ was computed after four calls to U(0,1). This was repeated four billion times. Figure 10 shows the results. For both graphs, the horizontal axis is the baseline bearing Θ in degrees. The left graph gives the measured difference $E[\Phi] - \Theta$ in degrees. The right graph gives the standard deviation of that difference in degrees.

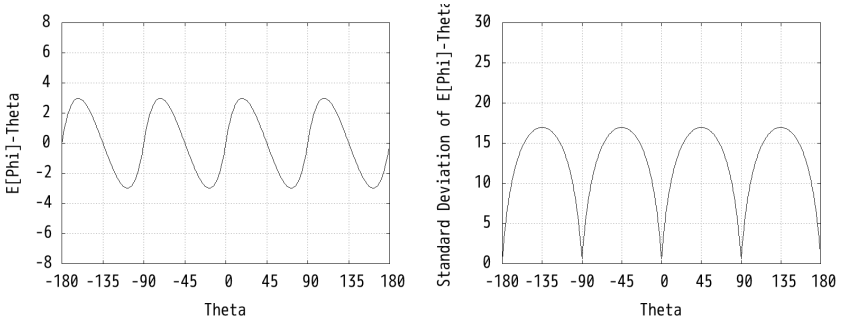

Figure 10: The expected angular bias (left) and the standard deviation (right) of PSO on Case 4, determined via simulation.

For the purposes of verification of our simulation, we were able to calculate the closed form of $E[\Phi] - \Theta$. Since it involves a quadruple integral, it is much

more difficult to solve than Case 3 and the details are shown in the Appendix. In this case the empirical simulation results have an average absolute error of only 0.003 degrees from theory – once again confirming the correctness of the simulation.

Figure 10 is very similar to Figure 9 and there is no qualitative change in our interpretation of these graphs.

5.5 Case 5

For the fifth case $\vec{X}_i \neq \vec{P}_i \neq \vec{G}$. This is also a situation encountered by non-global best particles. Φ can not be simplified:

$$\Phi = \arctan\left(\frac{r_{1,2}(P_{i,2} - X_{i,2}) + r_{2,2}(G_2 - X_{i,2})}{r_{1,1}(P_{i,1} - X_{i,1}) + r_{2,1}(G_1 - X_{i,1})}\right) \tag{16}$$

The baseline bearing is:

$$\Theta = \arctan\left(\frac{(P_{i,2} - X_{i,2}) + (G_2 - X_{i,2})}{(P_{i,1} - X_{i,1}) + (G_1 - X_{i,1})}\right) \tag{17}$$

The simulation results are shown in Figure 11. Again, for both graphs, the horizontal axis is the baseline bearing in degrees. The left graph gives the measured difference $E[\Phi] - \Theta$ in degrees. The right graph gives the standard deviation of that difference in degrees.

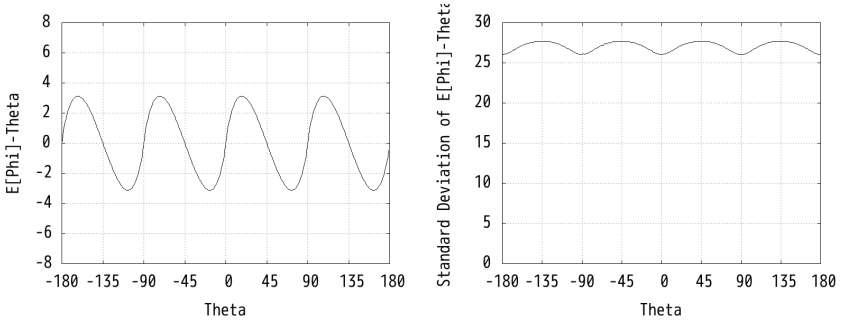

Figure 11: The expected angular bias (left) and the standard deviation (right) of PSO on Case 5, determined via simulation.

Once again the results are similar to those generated before, other than the standard deviation being higher. This is reasonable since we have the most degrees of freedom in Case 5.

In summary, the theoretical analysis indicates that there is an angular bias in the core PSO algorithm. The bias consists of two parts. The first part, skew, pushes particles towards bearings parallel with the diagonals. However, the second part, spread, indicates that diagonal directions are highly unstable. The combination of the two parts creates a PSO bias that favors particle bearings that are aligned with the coordinate axes.

6 Demonstration of the Skew and Spread Biases

In this section we demonstrate the skew and spread bias that was described in Section 5. We run PSO on the two-dimensional distance function again. Since we are minimizing, the optimum is at (0,0). A population of size 20 is used, $c_1 = c_2 = 2.0$, $\omega = 1.0$, and $V_{max} = 10.0$. A tight Gaussian cluster of particles is initialized at (300,0) with $\sigma = 2.0$. The PSO is run for 30 steps, to focus on the behavior before the optimum is found. The PSO is run for 5,000 trials and the trajectories of all the particles are drawn. This is shown in the top left picture of Figure 12. Note that the particles eventually home in on the origin (the center of the picture), but the spread bias is very clear.

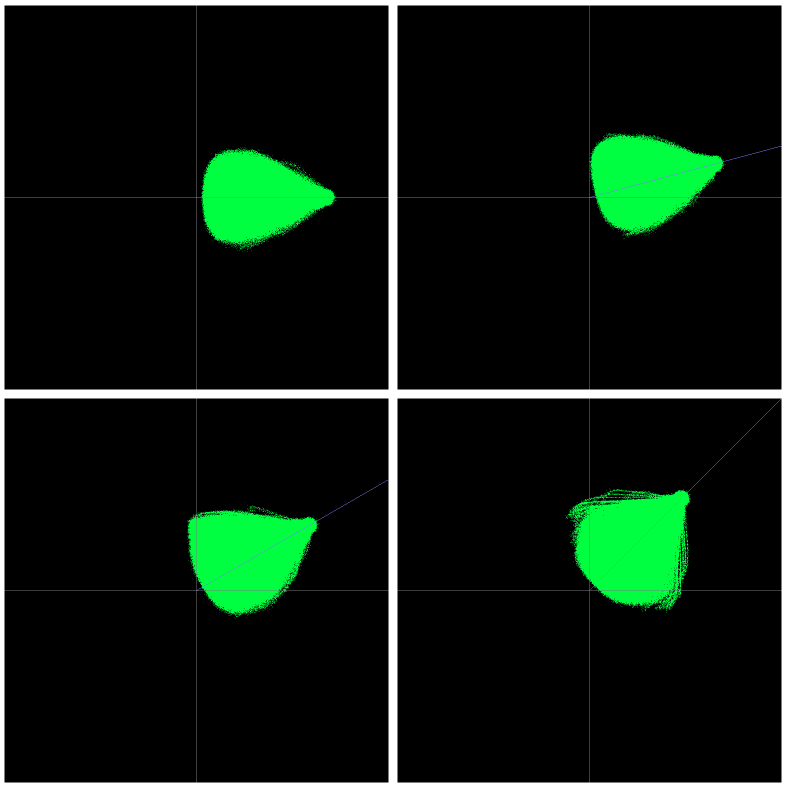

Figure 12: Skew and Spread of PSO.

Then the experiment is run again, but with the initial locations of the particles rotated 15° counterclockwise around the origin. It is important to note that we use the same random seed as before, so the random number sequence is identical. If the algorithm were rotationally invariant, the picture would look the same as before (other than the rotation). But since it is not, the image

is different (upper right). Note that the spread is increasing, and that more particles are skewed towards the 45° bearing.

The experiment was run again, with a rotation of 30°. Both the spread and skew biases are even more acute (lower left). Finally, in the lower right picture the angle of rotation is 45°. Now the skew bias is gone, but the spread bias is the greatest.

The only aspect that changed in the above experiments was the rotation of the global coordinate system. Since PSO is rotationally variant the behavior and performance of PSO depends on the angle of rotation of the system. We can see this performance difference more clearly in Figure 13. The performance that is monitored is the best fitness seen by 30 steps. For this experiment 200,000 trials were run for each angle of rotation. It is apparent that the best performance occurs when the system is rotated by −180°, −90°, 0°, 90°, and 180°. This is due to the lower spread and lack of skewness. As the system is rotated, and spread and skewness increase, performance becomes worse. However, at −135°, −45°, 45°, and 135°, performance becomes slightly better again. It is possible that the lack of skewness at 45° is causing this improvement. However, there is also another possibility, which we discuss next.

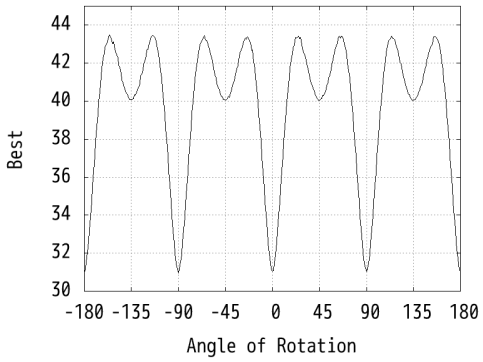

Figure 13: Performance of PSO as the system is rotated.

6.1 Magnitude Bias

To explain why the performance might improve at −135°, −45°, 45°, and 135°, we considered the expected magnitude of the change in velocity:

$$E\left[||c_1\vec{r_1} \odot (\vec{P_i} - \vec{X_i}(t)) + c_2\vec{r_2} \odot (\vec{G} - \vec{X_i}(t))||\right] \tag{18}$$

Assuming $c_1 = c_2 = c$:

$$cE\left[||\vec{r_1} \odot (\vec{P_i} - \vec{X_i}(t)) + \vec{r_2} \odot (\vec{G} - \vec{X_i}(t))||\right] \tag{19}$$

As before, we computed this expectation via simulation, for two dimensions. However, in this case the radius of the distribution of the initial particles plays an important role. For these experiments we used a uniform distribution within a circle of radius 10.0. The results are displayed for Cases 2, 3, and 5, in Figure 14. One million samples were generated for each of the three cases.

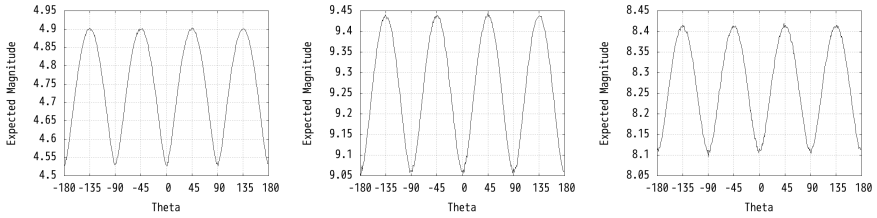

Figure 14: PSO magnitude bias for Case 3 (left), Case 4 (middle), and Case 5 (right).

As can be seen, there is a *magnitude* bias that depends on baseline angle Θ. The expected magnitude of the change in velocity vector is greater along bearings parallel with the diagonals at $-135°$, $-45°$, $45°$, and $135°$. Hence, on average, the particles move somewhat faster when traveling along those bearings. This appears to at least partially explain the mild improvement in the performance at $-135°$, $-45°$, $45°$, and $135°$ in Figure 13.

It should be noted that if the initialization radius is multiplied by a factor c, the results are also multiplied by that factor. So, what is important is the ratio of the expected maximum magnitude to the expected minimum. This ratio is approximately 1.083 for Case 3, 1.043 for Case 4, and 1.04 for Case 5.

The relative importance of the magnitude bias in comparison to skew and spread is unclear. The remainder of this paper will focus on creating fitness functions that will demonstrate how skew and spread can help or hinder PSO performance, depending on the angle of rotation. A further investigation of the magnitude bias will occur in future work.

7 Results with an Ellipse

Given the skew and spread bias, we decided to test the performance of PSO on a fitness function that is extremely radially asymmetric, namely, a two-dimensional ellipse. We also use the more conventional initialization of particles using a uniform distribution U(-1000,1000) for both the x and y location. Again we used a population size of 20, $c_1 = c_2 = 2.0$, and $V_{max} = 10.0$.

We used a simple hill-climber to adapt the semi-major and semi-minor axes of the ellipse, to maximize the difference between the best fitness at rotation angles $45°$ and $0°$ (see Figure 15).

The results are shown in Figure 16, averaged over 20,000 trials. The left graph shows the best performance for all rotation angles, after 500 steps. The

```
double FitnessEllipse(const double *position, const int dimensions) {
  double semiMajor = 0.000076;
  double semiMinor = 0.000001;
  double xt = position[0] / semiMajor;
  double yt = position[1] / semiMinor;
  return sqrt(xt * xt + yt * yt);
}
```

Figure 15: Ellipse Fitness Function

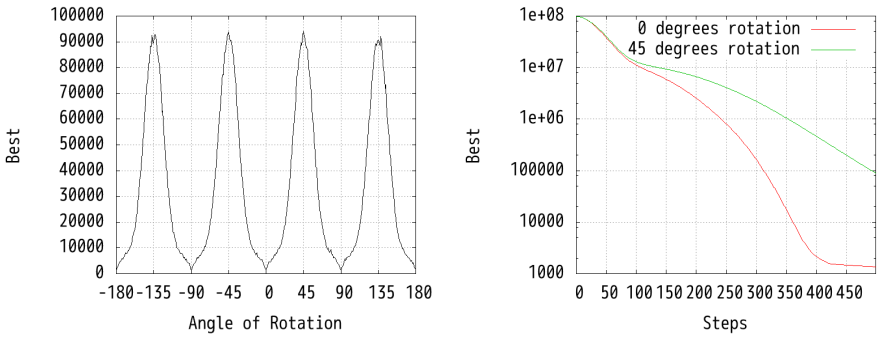

Figure 16: Performance of PSO as the system is rotated, for a very eccentric ellipse.

right graph shows the performance curves for both 0° and 45° rotations, over all 500 steps (note that a logarithmic scale is used for performance). The performance is approximately 70 times worse when the ellipse is rotated 45°.

8 Results with an Ellipse with a Mineshaft

Finally, we added a deep cylindrical mineshaft to the elliptical fitness function. The hill-climber adapted the semi-major and semi-minor axes, as well as the location and diameter of the mineshaft. The fitness of the mineshaft is -10,000. The goal was to maximize the difference between the probability of finding the mineshaft, at rotation angles 45° and 0°. Figure 17 shows the final fitness function.

```
double FitnessEllipseWithMineshaft(const double *position, const int dimensions) {
  double mineShaftX = 8.0;
  double mineShaftY = 0.0;
  double radius = 1.0;
  double delX = *(position + 0) − mineShaftX;
  double delY = *(position + 1) − mineShaftY;
  double dist = sqrt(delX * delX + delY * delY);
  if (dist <= radius) {
    fitness = −10000.0;// found the shaft!                                    10
  }
  else {
    double semiMajor = 1.0;
    double semiMinor = 0.0001;
    double xt = position[0] / semiMajor;
    double yt = position[1] / semiMinor;
    fitness = sqrt(xt * xt + yt * yt);
  }
  return fitness;
}                                                                            20
```

Figure 17: Ellipse with Mine Shaft Fitness Function

The results are shown in Figure 18, averaged over 20,000 trials. The left graph shows the best performance for all rotation angles, after 500 steps. This graph shows that the ellipse must be rotated to be aligned exactly with the coordinate axes in order for the optimum to be found (roughly 98% of the time). Even with a 1° rotation, the optimum is not found. The right graph shows the performance curves for both 0° and 45° rotations, over all 500 steps. It is clear that the PSO is making virtually no improvement when the system is rotated 45°. However, at 0° rotation the PSO performance improvement is quite smooth.

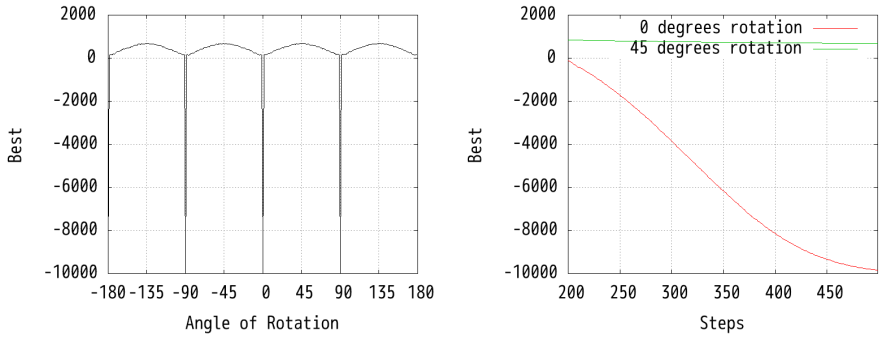

Figure 18: Performance of PSO as the system is rotated, for a very eccentric ellipse with a mineshaft.

9 Summary

In this paper we investigated the claim made by Janson and Middendorf [4] that when running the traditional PSO algorithm, "most movement steps occurred parallel to one of the coordinate axes". This claim is shown to be true. We then theoretically analyzed the dynamics of PSO by looking at its angular bias. This bias is composed of both skew and spread components – the two components combined produce the behavior that the bearings of particles tend to be parallel with coordinate axes. A third bias, called the magnitude bias, is also identified. All of these biases are manifestations of the rotational variance of standard PSO. These biases exist despite changes in standard parameter settings, the number of particles, and the number of dimensions.

We use our understanding of these biases to create fitness functions that can be easy or hard for PSO, depending on the angle of rotation. These functions are highly eccentric ellipses that create ridges (or troughs) in the fitness landscape. Our results indicate that PSO will have difficulty following ridges (troughs) that are not aligned with the coordinate axes. This occurs because both high spread and skewness tend to push particles off the ridges. This is important if following these ridges is necessary for good performance.

The fact that PSO is rotationally variant and hence has associated biases is not surprising. For example, evolutionary algorithms are also rotationally variant [13]. Yet, this does not discourage us from using these algorithms. The point of our paper is not that the standard PSO is "broken" or that it needs to be "fixed." Rather, as with all algorithms, our point is that we need to understand the biases of our algorithms, so that they can be matched to the characteristics of the problems. For example, if PSO is performing poorly, the user may wish to examine the landscape for linear features. If they exist, the landscape may need to be rotated.

Ultimately what we are seeing is a gradual relaxation of algorithmic properties that used to be deemed necessary. For example, old search algorithms

were always sound and complete. The introduction of incomplete algorithms such as PSO and evolutionary algorithms was met with considerable resistance, because, if a solution was not found, we could not say that one doesn't exist. Similarly, rotational invariance is a property that we used to take for granted. Yet, many of our new algorithms do not possess this property either. This is not necessarily a problem. The key is to analyze the data and see if it has features that do not match the algorithm [8, 5]. If so, the data needs to be recast, or the algorithm needs to be changed (or another algorithm should be chosen).

Acknowledgements

We thank John Hitchcock for showing us a more elegant solution to Case 3:

$$E[\Phi] - \Theta = \frac{\cot\Theta \log(\cos\Theta) - \tan\Theta \log(\sin\Theta)}{2}$$

.

References

[1] M. Clerc. The old bias. *personal web-site*, page 1, Dec 2001.

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

[3] R. Eberhart, Y. Shi, and J. Kennedy. *Swarm Intelligence*. Morgan Kaufmann, 2001.

[4] S. Janson and M. Middendorf. On trajectories of particles in PSO. *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, pages 150–155, 2007.

[5] K. A. D. Jong, W. M. Spears, and D. F. Gordon. Using markov chains to analyze gafos. In *Foundations of Genetic Algorithms 3*, pages 115–137. Morgan Kaufmann, 1995.

[6] Kennedy. personal communication. personal communication with Dr. W. Spears, 2007.

[7] J. Kennedy and R. Eberhart. Particle swarm optimization. *IEEE International Conference on Neural Networks*, 4:1942–1948, November-December 1995.

[8] J. Kennedy and W. M. Spears. Matching algorithms to problems: An experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator. In *In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC*, pages 78–83, 1998.

[9] C. Monson and K. Seppi. Exposing origin-seeking bias in PSO. *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, Jun 2005.

[10] E. Ozcan, S. Cad, and C. K. Mohan. Particle swarm optimization: Surfing the waves. *Proceedings of the IEEE Congress on Evolutionary Computation*, 3, Jul 1999.

[11] E. Ozcan and C. Mohan. Analysis of a simple particle swarm optimization system. *Intelligent Engineering Systems Through Artificial Neural Networks*, 8:253–258, Jun 1998.

[12] R. Poli. Mean and variance of the sampling distribution of particle swarm optimizers during stagnation. *IEEE Transactions on Evolutionary Computation*, 13(4):712–721, August 2009.

[13] R. Salomon. Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions. *BioSystems*, 39(3):263–278, 1995.

[14] Y. Shi and R. Eberhart. A modified particle swarm optimizer. *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 69–73, 1998.

[15] Y. Shi and R. Eberhart. Population diversity of particle swarms. In *IEEE Congress on Evolutionary Computation*, pages 1063–1067, 2008.

[16] Y. Shi and R. Eberhart. Monitoring of particle swarm optimization. *Frontiers of Computer Science in China*, 3(1):31–37, 2009.

[17] F. Vandenbergh and A. Engelbrecht. A study of particle swarm optimization particle trajectories. *Information Sciences*, 176(8):937–971, Apr 2006.

[18] D. Wilke. Analysis of the particle swarm optimization algorithm. Master's thesis, University of Pretoria, Pretoria, South Africa, Feb 2005.

APPENDIX: Computation of $E[\Phi]$ for Case 4

Recall that for Case 4:

$$\Phi = \arctan\left(\frac{(r_{1,2}+r_{2,2})(P_{i,2}-X_{i,2})}{(r_{1,1}+r_{2,1})(P_{i,1}-X_{i,1})}\right) \tag{20}$$

Let u denote $r_{1,2}$ and v denote $r_{2,2}$. Furthermore, let w denote $r_{1,1}$ and x denote $r_{2,1}$. Then, due to the independence of the four random variables $E[\Phi]$ is expressed as the integral:

$$E[\Phi] = \int_0^1\int_0^1\int_0^1\int_0^1 \arctan\left(\tan(\Theta)\frac{u+v}{w+x}\right) du\ dv\ dw\ dx \tag{21}$$

This quadruple integral can be separated into four double integrals by noting that the sum of two U(0,1) random variables is a T(0,1,2) triangular random variable. Let $y = u + v$ and $z = w + x$. Then

$$E[\Phi] = \int_0^1\int_0^1 \arctan\left(\tan(\Theta)\frac{y}{z}\right) y\ z\ dy\ dz + \tag{22}$$

$$\int_1^2\int_0^1 \arctan\left(\tan(\Theta)\frac{y}{z}\right) y\ (2-z)\ dy\ dz + \tag{23}$$

$$\int_0^1\int_1^2 \arctan\left(\tan(\Theta)\frac{y}{z}\right) (2-y)\ z\ dy\ dz + \tag{24}$$

$$\int_1^2\int_1^2 \arctan\left(\tan(\Theta)\frac{y}{z}\right) (2-y)\ (2-z)\ dy\ dz \tag{25}$$

The solution to the first integral is:

$$\left(\frac{1}{8tan^2(\theta)}\right)\left(\theta + tan^3(\theta) + 2\theta tan^2(\theta) - tan(\theta) - tan^4(\theta)atan\left(\frac{1}{tan(\theta)}\right)\right)$$

The solution to the second integral is:

$$-\frac{tan(\theta)}{8} - \frac{1}{8tan(\theta)} + \frac{tan(\theta)}{3}\log(tan^2(\theta)+4) - \frac{tan(\theta)}{3}\log(tan^2(\theta)+1) +$$

$$\left(-\frac{1}{8}tan^2(\theta) + \frac{2}{3tan^2(\theta)} + 1\right)atan\left(\frac{1}{2}tan(\theta)\right) +$$

$$\left(\frac{1}{8}tan^2(\theta) - \frac{5}{24tan^2(\theta)} - \frac{3}{4}\right)\theta$$

The solution to the third integral is:

$$-\frac{2}{3}tan^2(\theta)atan\left(\frac{1}{2tan(\theta)}\right)+\frac{5}{24}tan^2(\theta)atan\left(\frac{1}{\tan(\theta)}\right)-$$
$$\frac{\log(4tan^2(\theta)+1)}{3tan(\theta)}+\frac{\log(tan^2(\theta)+1)}{3tan(\theta)}+$$
$$\left(1-\frac{1}{8tan^2(\theta)}\right)atan(2tan(\theta))+\left(\frac{1}{8tan^2(\theta)}-\frac{3}{4}\right)\theta+\frac{1}{8}tan(\theta)+\frac{1}{8tan(\theta)}$$

The solution to the fourth integral is:

$$\left(-\frac{2}{3}tan(\theta)+\frac{4}{3tan(\theta)}\right)\log(tan^2(\theta)+4)+\left(\frac{4}{3}tan(\theta)-\frac{4}{3tan(\theta)}\right)\log(4tan^2(\theta)+4)+$$
$$\left(\frac{2}{3}tan(\theta)-\frac{2}{3tan(\theta)}\right)\log(tan^2(\theta)+1)+\left(-\frac{4}{3}tan(\theta)+\frac{2}{3tan(\theta)}\right)\log(4tan^2(\theta)+1)+$$
$$\left(\frac{1}{8tan(\theta)}-\frac{1}{8}tan(\theta)\right)+\left(\frac{5}{24}tan^2(\theta)+\frac{2}{3tan^2(\theta)}-3\right)atan\left(\frac{1}{2}tan(\theta)\right)+$$
$$\left(\frac{2}{3}tan^2(\theta)+\frac{5}{24tan^2(\theta)}-3\right)atan(2tan(\theta))+\left(-\frac{7}{8}tan^2(\theta)-\frac{7}{8tan^2(\theta)}+\frac{25}{4}\right)\theta$$