Diversity and frame invariance characteristics in Particle Swarm Optimization with and without digital pheromones

Vijay Kalivarapu  
_Iowa State University_, vkk2@iastate.edu

Eliot H. Winer  
_Iowa State University_, ewiner@iastate.edu

Follow this and additional works at: [http://lib.dr.iastate.edu/me_conf](http://lib.dr.iastate.edu/me_conf)

Part of the [Computer-Aided Engineering and Design Commons](http://lib.dr.iastate.edu/me_conf)

Recommended Citation
[http://lib.dr.iastate.edu/me_conf/157](http://lib.dr.iastate.edu/me_conf/157)

This Conference Proceeding is brought to you for free and open access by the Mechanical Engineering at Iowa State University Digital Repository. It has been accepted for inclusion in Mechanical Engineering Conference Presentations, Papers, and Proceedings by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
Diversity and frame invariance characteristics in Particle Swarm Optimization with and without digital pheromones

Abstract
Academic problems for testing optimization methods are widely criticized for not being a good representative of real world problems. Due to the unavailability of publishable proprietary information from industries they collaborate with, researchers tend to simulate complex problems by using ‘n-dimensional’ multimodal and/or multi-objective academic test problems for evaluating optimization methods developed. Most of these benchmarking test problems can be decomposed and solved as ‘n’ 1-dimensional optimization problems, rendering them as ineffective representation of real-world problems. However, studies show that coordinate rotation of test problems through an arbitrary angle makes design variables dependent on each other and cannot easily be decomposed into simpler problem chunks. Test problems formulated with coordinate rotation therefore will represent a realistic test bed for evaluating the performance of an optimization routine. However with coordinate rotation, the complexity of the problems potentially increases from $O(nn)$ to $O(\exp(n \ln n))$ imposing performance loss on the optimization method that solves the problem. In this paper, the authors attempted to investigate whether coordinate rotation affects the performance characteristics of the digital pheromone implementation of Particle Swarm Optimization (PSO). In particular, two characteristics - swarm diversity with different random number schemes for the velocity vector, and frame invariance with rotational problems are studied and reported. In other words, the authors intended to evaluate whether PSO with digital pheromones is truly capable of solving complex problems.

Keywords
Virtual Reality Applications Center, arbitrary angles, complex problems, design variables, digital pheromones, frame invariance, optimization problems, particle swarm optimization (PSO), shape optimization

Disciplines
Computer-Aided Engineering and Design | Mechanical Engineering

Comments
Diversity and Frame Invariance Characteristics in Particle Swarm Optimization with and without Digital Pheromones

Vijay Kalivarapu\textsuperscript{1} and Eliot Winer\textsuperscript{2}

\textit{Virtual Reality Applications Center, Iowa State University, Ames, IA, 50011, USA}

Academic problems for testing optimization methods are widely criticized for not being a good representative of real world problems. Due to the unavailability of publishable proprietary information from industries they collaborate with, researchers tend to simulate complex problems by using ‘n-dimensional’ multi-modal and/or multi-objective academic test problems for evaluating optimization methods developed. Most of these benchmarking test problems can be decomposed and solved as ‘n’ 1-dimensional optimization problems, rendering them as ineffective representation of real-world problems. However, studies show that coordinate rotation of test problems through an arbitrary angle makes design variables dependent on each other and cannot easily be decomposed into simpler problem chunks. Test problems formulated with coordinate rotation therefore will represent a realistic test bed for evaluating the performance of an optimization routine. However with coordinate rotation, the complexity of the problems potentially increases from $O(n^n)$ to $O(\exp(n \ln n))$ imposing performance loss on the optimization method that solves the problem. In this paper, the authors attempted to investigate whether coordinate rotation affects the performance characteristics of the digital pheromone implementation of Particle Swarm Optimization (PSO). In particular, two characteristics - swarm diversity with different random number schemes for the velocity vector, and frame invariance with rotational problems are studied and reported. In other words, the authors intended to evaluate whether PSO with digital pheromones is truly capable of solving complex problems.

I. Introduction

Particle Swarm Optimization (PSO)\textsuperscript{1,2} is a population based heuristic method retaining many characteristics of evolutionary search algorithms such as a GA. It is a recent addition to global search methods\textsuperscript{3} and one of its key features is its simplicity in implementation due to a small number of parameters to adjust\textsuperscript{4,5}. In a regular PSO, an initial randomly generated population swarm (a collection of particles) propagates towards the global optimum over a series of iterations. Each particle in the swarm explores the design space based on the information provided by two members – the best position of a swarm member in its history trail ($p\text{Best}$), and the best position attained by all particles ($g\text{Best}$) until that iteration. This information is used to generate a velocity vector indicating a search direction towards a promising design point, and the location of each swarm member is updated. The drawback of this approach is that information from these two members alone is not sufficient for the swarm to propagate toward the global optimum efficiently. This either could cause the swarm to lock into a local minimum or take a long time to approach the global optimum. Previous work by the authors demonstrated promising performance improvement of PSO in terms of increased solution efficiency, accuracy, and reliability through

\footnotesize
\textsuperscript{1} Postdoctoral Research Associate and author of correspondence, Department of Mechanical Engineering, Human Computer Interaction, Virtual Reality Applications Center, 2274 Howe Hall, Iowa State University, Ames, IA, 50011, USA, Student Member. Email: vkk2@iastate.edu, vijaykiran@gmail.com, Phone: 515-294-3092, Fax: 515-294-5530.

\textsuperscript{2} Associate Professor, Department of Mechanical Engineering, Human Computer Interaction, Virtual Reality Applications Center, 2274 Howe Hall, Iowa State University, Ames, IA, 50011, USA, Member.
implementing digital pheromones in PSO. Sections I-A through I-C explain the workings of a basic PSO and digital pheromone implementation of PSO.

A. Particle Swarm Optimization

PSO shares many characteristics of evolutionary search algorithms such as Genetic Algorithms (GA) and Simulated Annealing (SA) – a) Initialization with a population of random solutions, b) Design space search for optimum through updating generations and c) Update based on previous generations. The success of the algorithm has brought substantial attention among the research community in the recent past. The working of the algorithm is based on a simplified social model similar to the swarming behavior exhibited by insects and birds. In this analogy, a swarm member uses its own memory and the behavior of the rest of the swarm to determine the suitable location of food (global optimum). The algorithm iteratively updates the direction of the swarm movement toward the global optimum. The mathematical formulation of the method is given in Equations (1) and (2).

\[
V_{iter+1,i} = w_{iter} \times V_{iter,i} + c_1 \times \text{rand}_{p,i}^{iter} \times (pBest_i - X_{iter,i})
\]

\[
X_{iter+1,i} = X_{iter,i} + V_{iter+1,i}
\]

Equation (1) represents the velocity vector update, representing the direction and magnitude of the swarm member in a basic PSO in iteration \( \text{iter} \). Each successive iteration is represented by \( \text{iter} + 1 \). The square braces in equations (1) and (2) indicate an array meaning that the corresponding value (e.g., \( pBest_i \)) is computed for each design variable. \( \text{rand}_{p,i}^{iter} \) and \( \text{rand}_{g,i}^{iter} \) are unique random numbers generated between 0 and 1 for \( pBest \) and \( gBest \) components separately for each swarm member in each iteration. The parentheses in \( \text{rand}(\cdot) \) indicate that they are random number generating functions within the computer code. This ensures swarm diversity, meaning that the search is not linear in an n-dimensional space. More about swarm diversity can be obtained from. \( c_1 \) and \( c_2 \) are user definable confidence parameters. Typically, these are set to values of 2.0. \( pBest \) represents the best position of the particle in its history trail, and \( gBest \) represents the best particle location in the entire swarm. \( w_{iter} \) is termed “inertia” weight, and is used to control the impact of a particle’s previous velocity on the calculation of the current velocity vector. A large value for \( w_{iter} \) facilitates global exploration, which is particularly useful in the initial stages of an optimization. A small value allows for more localized searching, which is useful as the swarm moves toward the neighborhood of the optimum. These characteristics are attributed to the swarm by implementing a decay factor, \( \lambda \), for the inertia weight, as shown in equation (3). Equation (2) denotes the updated swarm location in the design space.

In addition to the originally developed PSO algorithm, significant enhancements have been proposed such as: a) mutation factors for better design space exploration, b) methods for constraint handling, c) parallel implementation, d) methods for solving multi-objective optimization problems, e) methods for solving mixed discrete, integer and continuous variables.

B. Digital Pheromones

Pheromones are chemical scents produced by insects to communicate with each other to find a suitable food source, nesting location, etc. The stronger the pheromone, the more the insects are attracted to the path. A digital pheromone is analogous to an insect generated pheromone in that they are the markers to determine whether or not an area is promising for further investigation. One of the well-known applications of digital pheromones is its use in the automatic adaptive swarm management of Unmanned Aerial Vehicles (UAVs). In this research, the UAVs are automatically guided towards a specific zone or target through releasing digital pheromones in a virtual environment, thereby reducing the requirement of humans physically controlling from ground stations. Other applications of digital pheromones include ant colony optimization for solving minimum cost paths in graphs solving network communication.
problems. The concept of digital pheromones is considerably new and has not yet been explored to its full potential for investigating n-dimensional design spaces for locating an optimum.

In a regular PSO algorithm, the swarm movement obtains design space information from only two components – pBest and gBest. When coupled with an additional pheromone component, the swarm is essentially presented with more information for design space exploration and has a potential to reach the global optimum faster.

C. Overview of Digital Pheromones in PSO

In a basic PSO algorithm, the velocity vector governs the swarm movement computed in Equation (1). Each swarm member uses information from its previous best and the best member in the entire swarm at any iteration.

\[ V_{i} = \omega V_{i-1} + c_1 r_1 (p_{Best} - P_{i}) + c_2 r_2 (g_{Best} - P_{i}) \]

Figure 1. (Left) Particle movement in basic PSO, (Right) Particle movement in digital pheromone PSO

However, multiple pheromones released by the swarm members could provide more information on promising locations within the design space when the information obtained from pBest and gBest are insufficient or inefficient. Figure 1 (left) displays a scenario of a swarm member’s movement whose direction is guided by pBest and gBest alone. The particle’s position in the previous iteration is denoted by \( P_{i-1} \), and the current position is denoted by \( P_{i} \). \( V_{i-1} \) indicates the particle’s velocity in the previous iteration and \( V_{i} \) denotes the velocity in the current iteration. If \( c_1 \gg c_2 \), the particle is attracted primarily towards its personal best position. On the other hand, if \( c_2 \gg c_1 \), the particle is strongly attracted to the gBest position. In the scenario dominated by \( c_2 \), as presented in figure 3 (left), neither pBest nor gBest leads the swarm member to the global optimum, at the very least, not in this iteration adding additional computation to find the optimum. Figure 1 (right) shows the effect of implementing digital pheromones into the velocity vector. An additional target pheromone component potentially causes the swarm member to result in a direction different from the combined influence of pBest and gBest thereby increasing the probability of finding the global optimum.

Figure 2 summarizes the general procedure for PSO, with steps involving digital pheromones highlighted. The method initialization is similar to a basic PSO except that 50% percent of the swarm within the design space is randomly selected to release pheromones in the first iteration. This parameter is user-defined, but experimentation has shown 50% to be a good default value. For subsequent iterations, each swarm member that realizes any improvement in the actual objective function value is allowed to release a pheromone.
Evaluate fitness value of each swarm member

Store pBest and gBest

Decay digital pheromones in the design space (if any)

Randomly chosen 50% of swarm release pheromones

1st iteration?

Only improved particles release pheromones

Merge pheromones based on relative distance between each

Find target pheromone toward which the swarm moves

Update velocity vector and position of the swarm

Converged?

STOP!

Figure 2 Overview of PSO with Digital Pheromones

Pheromones from the current as well as the past iterations that are close to each other in terms of the design variable value are merged into a new pheromone location. Therefore, a pheromone pattern across the design space is created, while keeping the number of pheromones manageable. In addition, the digital pheromones are decayed every iteration just as natural pheromones. Based on the current pheromone level and its position relative to a particle, a ranking process is used to select a target pheromone for each particle in the swarm. This target position towards which a particle will be attracted is called the target pheromone and added as an additional velocity vector component to pBest and gBest. This procedure is continued until a prescribed convergence criterion is satisfied. A detailed account of this procedure is fully explained in the previous work by authors 6, and is not described in this paper to maintain conciseness. The new velocity vector update equation is shown in equation (4).
and are unique random numbers generated between 0 and 1 for pBest, gBest and target pheromone components separately for each swarm member in each iteration. The parentheses in rand_{p,i}^{iter+1}, rand_{g,i}^{iter+1} and rand_{T,i}^{iter+1} indicate that they are random number generating functions within the computer code. These values are updated in each iteration within the velocity vector equation, resulting in improved swarm diversity. The square braces in equation (4) indicate an array meaning that the corresponding value (e.g., TargetPheromone) is computed for all design variables.

c_3 is a user defined confidence parameter for the pheromone component of the velocity vector similar to c_1 and c_2 in a basic PSO. c_3 combines the knowledge from the cognitive and social components of the velocity of a particle, and complements their deficiencies. In a basic PSO, the particle swarm does not have a memory of the entire path traversed in the design space apart from the best position of an individual particle (pBest) and the best member’s position in the entire swarm (gBest). The target pheromone component addresses this issue. It is a container that functionally stores the trail path of the swarm and utilizes the best features of pBest and gBest in steering towards a promising location in the design space. The confidence parameter c_3 determines the extent of influence a target pheromone can have on the swarm when the information from pBest and gBest alone are not sufficient or efficient to determine a particle’s next move. The use of the target pheromone relies heavily on pBest and gBest. If c_3 = 0, there is no influence of pheromones and the swarm behaves as if in a regular PSO. If either of c_1 or c_2 is 0 and c_3 > 0, then the target pheromone location is essentially determined only by the non-zero component of pBest or gBest and propagated into the velocity vector. This creates a bias thereby doubling the influence of non-zero pBest or gBest components on the swarm. This means that the swarm either explores or exploits the design space with double the intensity, either of which will prevent the swarm from converging. It is therefore essential that the influence of pBest and gBest be balanced (i.e. equal) for the pheromone component to provide accurate assistance in reaching the optimum.

Although analytical determination of a value for c_3 is out of the scope of this research, an empirical value has been determined through experimentation. A value between 2.0 and 5.0 has shown good performance characteristics and solved a variety of problems. An inertia weight, w_i of value 1.0 is initially chosen to preserve the influence of the velocity vector from previous iterations, and gradually decreased using an inertia weight decay factor similar to the one used in a basic PSO.

Digital pheromone implementation in PSO yielded promising performance improvements in terms of solution accuracy, efficiency and reliability on standard benchmarking test problems. Further enhancements in performance were realized through developing parallel cluster computing schemes 27, 28. Two different Graphics Processing Unit (GPU) implementations were also made 29, 30 to leverage the agility of current day graphics cards. Also, a quantitative assessment has also been made through statistical hypothesis testing 31.

D. Issues with Academic type Problems

Thus far, the methods developed by the authors were tested on purely academic type problems (e.g., De-Jong’s functions, Ackley’s path function, Griewangk’s function, Rastrigin’s function etc).

\[
F(X) = \sum_{i=0}^{n} f_i(x_i)
\]
Most of these test functions are decomposable into ‘n’ 1-D problems. Equation (5) is an illustration of an n-dimensional function that can be decomposed into ‘n’ functions not necessarily identical. The design variables represented by \( x_i \) in such functions can be termed as independent, making it possible to optimize the function in a series of ‘n’ independent optimization procedures. Although highly multi-modal academic type problems are quite complex, a closer look reveals that the function can be decomposed into ‘n’ 1-D problems, as shown in the Ackley’s path function in equation (6). The combination of the solutions obtained from solving each 1-D problem then forms the solution vector to the original \( F(X) \).

\[
F(x) = -a \cdot e^{-b \left( \frac{\sum_{i=1}^{n} x_i^2}{n} \right)} - e^{-\frac{\sum_{i=1}^{n} \cos(cx_i)}{n}} + a + e^1
\]

\( a = 20; \quad b = 0.2; \quad c = 2 \cdot PI; \quad i = 1 : n; \)

\[-32.768 \leq x_i \leq 32.768\]  

Real-world problems do not necessarily present themselves as decomposable, and therefore optimization methods should be robust enough to be able to solve them in spite of having dependent design variables. Since publishable industry problems are not always available to academicians and researchers, it will be useful to formulate academic problems to simulate real-world problems. One method to do this is through coordinate rotation.

**E. Coordinate Rotation**

Coordinate rotation is not a new concept. Matrix rotation operations, for instance, are extensively used in various fields of physics, computer graphics, etc. A two-dimensional rotational transformation is shown in equation (7), where two variables X and Y are rotated by an angle \( \theta \) resulting in transformed variables \( X' \) and \( Y' \).

\[
\begin{bmatrix}
X' \\
Y'
\end{bmatrix} = \begin{bmatrix}
\cos(\theta) & \sin(\theta) \\
-\sin(\theta) & \cos(\theta)
\end{bmatrix} \times \begin{bmatrix}
X \\
Y
\end{bmatrix}
\]

It might seem counter-intuitive how rotations can assist optimization routines. Figure (3) illustrates a hypothetical two variable objective function contour aligned to coordinate axes. The improvement in objective function values in this case is much simpler and faster as the optimization routine progresses. However, when the design variable axes are rotated by an angle (described by a rotation matrix \( M \)), the improvement in objective function values is much smaller and slower. This can be thought of as an optimization routine requiring more iterations and hence more objective function evaluations. In population based methods such as a GA or PSO, the complexity in solving such rotated problems is much higher when compared to problems that are decomposable and axes aligned.

![Figure 3](image-url)  
*Figure 3. (Left) Improvement in objective function value easier when design space is aligned to coordinate axes, (Right) Smaller improvements realized when coordinate axes are rotated*
Salomon \cite{salomon32} explains that the complexity in optimizing decomposable objective functions is only $O(n)$. He also explains that the complexity increases to $O(n \ln n)$ when a GA uses a small mutation probability, but given GA’s random nature, the increase is not very significant, and that the parameter settings in GAs are optimal to solve problems with independent design variables, i.e., decomposable problems. However, when a coordination rotation is introduced, the design variables become dependent meaning that a linear search in n-Dimensions will not suffice to arrive at the global optimum, the type of problems that are a representative of real world practical problems. The order of complexity for rotated problems could go as high as $O(\exp(n \ln n))$ inducing a severe performance loss in GA.

**F. Linear and Classical PSO**

Although a basic PSO is a simple algorithm to implement, there is a subtle implementation variation that can cause it to dramatically alter its performance metrics. Linear PSO, as Wilke in his notes on diversity \cite{wilke33} terms it, is different from a classical PSO only in the way random numbers for $pBest$ and $gBest$ are generated. Figure 4 explains the implementation difference.

```plaintext
for (i=0; i<population size; i++)
    do
        $R_1 = \text{random number: [0,1]}$
        $R_2 = \text{random number: [0,1]}$
        for (j=0; j<dimensions; j++)
            do
                velocity[i][j] =
                $W*velocity[i][j] +$
                $c_1*R_1*(pB[i][j] - X[i][j]) +$
                $c_2*R_2*(gB[i] - X[i][j])$
            end do
    end do
for (i=0; i<population size; i++)
    do
        for (j=0; j<dimensions; j++)
            do
                $R_1 = \text{random number: [0,1]}$
                $R_2 = \text{random number: [0,1]}$
                velocity[i][j] =
                $W*velocity[i][j] +$
                $c_1*R_1*(pB[i][j] - X[i][j]) +$
                $c_2*R_2*(gB[i] - X[i][j])$
            end do
    end do
```

**Figure 4.** (Left) Linear PSO, (Right) Classical PSO

The pseudo code in figure 4 illustrates a linear PSO (left) and a classical PSO (right). In a linear PSO, the random numbers scale $pBest$ (cognitive) and $gBest$ (social) components together, while their influence is individual and separate in a classical PSO. While both implementations are valid, a classical PSO causes the swarm search to be much more diversified than the linear PSO. Particle paths in a linear PSO collapse to line searches in n-dimensional space, i.e., the particle diversity is hampered. Wilke, in his second paper on linear and classical PSO comparison \cite{wilke34} establishes that a linear PSO is frame invariant while classical PSO is not. This means that a linear PSO, while incapable of being diverse, is able to solve problems with dependent design variables or those subject to coordinate rotation while a classical PSO does not.

**G. Diverse Rotationally Invariant (DRI) Scheme**

Although linear PSO (reduced diversity) and classical PSO (incapable of solving rotated problems) are subjected to a form of performance loss, Wilke claimed that they are not necessarily exclusive. He proposed a form of linear variant of PSO that can yet be diverse in a stochastic sense - Diverse Rotationally Invariant scheme \cite{wilke34}. There are other efforts in the literature that attempted to solve rotational problems. For example, Liang et al. \cite{liang35} attempted to use academic problems subject to coordinate rotation as test cases to evaluate comprehensive learning PSO method they developed.

Given that the digital pheromone implementation of PSO only adds another component to the velocity vector equation keeping other particle characteristics similar to a basic PSO, it is possible that the pheromone PSO also suffers similar setbacks as linear and classic variants of PSO. However, the authors theorize that the use of pheromones within a PSO induces significant diversity in the swarm to accurately, efficiently and reliably approach the neighborhood of the optimum. While analytical derivation was out of the scope of the paper, the authors chose to test digital pheromone PSO on a test bed similar to Wilke to get reasonable performance comparisons.

With perspectives from different researchers, it can be concluded that methods that can solve academic test problems subject to coordinate rotation are fairly comparable to solving practical complex optimization
problems. The authors attempted to evaluate PSO with digital pheromones to solve academic problems subject to coordinate rotation and presented the findings in this paper.

II. Methodology

A. Orthogonal and Rotational Matrices

An orthogonal matrix is a square matrix of the form $n \times n$ with its columns or rows forming unit orthogonal vectors. A matrix is considered orthogonal if it follows two rules: a) Its transpose is equal to its inverse, and b) its determinant is $\pm 1.0$ shown in equations (8) and (9).

\[
\]
\[
[M] = \pm 1.0
\] (8)

Those orthogonal matrices that result in a determinant of $+1.0$ are called as ‘special orthogonal group’ and can act as rotational matrix. More on orthogonal matrices can be found here\(^3\).

B. Orthogonal Rotational Matrix Construction

Literature shows multiple ways to construct orthogonal matrices\(^3\),\(^3\). The authors chose Salomon’s method to create an orthogonal matrix due to its simplicity in implementation. Although Salomon’s algorithm for constructing an orthogonal matrix was in Fortran programming language, it was altered to suit to C++ coding style, the programming environment the authors used. Figure 5 shows a pseudo code for creating a rotational orthogonal matrix.

```c
// Define three identity matrices of size equal to ‘n x n’
// n is the dimensionality of the test problem
Matrix ME, MTurn, MResult

// Initialize random seed
srand( (unsigned) time(0) )

// Functions
rotateMatrix(MTurn, i, j)
    alpha = (drand48() - 0.5)*PI*0.5
    MTurn[i, i] = cos(alpha)
    MTurn[j, j] = cos(alpha)
    MTurn[i, j] = sin(alpha)
    MTurn[j, i] = -sin(alpha)

// Main routine. MResult stores the rotated matrix
for(i=1; i<n-1; i++)
do
    rotateMatrix(MTurn, 0, i)
    MResult = MResult*MTurn
end do

for(i=1; i<n-2; i++)
do
    rotateMatrix(MTurn, i, n-1)
    MResult = MResult*MTurn
end do
```

**Figure 5.** Pseudo code for creating Salomon’s Rotational Matrix

Although Salomon’s algorithm constructs a pure rotational matrix, its generation is computationally intensive due to $(n-1)*(n-2)$ matrix-matrix multiplication. A computationally feasible alternative is
discussed in Wilke’s notes on frame invariance \(^{34}\), albeit lesser accurate than Salomon’s algorithm. For this paper, Salomon’s orthogonal matrix was used to create a pure rotation matrix once per trial run to rotate the coordinate frame of the objective function. Wilke’s rotational matrix was used once per iteration for each design variable to implement a Diverse Rotationally Invariant (DRI) scheme for PSO. The results section (Section III) explains the need for using DRI scheme.

C. Rotation of Objective Function Coordinate Frames

An orthogonal rotational matrix \(M\) of size \(n \times n\) was pre-constructed for each problem and stored, where ‘\(n\)’ indicates the dimensionality of the test problem. The design variable vector is pre-multiplied by the rotational matrix to result in a new set of rotated design variable vector, as shown in equation (10).

\[
\begin{bmatrix}
  m_{1x1} & \cdots & m_{1xn} \\
  \vdots & \ddots & \vdots \\
  m_{nx1} & \cdots & m_{nxn}
\end{bmatrix}
\begin{bmatrix}
  X_1 \\
  \vdots \\
  X_n
\end{bmatrix}
=
\begin{bmatrix}
  Y_1 \\
  \vdots \\
  Y_n
\end{bmatrix}
\]

Therefore if \([X] = [x_1, x_2, \ldots, x_n]^T\), the rotational matrix transforms the design variables to form \([Y] = [y_1, y_2, \ldots, y_n]^T\), where \(y_i = m_{ix1}x_1 + m_{ix2}x_2 + \ldots + m_{inx}x_n\), \(i = 1, 2, \ldots, n\). The matrix \([M]\) is pure rotation, and is constructed once per trial run with randomized initial values for rotational angles. The design variable vector \([Y]\) is used instead of \([X]\) every time the fitness value is computed in the optimization routine. The use of rotated design variable vector only changes the coordinate axes against which the objective function is aligned and does not alter the shape of the objective function in any manner.

III. Results

A. Evaluating Digital Pheromone PSO for Rotational Problems

Wilke has analytically established in his work that a classical implementation of basic PSO is rotationally variant. That means, it induces performance loss on problems with dependent design variables or rotational problems. Similarly, a linear implementation of basic PSO lacks diversity characteristics. That is, particle search reduce to line search in \(n\)-dimensional design spaces. Digital pheromone PSO is likely subjected to similar setbacks but the authors theorize that pheromones add significant diversity to the swarm to accurately, efficiently and reliably approach the neighborhood of the optimum. While analytical derivation was out of the scope of the paper, the authors chose to test digital pheromone PSO on a similar test bed as Wilke to get unbiased performance comparisons.

As such, a variety of scenarios were considered for testing. Parallels were drawn against linear and classic basic PSO implementations. Also, solution accuracies of basic and digital pheromone PSO were compared in rotated and un-rotated coordinate frames as well. In total, 24 scenarios were tested and figure 6 shows the layout.
B. PSO Heuristics

The heuristics: (a) Cap on the velocity vector values, (b) Particle position limits as a function of design variable bounds, and (c) dynamic inertia weight, may interfere with how PSO performs upon coordinate rotation. Therefore, one of the scenario for testing included no PSO heuristics for basic and digital pheromone implementations. A scenario with heuristics was also included to check how a fully functional digital pheromone PSO measured against basic PSO for rotational problems. Un-rotated functions were also included in the testing scenario to compare solution accuracies with rotated problems.

C. Test Problem Settings

Three 30 dimensional problems were used as test cases for executing all 24 scenarios listed in section III A. Table 1 Test problem Matrix shows the list of test problems along with their published solutions and design variable bounds. Full mathematical descriptions for these problems can be found in [38-40].

<table>
<thead>
<tr>
<th>Problem</th>
<th>Test Problem</th>
<th>Published Solution</th>
<th>Dimensions</th>
<th>Design variable bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Quadratic function</td>
<td>0.000</td>
<td>30</td>
<td>± 100.0</td>
</tr>
<tr>
<td>2</td>
<td>Ackley’s path function</td>
<td>0.000</td>
<td>30</td>
<td>± 30.0</td>
</tr>
<tr>
<td>3</td>
<td>Griewank function</td>
<td>0.000</td>
<td>30</td>
<td>± 500.0</td>
</tr>
</tbody>
</table>

The digital pheromone parameters $c_1 = 5.0$ and a pheromone decay $\lambda_p = 5\%$ as established previously by the authors were used in all scenarios. These parameter settings were upheld for scenarios with and without PSO heuristics because they are the underlying characteristics of a digital pheromone model. However, the move limit decay $\lambda_{ML}$ of 5%, which is used for velocity vector computation was not included in the scenario without PSO heuristics. A total of 20 trial runs were conducted for all scenarios and the swarm size was set as 20 to be consistent with Wilke’s implementation of PSO. The problems tested were considered converged when the difference in solutions was within a tolerance of 0.01 for ten consecutive iterations.

The computing platform used was an eight-core Intel Xeon processor (3.2 GHz) Linux workstation with RedHat Enterprise Linux 5 edition operating system. The code was however single threaded and only one
processor core was used for testing the performance. The system memory was 4GB DDR2. The PSO algorithm was implemented in C++ programming language.

D. Comparison of Results from Different Scenarios

a. Linear Implementation without Heuristics

Figure 7 (a) and (b) compares results between basic and pheromone linear implementations of PSO for un-rotated and rotated test problems without heuristics. Although un-rotated test cases do not assess the performance of digital pheromone or basic PSO for solving problems with dependent design variables, their results are gathered to provide valuable insights on how one implementation compares against the other.

![Figure 7](image)

**Figure 7.** Comparison of linear variants of Basic and Pheromone PSO without heuristics for (a) Unrotated, and (b) Rotated problems

Figure 7(a) shows the results of a linear implementation with no coordinate rotation on the test problems. Figure 7(b) Results from coordinate rotation of the design variables using Salomon’s orthogonal matrix for linear PSO.

It can be seen that basic and pheromone PSO performed almost equally well (12.95 and 12.38) without coordinate rotation. The difference in performance was much apparent in solving the Griewank’s function. However in both cases, the methods failed to reach the global optimum. A similar explanation can be attributed to the results from rotated test problems from figure 7(b) as well. On Ackley’s path function, it can be seen that the basic PSO (13.05) performed worse than pheromone PSO (11.62). However, pheromone PSO (46.57) failed to be of any merit when compared to basic PSO (26.62) on Griewank’s function.

For un-rotated problems, the quadric 30D test case averaged to 3981.6 for linear basic PSO, while it was 26571.3 for pheromone PSO. For rotated problems, the quadric 30D test case averaged to 4674.3 using basic PSO and 37280.1 using pheromone PSO.

Regardless of the un-rotated or rotated variants, the methods did not arrive at the neighborhood of the global minimum. This is expected because the linear implementation of PSO (basic and pheromone) lacks sufficient swarm diversity, and is consistent with the findings of Wilke.

b. Linear Implementation with Heuristics

Figure 7 (a) and (b) compares results between basic and pheromone linear implementations of PSO for un-rotated and rotated test problems with heuristics.
Figure 8. Comparison of linear variants of Basic and Pheromone PSO with PSO heuristics for (a) Unrotated, and (b) Rotated test problems

It can be seen from the figures 8(a) and (b) that the basic PSO failed to solve Ackley’s 30D and Griewank’s 30D problems. However, the pheromone PSO was able to arrive at the neighborhood of global optimum with a greater accuracy. These findings are consistent with Wilke’s observations that linear PSO is rotationally invariant. That means, although the solution accuracy is low, the solution values for un-rotated and rotated problems are fairly consistent.

On the other hand, incorporating PSO heuristics into the linear variant of pheromone PSO averaged at a solution value very close to the global minimum (2.18 and 2.15 on Ackley, 0.06 and 0.03 on Griewank). This suggests that heuristics such as limit on the maximum velocity vector, and particle positions have a positive influence on pheromone PSO in solving various test problems.

On the quadric 30D function, a similar pattern was observed. The basic implementation resulted in a solution of 11,0297.2 for the quadric function, while the pheromone PSO resulted in 850.15. A bar plot for the Quadric function was not included in figure 8 because the solution values were too high for all results to be legible on the plot. The linear implementation did not reach the global optimum but was consistent with Wilke’s notes on diversity.

c. Classical Implementation Without Heuristics

Figures 9 (a) and (b) show the classic implementation of basic and pheromone PSO without heuristics for un-rotated and rotated test problems.

Figure 9. Comparison of classic variants of Basic and Pheromone PSO without heuristics for (a) Unrotated, and (b) Rotated test problems

It can be seen from figures 9(a) and (b) that the solution accuracy characteristics on basic PSO for Ackley’s path function was better for un-rotated case (3.04) when compared to rotated variant (7.87). While this can be attributed to the reason that classic PSO is rotationally variant, the Griewank’s function was an exception where basic PSO was able to solve it with a high accuracy on both un-rotated and rotated variants. Although the solution values do not match too close to Wilke’s implementation for basic PSO, the general patterns of results agree with his findings. On the other hand, digital pheromone PSO failed to solve both Ackley and Griewank functions on both on un-rotated and rotated variants. On the Quadric function,
basic PSO was able to achieve an average solution of 1.36 without coordinate rotation and 0.01 with coordinate rotation. However, the pheromone PSO failed to solve it and returned an average 159383.9 without coordinate rotation and 181552.4 with coordinate rotation.

d. Classical Implementation With Heuristics

Figures 10 (a) and (b) compare the classic implementation of basic and pheromone PSO with heuristics for un-rotated and rotated test problems.

![Comparison of classic variants of Basic and Pheromone PSO with heuristics for (a) Un-rotated and (b) Rotated test problems](image)

Figure 10. Comparison of classic variants of Basic and Pheromone PSO with heuristics for (a) Un-rotated and (b) Rotated test problems

When PSO heuristics were introduced, the behavior of basic and pheromone PSO reversed in comparison to PSO without heuristics. The behavior followed a similar pattern as linear variants of PSO. While basic PSO failed to solve Ackley and Griewank’s function with and without heuristics, pheromone PSO was able to achieve superior solution accuracy both without and with coordinate rotation.

While the average solution values from two rotated test problems are not a representative of problems with dependent design variables in general, the results suggest that a classic implementation of pheromone PSO shows promise of having significant swarm diversity. That said, the pheromone implementation resulted in an average solution value of 1006.6 on Quadric problem without coordinate rotation. With coordinate rotation, the solution value achieved was 881.1. Although these values are not any close to the published solution of the Quadric function, they are quite superior than basic PSO that resulted in solutions of 112323.1 (un-rotated) and 106757.9 (rotated).

e. DRI Implementation Without Heuristics

Figures 11 (a) and (b) illustrate the results from DRI implementation within basic and pheromone PSO for un-rotated and rotated problems without heuristics.

![Comparison of DRI variants of Basic and Pheromone PSO without heuristics for (a) Un-rotated and (b) Rotated test problems](image)

Figure 11. Comparison of DRI variants of Basic and Pheromone PSO without heuristics for (a) Un-rotated and (b) Rotated test problems

The figures show that the DRI implementation significantly improved the solution accuracies of basic PSO when compared to its linear and classical counterparts. While the basic PSO in its classical implementation was at 20.1 for rotated problems, DRI implementation resulted in a solution accuracy of 4.30 for Ackley’s test problem. Griewank’s problem was solved with superior accuracy using basic PSO.
Just similar to the other scenarios, pheromone PSO was unable to solve Ackley and Griewank function. However, the solution accuracies improved by almost 25% on Griewank’s function (67.1 for linear and 50.18 for DRI). Similarly on the classical implementation without heuristics, the solution accuracy improvement was ~89% for DRI implementation. This suggests that the DRI implementation clearly improved the solution characteristics by increasing the swarm diversity within a pheromone PSO. The basic PSO also found similar improvement characteristics.

f. DRI Implementation With Heuristics

Figures 12 (a) and (b) illustrate the results from DRI implementation within basic and pheromone PSO for un-rotated and rotated problems with heuristics.

![Graphs showing comparison of DRI variants of basic and pheromone PSO with heuristics for (a) un-rotated, and (b) rotated test problems.]

Figure 12. Comparison of DRI variants of basic and pheromone PSO with heuristics for (a) un-rotated, and (b) rotated test problems

When heuristics are not added to the algorithm, DRI variant of basic PSO outperformed pheromone PSO, as seen in figure 11. However, pheromone PSO performed better with heuristics. This finding follows the same pattern as linear and classic implementation schemes discussed in sections III-D-a, III-D-b, III-D-c and III-D-d above.

It can also be observed from the figures above that the pheromone PSO is benefited by the stochastic diversity introduced by the DRI scheme with heuristics, although the classical implementation of pheromone PSO with heuristics arrived at a more accurate solution.

However, these findings for pheromone PSO are not consistent with Wilke’s findings with PSO heuristics turned off. When the heuristics are turned off, basic PSO consistently performed better in terms of solution accuracy than pheromone PSO in all of the cases considered until now.

IV. Discussion, Conclusions and Future work

When implemented without any heuristics, the behavior pattern of basic PSO closely followed Wilke’s findings on rotational invariance in all implementations – linear, classic and DRI. Linear basic and pheromone PSO mostly agreed with Wilke’s findings, meaning that the linear variant is insulated by coordinate orientation of the test problems, but lacked sufficient diversity to reach the neighborhood of global optimum. Wilke’s findings were only loosely followed by pheromone PSO, meaning that this pattern of behavior was not found on all test problems.

Implementing DRI scheme within PSO was clearly beneficial to both basic and pheromone PSO and has fairly improved swarm diversity and hence closer solution accuracy, albeit without any heuristics. While the heuristics proved to be detrimental to basic PSO, they did improve the solution accuracy characteristics of pheromone PSO by many orders of magnitude.

The results show that the use of digital pheromones improved the diversity in swarm (with and without coordinate rotation) only when PSO heuristics were introduced. In fact, in the two test problems – Ackley and Griewank, the classical implementation of pheromone PSO exceeded the solution accuracies compared to its linear and DRI counterparts. The pheromone PSO however failed to solve the quadric function with a reasonable accuracy, making it hard to draw a generalized conclusion, although it loosely followed the pattern of Ackley and Griewank test problems. Further, deducing a general conclusion is made complicated
by the fact that basic PSO showed superior performance capabilities only when the heuristics were turned off.

Coordinate rotation result in a new class of test problems than academicians and researchers are typically used to for testing the performance metrics of developed methods. Preliminary results suggest that digital pheromone PSO can address rotational test problems when PSO heuristics are used. The authors could draw a general conclusion that digital pheromones in PSO with the use of heuristics can better address rotational problems than a basic PSO. However, this study used limited number of test problems and further study with more decomposable test problems and with higher dimensions can help make the decision process easier, and hence is a short-term future work for this research.

References


