WCCI 2012 IEEE World Congress on Computational Intelligence
June, 10-15, 2012 - Brisbane, Australia IEEE CEC

Particle Swarm Optimization: Velocity Initialization

Andries Engelbrecht
Department of Computer Science
University of Pretoria
Pretoria, South Africa 0001
Email: engel@cs.up.ac.za

***Abstract*—Since its birth in 1995, particle swarm optimization (PSO) has been well studied and successfully applied. While a better understanding of PSO and particle behaviors have been obtained through theoretical and empirical analysis, some issues about the beavior of particles remain unanswered. One such issue is how velocities should be initialized. Though zero initial velocities have been advocated, a popular initialization strategy is to set initial weights to random values within the domain of the optimization problem. This article first illustrates that particles tend to leave the boundaries of the search space irrespective of the initialization approach, resulting in wasted search effort. It is also shown that random initialization increases the number of roaming particles, and that this has a negative impact on convergence time. It is also shown that enforcing a boundary constraint on personal best positions does not help much to address this problem. The main objective of the article is to show that the best approach is to initialize particles to zero, or random values close to zero, without imposing a personal best bound.**

I. Introduction

Numerous empirical [1], [2], [3], [4], [5] and theoretical [6], [2], [7], [8] analyses of particle swarm optimization (PSO) [9], [10] have been done. The main objective of these studies was to gain a better understanding of PSO, specifically the search behavior of particles, the influence of control parameters on performance, and the convergence properties of PSO. A very important aspect of PSO, and often overlooked, is how velocities should be initialized, and what the impact of different velocity initialization stratgies is on PSO performance.

While a popular approach followed in the literature is to initialize velocities to random values uniformly sampled within the domain of the decision variables of the optimization problem, the main objective of this article is to emphasize that initial velocities should be set to zero, or small random values close to zero. In reaching this objective, the article seeks to provide empirically derived answers to the following questions:

1) Do particles leave the boundaries of the search space, and to what extent does this happen?
2) What are the consequences of roaming particles on the feasibility of personal best and neighborhood best solutions?
3) Does the severity of roaming particles and their consequences depend on the way that velocities are initialized?
4) What are the effects of these roaming particles and boundary violations on the performance of PSO?
5) Does a boundary constraint on personal best positions help to address the negative effects of boundary violations, and specifically does it help to address the issues caused by random initialization.

The answers to the above questions result in the recommendation that a boundary constraint on the personal best positions should be avoided, and that particles should be initialized to zero, or small random values. Note that this is an empirical study, considering the standard global best PSO.

The rest of the article is organized as follows: A short overview of the global best PSO is given in Section II, to include only those aspects that are used for the purposes of this study. Sectoin III shows that particles leave the search space very early during the search irrespective of the initialization scheme, and discusses the influence of these roaming particles on performance. Based on the outcomes of this section, Section IV considers different velocity initialization approaches and their influence of swarm behavior and performance. Section V shows that a boundary constraint on the personal best positions does not help to address the issues with random initial velocities.

II. Global Best PSO

The basic behavior of each particle is to move towards two attractors, namely the best position found by the particle and the best position found by the particle's neighborhood. The best position is generally referred to as the personal best (pbest) position, while the best position within the neighborhood is refered to as the neighborhood best (nbest) position. For the global best PSO (gbest PSO), each particle's neighborhood is the entire swarm, in which case the nbest is referred to as the global best (gbest) position. This neighborhood topology is referred to as the star topology [11], [12]. Based on this topology, the velocity, $\mathbf{v}_i$, of the i-th particle is calculated as

$$\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1\mathbf{r}_1(t)(\mathbf{y}_i(t) - \mathbf{x}_i(t)) + c_2\mathbf{r}_2(t)(\hat{\mathbf{y}} - \mathbf{x}_i(t)) \tag{1}$$

where w is the inertia weight [13], c_1 and c_2 are the acceleration coefficients, $\mathbf{r}_1(t), \mathbf{r}_2(t) \sim U(0,1)^{n_x}$, n_x is the dimension of the search space, $\mathbf{x}_i(t)$ is the current position of the i-th particle, $\mathbf{y}_i(t)$ is the particle's pbest position, and $\hat{\mathbf{y}}(t)$ is the gbest position. Particle positions are updated using

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t) \tag{2}$$

A pbest position is updated only if the new position of the corresponding particle results in a better fitness than that of the current pbest position. For the purposes of this article, the gbest position is selected as that pbest position with the best fitness. This results in a memory-based update of the gbest position. An alternative, less often used approach, is to select gbest as the current particle with the best fitness.

In order to prevent step sizes from becoming too large, velocities can be clamped. However, the best clamping threshold, refered to as $\mathbf{V}_{max}$, is problem dependent. Velocity clamping was not used in this study.

Particle positions are initialized to random positions within the domain of the optimization problem, and such that the search space is uniformly covered. Different approaches have been followed in the current literature to initialize velocities, with some debates about the advantages and disadvantages of each:

1) Initialize velocities to zero, i.e. $\mathbf{v}_i(0) = \mathbf{0}$, for all $i = 1, \ldots, n_s$, where n_s is the number of particles. One of the criticisms given against zero initial velocities is that it limits the initial exploration ability of the swarm, and therefor that it limits the extend to which the surface is initially covered by the swarm of particles. As shown later, this is not the case. Remember that the initial particle positions are uniformly distributed throughout the search space, ensuring good initial coverage. It can also be reasoned, in relation to the origins of PSO, that physical objects, in their initial state, do not have any momentum – their initial velocities are zero. If initial velocities are large, then the momentum term (i.e. the inertia term) of equation (1) will result in large initial step sizes, which may be the cause of random initialization having more roaming particles.
2) Initialize velocities to random values withing the domain of the optimization problem, i.e. $\mathbf{v}_i(0) \sim U(-x_{min}, x_{max})^{n_x}$. This is an approach regularly being used in literature, and has caused some unpublished debates. Advocates of random initialization reason that random initial velocities help to improve the exploration ability of the swarm, due to the larger initial steps sizes. Consequently, as is argued, more accurate results will be obtained, and at better speed. While it is true that these larger step sizes increases the initial diversity of the swarm (as is illustrated later), it is also the case that these larger initial step sizes result in many (usualy most) of the particles violating the boundaries of the search space. To illustrate this point, if initial positions are initialized uniformly in the domain $[x_{min}, x_{max}]^{n_x}$, and velocities are also initialized in this domain, then the new positions after the first iteration will be distributed in the range $[2x_{min}, 2x_{max}]^{n_x}$! The consequence of these roaming particles is that many best positions will also violate the boundary constraint.
3) Initializing velocities to small random values. This proposal was made in response to the problem that large initial step sizes cause particles to leave the search. It is reasoned that small random values will not suffer the same problem, while still adding to the diversity of the swarm. However, as is indicated later in this article, particles still leave the search space. Also, a new problem is introduced, namely, what are appropriate small random values? Surely, the answer to this problem depends on the characteristics of the optimization problem.

As stated in the introduction, this article investigates these strategies to initialize velocities, and shows that zero initial velocities, or very small random values, are preferred.

III. Fruitless Wondering

Empirical work done by the author indicated that many particles leave the boundaries of the search space, very early in the optimization process. This behavior results in a number of potential problems, for example:

- Should a better solution be found outside of the defined bounds of the optimization problem, and no boundary constraint mechanism employed, personal best positions are also pulled outside the bounds of the search space. The consequence of these infeasible solutions is that global best positions are also pulled outside of the search space, resulting in an infeasible optimum being found.
- If roaming particles do not find better solutions outside of the search space, they are eventually pulled back into feasible space. However, this is done over a large number of iterations, wasting effort searching infeasible space.
- Roaming particles have a negative influence on swarm diversity calculations, since diversity increases as particles move further outside the bounds of the search space. This is a problem for PSO algorithms that use measures of diversity to control the search trajectories of particles. Such algorithms include the diversity-guided PSO [14], [15], amongst others [16], [17], [18].

This section provides results from an empirical analysis to show that, for all problems considered, a significant part of the swarm violates search space boundaries within the first few iterations, and that this results in best positions also being pulled outside of the search space.

While this study focuses on empirical analysis of the roaming behavior of particles, and the effect that initial velocity has, Helwig and Wanka [19] provided a theoretical analysis of the initial behavior of particles under different initial velocity initialization strategies. This theoretical study provided proofs that many particles leave the search space at the beginning of the optimization process. However, little empirical evidence has been provided to support these theoretical proofs. Therefore, this study has as its objective to provide empirical evidence to illustrate the roaming behavior of particles.

The remainder of this section is organized as follows: Section III-A provides a summary of the experimental procedure followed for this section and the remainder of the article. Section III-B presents empirical result to illustrate the roaming behavior of particles, and Section III-C discusses the consequences of this behavior.

A. Experimental Procedure

For the purposes of this study, the gbest PSO was applied to the functions listed in Table I. A swarm of 30 particles was used, an inertia weight of 0.729844, and acceleration coefficients of 1.496180. A memory-based global best selection strategy was used, with synchronous upadates of particles and best positions. Fifty independent runs were executed for each of the functions, for each initialization strategy.

B. Roaming Behavior

Using a gbest PSO as described in Section II, with parameter values as given in Section III-A, Figure 1 illustrates the percentage of particles that violated boundaries. Except for the Bukin 6 function, all over functions had most of the particles leaving the boundaries of the search space. At least 80% of the particles initially wandered outside the bounds, with an exponential decrease in the number of roaming particles as the number of iterations increased. However, note that it does take a significant number of iterations to reduce the number of violating particles. While the Bukin 6 function had less roaming particles than the other functions, note that the number of roaming particles converged to about 10% after 1000 iteration, failing to reach a zero percentage.

C. Consequences of Roaming Behavior

While Figure 1 does indicate that the number of roaming particles decreases over time, and some may argue that this indicates that roaming particles do not have negative consequences, this section shows, with empirical results, that roaming particles do have negative consequences:

- Best positions are pulled outside of the problem bounds. Figure 2 shows for the personal best positions the average (over the 50 independent runs) percentage of the personal best positions that do violate boundaries, and for the global best positions, the percentage of simulations for which the global best is outside of the problem boundaries. The effect of roaming particles is clearly illustrated with most of the personal best positions also leaving the search space, roaming outside of the bounds. As with the number of roaming particles, the number of roaming personal best positions does decrease over time, contributing to wasted effort searching infeasible space. Also note that, during the early stages of the search, a large percentage of simulations had the global best position also outside of the bounds. This has the consequence that more particles were pulled towards an infeasible global best position, and also may be the reason why so many particles were pulled outside of the bounds. It is the case, for the problems considered, that over time all simulations had their global best position inside of the bounds. Note that functions such as Ackley and Bukin 6 never mamanged to reduce the number of roaming personal best positions to zero during the limit of 1000 iterations, and Bukin 6 always had simulations with the global best positions out of bounds.
- Following on the observation that global best positions are pulled outside of the boudaries early on in the search process, if particles fail to find a solution of better fitness within the boundaries of the search space, an infeasible solution outside of the problem boundaries will be obtained.

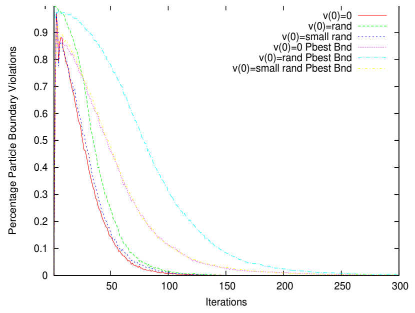
IV. Velocity Initialization Strategies

As indicated in Section II, three popular strategies to initialize velocities can be identified in the literature, namely zero velocities, uniform random values within the domain of the optimization problem, and small uniform random values. This section shows that initialization to random values within the domain of the problem is not a good idea. To aid this discussion, Figure 3 illustrates the rate at which the quality of the best found solution is improved, while Figure 4 illustrates diversity profiles. With reference to the figures, and the previous figures, the consequences of the different velocity initialization strategies are discussed per function. In the discusison below, the term *random initialization* refers to initial velocities sampled from a uniform distribution within the domain of the problem, *small random initialization* refers to initial velocities sampled from a uniform distribution in the range $[-0.1, 0.1]$, and *zero initialization* refers to initial velocities of zero.

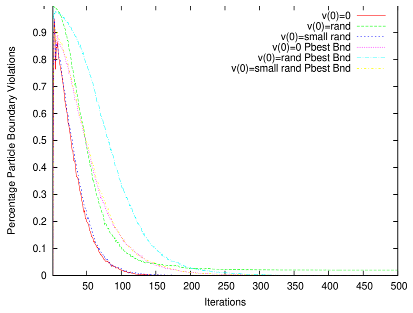
- **Absolute value, Griewank, Quadric**: Random initialization was slower in improving the fitness of the best solution compared to small random and zero initialization. Note that random initialization resulted in a larger diversity, more roaming particles, significantly more personal best positions initially leaving the search space, and a large number of simulations for which the global best position also left the search space. Note that diversity decreased slower than for small random initialization and zero initialization. Also, random initialization took signifactly longer to reduce the number of best position violations. An analysis of the rate at which individual decision variables converged within a threshold of 10^{-5} from the optimal revealed that random initialization is much slower than the other initialization strategies in increasing the number of converged decision variables. Note that small random initialization and zero initialization had very similar behaviors.
- **Ackley**: Random initialization is signifcantly slower in the rate of fitness improvement, despite significantly larger diversity. Note that diversity for random initialization increases after approximately 280 iterations, indicating that particles moved further apart. Random initialization also had significantltly more particles that violated the boundary constraint, never reaching 0% within the 1000 iterations. Personal best and global best position violatations were also higher, never reaching zero. Note that small random initialization and zero initialization showed similar trends, with a very fast reduction in the number of best position violations. Random initialization is again much slower in converging decision variables.

TABLE I
FUNCTIONS USED FOR EMPIRICAL ANALYSIS

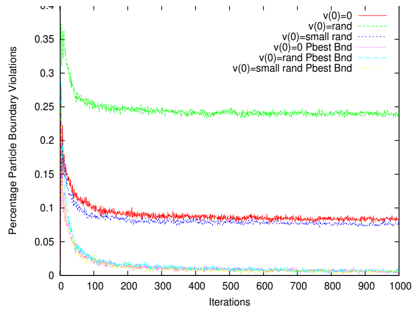
Function	Definition	Domain	Dimension
Absolute Value	$f(\mathbf{x}) = \sum_{j=1}^{n_x} \lvert x_i \rvert$	[-100,100]	30
Ackley	$f(\mathbf{x}) = -20e^{-0.2\sqrt{\frac{1}{n_x}\sum_{j=1}^{n_x} x_j^2}} - e^{\frac{1}{n_x}\sum_{j=1}^{n_x}\cos(2\pi x_j)} + 20 + e$	[-32.768,32.768]	30
Bukin 6	$f(\mathbf{x}) = 100\sqrt{\lvert x_2 - 0.01x_1^2\rvert} + 0.01\lvert x_1 + 10\rvert$	[-15,5],[-3,3]	2
Griewank	$f(\mathbf{x}) = 1 + \frac{1}{4000}\sum_{j=1}^{n_x} x_j^2 - \prod_{j=1}^{n_x}\cos\left(\frac{x_j}{\sqrt{j}}\right)$	[-600,600]	30
Quadric	$f(\mathbf{x}) = \sum_{l=1}^{n_x}\left(\sum_{j=1}^{l} x_j\right)^2$	[-100,100]	30
Rastrigin	$f(\mathbf{x}) = 10n_x + \sum_{j=1}^{n_x}\left(x_j^2 - 10\cos(2\pi x_j)\right)$	[-5.12,5.12]	30
Rosenbrock	$f(\mathbf{x}) = \sum_{j=1}^{n_x-1}\left(100(x_{j+1} - x_j^2)^2 + (x_j - 1)^2\right)$	[-2.048,2.048]	30



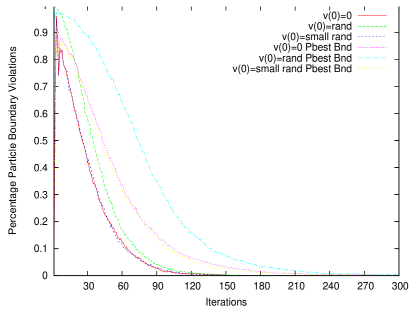
(a) Absolute Value



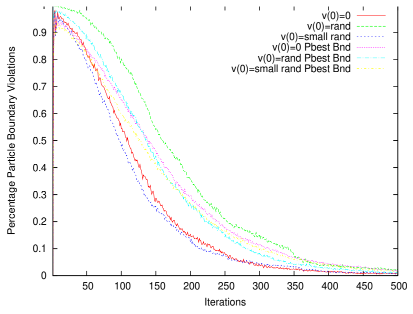
(b) Ackley



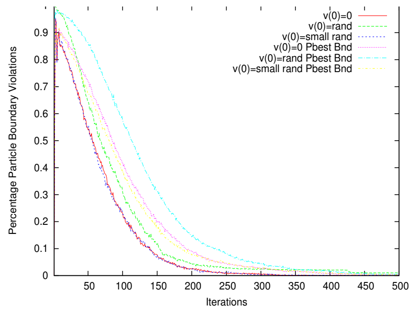
(c) Bukin 6



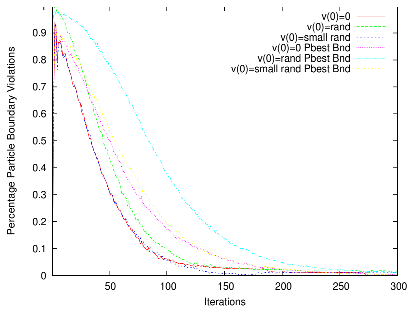
(d) Griewank



(e) Quadric



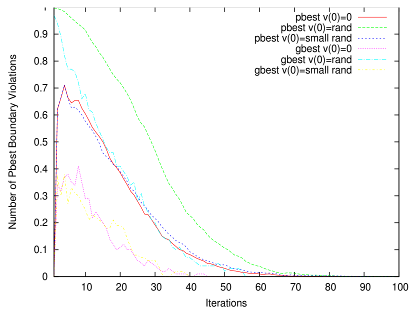
(f) Rastrigin



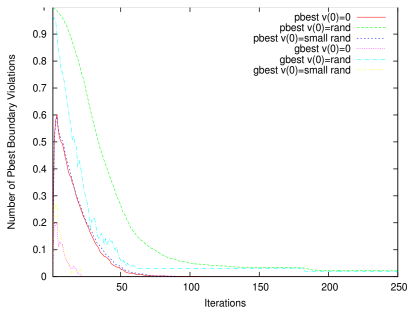
(g) Rosenbrock

Fig. 1. Percentage of Particles that Violate Boundary Constraints

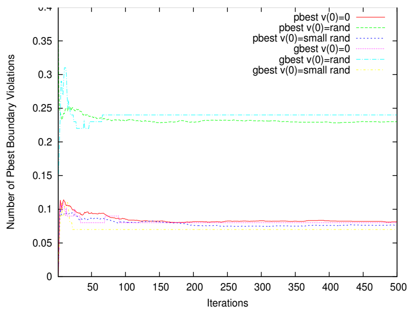
- **Bukin 6**: Random initialization was again slower, and although less severe than for the other functions, had significantly more particle and best position violations than small random initialization and zero initialization. Also note that random initialization maintained a higher diversity.
- **Rastrigin**: While random initialization was initially slower in improving fitness, performance was similar to small random and zero initialization after approximately 100 iterations. The diversity profile followed a similar trend. However, random initialization had signifcantly more particle and best position violations. Convergence



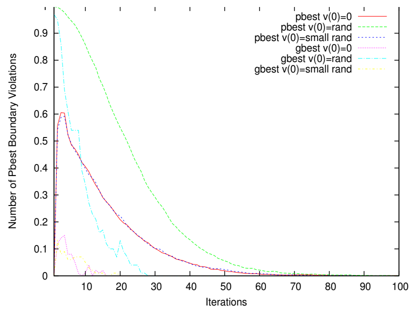
(a) Absolute Value



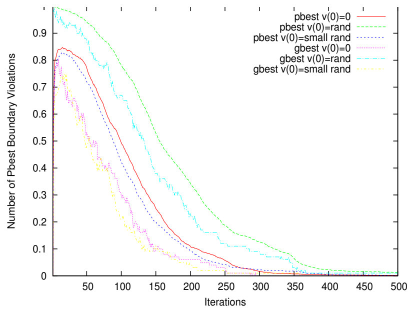
(b) Ackley



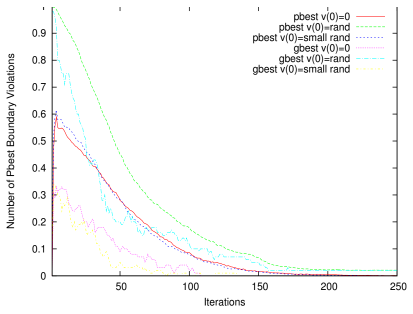
(c) Bukin 6



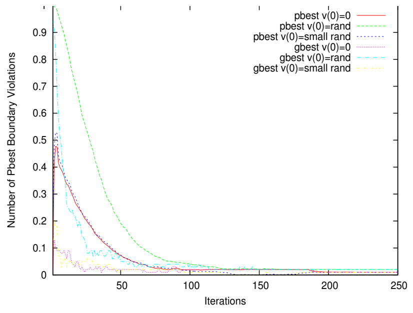
(d) Griewank



(e) Quadric



(f) Rastrigin



(g) Rosenbrock

Fig. 2. Personal Best and Global Best Boundary Violations

of decision variables was also slower than the other initialization strategies.

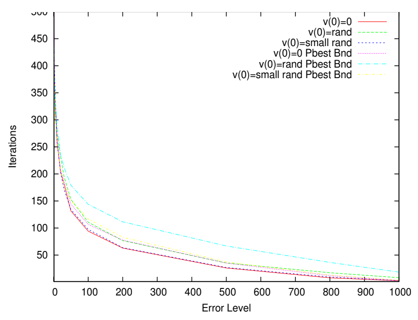
- **Rosenbrock**: Random initialization showed an initial slow improvement in best fitness, and had more particle violations. The number of particle violations and the diversity profile approached that of small random initialization and zero initialization after approximately 80 iterations. However, random initialization resulted in more best position violations throughout the 1000 iterations.

Returning to the opinion that random initialization should obtain better solutions due to its increased diversity levels throughout the optimization process (ass illustrated in Figure 4, Table II summarizes that average fitness of the global best solution after 1000 iterations for random initialization and for zero initialization. The table also contains the standard deviation over the 50 independent runs. It is only for the absolute value function that random initialization resulted in a better average fitness than zero initialization. For the rest of the considered functions, zero initialization haev resulted in the best average fitness. Notably is the significant difference in average fitness for the quadric function where zero initialization obtained an average fitness of 90.4 while random fitness managed to reach an average fitness of 180 only. These results provide clear evidence that random initialization could not produce better results than zero initialization for most of the functions.

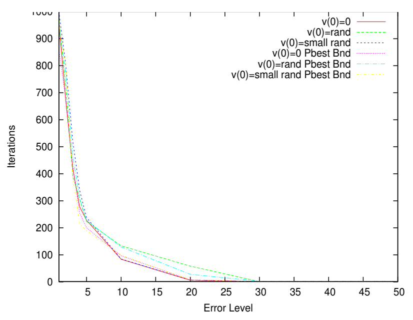
In conclusion, a strong correlation can be seen between random initialization and a slow improvement in fitness, high

TABLE II
AVERAGE FITNESS OF BEST SOLUTION AFTER 1000 ITERATIONS

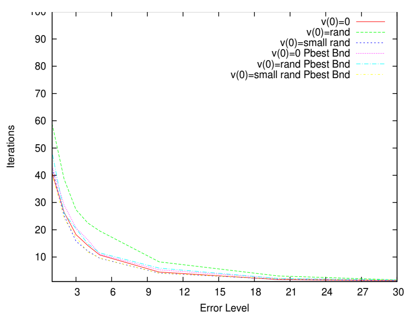
Function	**Zero Init No Pbest Bound**	**Random Init No Pbest Bound**
Absolute Value	3.53E-001±2.87E+000	2.46E-001±1.47E+000
Ackley	2.49E+000±1.35E+000	2.68E+000±2.67E+000
Bukin 6	6.20E-002±4.50E-002	6.65E-002±5.56E-002
Griewank	3.72E-002±5.26E-002	3.91E-002±5.57E-002
Quadric	9.04E+001±8.70E+001	1.80E+002±3.15E+002
Rastrigin	6.66E+001±1.71E+001	7.37E+001±2.16E+001
Rosenbrock	2.65E+001±1.53E+001	2.73E+001±1.66E+001



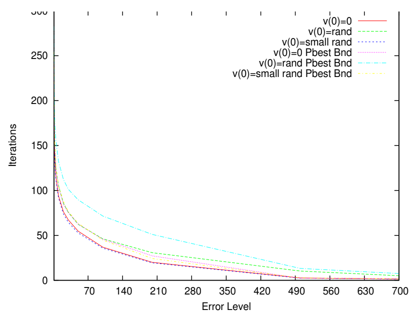
(a) Absolute Value



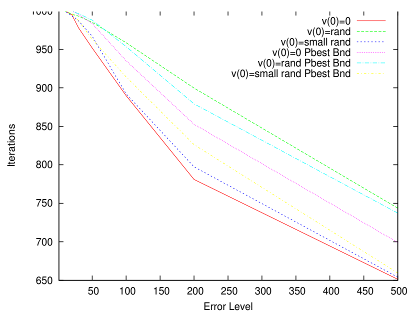
(b) Ackley



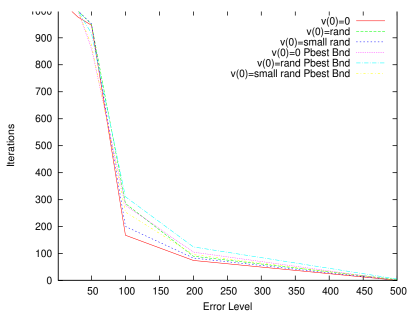
(c) Bukin 6



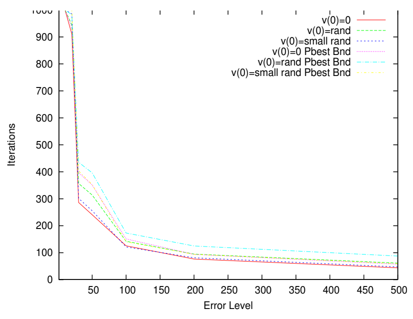
(d) Griewank



(e) Quadric



(f) Rastrigin



(g) Rosenbrock

Fig. 3. Fitness Reduction Profiles

diversity, large number of roaming particles, large number of personal best and global best position violations, and a slow convergence per decision variable. This leads to the main disadvantages of random initialization, namely much slower convergence and much more wasted effort searching outside the bounds of the optimization problem. The conclusion is therefore that random initialization should be avoided in favor of small random or zero initialization. Note that the latter two strategies exhibited similar behavior.

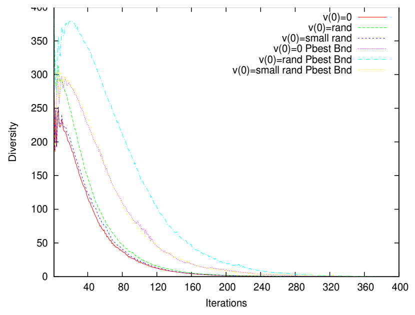
V. Personal Best Constraint

A first critique against the finding of Sections III and IV is that no boundary constraint mechanism has been used. Helwig and Wanka [19] showed that a boundary constraint mechanism should be used to prevent the roaming particles problem.

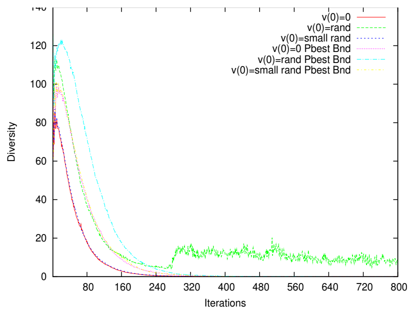
A very simple, and natural boundary constraint mechanism is to constrain personal best positions to always remain in the search space. Should a memory-based approach be used to select the global best position, then the global best position is guaranteed to be within bounds. This can be achieved with a simple change in the decision to update personal best positions: The personal best position of a particle is only updated if that particle has a better fitness than the current personal best position and if the particle is within bounds.

It may be argued that the effects of random initialization will be diminished should such a personal best boundary constraint be employed. This section investigates whether the personal best boundary constraint above does address the issues with random initialization. The previous figures also included the behaviors of gbest PSO for the three velocity initialization strategies under the personal best boundary constraint. With reference to the these figures, the following observations can be made:

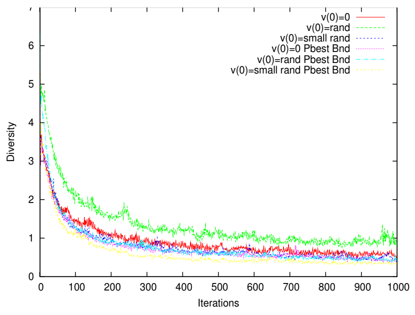
- In general, bounding of the personal best position for random initialization did not help with reference to the rate at which the fitness of the best solution improves nor the rate at which decision variables converges. Only for the Bukin 6 function did the random initialization with personal best bound performed better than without the personal best bound. However, random initialization



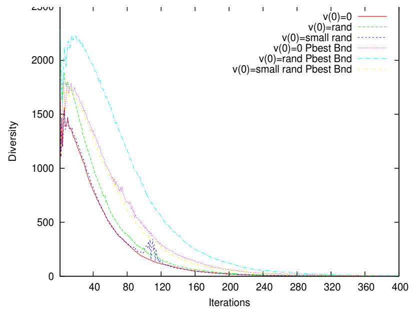
(a) Absolute Value



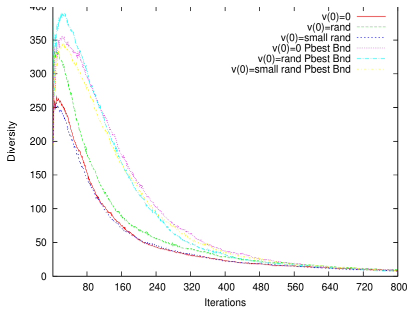
(b) Ackley



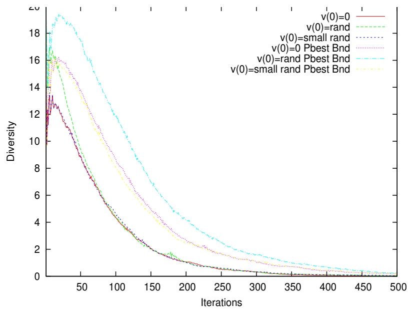
(c) Bukin 6



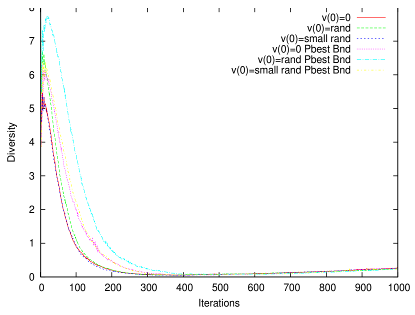
(d) Griewank



(e) Quadric



(f) Rastrigin



(g) Rosenbrock

Fig. 4. Diversity Profiles

with personal best bound still performed worse that small random and zero initialization without the personal best bound.

- The above observation is despite the increased diversity, and slower decrease in diversity of the personal best bound strategies. The increased diversity is likely due to the larger number of roaming particles. This might be due to the large distances between roaming particles and their bounded personal best and global best positions. These larger step sizes may result in more particles leaving the boundaries of the search space, and roaming particles to move further away from feasible space.
- Considering all of the velocity initialization strategies with a personal best bound, performance was worse than the non-bounded strategies. Overall, the small random and zero initialization without personal best bounds still performed best.

The above observations leads to the conclusion that a personal best bound does not address the issues with random initialization. It might be that other boundary constraint mechanisms will help, but such investigation is suggested or future research.

The outcome of this section is further confirmation of the conclusion that random initialization of velocities within the bounds of the optimization problem should be avoided, and that the best approach to follow is zero initialization or very small random values.

VI. Conclusion

The main objective of this paper was to show that initialization of velocities to uniform random values within the domain of the optimization problem is not an optimal approach to initialize velocities. It was shown empirically that fitness of the global best solution is improved at a much slower rate than zero initial velocities or small random initial velocities. Empirical results also showed that convergence per decision variable is delayed using random initialization. This is mainly due to the larger number of roaming particles, as well as the significantly larger number of roaming personal best positions and violating global best positions.

It was also illustrated that a boundary constraint on the personal best positions does not help to address the slow convergence of random initialization, nor the issue with the number of roaming particles. In fact, a personal best bound resulted in even slower convergence and more roaming particles.

As a side-effect of this study empirical results have clearly supported statements in previous theoretical studies [19] that most particles leave the search space early during the search process, irrespective of the initialization strategy.

Future research will expand this analysis to provide answers to the following questions:

- Does the swarm size have an effect on the findings of this article?
- What happens during the first iterations of PSO that result in most particles violating the boundary constraints?
- Do other PSO algorithms, such as the local best PSO, constriction PSO, and the Barebones PSO also suffer from roaming particles? Preliminary studies [20] with the charged PSO [21] have shown similar trends of roaming behavior. Furthermore, do the observations about random initialization also apply to these algorithms?
- To what extend will other boundary constraint mechanisms help to address the deficiencies of random initialization?

References

[1] A. Carlisle and G. Dozier, "An Off-the-Shelf PSO," in *Proceedings of the Workshop on Particle Swarm Optimization*, 2001, pp. 1–6.

[2] M. Clerc and J. Kennedy, "The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[3] Y. Shi and R. Eberhart, "Parameter Selection in Particle Swarm Optimization," in *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, 1998, pp. 591–600.

[4] R. Eberhart and Y. Shi, "Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 1, July 2000, pp. 84–88.

[5] F. van den Bergh and A. Engelbrecht, "Effects of Swarm Size on Cooperative Particle Swarm Optimisers," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2001, pp. 892–899.

[6] E. Ozcan and C. Mohan, "Analysis of a Simple Particle Swarm Optimization System," in *Intelligent Engineering Systems through Artificial Neural Networks*, 1998, pp. 253–258.

[7] I. Trelea, "The Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection," *Information Processing Letters*, vol. 85, no. 6, pp. 317–325, 2003.

[8] F. van den Bergh and A. Engelbrecht, "A Study of Particle Swarm Optimization Particle Trajectories," *Information Sciences*, vol. 176, no. 8, pp. 937–971, 2006.

[9] R. Eberhart and J. Kennedy, "A New Optimizer using Particle Swarm Theory," in *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, 1995, pp. 39–43.

[10] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proceedings of the IEEE International Joint Conference on Neural Networks*. IEEE Press, 1995, pp. 1942–1948.

[11] R. Eberhart, P. Simpson, and R. Dobbins, *Computational Intelligence PC Tools*, 1st ed. Academic Press Professional, 1996.

[12] J. Kennedy, "Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 3, July 1999, pp. 1931–1938.

[13] Y. Shi and R. Eberhart, "A Modified Particle Swarm Optimizer," in *Proceedings of the IEEE Congress on Evolutionary Computation*, May 1998, pp. 69–73.

[14] J. Riget and J. Vesterstrm, "A Diversity-Guided Particle Swarm Optimizer - the ARPSO," Department of Computer Science, University of Aarhus, Tech. Rep., 2002.

[15] M. Pant, T. Radha, and V. Singh, "A Simple Diversity Guided Particle Swarm Optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2007, pp. 3294–3299.

[16] T. Hendtlass, "Preserving Diversity in Particle Swarm Optimisation," *Lecture Notes in Computer Science*, no. 2718, pp. 31–40, 2003.

[17] Z. Li, R. Ngambusabongsopa, E. Mohammed, and N. Eustache, "A Novel Diversity Guided Particle Swarm Multi-objective Optimization Algorithm," *International Journal of Digital Content Technology and its Applications*, vol. 5, no. 1, pp. 269–278, 2011.

[18] J. Sun, W. Fang, and W. Xu, "A Quantum-Behaved Particle Swarm Optimization With Diversity-Guided Mutation for the Design of Two-Dimensional IIR Digital Filters," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, no. 2, pp. 141–145, 2010.

[19] S. Helwig and R. Wanka, "Theoretical Analysis of Initial Particle Swarm Behavior," in *Proceedings of the Tenth International Conference on Parallel Problem Solving from Nature*, 2008, pp. 889–898.

[20] B. Leonard, A. Engelbrecht, and A. van Wyk, "Heterogeneous Particle Swarms in Dynamic Environments," in *Proceedings of the IEEE Swarm Intelligence Symposium*, 2011.

[21] T. Blackwell and P. Bentley, "Dynamic Search with Charged Swarms," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2002, pp. 19–26.