Boundary Constraint Handling Techniques for Particle Swarm Optimization in High Dimensional Problem Spaces

Elre T. Oldewage[1,2], Andries P. Engelbrecht[1,3], and Christopher W. Cleghorn[1]

[1] Department of Computer Science, University of Pretoria, Pretoria, South Africa
vze.ezv@gmail.com, engel@cs.up.ac.za, ccleghorn@cs.up.ac.za
[2] Council for Scientific and Industrial Research, Pretoria, South Africa
[3] Institute for Big Data and Data Science, Pretoria, South Africa

Abstract. This paper investigates the use of boundary constraint handling mechanisms to prevent unwanted particle roaming behaviour in high dimensional spaces. The paper tests a range of strategies on a benchmark for large scale optimization. The empirical analysis shows that the hyperbolic strategy, which scales down a particle's velocity as it approaches the boundary, performs statistically significantly better than the other methods considered in terms of the best objective function value achieved. The hyperbolic strategy directly addresses the velocity explosion, thereby preventing unwanted roaming.

1 Introduction

Particle swarm optimization (PSO) is a stochastic, population-based optimization algorithm [9]. A swarm consists of a number of particles. Each particle's position in the search space represents a possible solution to an optimization problem. The particles move through the search space, guided by local and global information. This paper considers PSO with inertia weight [22].

Previous studies in literature have emphasized the importance of boundary handling techniques for PSO, especially in high dimensional spaces [14, 17]. As problem dimensionality increases, the particles become increasingly likely to leave the search space and exhibit unwanted roaming behaviour [14]. Application of boundary constraint handling techniques may mitigate particles' roaming behaviour and allow the search to continue even in high dimensional spaces.

A number of constraint handling strategies are examined that may be employed to mitigate particle roaming behaviour. Classical boundary constraint handling methods have been suggested [8, 19], but the methods utilize information about the optima locations and/or gradient information. For black-box optimization problems, such information is typically not available. The boundary constraint handling techniques considered in this paper do not make use of additional information about the optimization function.

Boundary constraint handling techniques bias the particles towards certain parts of the search space [10, 12]. Thus, the best choice in technique usually depends on the location of the optima. However, additional information about the

optima locations may not be known. This paper considers minimization problems that have been shifted by a random vector, distributed uniformly throughout the search space. There is thus no clear pattern regarding the optima locations, i.e. they are not near the boundaries or near the center of the search space in all dimensions. The paper discusses the performance of a selection of boundary handling techniques so that practitioners are guided in choosing a strategy when the locations of the optima are unknown or differ widely among dimensions.

The paper proceeds as follows: Section 2 discusses the boundary constraint handling techniques being considered. Section 3 describes the experimental method. Section 4 presents the empirical results and Section 5 concludes the paper.

2 Background

This section discusses a number of the most common boundary constraint handling techniques. Section 2.1 discusses position repair methods. Section 2.2 introduces velocity repair strategies that can be combined with the position repair strategies. Section 2.3 discusses techniques that do not fall into either category.

2.1 Position Repair Strategies

This section lists position repair strategies, which modify a particle's position so that it no longer violates the boundary.

Infinity: The first strategy, suggested by [3] only modifies the PSO algorithm by constraining the personal and global best positions to be within the search space. Thus, the particles may leave the search space, but their local and global attractors will always be within bounds, thereby encouraging the particles to return to the search space. There have been suggestions in literature to make this approach standard practice [3]. This method is also known as the "invisible wall" [20]. The *infinity* method has the advantage of not modifying the velocity or position vectors directly, thereby preventing the algorithm from becoming biased as a side effect of the boundary handling technique.

Random: The *random* method [4, 13, 18] re-initializes any invalid position component uniformly within the search space. A possible side effect of this method is to inject diversity into the swarm by randomly selecting position components that particles would have been unlikely to encounter otherwise. Another variant, *random-half* [21], re-initializes any invalid position components within the half of the search space nearest the violated boundary.

Absorb: The *absorb* strategy repairs a particle's position by moving it back onto the boundary in every violated dimension. This approach biases particles towards solutions that are on the boundary of the search space. The *absorb* strategy is also known as *truncate* [1], *nearest* [11], or *boundary* [18].

Exponential: The *exponential* method as originally proposed [1] repairs a particle's position in every dimension by moving the particle to a point between its previous position and the violated boundary. The new position is sampled from a truncated exponential distribution, oriented so that there is

a higher probability of sampling a position near the boundary. An alternative method [18] samples from a truncated exponential distribution spread across the entire search space in the violated dimension (oriented so that positions near the violated boundary are more likely). The original method is referred to as *exponential-confined* and the latter as *exponential-spread*. The *exponential* method introduces less artificial diversity than the *random* and *random-half* methods, and also preserves search information about good solutions near the boundaries by sampling from a biased distribution.

2.2 Velocity Repair Methods

A particle's velocity vector contains information about favourable search directions in relation to its local and global attractors. If the particle is relocated, then the relative direction of its attractors change. Modifying a particle's position without also adjusting its velocity may cause the particle to move in directions that are unfavourable due to its outdated momentum component. Additionally, if the particle left the search space due to high velocity in a given dimension, then the particle is likely to leave the search space again after being moved inside the search space (because it still has a large, outward momentum component). A variety of velocity repair methods are discussed below:

Zero: Set the velocity to zero in violated dimensions.

Adjust: The *adjust* strategy [11], [13] performs a backward calculation to obtain the repaired velocity after applying a position repair strategy, $\mathbf{v}_i^{t+1} = \mathbf{x}_i^{t+1} - \mathbf{x}_i^t$, where $\mathbf{v}_i^{t+1}$ denotes the velocity and $\mathbf{x}_i^{t+1}$ denotes the position of the i-th particle at iteration $t+1$. This strategy records a particle's movement to a feasible location. For certain position repair strategies, this may help to prevent the particle from leaving the search space again or from moving in unfavourable directions due to an outdated momentum component.

Reflect: The *reflect* strategy reflects the particle's velocity in the violated dimension. Reflection ensures that particles do not have large, outward momentum components after their positions have been repaired, with the aim of reducing their propensity to leave in the following iteration.

Random Damping: *Random damping*[15] is a hybrid between reflection and absorption. If a particle exceeds the boundary in a given dimension, its velocity is partially reflected and partially absorbed. This forces the particle back into valid space and decreases its velocity. The fraction of the velocity to be reflected or absorbed is uniform random.

Damping: *Damping* is a deterministic version of the *random damping* method in [15]. The parameter λ is used to determine how much of the velocity is reflected or absorbed. λ may be a constant value or, as in [16], may depend on the particle's distance from the boundary. In this paper, $\lambda = 0.5$.

2.3 Other Strategies

This section lists strategies that prescribe how a particle's position and velocity should be repaired or interpreted when a boundary is violated.

PBest: The *pBest* method, as proposed in [16], re-positions a particle to its personal best position and sets its velocity to **0** if it leaves the search space (in any dimension). Since particles frequently leave the search space in high dimensions, this may lead to most of the swarm being relocated frequently. Relocating a particle encourages highly exploitative behaviour: in the following iteration, the velocity's cognitive component will be zero and the momentum component is zero, since the velocity was zeroed. Thus, the particle's movement depends only on its social component, causing the particles to move towards the global best position. In this paper, another version of the *pBest* strategy is suggested in which only violated dimensions are reset. This method is referred to as *pBest-dim*. Resetting only violated dimensions will reduce the chances of premature convergence since fewer dimensions will rely on social-only velocity updates.

Hyperbolic: The *hyperbolic* strategy [6] prevents a particle from ever reaching the boundary by scaling its velocity. The closer a particle is to the boundary, the smaller its scaled velocity is. Scaling is performed as follows:

$$v_{i,j}^t = \begin{cases} \frac{v_{i,j}^t}{1+|v_{i,j}^t/(U_j - x_{i,j}^t)|} & \text{if } v_{ij}^t > \frac{U_j+L_j}{2} \\ \frac{v_{i,j}^t}{1+|v_{i,j}^t/(x_{i,j}^t - L_j)|} & \text{if } v_{ij}^t \leq \frac{U_j+L_j}{2} \end{cases} \tag{1}$$

where U_j and L_j denote the upper and lower boundaries in the j-th dimension.

Resampling Stochastic Scalars: The *RES* or *resampling* method [1] resamples the stochastic scalars $r_{1,j}$ and $r_{2,j}$ in every dimension until the resulting velocity does not cause the particle to leave the search space. This strategy will have a non-deterministic run time. Particles close to the boundary may have to draw many random numbers to obtain satisfactory values for $r_{1,j}$ and $r_{2,j}$.

Periodic PSO: The *periodic* strategy [24] does not modify particle positions or velocities. Instead, the search space is extended with infinitely many copies that the particles can traverse without further consideration to the boundaries. A particle's position is mapped back to the original search space for evaluation, where each dimension is mapped by M_j as described below:

$$x_{i,j}^t \overset{M_j}{\mapsto} L_j + (x_{i,j}^t \% (U_j - L_j)) \tag{2}$$

where % is the modulo operator. A particle's score or fitness is given by $f(M(\mathbf{x}_i^t))$, where f denotes the objective function. Although the strategy works well on some search spaces, adjoining copies of the search space may introduce sharp discontinuities, which make the search space more difficult to traverse.

3 Experimental Method

This section describes the empirical method. The experiments used PSO with inertia weight [22] with the global best topology. The selected inertia weight, $w = 0.7298$ and the acceleration coefficients $c_1 = c_2 = 1.49618$ are known good values suggested by Clerc [7] that guarantee convergence of the swarm (in terms of expectation and variance of particle positions [5]). Each swarm consisted of 50

particles. The different boundary-handling techniques were tested on problems from the CEC 2010 Benchmark Suite for Large Scale Optimization [23] with dimensionality of 1000 ($n = 1000$). The suite consists of minimization problems that includes separable, non-separable, and partially separable problems. The degree of separability is controlled by a parameter m which was set to 10 for these experiments. The original definition of the benchmark suite uses a vector of random numbers distributed normally throughout the search space (in each dimension). However, this will bias the location of the optima to be near the center of the search space. In order to prevent such bias, the shift vectors used in this paper are distributed uniformly throughout the search space. Every boundary-handling technique was run on each of the 20 benchmark problems 30 times for statistical significance. Every simulation was allowed 5000 iterations.

4 Results

Every boundary handling technique was assigned a rank score that depends on the best score achieved over all simulations as proposed in [2]. These scores were normalized so that the best rank score is 1 and the worst is 0 (as shown in Figure 1). A score is related to the number of statistically significant "wins" when a strategy is compared in a pairwise manner to all the other strategies across all benchmark functions (in terms of solution accuracy). Comparisons are performed using a Mann-Whitney U test with $p = 0.05$. Additionally, an average normalized fitness was calculated for each technique according to:

$$\frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \frac{\overline{f(\mathbf{y})}}{\widetilde{f(\mathbf{y})}} \tag{3}$$

where $\mathcal{F}$ denotes the set of benchmark functions, $|.|$ denotes set cardinality, $\overline{f(\mathbf{y})}$ denotes the best fitness attained by the strategy on function f, averaged across all runs and $\widetilde{f(\mathbf{y})}$ denotes the worst average fitness attained by any strategy on function f. Thus, if a strategy always performed the worst on all functions, it would receive an average normalized fitness of 1.

Figure 1, which plots the rank score and average normalized fitness, shows that the *hyperbolic* strategy exhibits the best performance in terms of both measures. *Hyperbolic* also performed statistically significantly better than the other four best strategies on 16 out of the 20 benchmark functions. It is known from literature [14, 17] that a large factor in PSO's poor performance in high dimensional problem spaces is the initial velocity explosion and the consequent roaming behaviour. The *hyperbolic* strategy completely mitigates the effects of the velocity explosion, since the velocity is scaled down to ensure that the particles remain in valid space. Although this prevents particles from attaining optima that are on the boundaries, the strategy does not affect the particles' search direction or artificially introduce or inhibit swarm diversity. The reduction in the velocity explosion is apparent in Figures 2 and 3 which plot the average velocity magnitude and the average variance in velocity for the five best-performing strategies.

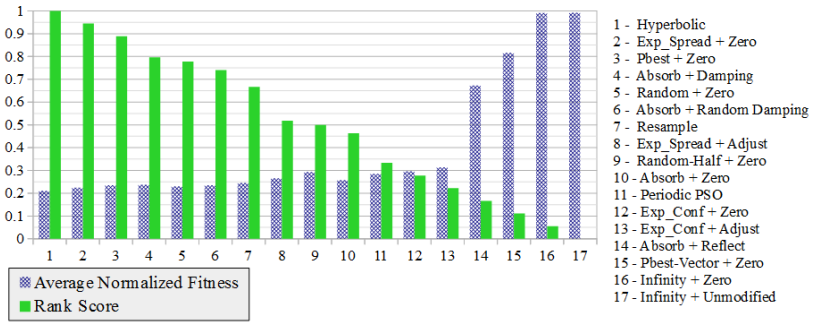

Fig. 1. Average Normalized Fitness and Rank Scores

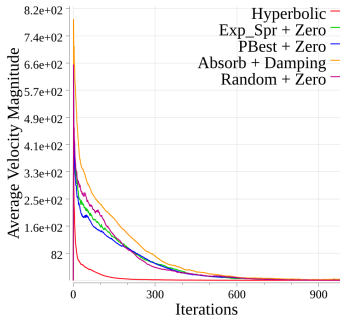

Fig. 2. Velocity Magnitude (Top 5)

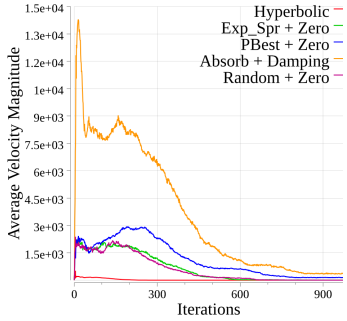

Fig. 3. Velocity Variance (Top 5)

For 15 out of the 19 strategies, the average normalized fitness was between 0.21 and 0.32. Therefore, although the difference in performance among the strategies were statistically significant, the actual difference in fitness between most of the strategies was not very large. This is likely due to the problem-dependent nature of boundary handling strategies, which is known in literature.

Due to space limitations, only a few of the strategies' behaviour are discussed in detail. Although it may be expected that the *pBest* strategy will converge prematurely, Figure 4 shows that the *pBest* strategy failed to converge. Instead, the swarm's diversity oscillates with every iteration, as the particles attempt to explore, leave the search space immediately and are reset. Since no searching could take place, the personal bests were almost never updated and thus the global best was almost never updated. In contrast, the *pBest-dim* strategy performed quite well and achieved the 3-rd best score. *pBest-dim* also performed better than randomly re-initializing, since resetting the position to a known good location encourages the search to exploit within a known good region.

The two strategies that performed the worst were *infinity + zero* and *infinity + unmodified*. All of the particles left the search space and remained out of bounds for the remainder of the search. The average number of violated boundaries was fewer for *infinity + zero* than for *infinity + unmodified* (see Figure 5).

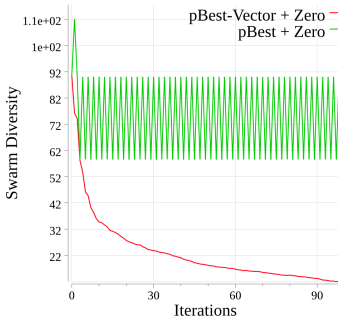

Fig. 4. Swarm Diversity of PBest Strategies on F10 (First 100 Iterations)

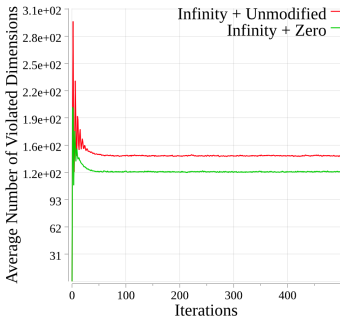

Fig. 5. Average Number of Violated Dimensions on F11 (First 500 Iterations)

Thus, zeroing the velocity component when a particle is out of bounds does improve the particle's ability to return to the search space to some extent.

In all cases where comparison was possible, zeroing the velocity performed better than adjusting or reflection. *Damping* and *random damping* performed better than *zero*. However, *damping* and *random-damping* are not applicable for many of the position repair strategies. All three of the best-performing velocity repair strategies reduce the velocity in some manner, thereby reducing the velocity explosion and the consequent roaming behaviour.

5 Conclusion

This paper tested PSO with a variety of boundary constraint handling techniques on high dimensional problems with optima that were distributed uniformly throughout the search space. The best-performing strategy was *hyperbolic*, which rescales a particle's velocity so that the particle can never reach the boundary, thereby preventing the velocity explosion and subsequent unwanted roaming. The five best performing strategies were *hyperbolic*, *exp_spread + zero*, *pBest + zero*, *absorb + damping* and *random + zero*. Although the difference in performance of these strategies are statistically significant, their performance is highly problem-dependent and their average normalized fitness values were similar. In general, velocity repair strategies such as *zero* and *damping* performed better than the other velocity repair strategies. The worst strategies were *infinity+zero* and *infinity+unmodified*, which were also the least restrictive.

Acknowledgments. This work is based on the research supported by the National Research Foundation (NRF) of South Africa (Grant Number 46712). The opinions, findings and conclusions or recommendations expressed in this article is that of the author(s) alone, and not that of the NRF. The NRF accepts no liability whatsoever in this regard.

References

1. Alvarez-Benitez, J.E., Everson, R.M., Fieldsend, J.E.: A mopso algorithm based exclusively on pareto dominance concepts. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) In Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization. pp. 459–473. Springer Berlin Heidelberg (Mar 2005). https://doi.org/10.1007/978-3-540-31880-4_32
2. Bonyadi, M.R., Michalewicz, Z.: Impacts of coefficients on movement patterns in the particle swarm optimization algorithm. IEEE Transactions on Evolutionary Computation **21**(3), 378–390 (June 2017). https://doi.org/10.1109/TEVC.2016.2605668
3. Bratton, D., Kennedy, J.: Defining a standard for particle swarm optimization. In: Proceedings of the IEEE Swarm Intelligence Symposium. pp. 120–127. IEEE Computer Society (2007). https://doi.org/10.1109/SIS.2007.368035
4. Chu, W., Gao, X., Sorooshian, S.: Handling boundary constraints for particle swarm optimization in high-dimensional search space. Information Science **181**(20), 4569–4581 (Oct 2011). https://doi.org/10.1016/j.ins.2010.11.030
5. Cleghorn, C.W., Engelbrecht, A.P.: Particle swarm stability: a theoretical extension using the non-stagnate distribution assumption. Swarm Intelligence (Sep 2017). https://doi.org/10.1007/s11721-017-0141-x
6. Clerc, M.: Confinements and biases in particle swarm optimization (March 2006), http://clerc.maurice.free.fr/pso/, [Online; posted 12-March-2006]
7. Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation **6**(1), 58–73 (Feb 2002). https://doi.org/10.1109/4235.985692
8. Deb, K.: Optimization for Engineering Design: Algorithms and Examples. Prentice-Hall (1995)
9. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science. pp. 39–43 (Oct 1995). https://doi.org/10.1109/MHS.1995.494215
10. Helwig, S., Branke, J., Mostaghim, S.: Experimental analysis of bound handling techniques in particle swarm optimization. IEEE Transactions on Evolutionary Computation **17**(2), 259–271 (April 2013). https://doi.org/10.1109/TEVC.2012.2189404
11. Helwig, S., Branke, J., Mostaghim, S.: Experimental analysis of bound handling techniques in particle swarm optimization. Transactions on Evolutionary Computing **17**(2), 259–271 (April 2013). https://doi.org/10.1109/TEVC.2012.2189404, http://dx.doi.org/10.1109/TEVC.2012.2189404
12. Helwig, S., Wanka, R.: Particle swarm optimization in high-dimensional bounded search spaces. In: 2007 IEEE Swarm Intelligence Symposium. pp. 198–205 (April 2007). https://doi.org/10.1109/SIS.2007.368046
13. Helwig, S., Wanka, R.: Particle swarm optimization in high-dimensional bounded search spaces. In: Proceedings of the IEEE Swarm Intelligence Symposium. pp. 198–205. IEEE Computer Society (2007). https://doi.org/10.1109/SIS.2007.368046
14. Helwig, S., Wanka, R.: Theoretical analysis of initial particle swarm behavior. In: Proceedings of the 10th International Conference on Parallel Problem Solving from Nature - Volume 5199. pp. 889–898. Springer-Verlag New York, Inc. (2008). https://doi.org/10.1007/978-3-540-87700-4_88

15. Huang, T., Mohan, A.S.: A hybrid boundary condition for robust particle swarm optimization. IEEE Antennas and Wireless Propagation Letters **4**, 112–117 (2005). https://doi.org/10.1109/LAWP.2005.846166
16. Mostaghim, S., Mostaghim, S., Halter, W., Wille, A.: Linear Multi-Objective Particle Swarm Optimization, pp. 209–238. Springer Berlin Heidelberg, Berlin, Heidelberg (2006). https://doi.org/10.1007/978-3-540-34690-6_9
17. Oldewage, E.: The Perils of Particle Swarm Optimisation in High Dimensional Problem Spaces. Master's thesis, University of Pretoria, Pretoria, South Africa (2018)
18. Padhye, N., Deb, K., Mittal, P.: Boundary handling approaches in particle swarm optimization. In: Bansal, J.C., Singh, P.K., Deep, K., Pant, M., Nagar, A.K. (eds.) Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications. vol. 1, pp. 287–298. Springer India (2013). https://doi.org/10.1007/978-81-322-1038-2_25
19. Reklaitis, G., Ravindran, A., Ragsdell, K.: Engineering Optimization Methods and Applications. Wiley (1983)
20. Robinson, J., Rahmat-Samii, Y.: Particle swarm optimization in electromagnetics. IEEE Transactions on Antennas and Propagation **52**(2), 397–407 (Feb 2004). https://doi.org/10.1109/TAP.2004.823969
21. Shi, Y., Cheng, S., Qin, Q.: Experimental study on boundary constraints handling in particle swarm optimization: From population diversity perspective. International Journal of Swarm Intelligence Research **2**(3), 43–69 (Jul 2011). https://doi.org/10.4018/jsir.2011070104
22. Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: Proceedings of the IEEE International Conference on Evolutionary Computation. pp. 69–73 (May 1998). https://doi.org/10.1109/ICEC.1998.699146
23. Tang, K., Li, X., Suganthan, P.N., Yang, Z., Weise, T.: Benchmark functions for the cec2010 special session and competition on large-scale global optimization. Tech. rep., Nature Inspired Computation and Applications Laboratory (2009)
24. Zhang, W., Xie, X., Bi, D.: Handling boundary constraints for numerical optimization by particle swarm flying in periodic search space. CoRR **abs/cs/0505069** (2005), http://arxiv.org/abs/cs/0505069