Particle Swarm Optimization for Large-Scale Optimization

Willie Loftie-Eaton
Department of Computer Science
Stellenbosch University
Student Number: 24717274
Email: 24717274@sun.ac.za

Abstract—**Particle swarm optimization (PSO) is known to suffer under the curse of dimensionality. This paper investigates two known approaches for high dimensional problems, and explores the effectiveness of a hybrid approach that initializes particles using subspace initialization and stochastic scaling.**

I. Introduction

Many optimization algorithms suffer from the "curse of dimensionality", which implies that their performance deteriorates as the dimensionality of the search space increases. [1] provides an explanation for why this can be made intuitive (to briefly summarize: search space volume increases exponentially as the number of dimensions increase, hence the hypercube near the optimum rapidly shrinks in size relative to the search space, thus the global optimum becomes harder to find (for a uniform sampling method)).

From [2] and [3], it is clear that the degree of stochasticity in the position updates of particles in a particle swarm optimizer (PSO) has a marked influence on the performance of a PSO. This is because low stochasticity in particle position updates may result in the subspace containing the optimum never being reached by particles in the swarm. One approach to increase the stochasticity of particle position updates is to linearly increase the number of dimensions in which a particle's position is scaled in a stochastic manner. We shall refer to this method as the 'stochastic scaling' approach.

One approach to addressing the curse of dimensionality was put forth in [4] which suggested initializing particles within a small subspace of the search space, focusing on exploitation to find fairly good solutions. We shall refer to this method as as the 'subspace initialization' approach.

Both the stochastic scaling approach and subspace initialization approach have been evaluated independently. This paper proposes a hybrid approach, and explores the effectiveness of a hybrid approach that initializes particles using subspace initialization and stochastic scaling, specifically with linearly increasing number of groups.

The remainder of the paper is structured as follows: Section II provides a brief overview of the basic PSO algorithm, stochastic scaling and subspace initialization approaches, and briefly describes each of the strategies against which the proposed hybrid approach will be compared. Section III introduces the hybrid approach for large scale optimization problems (LSOPs). Section IV describes the experimental method. Section V provides the results of the experiments and discusses the observed behavior. Section VI concludes the paper.

II. Background

A. Overview of PSO

Let s denote the swarm size. Each individual $1 \leq i \leq s$ has the following attributes. A current position in the search space $\mathbf{x}_i$, a current velocity $\mathbf{v}_i$, and a personal best position in the search space $\mathbf{y}_i$. During each iteration, each particle in the swarm is updated using (2) and (1). Assuming that the function f is to be minimized, that the swarm consists of n-dimensional particles, and that $\mathbf{r}_1 \sim U(0,1)^n, \mathbf{r}_2 \sim U(0,1)^n$, then velocity is updated by

$$\begin{aligned} v_{i,j}(t+1) = {} & w v_{i,j}(t) \\ & + c_1 r_{1,j}(t)[y_{i,j}(t) - x_{i,j}(t)] \\ & + c_2 r_{2,j}(t)[\hat{y}_{i,j}(t) - x_{i,j}(t)] \end{aligned} \tag{1}$$

for all $j \in 1, \ldots, n$. c_1 and c_2 denote the acceleration coefficients if the cognitive and social components, respectively. w denotes the inertial weight. Thus $v_{i,j}$ is the velocity of the jth dimension of the ith particle. The new particle position is given by

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

The personal best position, $\mathbf{y}_i$, of each particle is updated using

$$\mathbf{y}_i(t+1) = \begin{cases} \mathbf{y}_i(t), & \text{if } f(\mathbf{x}_i(t+1)) \geq f(\mathbf{y}_i(t)) \\ \mathbf{x}_i(t+1), & \text{if } f(\mathbf{x}_i(t+1)) < f(\mathbf{y}_i(t)) \end{cases} \tag{3}$$

and the global best position found by any particle during all previous steps, $\hat{\mathbf{y}}_i$, is defined as

$$\hat{\mathbf{y}}(t+1) = \arg\min_{\mathbf{y}_i} f(\mathbf{y}_i(t+1)), \quad 1 \leq i \leq s. \tag{4}$$

B. Group-Based Stochastic Scaling

Traditional PSO approaches typically scale the cognitive and social components of the velocity update equation using vectors of random values applied component-wise to each decision variable, as shown in (1). While this allows for broad exploration, it can lead to inefficiencies in LSOPs due to the curse of dimensionality.

Group-based stochastic scaling offers a compromise by dividing the problem's decision variables into groups and applying a single random scaling factor to all variables within a group. This constrains particle movement to lower-dimensional subspaces, encouraging more focused exploitation.

The linearly increasing group strategy, introduced in [5], starts the optimization process with a single group — the stochastic components in (1), $\mathbf{r}_1$ and $\mathbf{r}_1$, are scalar vectors. This limits movement to the subspace defined by the swarm's initial positions. Gradually, the number of groups are increased with each iteration until each variable can be scaled independently (as in (1)). This progression provides a beam-search-like behavior, allowing the swarm to initially explore a narrow but promising region of the search space before gradually expanding its freedom to escape poor initial subspaces.

Compared to:

- Fixed group number strategies, which use a constant number of groups throughout the search, this method avoids premature convergence by dynamically increasing exploration.
- Decreasing group number strategies, which start with full flexibility and become more constrained over time, the increasing group strategy exhibits better control over exploration, especially early on when poor decisions can irreversibly mislead the search in LSOPs.

Algorithm 1 gives the pseudocode for the linearly increasing group-based stochastic scaling PSO (LIGPSO).

C. Subspace Initialization

As mentioned in the introduction, may PSO approaches suffer from the curse of dimensionality for LSOPs - these include initialization strategies such as uniform random sampling or quasi-random sequences (e.g., Sobol or Halton) [4], which aim to cover the search space as evenly as possible. The curse of dimensionality implies even large swarms will cover only an exponentially shrinking fraction of the search space. The likelihood of initializing particles near good solutions is hence low.

The subspace initialization strategy addresses this challenge by deliberately restricting the initial particle positions to a lower-dimensional subspace within the full search space. Each particle is initialized on a line or plane through the center of the search space, defined as a random linear combination of a small set of orthogonal basis vectors (the seed set). The reduced dimensionality of the initialization region increases the probability that multiple particles are initialized in proximity to each other, promoting exploitation.

Empirical results in [4] show that, in high dimensions, subspace initialization often yields better-quality solutions than global initialization techniques.

Algorithm 2 gives the pseudocode for the subspace initialization PSO (SIPSO).

D. PSOs for LSOPs

This paper compares SIPSO and LIGPSO to

- the standard PSO with fully stochastic components, which is referred to simply as PSO

Algorithm 1 PSO with Linearly Increasing Group-Based Stochastic Scaling

```
Require: Objective function $f$, dimensionality $n$, swarm size $s$, bounds $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$, max iterations $T$
Ensure: Best solution $\hat{\mathbf{y}}(T)$ found
 1: Initialize positions $\mathbf{x}_i(0)$ and velocities $\mathbf{v}_i(0)$ for $i = 1, \ldots, s$
 2: Set personal bests: $\mathbf{y}_i(0) \leftarrow \mathbf{x}_i(0)$
 3: Set global best: $\hat{\mathbf{y}}(0) \leftarrow \arg\min_{\mathbf{y}_i(0)} f(\mathbf{y}_i(0))$
 4: for $t = 0$ to $T - 1$ do
 5:     $g(t) \leftarrow \left\lfloor \frac{(n-1)t}{T-1} \right\rfloor + 1$        ▷ Number of groups
 6:     Randomly permute $\{1, \ldots, n\}$ and partition into groups $G_1, \ldots, G_{g(t)}$
 7:     Sample group-level random vectors: $\mathbf{r}_1(t), \mathbf{r}_2(t) \sim \mathcal{U}(0,1)^{g(t)}$
 8:     for each particle $i = 1$ to $s$ do
 9:         for each group $k = 1$ to $g(t)$ do
10:             for each dimension $j \in G_k$ do
11:                 Update velocity component:
                    $$\begin{aligned} v_{i,j}(t+1) = {} & w \cdot v_{i,j}(t) \\ & + c_1 \cdot r_{1,k}(t) \cdot [y_{i,j}(t) - x_{i,j}(t)] \\ & + c_2 \cdot r_{2,k}(t) \cdot [\hat{y}_j(t) - x_{i,j}(t)] \end{aligned} \quad (5)$$
12:             end for
13:         end for
14:         Update position: $\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1)$
15:         Clip $\mathbf{x}_i(t+1)$ to $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$
16:         if $f(\mathbf{x}_i(t+1)) < f(\mathbf{y}_i(t))$ then
17:             $\mathbf{y}_i(t+1) \leftarrow \mathbf{x}_i(t+1)$
18:             if $f(\mathbf{y}_i(t+1)) < f(\hat{\mathbf{y}}(t))$ then
19:                 $\hat{\mathbf{y}}(t+1) \leftarrow \mathbf{y}_i(t+1)$
20:             end if
21:         else
22:             $\mathbf{y}_i(t+1) \leftarrow \mathbf{y}_i(t)$
23:         end if
24:     end for
25: end for
26: return $\hat{\mathbf{y}}(T)$
```

- the standard PSO with scalar vectors as stochastic components, which is referred to as PSO_S

The next section will introduce a hybrid approach between SIPSO and LIGPSO.

III. Methodology

The hybrid approach between SIPSO and LIGPSO simply initializes particles using SIPSO, and then updates particles similar to LIGPSO. This approach is called the subspace initialization linearly increasing group-based stochastic scaling PSO (SILIGPSO).

IV. Empirical Procedure

Swarms were connected using the star topology. The inertial weight and acceleration coefficients for all approaches were

Algorithm 2 PSO with Subspace Initialization

```
Require: Swarm size $s$, dimension $n$, subspace size $u$, bounds $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$, fitness function $f$
1:  Generate $u$ random linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_u \in \mathbb{R}^n$
2:  Orthogonalize using Modified Gram-Schmidt to get $\mathbf{b}_1, \ldots, \mathbf{b}_u$
3:  Compute search space center $\mathbf{c} = \frac{1}{2}(\mathbf{x}_{\min} + \mathbf{x}_{\max})$
4:  for each particle $i = 1$ to $s$ do
5:      Sample $c_1, \ldots, c_u \sim \mathcal{U}(0, 1)$
6:      $\mathbf{d}_i \leftarrow \sum_{j=1}^{u} c_j \mathbf{b}_j$
7:      Determine $[t_{\min}, t_{\max}]$ such that $\mathbf{p}_i = t\mathbf{d}_i + \mathbf{c}$ lies (approximately) within bounds
8:      Sample $t \sim \mathcal{U}(t_{\min}, t_{\max})$
9:      Initialize $\mathbf{x}_i(0) \leftarrow t\mathbf{d}_i + \mathbf{c}$
10:     Initialize $\mathbf{v}_i(0) \sim \mathcal{U}(-|\mathbf{x}_{\max} - \mathbf{x}_{\min}|, |\mathbf{x}_{\max} - \mathbf{x}_{\min}|)^n$
11:     $\mathbf{y}_i(0) \leftarrow \mathbf{x}_i(0)$
12: end for
13: $\hat{\mathbf{y}}(0) \leftarrow \arg\min_{\mathbf{y}_i(0)} f(\mathbf{y}_i(0))$
14: for $t = 0$ to $T - 1$ do
15:     for each particle $i = 1$ to $s$ do
16:         Sample $\mathbf{r}_1(t), \mathbf{r}_2(t) \sim \mathcal{U}(0, 1)^n$
17:         Update velocity $\mathbf{v}_i(t + 1)$ using (1)
18:         Update position $\mathbf{x}_i(t + 1)$ using (2)
19:         Clamp $\mathbf{x}_i(t + 1)$ to bounds $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$
20:         Update personal best $\mathbf{y}_i(t + 1)$ using (3)
21:     end for
22:     Update global best $\hat{\mathbf{y}}(t + 1)$ using (4)
23: end for
24: return $\hat{\mathbf{y}}(T)$
```

set to $w = 0.729$, $c_1 = 1.49445$, and $c_2 = 1.49445$. There were 30 particles in each swarm. In the case of the SIPSO and SILIGPSOs, the dimension of the subspace was the minimum of the dimension of the search space, and 25. Particles had initial velocities of zero, and velocity clamping was not applied.

Table I lists the functions on which the different approaches were evaluated. These functions were tested because their optimum was known, hence more analysis could be done. The "Source" column of Table I lists the identifier of each function according to its source. Function i from [6] is denoted by f_i.

Each approach was run on each of the benchmark problems once, for variable number of iterations, and over a variable number of dimensions from $\{5, 500, 1000, 2000\}$.

TABLE I: Benchmark Functions

Function	Source
Absolute value	f_1
Ackley	f_2
Hyperellipsoid	f_7
Quadric	f_{10}
Quartic	f_{11}
Schwefel 1.2	f_{16}
Spherical	f_{22}

To compare the performance of the approaches, the following was observed:

- The **average proximity to the global best** was tracked by calculating the average distance of all particles in the swarm to the location of the best fitness value found across all iterations.
- **Swarm performance** was characterized by taking the lbest ($\hat{\mathbf{y}}$ in (1)) fitness for each iteration.
- **Swarm diversity** was characterized by means of the average distance from the swarm centre as suggested in [7].

V. Results

A. Average distance

Figures 1, 2, and 3 demonstrate that, for large dimensions, SIPSO and SILIGPSO have the smallest distance from the global optimum, while the simple PSO is best for fewer dimensions.

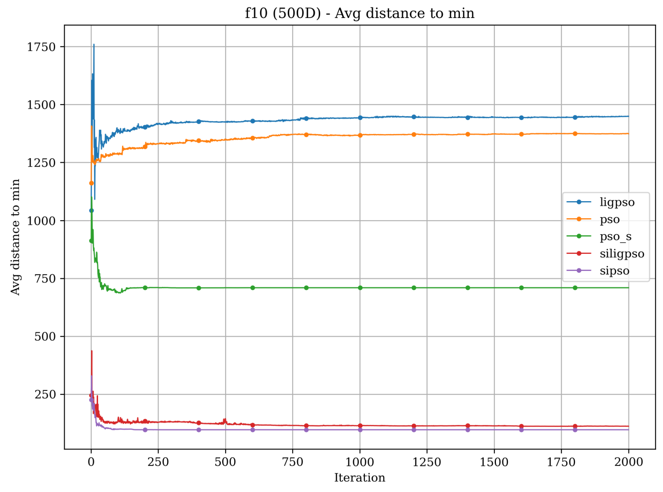

Fig. 1: Average distance to minimum over time for f10, 500D

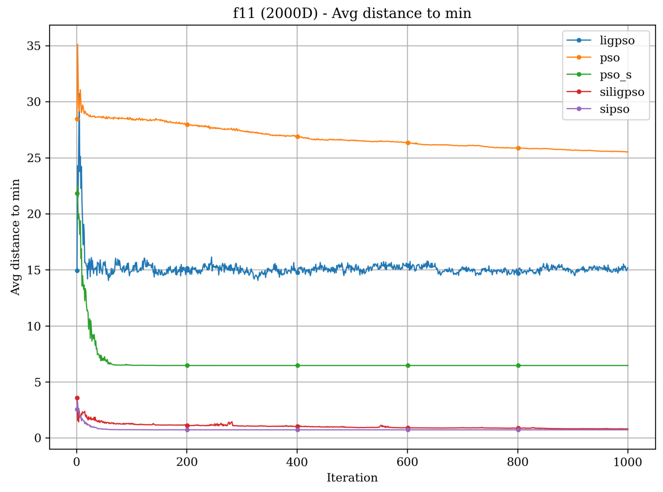

Fig. 2: Average distance to minimum over time for f11, 2000D

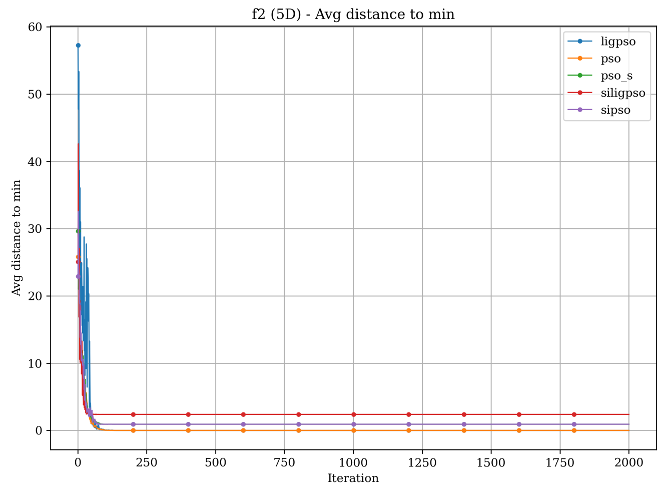

Fig. 3: Average distance to minimum over time for f2, 5D

B. *Best fitness*

Figures 4, 5, and 6 demonstrate that, for large dimensions, SIPSO and SILIGPSO are once again very good compared to the other PSOs.

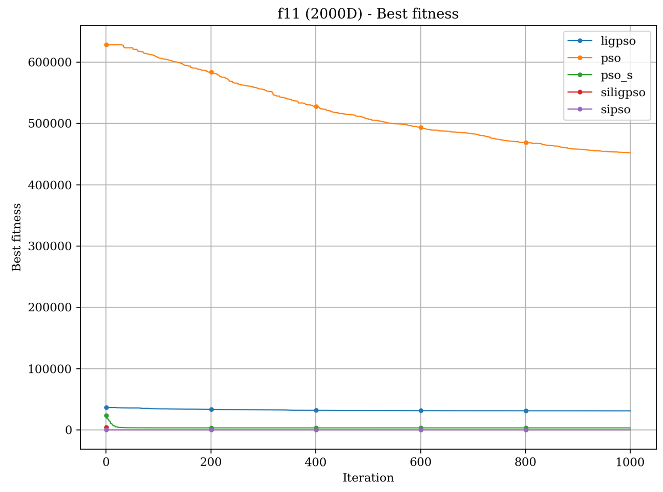

Fig. 4: Best fitness over time for f11, 2000D

C. *Diversity*

Figure 9 illustrates that all PSOs quickly lose diversity for fewer dimensions.

VI. Conclusion

This paper concurs with [4]: SIPSO is effective at finding optima for LSOPs.

The hybrid approach between LIGPSO and SIPSO, SILIGPSO, also performs well, but more analysis is needed. More functions and variations of functions should be tested with the SILIGPSO.

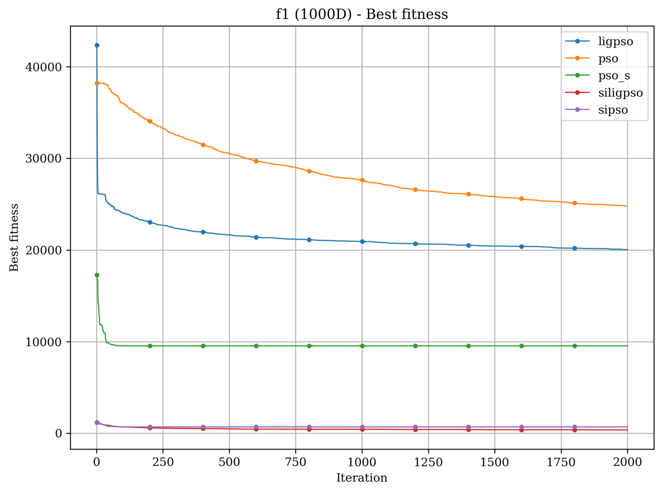

Fig. 5: Best fitness over time for f1, 1000D

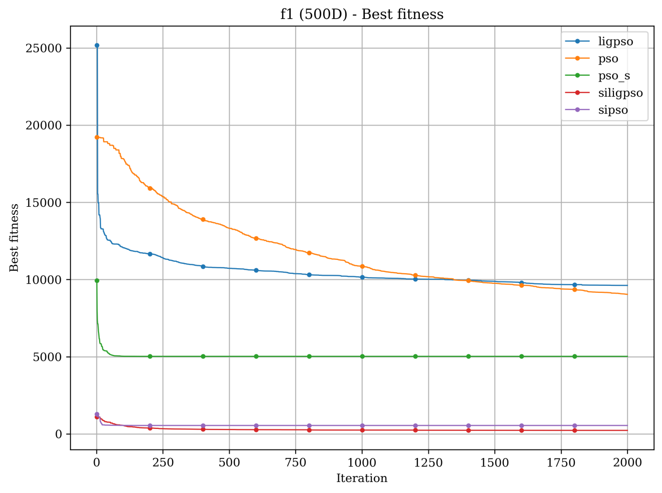

Fig. 6: Best fitness over time for f1, 500D

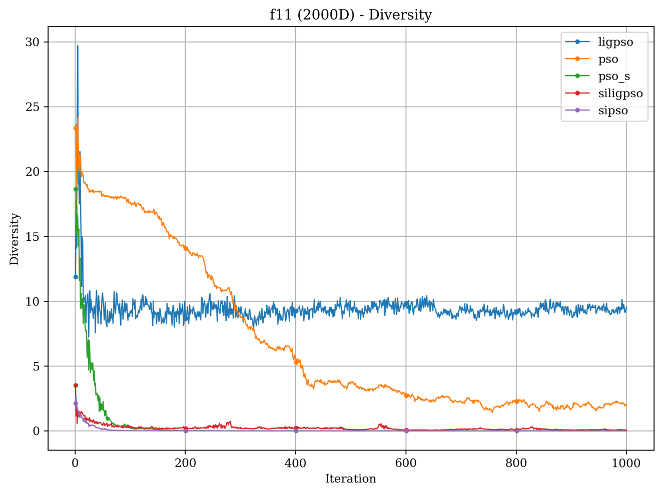

Fig. 7: Diversity over time for f11, 2000D

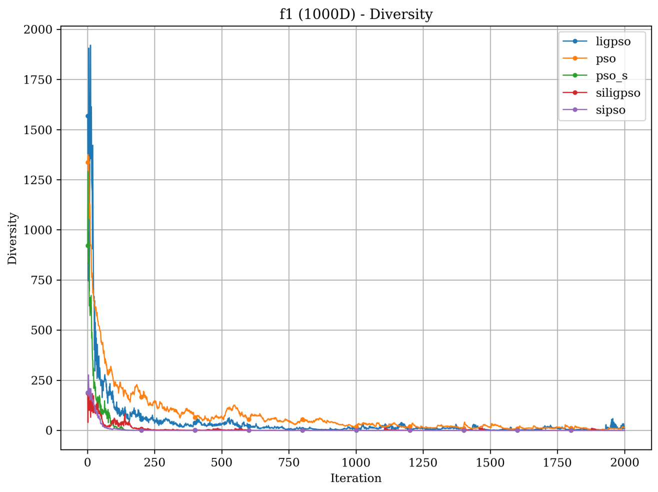

Fig. 8: Diversity over time for f1, 1000D

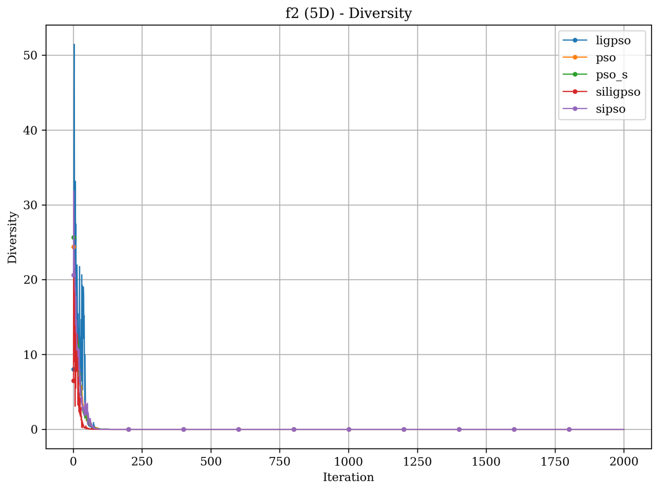

Fig. 9: Diversity over time for f2, 5D

REFERENCES

[1] F. van den Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[2] E. T. Oldewage, A. P. Engelbrecht, and C. W. Cleghorn, "The importance of component-wise stochasticity in particle swarm optimization," in *Swarm Intelligence*, M. Dorigo, M. Birattari, C. Blum, A. L. Christensen, A. Reina, and V. Trianni, Eds. Cham: Springer International Publishing, 2018, pp. 264–276.

[3] ——, "Degrees of stochasticity in particle swarm optimization," *Swarm intelligence*, vol. 13, no. 3-4, pp. 193–215, 2019.

[4] E. van Zyl and A. Engelbrecht, "A subspace-based method for pso initialization," in *2015 IEEE Symposium Series on Computational Intelligence*, 2015, pp. 226–233.

[5] E. T. van Zyl and A. P. Engelbrecht, "Group-based stochastic scaling for pso velocities," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 1862–1868.

[6] A. Engelbrecht, "Particle swarm optimization: Global best or local best?" in *2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence*, 2013, pp. 124–135.

[7] O. Olorunda and A. P. Engelbrecht, "Measuring exploration/exploitation in particle swarms using swarm diversity," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, 2008, pp. 1128–1134.