Particle Swarm Optimization with an Oscillating Inertia Weight

Kyriakos Kentzoglanakis
School of Computing
University of Portsmouth
UK
kyriakos.kentzoglanakis@port.ac.uk

Matthew Poole
School of Computing
University of Portsmouth
UK
matthew.poole@port.ac.uk

ABSTRACT
In this paper, we propose an alternative strategy of adapting the inertia weight parameter during the course of particle swarm optimization. Three oscillating inertia weight functions of time are presented and their influence on the performance and success of the algorithm is investigated by optimizing six standard benchmark functions. Instances of the algorithm that make use of inertia weight adaptation schemes from the literature are also tested for comparison purposes. The results show that the proposed functions are competitive and, in some cases, outperform the established inertia weight functions, in terms of consistency and speed of convergence.

Conference Track
Ant Colony Optimization and Swarm Intelligence

Categories and Subject Descriptors
G.1.6 [Numerical Analysis]: Optimization—Global Optimization; I.2.8 [Artificial Intelligence]: Problem Solving, Control Methods and Search—Heuristic methods

General Terms
Algorithms, Performance, Experimentation

Keywords
particle swarm optimization, inertia weight

1. INTRODUCTION
Early simulations of movement of individual entities in animal societies, such as simulated bird flocks [18, 9] and fish schools [10], demonstrated that collective behaviour and self-organization in such systems emerge from local interactions between these simple entities, rather than from centralized control.

Particle swarm optimization (PSO) is a stochastic, global optimization technique that exploits this empirical observation, for the purposes of problem solving [12]. As such, the progression of the swarm towards finding better solutions is not expressed algorithmically but, rather, emerges as a result of simple particle interactions.

Despite the fact that what can be considered as a general, standard PSO version [2] is, essentially, parameter free, there have been numerous efforts to investigate the behaviour of the algorithm under different parametric conditions. Two of the most studied aspects of PSO are the swarm communication topology and the inertia weight parameter.

The swarm’s communication topology defines neighborhoods, where information concerning discovered solutions is shared among particles and is used to calculate their subsequent positions in the search space.

The inertia weight parameter, introduced in [19], defines the impact of each particle’s previous velocity to the current one and, according to Poli et al., it can be interpreted as the fluidity of the medium in which particles move [17].

The inertia weight effectively controls the scope of the search [20], what is frequently referred to as the balance between searching for an optimal region (exploration) and searching for an optimal point within the region (exploitation) [13].

In this paper, we investigate the impact of a non-monotonic, time varying inertia weight parameter to the objective of guiding the search process, so as to reduce the likelihood of premature convergence by controlling the dynamics of the balance between global and local search, exploration and exploitation.

Two sinusoidal inertia weight functions of time are proposed; one with a constant amplitude and one with a linearly decreasing amplitude. Also the usage of a step function with the same frequency characteristics as the sinusoidal functions is explored.

The performance of the proposed inertia weight functions of time is assessed by optimizing six well-known benchmark functions. Results are also provided using standard methods of adapting the inertia weight, for the purpose of comparison.

Similar approaches have been implemented in the context of simulated annealing [14], a stochastic global optimization technique inspired by the physical process of annealing, i.e. the heating and subsequent cooling of solid materials with the objective of coercing them into a high order, low energy state. The behaviour of the algorithm depends on the temperature parameter, which typically decreases over
Table 1: The inertia weight update functions \( \omega(t) \) from the literature, that were also tested in the experiments. PSO iterations are denoted by \( t \) and the total number of iterations for which the inertia weight is allowed to vary, by \( S \).

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant ( \omega ) (value from [19])</td>
<td>const</td>
<td>( \omega(t) = 0.7298 )</td>
</tr>
<tr>
<td>Linearly decreasing ( \omega ) [20]</td>
<td>lin dec</td>
<td>( \omega(t) = \omega_{\text{min}} + (\omega_{\text{max}} - \omega_{\text{min}}) \frac{t}{t_{\text{max}}} )</td>
</tr>
<tr>
<td>Nonlinearly decreasing ( \omega ) [3]</td>
<td>nonlin dec</td>
<td>( \omega(t) = \omega_{\text{min}} + (\omega_{\text{max}} - \omega_{\text{min}}) \left( \frac{t}{t_{\text{max}}} \right)^n ) with ( n = 1.2 )</td>
</tr>
<tr>
<td>Linearly increasing ( \omega ) [22]</td>
<td>lin inc</td>
<td>( \omega(t) = \omega_{\text{max}} - (\omega_{\text{max}} - \omega_{\text{min}}) \frac{t}{t_{\text{max}}} )</td>
</tr>
<tr>
<td>Random ( \omega ) [7]</td>
<td>rnd</td>
<td>( \omega(t) = U(\omega_{\text{min}}, \omega_{\text{max}}) )</td>
</tr>
</tbody>
</table>

where \( U(a,b) \) a random number drawn from a uniform distribution in \([a,b]\).
the refinement of discovered solutions (local search) near the end.

The same authors also showed that a PSO algorithm with a random inertia weight component was able to track a nonlinear dynamic system [7]. In optimizing such systems, an informed decision can not be made on whether exploration or exploitation is preferable at any given time. As such, a random inertia weight, uniformly distributed within a proper range, makes this decision stochastic.

A somewhat different interpretation suggests that the value of the inertia weight is not associated with the scope but rather with the stability of the search [22]. A linearly increasing time-varying inertia weight, in this context, serves to guide the algorithm from an initial turbulent state (low inertia weight) towards stability (high inertia weight) at later stages.

Other approaches to adjusting the inertia weight include such as a random inertia weight component was able to track a nonlinear dynamic system [3] and a fuzzy system for dynamic adaptation of $\omega$ [21].

3. APPROACH

The concept of an oscillating inertia weight implements a strategy whereby a wave of exploration (global search), followed by a wave of exploitation (local search), form a cycle which is repeated during the optimization process. This way, the swarm periodically transitions from exploratory to exploitative states of search.

This temporal behaviour can be implemented by means of an inertia weight function $\omega(t)$, where $t$ denotes PSO iterations, that oscillates between an upper bound $\omega_{\text{max}}$ and a lower bound $\omega_{\text{min}}$.

In this paper, we propose and investigate the behaviour of three such functions: a sinusoidal function with a constant amplitude, a sinusoidal function with a linearly decreasing amplitude and a step function with the same frequency characteristics as the sinusoidal functions.

The first of the three, a sinusoidal with a constant amplitude is expressed by the following equation:

$$\omega(t) = \frac{\omega_{\text{min}} + \omega_{\text{max}}}{2} + \frac{\omega_{\text{max}} - \omega_{\text{min}}}{2} \cos\left(\frac{2\pi t}{T}\right)$$  (5)

As implied in the previous section, a typical requirement is that the algorithm starts at $\omega_{\text{max}}$ with high exploration capabilities and ends at $\omega_{\text{min}}$ with high exploitation tendencies, hence such a wave function should complete $\frac{3}{2} + k$ cycles within a PSO run, where $k \in \mathbb{N}$ is a parameter that controls the frequency of oscillation.

Consequently, the period $T$ of oscillation, i.e. the number of PSO iterations within which the inertia weight function completes a full cycle, is given by:

$$T = \frac{2S_1}{3 + 2k}$$  (6)

where $S_1$ is the number of iterations for which the inertia weight is allowed to oscillate. We set $S_1 = 3S/4$ iterations, where $S$ is the total number of iterations for a single PSO run. During the final $S_2 = S/4$ iterations, the inertia weight value is kept constant at $\omega(t) = \omega_{\text{min}}$, in order to promote the refinement of discovered solutions. This behaviour is displayed in figure 1 (on the left).

The second proposed function is a sinusoidal with a linearly decreasing amplitude, so that the swarm’s transition between global and local search becomes subtler over the course of optimization. Effectively, $\omega_{\text{max}}$ in equation 5, becomes a linearly decreasing function $\omega'_{\text{max}}$ of time:

$$\omega'_{\text{max}}(t) = (1 - \frac{t}{S})\omega_{\text{max}}$$  (7)

and the value of inertia weight now oscillates between a static $\omega_{\text{min}}$ and a dynamic $\omega'_{\text{max}}(t)$, according to the equation:

$$\omega(t) = \frac{\omega_{\text{min}} + \omega'_{\text{max}}(t)}{2} + \frac{\omega'_{\text{max}}(t) - \omega_{\text{min}}}{2} \cos\left(\frac{2\pi t}{T}\right)$$  (8)

The third proposed inertia weight function is a step function with the same frequency characteristics as in the sinusoidal case. In this case, exploration and exploitation tendencies are discretely modelled, according to the equation:

$$\omega(t) = \begin{cases} 
\omega_{\text{max}} & \text{if } \cos\left(\frac{2\pi t}{T}\right) \geq 0 \\
\omega_{\text{min}} & \text{otherwise}
\end{cases}$$  (9)
and PSO was allowed to run for 4.

## 4. METHODS

The conducted tests investigate the influence of the proposed oscillating inertia weight functions, described by equations 5 (osc), 8 (osc dec) and 9 (step) to PSO. For comparison purposes, five inertia weight functions from the literature, summarized in table 1, were included in the tests. These tests consist of optimizing six standard benchmark functions, presented in table 2.

For the experiments, the swarm population size was set to 20 particles, particle positions were assymetrically initialized and PSO was allowed to run for \( S = 10000 \) iterations. For the first set \( S_1 = 3S/4 = 7500 \) iterations, the inertia weight was allowed to vary according to the equation under consideration, while during the final \( S_2 = S/4 = 2500 \) iterations, the inertia weight was kept constant at its last assumed value (except for the random inertia weight function where the value \( \omega_{\text{min}} \) was used for this final stage).

The empirical values \( \omega_{\text{min}} = 0.3 \) and \( \omega_{\text{max}} = 0.9 \) were chosen as the range of the inertia weight. The acceleration coefficients were set to \( C_1 = C_2 = 1.496 \), following Clerc and Kennedy’s recommendation [5].

Enforcement of \( x_{\text{max}} \) was also performed, according to which if a particle position vector component exceeds the objective function’s feasible bounds, it is brought back to the exceeded boundary and the corresponding velocity vector component is set to zero.

Two different sets of experiments were conducted, each varying the swarm topology in use. The first set uses a static local best topology, considered as standard [2], according to which the swarm forms a ring lattice and each particle communicates with its \( K = 2 \) adjacent particles.

The second set uses a dynamic random neighborhood selection strategy, whereby at each iteration, a particle randomly chooses \( K \) other particles to form its information neighborhood. For the results presented here, the value \( K = 3 \) is used.

Table 3 summarizes the results, which are averaged across 50 independent PSO runs. The table records the average optimization success rate according to the function goals defined in table 2, as well as the average number of iterations needed to satisfy these goals. Parameter \( k \) of equation 6 has been set to \( k = 7 \) for the three oscillatory inertia weight functions (osc, osc dec and step).

### Table 2: The six benchmarks functions that were used in the tests.

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Feasible Bounds</th>
<th>Initialization</th>
<th>D</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>sphere</td>
<td>( f_1 = \sum_{i=1}^{D} x_i^2 )</td>
<td>((-100, 100)^D)</td>
<td>((50, 100)^D)</td>
<td>30</td>
<td>0.01</td>
</tr>
<tr>
<td>rosenbrock</td>
<td>( f_2 = \sum_{i=1}^{D-1} \left( 100(x_{i+1} - x_i)^2 + (x_i - 1)^2 \right) )</td>
<td>((-30, 30)^D)</td>
<td>((15, 30)^D)</td>
<td>30</td>
<td>100</td>
</tr>
<tr>
<td>rastrigin</td>
<td>( f_3 = \sum_{i=1}^{D}</td>
<td>x_i^2 - 10 \cos(2\pi x_i) + 10</td>
<td>)</td>
<td>((-5.12, 5.12)^D)</td>
<td>((2.56, 5.12)^D)</td>
</tr>
<tr>
<td>grieswank</td>
<td>( f_4 = \frac{1}{10000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 )</td>
<td>((-600, 600)^D)</td>
<td>((300, 600)^D)</td>
<td>30</td>
<td>0.05</td>
</tr>
<tr>
<td>schaffer’s f6</td>
<td>( f_5 = 0.5 + \frac{\sin(\sqrt{x_1^2 + x_2^2}) - 0.5}{\left(1 + 0.001(x_1^2 + x_2^2)\right)^2} )</td>
<td>((-100, 100)^2)</td>
<td>((50, 100)^2)</td>
<td>2</td>
<td>(10^{-5})</td>
</tr>
<tr>
<td>schweifel 1.2</td>
<td>( f_6 = \sum_{i=1}^{D} (x_i^2)^2 )</td>
<td>((-100, 100)^D)</td>
<td>((50, 100)^D)</td>
<td>30</td>
<td>100</td>
</tr>
</tbody>
</table>

### Table 3: Average optimization success rates across 50 PSO runs, for the function goals defined in table 2. The average number of iterations required to reach the optimization goal is shown in parentheses. For the results shown at the top panel, a local best (ring) topology is used. For the results shown at the bottom panel, a random neighborhood topology of size 3 is used. The frequency parameter \( k \) from equation 6 is set to \( k = 7 \) for the oscillating functions.

Table 5: DISCUSSION
Figure 2: Examples that showcase the effects of different non-monotonic inertia weight functions on the average particle velocity vector magnitude, for the rosenbrock function. From left to right and from top to bottom, parameter $k$ from equation 6 is set to 1, 3, 5 and 7 respectively.

The results indicate that the performance of the oscillating functions is generally competitive to the inertia weight functions from the literature that were also tested. Among the latter, the constant (const) and decreasing (lin dec and nonlin dec) functions achieve satisfactory success rates with good convergence speeds, especially when the ring topology is in use (with the exception of the rastrigin function). Among the decreasing functions, the nonlinear version is slightly better than the linear one. The behaviour of the linearly increasing function is somewhat erratic, while the random inertia weight appears to be more successful when the random neighborhood topology is deployed.

Among the oscillating functions, the performance of osc and osc dec are quite comparable and competitive to the decreasing functions. The step function is less consistent in its success rates but performs particularly well on the rastrigin problem, unlike all other inertia weight functions, especially when using the ring topology.

The switch from the ring topology to the random neighborhood has the effect of generally enhancing the consistency of optimization success rates across all benchmark functions. The success rates for the rastrigin function in particular are significantly increased, compared to those using the ring topology.

Considering the trade-offs between optimization success rates, speed of convergence and consistency of performance across all benchmark functions, the sinusoidal inertia weight function with a constant amplitude and the nonlinearly decreasing function produce the best set of results, when using a random neighborhood topology, with the sinusoidal function converging rather faster.

A comparison between the number of iterations required to satisfy the set goals (shown in parentheses in table 3) and the inertia weight oscillation cycles in figure 1 reveals that, in many cases, these goals are achieved before the oscillatory function completes a full cycle. Indeed, in such cases, an oscillation half-cycle corresponds to a rapid nonlinear inertia weight decrease (exploitation), which appears to be enough to satisfy the required objective function goal.

The influence of the repeated oscillation cycles to the average best particle fitness is displayed in figure 3, for all benchmark functions. In addition to the oscillating functions, the average best particle fitness using the nonlinearly decreasing function has also been plotted for comparison purposes.

The inertia weight adaptation strategy is one the factors influencing the momentum of particles as they fly in the search space attracted by their own and their neighborhood’s best solutions. In this context, an oscillating inertia weight
Figure 3: Average best particle fitness across 50 PSO runs for the six benchmark functions, using the random neighborhood topology, for selected inertia weight functions. The frequency parameter $k$ from equation 6 is set to $k = 7$ for the oscillating functions.
serves the purpose of periodically promoting and suspending particle momentum, or driving the swarm towards global and local search, respectively. This accelerate/decelerate effect is demonstrated in figure 2, where average particle velocity vector magnitude cycles can be seen to correspond to inertia weight function cycles.

Different frequency settings of inertia weight oscillation, by adjusting parameter $k$ in equation 6, were tested but no definite conclusions could be drawn. Variations to the value of $k$ did not appear to have a noticeable impact to the results, as presented in table 3. Only the step inertia weight function benefited from a frequency increase and improved its success rates considerably, when the random neighborhood topology was in use.

Naturally, the momentum of particles is also dependent on the structure and possible dynamics of the swarm’s topology, i.e. the kind of information communicated between particles within a neighborhood, what Kennedy and Mendes call neighborhood consensus [13]. Inertia weight adaptation strategies can, thus, be viewed as a set of global, secondary filters that scale the results of individual particle interactions in different ways.

Finally, we note that none of the adaptation strategies presented and tested in this paper can be considered as being intelligent, in the sense that they do not dynamically respond to particular conditions that would suggest a shift between a local and a global mode of search, but rather enforce a scheme of a globally predefined balance between the two.

6. CONCLUSION

Inertia weight adaptation during the course of optimization is a significant aspect affecting the behaviour of PSO, both in terms of the ability to satisfy set goals and the speed of convergence.

In this paper, we proposed a set of three oscillating inertia weight functions of time that periodically alternate between global and local search waves, implementing a different kind of balance between the swarm’s tendencies towards exploration and exploitation of the search space.

The proposed functions, as well as a number of inertia weight functions from the literature, were tested on the optimization of six standard benchmark problems. The oscillating functions were found to be generally competitive and, in some cases, outperformed established inertia weight adaptation schemes. When considering particular experimental settings, such as a random neighborhood communication topology, the sinusoidal functions achieved very good success rates and convergence speeds, consistently across all benchmark functions.

Further work is needed to investigate how and why the swarm responds to different inertia weight frequency settings, as regulated by parameter $k$ in equation 6. In the suite of tests performed for the purposes of this paper, no definite conclusions could be drawn regarding this issue.

Other non-monotonic inertia weight functions, perhaps oscillating between dynamic upper and lower bounds can also be developed with the possible incorporation of heuristics such as gradient estimations [4] or best performance evaluations [21], so that the inertia weight can adapt according to emerging swarm conditions.

As is the case with the swarm’s communication topology, there seems to be no one-size-fits-all, optimal strategy of adapting the inertia weight during the course of optimization and problem-specific characteristics have to be considered before a decision regarding the selection of a such strategy is made.

7. REFERENCES


